A Flexible Explanation System for Case-Based Reasoning

Conor Nugent

A thesis submitted to the University of Dublin, Trinity College

in fulfillment of the requirements for the degree of

Doctor of Philosophy

October 2005

Declaration

I, the undersigned, declare that this work has not previously been submitted to this or any other University, and that unless otherwise stated, it is entirely my own work. This thesis may be borrowed or copied upon request with the permission of the Librarian, University of Dublin, Trinity College.

Conor Nugent

Dated: March 14, 2006

Acknowledgements

Firstly, I would like to express my gratitude to my supervisor Prof. Pádraig Cunningham for having faith in me and giving me this great opportunity. His encouragement, patience, help and expert advice over the years has been invaluable.

Secondly, I would like to thank all the members of the MLG; Nadia Bolshakova, Kenneth Bryan, Peter Byrne, Michael Carney, Patrick Clerkin, Lorcan Coyle, Michael Davy, Sarah Jane Delany, Chris Fairclough, Derek Greene, Marco Grimaldi, Conor Hayes, Deirdre Hogan, Brian Mac Namee, Neil McDonnell, François-Xavier Pasquier, Matthew Sammon, Alexey Tsymbal and Anton Zamolotskikh. The best thing about working in the MLG has been the people in it. The last three years have been tremendous fun and it has been a pleasure to work and socialise with such a nice bunch. I would like to thank Lorcan, Brian, Marco and Pat for welcoming me into the group and making the first year such good fun. In particular I would like to thank Lorcan for all his help, encouragement and advice over the years.

John, Dónal and Myself all started in the group together. I count myself lucky to have met two such great guys who have been fantastic friends. I can't thank you both enough for all that you have done for me in the last few years.

More than anyone else I owe a debt of gratitude to my family and in particular my parents for their fantastic support throughout my life. You've both been an inspiration to me and this thesis is dedicated entirely to you both.

Finally I would like to thank Lynsey for her help and support and for making what could have been a pretty difficult year a very special one.

Conor Nugent

University of Dublin, Trinity College

October 2005

Abstract

Artificial Intelligence systems have reached a level of sophistication and accuracy which means that they can now be confidently used as decision aids in real world situations. However, although these systems are accurate, users are still reluctant to use them. Unlike domain experts such systems are unable to present convincing explanations in support of the recommendations that they make. This is a major shortcoming, as without this reassurance, people are naturally reluctant to accept unsupported recommendations.

This is an issue which has plagued the Artificial Intelligence community since the early work on Expert Systems in the 1970s. Early solutions to this problem involved the presentation of rules used in the reasoning mechanism of the system to the user as a means of explanation. Such efforts were of limited success, as the rules used by the system were not easily interpretable and served as poor explanations. However, despite this, Rule-based explanations are still commonly used today.

Case-based Reasoning explanations represent an alternative approach which has inherent advantages in terms of transparency and user acceptability. Case-based reasoning explanations are based on a naturalistic concept of presenting similar past examples in support of and as justification for recommendations made. However, the traditional approach in Case-based Reasoning applications of simply supplying the nearest neighbour has been found to have shortcomings in terms of providing satisfactory explanations. The relevance of the explanation case may not be clear to the end user as it is retrieved using domain knowledge which they themselves may not have.

In this thesis we describe a framework for generating convincing Case-based explanations which addresses the shortcomings of the traditional approach in a flexible and adaptable manner. Our Framework finds cases that form the most convincing arguments and is able to explain the relevance of the selected case in terms of discursive text. By providing useful explanatory feedback we hope to instil greater confidence in the user of

CBR system's recommendations. The framework we have developed uses a localised solution which is in keeping with the CBR philosophy and uses Logistic Regression models to extract information useful to the explanation process. This approach supports continued incremental learning and eliminates the need for expert domain knowledge. We have carried out a user evaluation of our explanation framework to establish its effectiveness.

Another issue with the deployment of decision support systems is the maintenance of user confidence in the long term. Although the provision of convincing explanations may help instil confidence in the user in the short term, if the system makes mistakes that confidence will be quickly lost. As a means of ameliorating this damage, measures of confidence in a systems prediction can be given to alert the user to when the system might be making a mistake. The Framework we have developed also has potential in terms of assessing the level of confidence that should be placed in recommendations.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

"Computers are useless. They can only give you answers." Pablo Picasso

Computer technology has come a long way from the time when it simply comprised of giant machines used to crunch numbers and churn out solutions to calculations that were too tedious and time-consuming to be carried out by people. Computer systems are now routinely used to perform tasks that we ourselves don't fully understand and in contexts we would never have before envisioned. As computer systems are entrusted with ever more critical tasks and play more prominent roles in everyday life there has been a demand for such systems to do more than merely produce solutions and carry out their duties silently. In many cases, people using such systems are suspicious of them and reluctant to accept without question the solutions they produce. This problem is typified in a very dramatic fashion in Douglas Adam's The Hitchhikers Guide to the Galaxy. A massively powerful computer called Deep Thought is designed and set the task of calculating the answer to the Ultimate Question of Life, the Universe, and Everything. When Deep Thought returns its answer, "42", its creators are understandably irate, confused and disappointed. They immediately seek an explanation as to what the question was so that they can understand the answer. Although extreme, Deep Thoughts response exemplifies a problem which is common in many computer systems today. As the tasks set to be performed by computer systems have become more sophisticated and ambitious, the emphasis has shifted away from their ability to simply produce solutions to their ability to explain and justify those solutions.

Machine learning research has proven successful in developing increasingly more accurate decision support systems capable of reliably acting as decision aids for human users.

Although more reliable systems are welcomed, the issue of providing useful human interpretable explanations has been largely ignored. Where explanations are provided by decision aid systems they are usually in the form of rules. Such rules, although useful in the internal reasoning mechanism of the system, are often not interpretable by users and serve as poor explanations. This, not surprisingly, has proven to be a major stumbling block in the application of such systems to real world scenarios (Andrews et al. 1995). In this thesis we describe a framework for generating convincing explanations for Case-based Reasoning (CBR) systems. By providing useful explanatory feedback we hope to instil greater confidence in the CBR system's recommendations in the user.

1.1 Case-based Reasoning

Case-based Reasoning is an artificial intelligence methodology for solving problems by using or adapting solutions to old problems (Riesbeck and Schank 1989). Unlike other machine learning techniques CBR does not try to model the problem domain directly, instead a set of past examples called cases are retained. Each case is made up of a description of a past example or experience and its attached solution. The full set of past experiences encapsulated in cases is called the case-base. When a new problem is presented the case-base is searched and similar past examples are used to find a solution.

CBR is referred to as a *lazy* learning technique as the "learning" process isn't carried out until a problem is presented and the case-base is searched. Other machine learning techniques are referred to as *eager* learners as they build models of the problem domain in advance of any problem being presented and use the model to make future recommendations. The approach to learning taken by CBR has a number of advantages that distinguish it from other machine learning techniques.

The CBR methodology also supports incremental learning. Mistakes the system makes can be quickly corrected by adding a case representing the correct solution to a problem. Other techniques would require that the domain model be recompiled. CBR has also proven to be successful in solving *weak-theory* problems. These are problems for which little insight exists and where the problem domain may be complex. Another important advantage of CBR that is often cited is in terms of user acceptance, Leake (1996):

> *"...neural network systems cannot provide explanations of their decisions and rule-based systems must explain their decisions by reference to their rules,*

> *which the user may not fully understand or accept. On the other hand, the results of CBR systems are based on actual prior cases that can be presented to the user to provide compelling support for the system's conclusions."*

However, although CBR does inherently have advantages in terms of providing user feedback, the case-based solution isn't without fault and there are still outstanding issues that need to be addressed.

1.2 Contributions of this Thesis

The primary contribution of this thesis is the development of a flexible explanation framework for Case-Based Reasoning. The Framework was developed in response to the increased demand for machine learning tools that can offer explanations of their recommendations and so instil greater confidence in those recommendations in the user. Although CBR has inherent advantages in terms of providing explanatory feedback it has been recognised that the traditional approach to CBR explanations is not a perfect solution and that there are issues to be addressed. The framework we have developed addresses these issues in binary classification tasks while maintaining the strengths of the CBR approach. Our approach has a number of novel features;

- it applies a *lazy* approach to explanation production that utilises localised logistic regression models,
- it selects cases that form the most convincing arguments in an automated way and without domain knowledge,
- it provides discursive texts describing the effects of differences in feature-values between the Query Case and the Explanation Case without prior domain knowledge.
- it provides a measure of confidence in the system's prediction to alert the user to when there is doubt about a system decision,
- it is not resticted purly to CBR systems and we demonstrate that Case-based reasoning explanations can be used to provide explanaions for Black Box machine learning algorithms.

1.3 Publications Related to this Thesis

- Conor Nugent, Pádraig Cunningham and Dónal Doyle.: 2005, The Best Way to Instil Confidence is by Being Right; An Evaluation of the Effectiveness of Case-Based Explanations in providing User Confidence, *6th International Conference on Case-Based Reasoning,* eds. H.Muñoz-Avila and F. Ricci, pp368-381, Springer LNAI 3620.

- Conor Nugent, Dónal Doyle and Pádraig Cunningham.: Gaining Insight through Case-Based Explanation. To appear in *CBR in Knowledge Discovery and Data Mining*, eds. S. Pal, D. Aha and K. Moy Gupta, Wiley.

- Conor Nugent and Pádraig Cunningham.: A Case-Based Explanation System for Black Box Systems. To appear in *Artificial Intelligence Review* , ed. D. McSherry, D. Leake.

- Dónal Doyle, John Loughrey, Conor Nugent, Lorcan Coyle, Pádraig Cunningham.: 2005, Fionn: A Framework for Developing CBR Systems, in *Expert Update* 8(1), 11-14.

- Conor Nugent and Pádraig Cunningham.: 2004, A Case-Based Explanation System for 'Black Box Systems'; In Gervs, P. & Gupta, K.M. (eds.) Proceedings of the ECCBR 2004 Workshops, Technical Report 142-04, Departamento de Sistemas Informticos y Programacin, Universidad Complutense de Madrid, Madrid, Spain. 165-174.

1.4 Summary and Structure of this Thesis

In brief: Chapter 2 describes how explanations are produced in other machine learning techniques. Chapter 3 describes the case-based reasoning methodology, reviews the history of explanations in machine learning and in particular CBR. Chapter 4 describes the Explanation Framework we have developed. Chapter 5 describes how our Explanation Framework was implemented. Chapter 6 contains evaluations of our Explanation Framework and the thesis is concluded with Chapter 7.

In more detail: In Chapter 2 we briefly describe how providing explanations has been approached in other machine learning technologies as well as highlighting the shortcomings and limitations of such approaches.

In Chapter 3: we describe the basic CBR methodology with a particular focus on its potential in terms of providing explanations. We then present a review of explanations in CBR. We highlight the advantages of the CBR approach in contrast with others but also highlights issues of this approach that need to be addressed.

Chapter 4 describes the Explanation Framework that we have developed. It points out the issues that the Framework attempts to address and describes the approach taken in addressing these issues. It provides step by step examples of how each of the component parts of the Framework are carried out and examples of the explanations produced by the Framework.

Chapter 5 describes *Fionn*, the machine learning workbench that we have developed and in which the Explanation Framework was implemented. We outline the basic structure of the *Fionn* workbench and describe the case representation technology underpinning it, CBML. We then outline briefly how two classifiers, *k-NN* and Logistic Regression, were implemented as part of the *Fionn* workbench. Once we've described these basic elements we outline how they were used to implement the Framework.

Chapter 6 contains the evaluations that we have carried out on the various aspects of our Explanation Framework. We present the findings of a user evaluation we carried out on the Framework to determine its effectiveness in improving user confidence. We also evaluate the localised logistic regression as a classification method in comparision with *K-NN* and globalised approaches. Finally we investigate the potential effectiveness of the estimates of confidence produced by our Framework.

Finally Chapter 7 concludes the thesis and describes some further work that could be investigated.

Chapter 2

Machine Learning and Explanations

In this Chapter we briefly review how explanations are produced by other machine learning technologies. We highlight the shortcomings of some of the approaches taken and the lessons that have been learnt in those past attempts. The earliest attempts at providing users with explanations stemmed from work that was done on Expert Systems in the 1970s. This work first highlighted the need for explanations and the weaknesses of artificial intelligence techniques in providing them. Importantly, it also generated discussion on what a good explanation would be and how it could be provided. Although the Rule-based approach used in Expert Systems was found to be of limited use, it is this form of explanation that has persisted in the machine learning community. Decision Trees are often used to produce explanations as they are deemed to be interpretable models which can be easily transformed into rule form. Other machine learning systems lack any transparency in the way they operate and are so complex that they are deemed to be Black Boxes. The provision of an explanatory component for such systems has proven to be a major problem. Instead of provding explanations based directly on the system, special explanation components are developed. Many of these explanatory systems are again rule-based approaches. We will begin this chapter with a review of the work on Expert Systems before moving on to Decision Tree and Black Box approaches.

2.1 Expert Systems

Expert Systems seek to encapsulate the knowledge and expertise of a particular domain in the form of rules. Using this knowledge the Expert System can then assist users in making decisions in an interactive way. One of the earliest examples of such a system is MYCIN (Shortliffe 1976) which was developed in the 1970's and inspired many other Expert Systems (Jackson 1986). MYCIN is an interactive program that diagnoses certain infectious diseases, prescribes anti-microbial therapy, and can explain its reasoning in detail. The ability to explain its reasoning was added to offer a greater degree of transparency into the reasoning process of the system and so instil greater trust in it's recommendations. Users could ask the system why it was asking for certain information.

However, the explanations offered by the system were soon found to be inadequate and confusing. The explanations offered by MYCIN were formulated in terms of the rule-based structure used internally by the system for reasoning and weren't readily interpretable by users. Later Expert Systems such as NEOMYCIN and XPLAIN sought to improve on this short-coming (Clancy and Bock 1982, Swartout 1983). The XPLAIN system supplemented its explanations with background information and references to literature thus giving greater credence to its actions.

Although efforts were made to improve the explanations produced by early Expert Systems these efforts largely failed as the rule-based explanations they produced were incomprehensible to users. It was realised that the requirements of users in requesting explanations were not being properly addressed (Majchrzak and Gasser 1991). This led many people to investigate what the requirements of users seeking explanations were and what criteria explanations produced must meet in order to fulfil those requirements.

Swartout and Moore (1993) made the first attempt to address the issue and proposed five requirements for explanations; *fidelity*, *low construction overhead*, *efficiency* and that they are *understandable* and *sufficient*. The requirements of low construction overhead and efficiency refer to how easily explanations can be generated at runtime and reflect concerns about the level of computational power available which although relevant at that time can be largely ignored now. The requirements of *fidelity* and being *sufficient* refer to the belief that the explanations produced should reflect exactly the knowledge stored in the system and that there should be sufficient knowledge within the system to answer any question users might have. Most of the requirements refer to the mechanics of producing an explanation with the exceptions that explanations be *understandable* and have *fidelity* to the Expert System. These two requirements would seem to be in conflict with each other as the information stored in an Expert System is often incomprehensible to users.

Wick and Thompson (1992) take a much more user centric approach and they argue strongly against fidelity being a requirement of an explanation. They view the explanation provision task as being separate to that of the Expert System's diagnostic task. They propose three goals which explanations produced by a system might serve which reflect the requirements of three very different types of user;

- **Verification:** to help the end user verify the knowledge used by the system. This form of explanation is intended for knowledge engineers who are interested in and comfortable with the inner workings of the Expert System.
- **Duplication:** reproduce the knowledge in the system in a form that is acceptable to domain Experts.
- **Ratification:** the explanation should increase the end user's confidence in the system's recommendation.

Importantly Wick and Thompson realised that the explanation process can and should in some cases be decoupled from the reasoning process. The single greatest requirement of any explanation should be the needs and goals of the user and that this should be the primary criterion for any explanation. Presenting explanations based purely on the reasoning process in some cases will fail to meet these needs.

Construction of the knowledge base of rules on which Expert Systems rely is a time-consuming task that requires continued access to domain experts and consultation with Knowledge Engineers. Such access may not always be possible. In the next section we discuss Decision Trees which can be used to automatically extract rules from raw data.

2.2 Decision Trees

Decision Trees are an extremely popular machine learning technique that model classification tasks in a natural and intuitive form. The technique models the decision process as a series of linked test conditions. Each test condition forms a node in the tree structure and depending on the outcome of that condition the decision process continues along a branch from that node to the next. This process is continued until a terminating node is reached and the decision process is completed. For example, Figure 2.1 depicts a Decision Tree that models the decision as to whether to play tennis or not (Mitchell 1997).

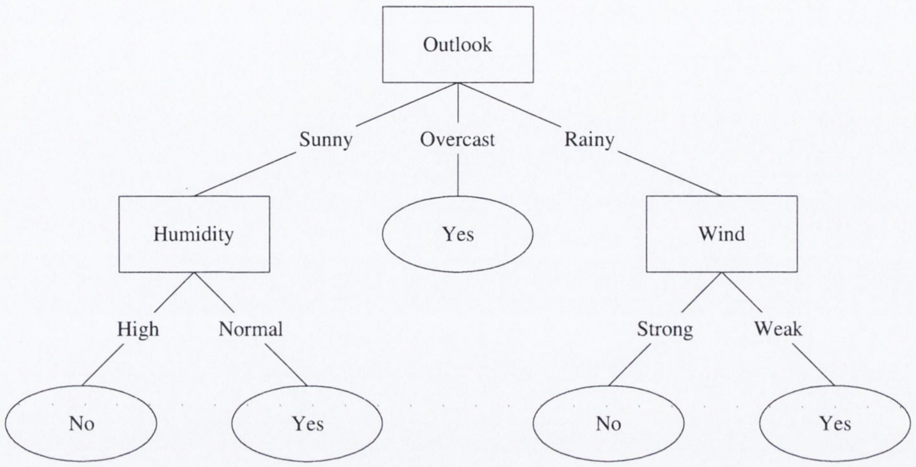


Figure 2.1: An example of a Decision Tree for deciding whether to play tennis or not

The process begins with the root node where the `outlook` attribute is tested. There are three possible outcomes to this test and these are represented by the branches. If the `outlook` attribue was `sunny` then the decision process would move on to a second node and the `humidity` node would be tested. Finally based on the result of that test a decsion would be made as to whether to play tennis or not.

Decision Trees must first be contructed from raw data using some form of induction algorithm. In designing a Tree induction algorithm a number of choices must be made. For example, the type of conditions contained in the nodes and selection criteria to be used in chosing which attributes to test. The choices made for such decisions distinguish the one Decision Tree algorithm from the next. The fundamental choices in the design of Decision Tree induction algorithm can be summarised as;

- **The Splitting Function:** This is the scheme used to split the sample space at each successive node. Normally a single attribute value is tested as in Figure 2.1 but other more complex multivariable schemes have been used (see Section 3.2.3). Schemes using a single variable are called univariate while the others are known as multivariate.

- **Splitting Criterion:** Once a splitting function has been defined the manner in which the various possible function parameters are selected needs to be considered. Most Tree algorithms operate by constructing the Tree in a greedy non-back tracking manner. This makes the choice of splitting parameters critical since choices cannot later be revised. The splitting parameters are usually selected based on some statistical measure of how likely the resulting split is to lead to a successful terminal split.

- **Leaf Node Function:** How data that reaches a terminal node is represented is another point where Tree induction algorithms can deviate. For classification Trees the leaf node is usually assigned the majority class of the samples that reach that node but *k-NN* algorithms have also been used. For regression tasks the leaf node is usually assigned the average value of all samples that reach it. However, some algorithms may use simple representational models of the data that reaches a leaf node. In such cases simple models such as linear regression equations are used to approximate regions of the sample space.

- **Stopping Criteria:** Trees can be potentially grown until there are nearly as many leaf nodes as there are data samples. To do so would drastically affect the Trees interpretability and generalisation properties. So some measures are needed to control the growth of the Tree. These measures can either be global, looking at the overall structure of the Tree, or local, applied at the level of node partitioning. Global schemes usually predetermine the overall structure of the Tree; they define how many internal nodes a Tree might have. Local schemes might limit the further expansion of a node if the extra accuracy gained on the training set is below a certain threshold. In practice, such stopping measures may miss unpromising Tree expansions that later lead to useful informative expansions. To counteract such an event, Trees are usually grown liberally and later pruned back to a desired level of complexity.

- **Pruning Scheme:** Once a Tree has been grown it may be necessary to collapse back the nodes so as to improve the generalisation and interpretability of the model. The manner in which this is done, the criterion for determining the nodes for collapsing, is a further point where induction algorithms may differ.

Some popular algorithms include for classification tasks include ID3 (Quinlan 1986), its successor C4.5 (Quinlan 1993) and ASSISTANT (Kononenko et al. 1984, Cestnik et al.

1987). Descision Tree algorithms have also been developed for regression tasks such as CART (Breiman et al. (1984)) and Quinlan's M5 (Quinlan (1992)).

One of the major reasons for the popularity of Decision Trees is their advantages in terms of interpretability. As each path from the root to a leaf corresponds to one conjunction of tests the paths can be written down as a set of *IF-THEN* rules. One method of producing rules is C4.5 (Quinlan 1993). These rules can then be used as the basis for an explanation. For example consider a hot humid sunny day with a high temperature and weak winds. From Figure 2.1 we can see that this type of day is not suitable for playing tennis. The leaf node for this decision is the bottom left node of the Decision Tree. Therefore an example rule that can be used as an explanation for this path is:

> *Rule: IF(Outlook=Sunny) AND (Humidity=High) THEN NO*
>
> *Explanation: As the outlook is sunny and humidity is high it is not suitable to play tennis today.*

Explanations of this form are not always as simple for a user to understand as that shown in Figure 2.1. As the depth of a Decision Tree increases so to does the complexity of the generated rules. Complex problems can quickly lead to Decision Trees that are so large and complex that they interpretable qualities are completely lost. As we shall see in Section 2.3.1 this can often be the case.

2.3 Explanations for Black Box Systems

In machine learning research the quest for increasingly more accurate and stable classifiers has led to ever more complicated algorithms. Ensemble approaches and algorithms such as Support Vector Machines and Neural Networks have reached a level of complexity where they are not readily interpretable. Such approaches are commonly referred to as black-box algorithms owing to their lack of transparency with regard to the reasoning behind the predictions they make.

Although increases in accuracy are welcomed, research has highlighted the need for interpretability and transparency as a critical aspect in the implementation of machine learning techniques in real world applications (Andrews et al. 1995). People are understandably reluctant to accept without question predictions from black-box systems.

This has led to the development of explanation systems that strive to offer an insight into the workings of the black-box systems. Many different approaches have been taken but commonly the explanation systems try to build machine learning systems that are inherently interpretable such as Tree-based or Rule-based systems that describe the underlying Black Box e.g. (Andrews et al. 1995, Tickle et al. 1998, Zhou and Jiang 2003). The relevant rules or a Tree structure is then used as evidence in support of the black box's prediction. Such systems use the Black Box as an oracle capable of supplying an unlimited amount of training data. The hope is that, with an abundance of training data, the explanation system should offer a good description of the underlying black-box system.

However, in reality such systems are limited in the level of fidelity that they can achieve while maintaining some level of interpretability. The differing bias of the black-box algorithm and that of the one being used for explanations means that it can be difficult to fully capture the operation of the black-box system. Domingos (1998) focused on how well an explanation facility captured the improvements gained through the use of ensemble techniques. He found that it retained just 60% of the gains. More accurate descriptions of the operation of the black box often come at the cost of increasingly more complex Tree and rule-based systems. This trade off in interpretability versus fidelity means that such approaches are of limited use as a convincing explanation system when the underlying problem is complex and the credibility of the system can be damaged by bad, inaccurate or convoluted explanations (Majchrzak and Gasser 1991). An example of such an approach is the Decision Tree based approach called TREPAN.

2.3.1 TREPAN

TREPAN is a Decision Tree based approach to providing explanations for Black Box Systems (Craven and Shavlik 1996). TREPAN seeks to extract the information trapped in the Black Box and deliver it in a more intrepretable form, as a Decision Tree. TREPAN is an algorithm that maintains a pedagogical approach, utilising the Black Box as an oracle from which extra task specific information can be gleaned during the learning process. TREPAN differs significantly from other popular Decision Tree algorithms such as CART and C4.5 in a number of ways;

- **Membership Queries and the Oracle:** TREPAN queries the network as to the class of various instances. Theses queries are used in two ways. Initially they are used to establish the classes of the training instances. This is carried out to ensure that the data used to train the Decision Tree reflects the function that has been learnt by the Black Box.The data is relabelled depending on how the Black Box would label it.

Secondly, TREPAN ensures that at any point it has a predefined number of instances at a node before assigning it a class or defining a splitting criterion. If a situation is reached where there are not enough instances the network is pooled with relevant queries until the deficit is depleted. The relevant queries are based on the set of constraints needed to reach the given point in the tree.

- **Tree Expansion:** TREPAN uses a best-first criterion to expand the Tree, unlike most Decision Tree algorithms which are based on a depth-first greedy search. The notion of the best node, in this case, is the node at which there is the greatest potential to increase the fidelity of the extracted Tree to the network.

- **Splitting Tests:** Each non-leaf node in a Tree contains a test that is used to split the instance space at that node. C4.5 and CART use simple single feature tests whereas TREPAN uses *M-of-N* type expressions as splitting criteria. An *M-of-N* rule is a Boolean expression that is specified by an integer threshold, M, and a set of n Boolean literals. The rule is true if any *M* of the *N* Boolean literals are true. By using *M-of-N* tests its hoped that a more concise Tree will be produced.

TREPAN has been applied successfully to a number classification problems including ensembles of Neural Networks. Although great care is taken to try and ensure that the Decision Trees produced are small and concise the Trees produced are not easliy interpretable.

An example of a Decision Tree produced by the Trepan algorithm for a predicting whether stock prices are going to go up or down can be seen in Figure 2.2. In this example there is a large root node which contains a large set of *M of N* conditions. Although these nodes are especially designed to produce a concise interpretable Tree it is difficult to see if any real insight into the domain can be garnered by such Trees.

The second example in Figure 2.3 is of the Tree produced to predict which way a voter is likely vote, either democrat or republican. Here, although the functions contained within each node are relatively simpler than those in Figure 2.2, the overall Tree structure

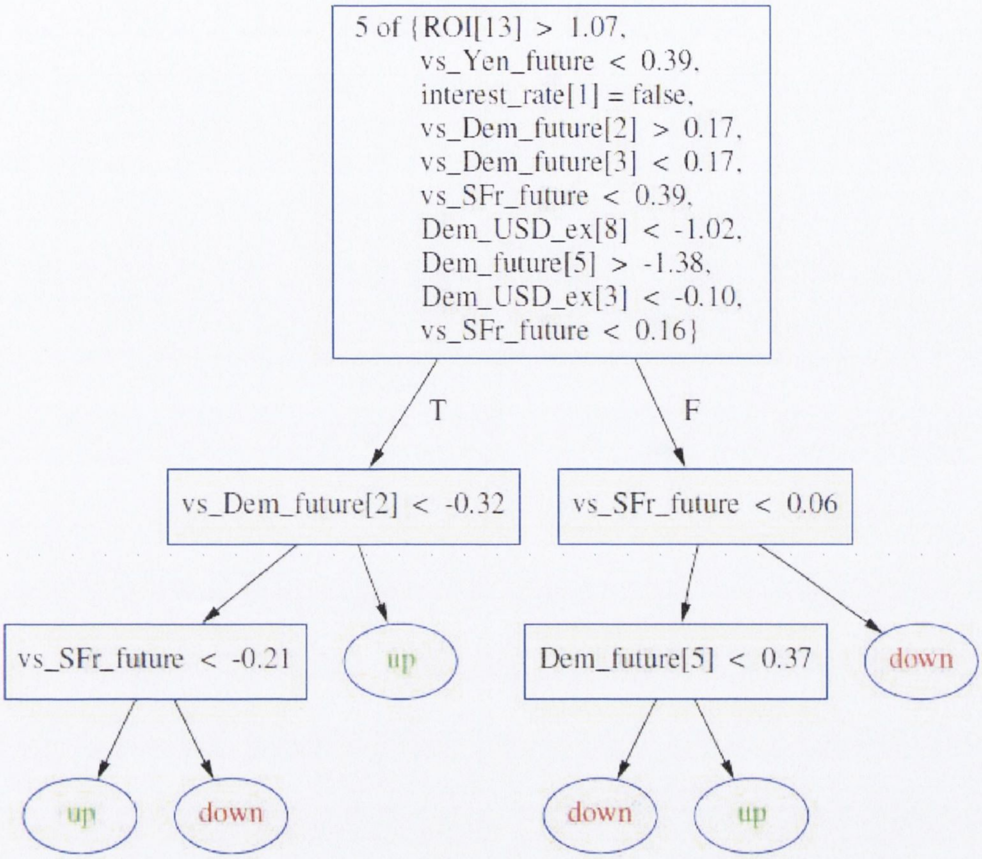


Figure 2.2: A Decision Tree produced by Trepan

is much larger. Again it is difficult to see whether such Trees or portions of them could serve as convincing explanations to users who aren't themselves experts in both the domain in question and also in machine learning itself.

2.4 Summary

It is clear that there has been a demand for interpretable explanations in Artificial Intelligence technologies from a very early stage in the development of the field. However, as the field has grown and more sophisticated techniques have been developed, the need for explanations has been largely ignored. Decision Trees do have explanatory potential which quickly breaks down as problems become more complex and the Tree structure grows.

Importantly, as Wick and Thompson (1992) point out the user's demands and goals must be taken into account in designing explanations. Rule-based approaches represent at-

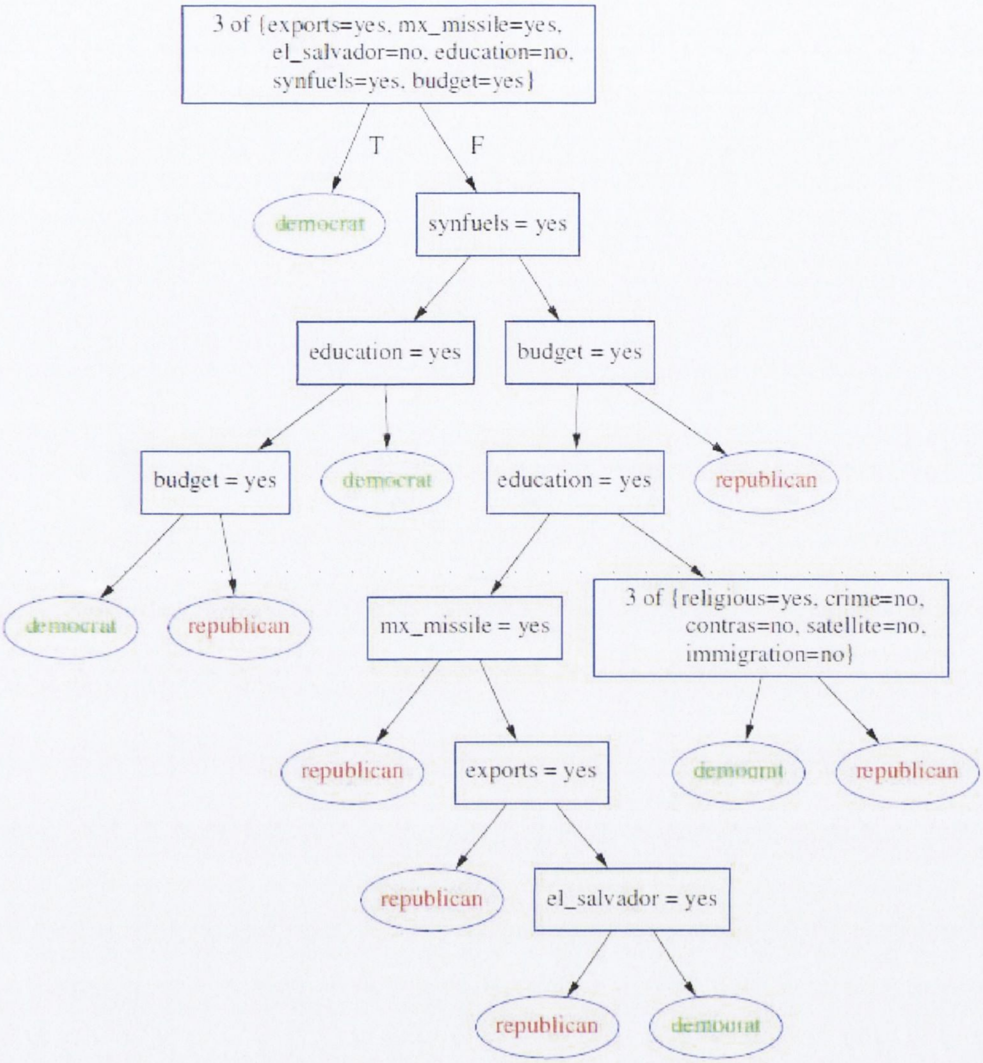


Figure 2.3: A Decision Tree produced by Trepan

tempts by machine-learning designers to build interpretable models but these approaches have not been designed with the intention of explaining individual decisions or recommendations. This task is different and the use of Decision Trees in this context represents a poor solution. The explanations produced by such systems have been found to be too complex for users to understand. Case-based reasoning has an inherent advantage in terms of providing humanly interpretable explanations and in the next Chapter we will discuss the CBR methodology as well existing research in CBR in providing explanations.

Chapter 3

Case-based Reasoning and Explanations

Case-Based reasoning (CBR) is a machine learning technique that differs fundamentally in its approach to problem solving in comparison with other techniques. Most other techniques such as Decision Trees and Neural Networks are model-based and try to generate general models of the problem domain which can then be used to solve future problems. In the case of Neural Networks the model is encoded in the structure of its neurons and their attached weighted connections. In Tree-based and Rule-based systems the knowledge is described by an explicit set of rules. CBR in contrast contains no explicit model of the problem domain, instead a set of past examples called cases are retained. Each case is made up of a description of a past example or experience and its attached solution. The full set of past experiences encapsulated in individual cases is called the case-base. As an example of a simple case please consider the one shown in Table 3.1 which is taken from the Blood Alcohol Content (BAC) case-base (Cunningham, Doyle and Loughrey 2003). The task in this domain is to predict whether an individual is over the legal drink driving limit or not. The features such as `Weight` and `Meal` are used to describe the past examples and characterise the present problem. Each case in the case-base has its solution attached which in this case is just the label `Under`, reflecting that the individual was found to be under the drink-driving limit.

In CBR problems are solved "by using or adapting solutions to old problems" (Riesbeck and Schank 1989). When a new problem is presented, the case-base is searched and similar past examples are found and used to solve that problem. The need to detect and model

Table 3.1: Example from the Breathalyser Domain

Features	Sample Case
Weight	76
Duration	60
Gender	Male
Meal	Full
Units	2.9
BAC	Under

general patterns in the problem space is avoided. For example in the BAC domain, to determine whether a particular individual is under or over the drink-driving limit their characteristics such as `Weight` and what kind of meal they've had are recorded and the case-base is searched for similar past cases. Using these cases a classification for the present problem is determined.

The very different approach taken to problem solving in CBR has a number of advantages. CBR has proven to be successful in solving *weak-theory* problems, problems for which little insight into the problem exists and the problem domain may be complex. Instead of having to formulate a set of rules which can be applied to solving the problem a set of past examples is used. In CBR problems are solved using the cases that are most relevant to that problem. This localised approached taken in CBR means that it is suited to problems that are complex and non-linear in nature. Cases can often be found readily available in form of records in a database. It is for this reason that CBR can result in lower knowledge engineering cost in comparison to other techniques (Cunningham 1998). Where rules can be generated reasonably easily, for example by Decision Trees, CBR still maintains a number of advantages. One particular advantage is in terms of maintenance. CBR is a lazy learning technique which means that it defers all problem solving effort until runtime. Given the lazy learning approach taken and given that one of the main sources of knowledge in CBR system is the case-base, the system can easily be updated when new information becomes available by simply adding new cases to the case-base. Other techniques would require that the entire model is recompiled. The CBR methodology also supports incremental learning. Mistakes the system makes can be quickly corrected by adding a case representing the correct solution to a problem. However, in reality in order to ensure the continued accuracy of a case-base system care must be taken as to how it is maintained. This includes examining the contribution of each case within the case-base to the system's performance and altering the case-base accordingly as well as looking at the other areas of knowledge within the CBR system. Schemes for performing such tasks have become an important area of research with the CBR community (Smyth and McKenna 2001, Delany et al. 2005, Delany and Cunningham 2004b).

Another advantage of CBR that is often cited is in terms of user acceptance (Leake 1996). The lack of ability of many machine learning techniques to provide users with explanations has proven to be a major failing when used in real world environments. It has previously been taken for granted that CBR does not suffer this same shortcoming. However, although CBR does inherently have advantages in terms of providing user feedback the issue is less straightforward than previously thought and has recently become the focus of much interest in the research community. Addressing the issues surrounding CBR explanations in classification forms the main focus of this thesis and we will review the issues in this area along with the relevent research in Section 3.4. Before we look at CBR explanations we will first outline in more detail the important aspects of the CBR methodology with a particular emphasis on CBR as a classification technique and also look at the historical origins of CBR in Sections 3.1 and 3.2 .

3.1 Origins Of CBR

Ideas on the role of analogy and the use of past experiences in reasoning can be traced back to the work of Thagard, Gentner, Schank and Wittenstein among others (Holyoak and Thagard 1989, Gentner 1983, Schank and Abelson 1975, Wittgenstein 1943). For instance Wittenstein observed that natural concepts such as tables and chairs are in fact polymorphic and cannot be classified by a single set of necessary and sufficient features but instead can be defined by a set of instances cases with family resemblances (Wittgenstein 1943). This work has been cited by Aamodt and Plaza as the philosophical basis for CBR and suggests the need for a case-based approach in *weak-theory* domains (Aamodt and Plaza 1994).

The work of Thagard, Gentner and Schank all comes from the cognitive science field. Thagard and Gentner focused on a cognitive model of thinking based on analogy and Thagard also went on to investigate the important role of analogy in explanations (Thagard 1989). Gentner's research has mainly focused on the role of analogy in learning (Gentner et al. 2001). Although the research into the role of analogy in reasoning is very much related to CBR the work of Roger Schank and his research group very much defined CBR as it is known today (Schank 1999).

3.2 The CBR Methodology

The CBR methodology has a logical step-like structure that can be broken down into four distinct phases. These phases are often referred to as the 'Four REs' cycle as described by (Aamodt and Plaza 1994). Figure 3.1 shows a diagram of this system. This cycle contains four phases which we will first list and describe very briefly before discussing each phase in greater detail:

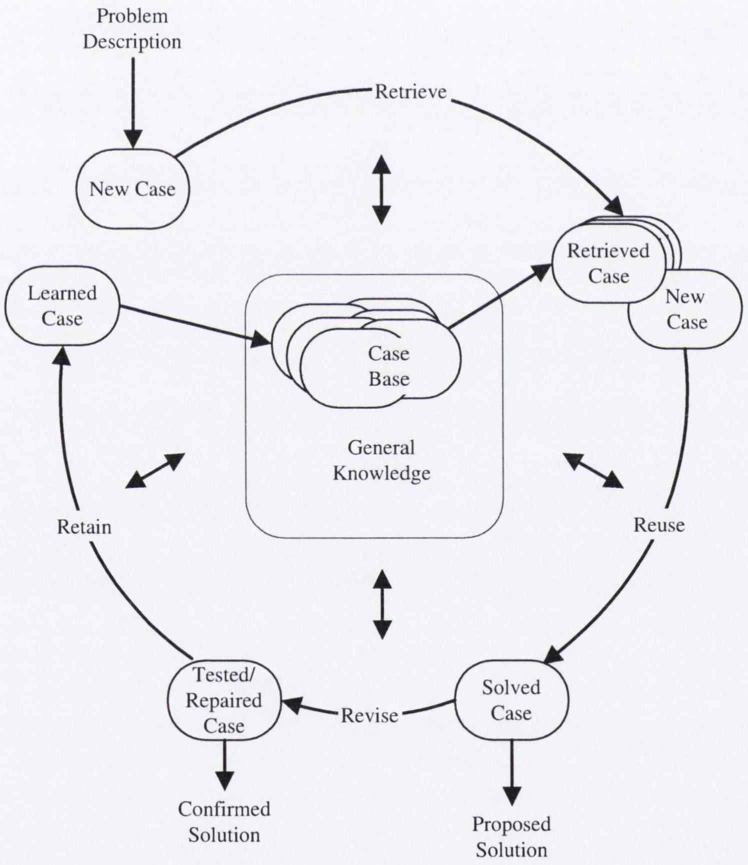


Figure 3.1: The CBR Cycle (Aamodt and Plaza 1994)

Retrieval To begin with, the case-base is searched and the most similar cases to present problem description are sought out and retrieved.

Reuse The retrieved cases are then used to generate a solution to the presented problem description.

Revision The proposed solution is evaluated and validated.

Retention Once a satisfactory solution has been reached the problem description and solution are then added to the case-base.

3.2.1 Retrieval

The retrieval stage is generally held to be the most important phase of the CBR methodology as so much is dependent on finding a good set of cases given the problem case. There are a number of ways in which this can be done. Some apporaches taken include indexed-based retrieval and use database methods but the method most commonly used in classification tasks is the k Nearest Neighbours (k-NN) retrieval algorithm. In the k-NN algorithm a case representing the present problem is compared with each case in the case-base and a similarity score is calculated. The similarity score calculated for two cases is based on the amalgamation of the similarities scores for each of the features that describe a case. For clarity we will refer to similarities between features as local similarities. Local similarity scores can be calculated using simple functions such as the Euclidean distance in the case of real-values features or exact match criteria for nominal features. However, more sophisticated measures are commonly used. For example in the case of unordered symbolic features a similarity table can be defined or a value difference metric can be defined (Salzberg 1991). There has also been much research on learning similarity measures (Wettschereck et al. 1997, Stahl and Gabel 2003, Stahl 2002). In this case the similarity function is learnt in much the same way that many other machine learning tasks are.

Once a satisfactory set of local similarity measures have been decided they can be used in the amalgamation function to define the overall similarity between two cases. Similarity between two cases Q and C is usually defined as the sum of the local similarity values multiplied by their relative importance:

$$Sim\left(Q,C\right) = \sum_{f\in F} w_f \times \sigma_f \tag{3.1}$$

where σ_f is the local similarity function for feature f and w_f reflects its *weight* (or importance). F is the set of all features in Q. Again the determination of what weight to attach to each feature is a very important issue. The k-NN algorithm is very sensitive to the presence of irrelevant or noisy features. Finding a good set of feature weights can greatly improves the performance of the CBR system and it is a topic that has received a lot of attention (Stahl 2001, Aha 1997).

3.2.2 Reuse

After the retrieval phase the CBR system is left with one case or more from which to derive the solution. In the case of classification tasks this is usually a relatively trivial task. Either the most similar case's solution is used or the majority solution found within a certain number of the most similar cases is used. When CBR is used in knowledge intensive domains the returned solutions are often more complex then a simple class label. A solution may constitute a set of actions to be taken or a set of ingredients to manufacture viable tablets (Bergmann 1993, Craw et al. 1998). In these cases the solutions often have to be adapted to take into account differences in the problem specification and this can be a complex task. It often requires considerable domain knowledge in order to perform the adaptation process and information may not be easily encoded in *weak-theory* domains. This knowledge gap has led Watson (1997) to refer to the adaptation process as the Achilles' heel of CBR.

3.2.3 Revision

CBR systems inevitably make mistakes and before any suggested solution is added to the case-base it is first validated. In this way the CBR system can fine tune and improve its performance in a real time learning scenario.

3.2.4 Retention

The retention phase involves updating the CBR system based on any feedback received. This normally means the addition of a new case when the system has made an error but can also mean adjustments to other elements of the system such as altering the similarity measures. In some systems operating in complex domains the case representation that is added to the system can also include additional information on the outcome of the solution, which may also include fine-grained information on how well the solution addressed systems goals Goel et al. (1991). Similarly other systems record more than just the result and record the problem solving process itself Veloso and Carbonell (1994).

3.3 The Knowledge Contained in CBR Systems

Unlike rule-based systems but like techniques such as neural networks, all the knowledge contained in a CBR system isn't stored in one easily identifiable place. Although cases form the most obvious source of knowledge much other knowledge can be distributed across other elements of the system. In providing explanations we need to draw on knowledge stored within the system and it is important to realise where this knowledge can lie. Richter has identified four different ways in which knowledge can be represented in a CBR system (Richter 1995, 1998). He has named these *knowledge containers* and they have met wide acceptance as a natural organisation of knowledge in CBR. Richter's knowledge containers are:

Vocabulary Knowledge The attributes and predicates that can be used to describe the cases.

Case Knowledge consists of the problem episodes or instances represented as cases that can be used to solve similar problems in the future.

Similarity Knowledge represents the similarity measures which are used to match cases in a particular domain.

Adaptation Knowledge is knowledge used to adapt the solution of the matching case for the target problem.

The knowledge for the vocabulary, similarity measure and solution transform is structured and used at compile time and the knowledge in the case-base plays its role only at run time. Richter suggests that this is the major advantage of CBR because the knowledge acquisition of cases is easy. Although the acquisition of knowledge in the other containers is more difficult to obtain, shifting knowledge from the case-base to another container can lead to significant improvements in the system. As highlighted in Section 3.2.1 much of a CBR system's performance is dependent on its similarity measures and much effort can go into ensuring that these measures are effective. This means that much of the knowledge in a CBR system can end up being stored as similarity knowledge. It is clear that knowledge encapsulated in a CBR system may not be explicitly expressed to the user in a classification task. The user may see the benefits of case and similarity knowledge in the classification presented to them, however this knowledge is still hidden away from the end user. This is a particular problem in case-base explanations as we shall see in the next section.

3.4 Case-based Explanations

In Chapter 2 we discussed how explanations are produced by other machine learning techniques. It was clear that many machine learning techniques lacked any inherent ability to produce explanations while those that do are limited in the scope in which their explanations are applicable. Conversely CBR systems have an inherent transparency that has particular advantages for explanations as Leake (1996) points out:

> *"...neural network systems cannot provide explanations of their decisions and rule-based systems must explain their decisions by reference to their rules, which the user may not fully understand or accept. On the other hand, the results of CBR systems are based on actual prior cases that can be presented to the user to provide compelling support for the system's conclusions."*

At this stage it is useful to make the distinction between knowledge-intensive and knowledge-light explanations. Knowledge-intensive approaches involve designing the case structure with explanations in mind and inserting explanation patterns (Schank 1986). The most notable and influential work was that by Leake (1992) on SWALE. The SWALE project investigated the potential of CBR to produce creative explanations of anomalous events. It produces explanations based on the retrieval and application of especially constructed cases storing explanations to prior events. These explanation patterns are then retrieved and used to explain new events. The name SWALE came from a young successful race horse that died suddenly and unexpectedly in 1984. After the horse's death many people hypothesised as to the cause of death. Providing many different interesting explanations of this event served as the test scenario for the SWALE system. For instance given the *Janis Joplin* explanation pattern stored within the system, SWALE hypothesises that a possible cause of death was a drugs over-dose. Another more recent example of a knowledge intensive approach to explanations is the DIRAS system (Armengol et al. 2001).

In contrast to the Knowldge-Itensive approach the Knowledge-light explanations rely on revealing the cases used in the reasoning process as they are. The focus of this thesis is on explanations produced in an automated fashion which is in keeping with the knowledge-light approach. The focus of the rest of this Section will be on knowledge-light explanations.

3.4.1 Goals, Advantages and Limitations in CBR Explanations

The concept of an explanation is intuitively understood by most people, we use them effortlessly throughout our everyday lives. However the vast array of contexts and ways in which explanations are used means that the concept of an explanation is an extremely difficult one to concretely define. What might be considered a valid and good explanation in one context might have no meaning in another. The slippery nature of the explanation concept of course impacts on the way in which explanations can be evaluated. There is no one fixed criteria on which all explanations can be judged. Explanations can only really be evaluated in terms of how effectively they meet the objectives that necessitated the creation of the explanation.

However, within the machine learning community explanations are usually used within a restricted range of contexts and with clear objectives. Sormo and Cassens (2004) define a set of goals that explanations might serve in CBR systems that was inspired by the categorisation proposed by Swartout and Smoliar for explanations in early rule-based systems (Section 2.1). They defined five categories based on the goals that the explanation might serve:

- **Transparency:** The goal of the explanation is to impart an understanding of how the system found an answer.
- **Justification:** This is the goal of increasing the confidence in the advice or solution offered by the system by giving some kind of support for the conclusion suggested by the system.
- **Relevance:** An explanation of this type would have to justify the strategy pursued by the system.
- **Conceptualization:** The explanation serves to make clear to the user the knowledge and vocabulary used by the system.
- **Learning:** The primary role of an explanation is to impart knowledge to the user.

The goal that most explanation systems in knowledge-light classification tasks serve is primarily justification. The typical form of explanation used in the knowledge-light domain is presenting the user with the most similar case. For example, consider Table 3.2. Here we have a*Target Case* representing the present problem for which the system has predicted the subject is under the drink-drive limit. Beside the *Target Case* we can see the *Explanation Case* presented to the user as support for the prediction given. Implicit in this form of explanation is the argument that given the precedent of the *Explanation Case* and its similarity to the present situation the proposed solution is justified.

Table 3.2: Sample knowledge-light explanation from the BAC Domain

Features	Target Case	Explanation Case
Weight	76	76
Duration	60	60
Gender	Male	Male
Meal	Full	Full
Units	2.9	2.6
BAC	Predicted Under	Under

This type of explanations and the insight it offers to the end user differ considerably from those found in rule-based and other approaches in a number of ways:

- **Natural Form of Explanation:** Research in cognitive science and other areas suggests that explanation by analogy is a natural form of explanation in some domains and one people can quickly relate to. Cunningham, Doyle and Loughrey (2003) conducted a trial in which they compared users' satisfaction with simple case-based explanations as seen in Table 3.2 and alternatively with rule-based explanations. They found that in general users had a preference for case-based explanations. Gentner et al. (2003) investigated the role of analogy in learning and although referring to more elaborate text based cases argue cases and examples are concrete, they are more engaging and more easily understood than abstract, domain-general principles.

- **Use of Real Evidence:** In CBR the user is presented with actual cases that represent past experiences. In most applications these cases are undoubtedly true and so their validity isn't in question, this is the great strength of case-based explanations. Users who are unfamiliar or suspicious of a system are more likely to be convinced by explanations that contain factual evidence than by unsupported rules.

- **Fixed and Simple Form of Explanation:** CBR explanations avoid the interpretability versus fidelity trade off that can plague some other techniques. In other forms of explanation such as rule-based, the explanations can grow more and more complex in line with the complexity of the problem. This can lead to explanations that aren't intrepretable. In case-based explanations the explanation presented to the user, a case, is independent of the complexity of the problem.

Although simply presenting the most similar case can be perfectly adequate as an explanation, there has been a growing realisation among the CBR community that the case alone might not always fully suffice as a satisfactory explanation. The major challenge with case-based explanations lies in ensuring the perceived appropriateness of the presented cases to the validity of the prediction. The task of ensuring that the cases are deemed appropriate and convincing can be broken down into two challenges:

- **Selecting the Best Case to Present to the User:** The major driving force for the provision of explanations is to offer a justification for a prediction. The goal of providing a convincing argument may not always be best served by supplying the user with the nearest neighbours. Convincing explanations are domain and user dependent (Sormo and Cassens 2004), and this should be reflected in the case retrieval process. As discussed in Section 3.3 the utility for which the cases are to be used should be reflected in the similarity measures used. Taking the domain and user details into account, the retrieval process should be adjusted to select the cases that form the most convincing argument. We will describe work that addresses the retrieval process in this way in Section 3.4.2.

- **Explaining the details and relevance of retrieved cases:** This is an issue that has recently received a lot of attention in the CBR community (Nugent and Cunningham 2005, McSherry 2004, Sormo and Cassens 2004). In CBR explanations, the ability of the user to make meaningful comparisons between feature values in the query and the retrieved explanation cases is of critical importance to the success of the explanation. CBR systems are not wholly transparent and much domain knowledge can be contained within the similarity metrics used in the system as highlighted in Section 3.2.1. It is implicitly assumed in simple CBR explanations system that the user has this same domain knowledge and so the appropriateness of the explanation case is clear. However, this may not be the case and the relevance of the retrieved case may be lost on novice users. By providing users with extra explanatory feedback further insights into the problem are given in addition to further reassuring the user.

A further challenge to using explanations to foster user confidence is to do so in the long run. The primary motivation in providing users of CBR systems with interpretable explanations is to increase their confidence in the system. However, as is pointed out by Cheetham and Price (2004), people can quickly lose confidence in a system if it makes predictions which then turn out to be incorrect. This may well be further compounded if the suggested incorrect solution was supported by an explanation (Watson 1997). To address this issue Cheetham and Price propose using confidence measures so as to alert the user to when a system may be making a mistake.

Already these issues have begun to be addressed by the CBR research community and in the next section we will highlight and discuss some systems that address these issues.

3.4.2 CBR Explanation Systems

ProCon

The issue of transparency is one that McSherry has addressed in his ProCon System (McSherry 2003). McSherry has focused on making the relationship between the feature values within a case and its predicted value explicit. He argues that simply presenting the feature values in the most similar cases may be misleading. The relationship between feature values and the predicted value may not always be a positive one; the presence of some feature values may in fact be evidence against the prediction. Simply supplying the user with a case may lead them to incorrectly infer the relationship between feature-values and the prediction. To combat this McSherry provides the user with extra relational information about the case feature-values and the predicted class-value. To infer the feature-values to class-value relationships, a Naïve Bayes model is built on the entire training set and from this the relational information is derived. Using the Naïve Bayes model it is possible to infer the effect of different feature-values and so inform the user whether a particular feature-value is a *supporter* or *opposer* of a given prediction. In table 3.3 we can see an example of the ProCon-2 system's output as seen in (McSherry 2003).

Although this approach does succeed in providing the end user with information on the relationships between feature-values and the predicted outcome, the manner in which it

Table 3.3: Sample ProCon-2 Output from (McSherry 2003)

ProCon-2:	Please describe the target problem.
User:	weight=79, duration=90, sex=male, meal=full, units=10.1
ProCon-2:	The 3 most similar cases are: Case 13: 79, 240, male, full, 9.6, over-limit(0.97) Case 26: 73, 120, male, full, 9, not-over-limit(0.96) Case 56: 79, 120, male, full, 7.2, not-over-limit(0.96) The predicted outcome is: over-limit.
User:	**why**

ProCon-2:	Features of the target problem that SUPPORT the prediction:
	
	units=10.1
	
	Features of the target problem that OPPOSE the prediction:
	
	weight=79
	duration=90
	sex=male
	meal=full
	
	The most similar case is:
	Case 13:79, 240, male, full, 9.6, over-limit
	
	The outcome in this case was over-limit in spite of the oposing evidence that:
	
	weight=79
	sex=male
	meal=full

is done may limit it to very simple problems. The Naïve Bayes approach induces a global model of interactions and this may not always be appropriate as the influence of features may vary over the problem space. Indeed, it might be argued that global models are not in the spirit of CBR which is a local approach to learning. As highlighted earlier, much of the strength of CBR lies in its abilities to deal with weak-theory and complex domains and support incremental learning. Naïve Bayes models are also extremely simple and view each feature in isolation. The conditional probability of a given class label is calculated for each feature given its value. No interaction between features is taken into account. Bayes rule is then used to calculate the joint probability and make a prediction. It is very likely that in more complex domains the explanations produced may seem counter intuitive and the derived relational information may not reflect the reality in the problem space. Some of the limitations of this approach are discussed in Doyle (2005).

A Fortiori Arguments

As highlighted in Section 3.4.1 The nearest neighbour may not always form the most convincing argument. This is an issue that Doyle et al. (2004) have addressed. They use specially designed similarity meaures to form *a fortiori* arguments in favour of the CBR system's prediction. Most parents are familiar with the use of *a fortiori* arguments by children[1]. *A fortiori* arguments are used to argue a case beyond reasonable doubt. Let us consider an example of a child using an *a fortiori* argument to plead their case to see the latest Harry Potter movie. Figure 3.2 shows an example of a child called Mark (the triangle) who wants to see the latest Harry Potter movie. The circles represent the children who have seen the movie and the squares the children who have not. Mark knows that Kate is the closest in age to him and she has seen the movie. But Mark knows that the older you are the more likely you are to be allowed to see the movie. If Mark were to use Kate as an argument to convince his parents to let him go to the movie, there is a possibility that Mark's parents can argue that Mark is still a little too young to go. However Mark knows that if he uses John who is younger than him as his argument to see the movie, he has a stronger case.

Doyle et al argue that the same principle applies in classification tasks whereby cases that are between the query case and the decision boundary provide more convincing explanations. That is, cases that are more marginal on the important criteria are more convincing. With such cases the user is better able to assess whether the classification of the target case is justified. To form such arguments Doyle et al. have developed a system that attempts to select a case nearer the decision boundary than the nearest meighbour. A major step in this process is to use explanation utility measures (Doyle et al. 2004). These measures are dependent on the classification of the case being explained and based on domain knowledge of the effects of feature-values on the class value. Figure 3.3 shows

[1] "*a fortiori* - adv. for similar but more convincing reasons: if Britain cannot afford a space program, then, a fortiori, neither can India." - Collins English Dictionary

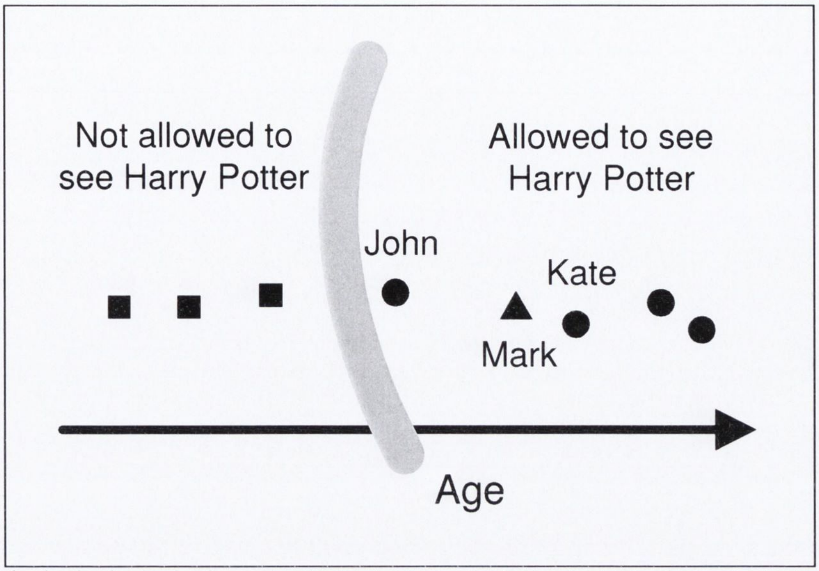


Figure 3.2: Using *a fortiori* arguments

the explanation utility measure for the feature Age when the classification is Allowed to See Harry Potter. When calculating the utility between a case x and a query case q, if the age of x is older than q, the utility measure for age is in the range of 0-1. However if q is older than x the utility is 1. Therefore the utility for the *Allow* argument works by favouring younger cases than the query case. Alternatively when trying to argue that someone should not be allowed to the cinema an alternative graph that favours older cases would be used.

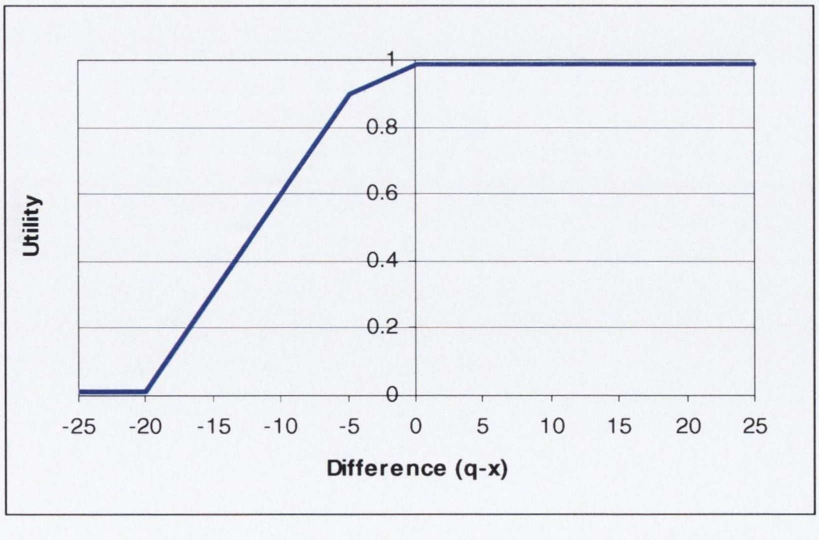


Figure 3.3: Explanation Utility graph for the Age feature and the 'Allowed'.

Once the utility measures for each feature and classification combination has been defined, the most convincing case to support a particular classification is selected using the following process:

1. Get Nearest Neighbours.
2. Perform Classification using Nearest Neighbours.
3. Select explanation utility measures to use based on classification.
4. Reorder Nearest Neighbours of the same class using selected explanation utility measures.

As an example of the kind explanations produced by this system please consider Table 3.4. Here we can see both the nearest neighbour which would normally be selected as the explanation case and the actual explanation case selected using especially designed utility measures. In this case the implicit argument is; given that a previous individual was both lighter, had consumed more units of alcohol and was discovered to be under the limit then it is reasonable to assume that the target case is too. This does form a more convincing argument but it is worth noting that it requires domain information in order to design the utility measures and it is dependent on the user having that same domain knowledge. Without such domain knowledge the argument may be lost on novice users and the effects of differences between the feature values in the explanation case and the target case may be off-putting. This is an issue Doyle et al. later addressed by supplementing their explanations with texts explaining the effects of feature-value differences.

Table 3.4: Sample a- Fortiori Explanation

Features	Target Case	Nearest Case	Explanation Case
Weight	76	76	73
Duration	60	60	60
Gender	Male	Male	Male
Meal	Full	Full	Full
Units	2.9	2.6	5.2
BAC	Under	Under	Under

CBR Explanations for Black Box Systems

As we saw in Section 2.3 providing interpretable explanations for Black Box systems is a difficult task. We investigated the prospect of introducing CBR explanations as a means of explaining the output of Black-Box Systems such as Ensembles, Neural Networks and Support Vector Machines (Nugent and Cunningham 2005). Such systems have proved to be effective in modelling non-linear problems but are plagued by a lack of interpretability in application to real-world problems. As we saw in Section 2.3 various rule-based and decision tree systems have been developed but these often fail to capture the workings of the Black Box System and can led to poor and convoluted explanations (Andrews et al. 1995, Tickle et al. 1998, Zhou and Jiang 2003).

Given the characteristics of case-based explanations discussed in Section 3.4.1 we developed a CBR explanation system for Black-Box Systems used in regression tasks. Key components in our approach were the use of the Black-Box as an oracle and the use of local models to describe the feature-space in the area of interest. In using the Black Box as an oracle we simply present it with sets of feature values similar to those of the target problem and record its output. In this way we can build up an artificial case-base around the point of interest in the feature space. To try and capture the information stored in this artificial case-base about feature importance and influences we build a locally weighted linear regression model on the data (Atkeson and Moore 1997). Once we have this information we can then use it in selecting an explanation case and offering the user an insight into the influences of the different features.

To make this process a little clearer consider the following simple example. Imagine we have a neural network model that predicts the Blood-Alcohol content (BAC) in a person's blood after they have consumed a certain number of units of alcohol and stopped drinking. The graph of the function learnt by the neural network (NN) might look something like the one in Figure 3.4. As the consumed units are absorbed into the body the BAC value increases until it has reached a maximum value from where the level then begins to fall back down as the body processes the alcohol.

The function learnt by the NN is of course unknown to us and so when we ask it to provide a prediction for the BAC level for time T we will simply be presented with a prediction P(T) with no insight on how this prediction was derived. We can then begin to interrogate the NN with cases similar to our query case (QP) and build a case base that

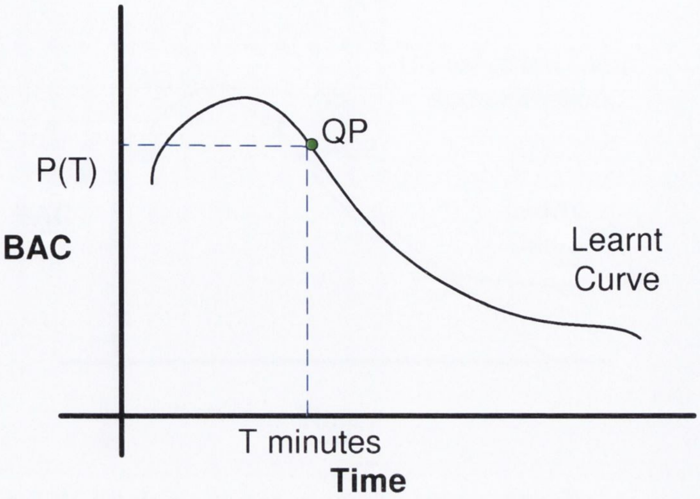


Figure 3.4: The function learnt by the NN-BAC vs. Time

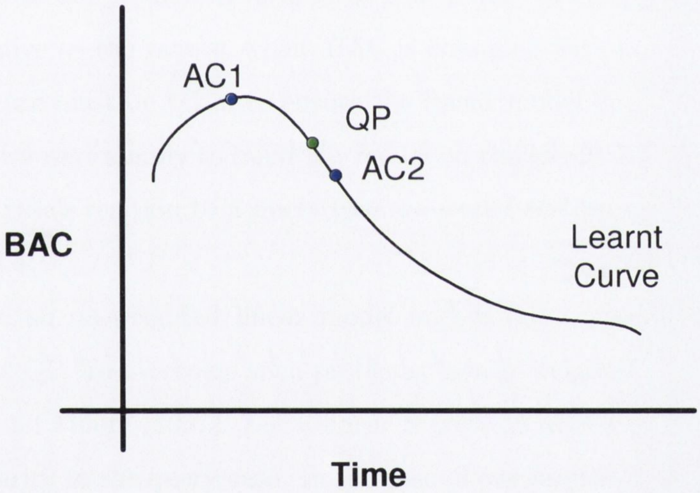


Figure 3.5: Artificial Data Points AC1 and AC2 are created around QC

describes the NN's function around QP as seen in Figure 3.5.

Once we have built an artificial case-base around QP that accurately describes the black box's function in that area we are then left with the problem of how best to extract feature rankings from it. For regression tasks, multivariate linear regression models would seem to be the best candidate for deriving such information. A linear regression model provides us with a set of coefficients for each feature that can then be used to infer how sensitive the prediction is to changes in each feature's value and so its relative importance.

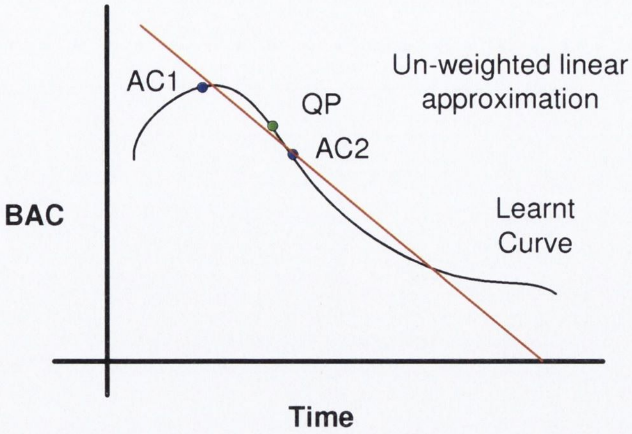


Figure 3.6: Fitting a linear model to the artificially created data

The coefficients also provide information about whether a feature is negatively or positively correlated with the prediction variable at that point. In our particular example the coefficient would give us the rate at which BAC is changing with time at that particular point. However, care must be taken to ensure the linear model derived truly reflects the NN's function. If we were simply to build our model on the locally built case base without attention to each case's relation to a query case we would end up with a model like that shown in Figure 3.6.

This would be an un-weighted linear model and is not a good model of the NN's behaviour at point QP. To overcome such problems locally weighted linear regression can be used Atkeson and Moore (1997). Local linear regression allows us to weight each case based on its similarity to the query case. In the case of our implementation we use an un-weighted Euclidean distance measure as the weighting function. For instance AC1 would be given a lower weight than AC2 and so would have less of an impact on the derived model. This gives us a model that is close to a tangent to the curve at QP and gives us a slope value that truly reflects the NN's function as can be seen in Figure 3.7.

The above example is quite simple and the information extracted may not seem to be very useful, but in a multi-dimensional problem such information is extremely useful. In such a case, a hyperplane is produced and each coefficient of that model gives us a sense of how each feature relates to the predicted value. How the feature salience information is used in the final explanation stage is very much dependent on the context the system is being used in; on what is deemed most effective and useful for a particular domain and the

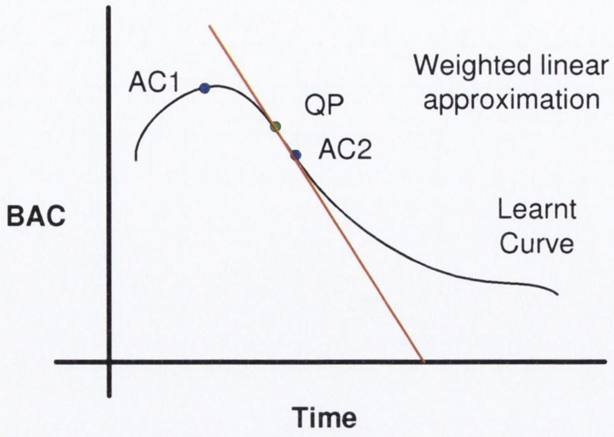


Figure 3.7: Fitting a locally weighted linear model

Table 3.5: An Example of CBR Explanation of a Black Box System

	Query Case	**Explanation Case**
Weight (kgs)	85	80
Duration (mins)	60	60
Gender	Male	Male
Meal	Lunch	Lunch
Amount (Units)	4.9	4.9
BAC	12.0	14.0

Explanation:

The important features in determining this preidction, listed in order of impact, are: `Amount, Duration, Gender, Weight` and `Meal`

`Weight` being smaller in the explanation case has the effect of increasing its `BAC` value

users in that domain. As an example of the type of explanation produced by this system consider the explanation displayed in Table 3.5. This case-based from of explanation has advantages in terms user acceptability and the use of real evidence as outlined in Section 3.4.1. The addition of discursive text helps inform the user as to the relevance of the retrieved Explanation Case and explains the effects of any differences that might exist between it and the Query Case.

3.4.3 Assessing Confidence in CBR

Predicting confidence in a system's predictions is not a new idea and in fact it was a feature in some early Knowledge-Based and Expert Systems (Davis (1982), Watson and Gardingen (1985)). Systems designers have long realised that it is better for the system to admit that it simply doesn't have enough information to answer confidently then to speculate wildly. Such systems were designed to recognise and handle situations where the system is inadequate. Information on how to recognise such situations was encoded in the meta-level knowledge of the system which controls how the system operates by the system designers.

Many CBR systems act as decision aids too and as mentioned in Section 3.4.1 predicting confidence is an issue for CBR Systems too. McLaren and Ashley recently addressed the issue in CBR with an approach that is very similar to that taken in Knowledge-Based and Expert Systems. Their system uses Meta-Rules as a means of detecting possible errors. If the conditions of the rules are met than the system is deemed to be going beyond its capabilities. Their System, SIROCCO, operates in an engineering ethics domain which highlights some of the error sensitive domains for which CBR systems can be used.

Alternatively other researchers have taken a far more CBR-light orientated approach to the task (Cheetham and Price 2004, Delaney et al. 2005a). Both have looked at using measures based on similarity information derived from the cases used in any decision as indicators of confidence. Examples of the kind of measures considered are:

- The similarity distance between the target case and its nearest neighbour.
- Percentage of number of cases retrieved with the predicted class value.
- Average similarity of cases with the predicted class value.
- Sum of the similarities of retrieved cases with the predicted class.
- Average similarity of retrieved cases without the predicted class value.

It is easy to imagine that many such indicators could be created and indeed Cheetham and Price (2004) propose 12 such measures. However both Cheetham et al. and Delaney et al. have found such measures used individually to be poor confidence measures. However both sets of researchers have looked at means of combining the measures so that a more robust aggregate confidence measure can be produced. Cheetham et al. used a decision tree to learn when it is best to use certain measures and then combined the selected measures. Alternatively Delaney et al. used an ensemble-like approach to combine a set of simple base indicators. While Cheetham et al. found their approach to be of limited success Delaney et al. found their approach to be very promising in the very sensitive spam domain. However it is clear that confidence measures based on case similarities don't seem to offer any real insight in system confidence. Most of the simple measures strive to capture the marginality of a prediction, to describe how close a case was to the decision boundary. None of the measures so far considered seem to reliably reflect the case's marginality and so fail as confidence measures. Using combinations of indicators has been found to be more effective but perhaps there is a more direct way of reflecting case the marginality of a case.

3.5 Conclusion

We have reviewed the CBR methodology and highlighted where knowledge is stored in the CBR system. It is clear that although aspects of the CBR approach are transparent the methodology isn't wholly so and this limits the effectiveness of traditional CBR-light explanations. We have reviewed research in the area that has addressed these issues and highlighted some limitations. Our review has made it clear that a successful CBR explanation framework would have to contain

- a means of providing user feedback that reflects the CBR methodology and the knowledge in the system,
- a means of selecting good explanation cases,
- a means of providing reliable confidence measures.

In the next section we discuss the development of our Explanation Framework and how it addressed these issues.

Chapter 4

Explanation Framework

The general realisation within the machine learning community that the tools that we build need to be interpretable to real life users has spurred a renewed interest in designing systems that provide explanatory output. As is clear from Section 3.4 Case-based Reasoning has a legacy of research and many favourable characteristics in terms of providing explanations. However, the traditional approach in CBR-light applications of simply supplying the nearest neighbour has been found to have shortcomings in terms of providing satisfactory explanations. We have highlighted two issues in particular that need to be addresed in order to create satisfactory and convincing explanations:

- The selection of cases to present to the user
- Explaining the details and relevance of retrieved cases

The primary motivation in providing users of CBR systems with interpretable explanations is to increase their confidence in the system. However, as is pointed out by Cheetham and Price (2004), people can quickly lose confidence in a system if it makes recommendations which then turn out to be incorrect. As a means of combating the possible loss of user confidence, researchers have tried to devise mechanisms capable of determining when a the system might be making a mistake. Such mechanisms strive to determine the level of certainity which the system can attach to its recommendation. This information is then delivered to the end-user is terms of a confidence measure. By alerting users to when the system might be making an error, long-term user confidence can be maintained even when the system does make mistakes.

As we have seen in Section 3.4, these are all issues which have recently been addressed by researchers. However, we wished to develop a single unified solution to all of these problems which was in keeping with the characteristics that set CBR apart from other machine learning techniques. We also wished to develop a solution which could potentially be applied to many different domains without specialist knowledge from that domain having to be encoded into the system.The system as developed by Doyle et al. (2004) required that a domain specialist be consulted so that explanation utility measures could be designed and *a fortiori* cases found. The system we have developed strives to avoid this effort. To realise these aims we have developed a solution that tackles all three problems in a localised way. We envisioned a solution that, like our work in CBR explanations for Black Box systems, would build local models on selected small portions of the case-base (Nugent and Cunningham 2005). By applying this localised approach we can provide users with informative feedback that reflects the case-base in the relevant region of the feature-space and at that time.

CBR is particularly suited to *weak theory* domains. Such domains are characterised by the fact that simple generalised principles don't apply universally across the entire solution space. By applying local models we can generate explanations that reflect the nature and charcteristics of such domains. The framework we have developed has three key facets:

- It selects cases that form *a fortiori* arguments in an automated way and without domain knowledge,

- It provides discursive texts describing the effects of differences in feature-values between the Query Case and the Explanation Case,

- It provides a measure of confidence in the system's recommendation to alert the user to when there is doubt about a system decision.

This Chapter describes the design of our general framework for case-based explanations and each of the component stages of that process. We begin in the next section with a general outline of the approach taken before discussing each stage in the process in more detail

4.1 Design Of General Framework

Each explanation produced by our framework is especially tailored to each recommendation made by the underlying CBR recommendation system. The flow of execution of the explanation process carried out by our framework can be broken down into five distinct phases as can be seen in fig. 4.1. In each of these stages, particular tasks necessary for producing the explanation are carried out. We will now discuss each stage in turn:

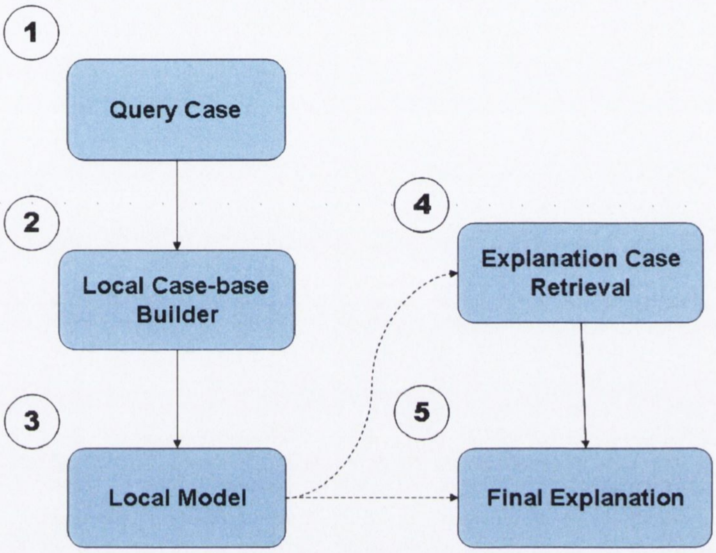


Figure 4.1: Flow diagram of the Operation of the Framework

1. **Query Case:** Each explanation produced is tailored to the particular set of inputs on which the underlying CBR system has made a recommendation. These set of inputs along with the systems recommendation form a Query Case and this case is then used to seed the rest of the explanation process.

2. **Local Case-base Builder:** In the second phase we wish to capture the information that lies in the region of the case-base around the Query Case. We do this by building a local case-base. The local case-base is a subset of cases from the original case-base that traverse the decision boundary at that point in the feature space. This ensures that we have captured information about the local relationships between feature-values and the class label. The manner in which we do this will be discussed in more detail in Section 4.1.2.

3. **Local Model** The next task is to transform the information stored in the local case-base into a more interpretable and useful form. To do this we build a model on the local case-base which will offer us an insight into the information stored within. The choice of model we use is very much determined by the information we need to extract and the tasks that we would like to use the model to drive. There are three tasks for which we would like to use our localised model:

   A Selection of an explanation case

   B Explaining the effects of feature differences

   C Determining how certain a recommendation is

   The model we have selected to perform these tasks is the Logistic Regression model. It is a simple but powerful statistical model which is probabilistic and offers insights into the feature-value relationships in a natural and intuitive way. We will discuss this model, why we selected it and how we use it further in Sections 4.1.1, 4.1.3 and 4.1.4.

4. **Explanation Case Retrieval:** In the case retrieval process we wish to select the case that forms the most convincing argument in favour of the system's recommendation. As we saw in Section 3.4 selecting cases that form a fortiori arguments is an effective way of achieving this. This means finding cases that are more marginal, that lie nearer the decision boundary. Previously this has been done by using specially designed similarity metrics which encapsulate domain knowledge about the relationship between features and class labels. We find a fortiori cases in an automated way which avoids the insertion of domain knowledge. We do this by using the Logistic Regression model's probabilistic characteristics to determine which cases are the most marginal. We will explain this process in more detail in Section 4.1.3.

5. **Final Explanation Stage:** In the final explanation stage two important tasks are carried out; feature-value differences between the explanation and the Query Case are explained and the confidence in the underlying CBR system's recommendation is determined. To explain the feature-value differences the Logistic Regression model is again employed. The characteristics of the design of the logistic regression model mean that it can explain the effects of such differences in a natural way. The design of the logistic regression model is explained in Section 4.1.1 and how we use it to

explain the feature difference in Section 4.1.4. Since the logistic regression model is probabilistic, it is a trivial task to generate a probability of a given Query Case being of a certain class. If the probability is below a certain threshold then we can determine that confidence in a recommendation is low or alternatively that there is a high degreee of confidence in the recommendation. How such a threshold can be determined is discussed in Section 4.1.5.

This process is carried out at run-time and for each recommendation that requires an explanation. A key component of out framework is the use of Logistic Regression as our local model as it is then used in many of the other processes in the framework. In the next section we will discuss the selection of this model, its key characterisitics and how it works.

4.1.1 Choice of Local Model

At first glance the approach that we have taken in fitting a statistical model to the data in the case-base and using it to determine the effects of feature differences may look similar to the approach taken by McSherry which we described in Section 3.4.2. However, there are a number of important differences in both the model that we use and in the manner in which it is employed that mean that the approaches are in fact quite different. Firstly we do not attempt to fit the model to the entire case-base. We argue that fitting a gobal model over a local learning system isn't in keeping with the CBR philosophy. As we highlighted in the introduction to Chapter 3 some of the main characteristics that distinguish CBR from other machine learning techniques are:

- it is a lazy learning method that supports incremental learning
- it is effective at solving *weak-theory* problems

Fitting a model to the entire case-base prior to the system being used is to fit an eager learning method over a lazy method. The explanations produced by such a system may not reflect the case-base if it is updated. Furthermore, CBR has advantages in terms of tackling *weak-theory* problems and complex problems. These are problems in which simple statistical models fail to capture the underlying patterns in the data. These problems have a complexity and non-linear nature which violates many of the assumptions made in simpler statistical methods. To overcome these problems we apply our model on small portions of the case-base at runtime. This ensures that we are not only using the most up-to-date data available but also that we are not fitting a simple model over complex data. Although simple models may not be able capture the trends throughout the case-base they are adequate when applied to localised regions of the case-base. This localised approach is used extensively in statistics and we have used it previously in our work on CBR explanations for black-box systems. In fact, localised logistic regression techniques have been used in text classification (Zhang and Yang 2004).

In terms of statistical models, Naïve Bayes and Logistic Regression represent two fundamentally different approaches to problem solving. Naïve Bayes is a generative statistical method while Logistic Regression is a discriminative method. Generative classifiers learn a model of the joint probability of the features and the corresponding class label and make decisions by using Bayes rule to compute the posterior probability of the class variable. In the case of Naïve Bayes this means building a separate probability distribution for each feature that relates it to the class label. For a given set of inputs the probability of a given class is calculated for each feature-value and then Bayes rule is used to combine the individual probabilities and make a recommendation. Logistic regression is a discriminative method and this means that it tries to calculate the probability of a class directly given the the full set of inputs. Thus feature relations are taken into account and noisy features are naturally tuned out. The logistic regression model is also designed with the interpretation of relationships between feature-values and class value in mind. We have discussed, in broad terms, the strengths of the logistic regression model and how it is employed within our framework without really describing the model itself. We will now discuss the model in greater detail before describing each of the ways in which the model is used within the framework.

A Local Model: Logistic Regression

Hosmer and Lemeshaw have written an excellent and comprehensive book on the subject of Logistic Regression (Hosmer and Lemeshow 2000). In the preface to the second edition they point out the huge increase in the use of the modeling technique from its original use within epidemiologic research to use within fields as diverse as *"biomedical research, business and fiance, criminology, ecology, engineering, health policy, linguistics and wildlife biology"*. Logistic regression is a data analysis technique that offers an insight into the relationship between input variables and a target, or class variable. It is specifically designed for binary classification problems and the increase in popularity of the modeling technique is understandable as it offers powerful insights while maintaining model simplicity.

Logistic regression, like linear regression, produces a set of coefficients from which the relationship of an input variable to the target class variable can be deduced. However, unlike linear regression, logistic regression coefficients don't directly correspond to slope values in the same way. In logistic regression tasks, the two possible class values are coded as being either 0 or 1. Because the value predicted by the model, the conditional mean, is no longer an unbounded value as in linear regression but a value between 0 and 1, the data is fitted to a distribution that ensures the outputted value always meets this bounding criteria. To do this, the logistic distribution is applied as can be seen below (4.1).

$$Y(x) = \frac{e^{\beta_0+\beta_1 x}}{1+e^{\beta_0+\beta_1 x}} \tag{4.1}$$

Here $Y(x)$ is the conditional mean for a particular value of x while β_0 and β_1 are the model parameters. The distribution produces the conditional mean, a value between 0 and 1, for any given inputted value of x. Importantly, for binary problems the conditional mean is in fact the probability of class 1 given x.

At first glance this model looks quite intimidating and seems to offer no hope of offering an insight into the relationship between x and our class variable. However, the logistic distribution is chosen because it can be easily transformed into another form which has many of the desirable properties of a linear regression model. By applying the logit transform, equation 4.2, we end up with a simple and interpretable model, the logit (4.3).

$$g(x) = \ln \frac{Y(x)}{1-Y(x)} \tag{4.2}$$

$$g(x) = \beta_0 + \beta_1 x \tag{4.3}$$

The parameters of the logit model can easily be converted into odds ratios. The odds ratio of an event is the odds of that event occurring over the odds of it not happening. For instance, if someone were to state the odds ratio of smokers to non-smokers getting cancer is 2, then this would mean smokers are twice as likely to develop cancer as non-smokers. Alternatively, if we looked at the relationship the other way round, non-smokers to smokers, we would get a odds ratio of 0.5. This means that non-smokers are half as likely to get cancer. In general an odds ratio greater then one for possibility A over possibility B means A makes the event more likely than the alternative while and odds ratio of less then one means it makes it less likely. The logistic regression model makes the calculation of odds ratios quite easy and this is extremely useful and informative. It is this simple relationship between the model coefficients and the odds ratio and their natural interpretation that has made logistic regression such a popular tool. We will first discuss in a very general sense how this is done as it will be of use in Section 4.1.4 and then focus on a particular example that highlights why logistic regression has proved so popular.

In order to extract the odds ratio, two steps are taken. First the logit difference is found. Imagine we are interested in the odds ratio of two different events, $x = c$ and $x = d$. the logit difference can be calculated as in equation 4.4. The logit difference, ld, is simply the difference in the logit function for the two values of x we are interested in. Once this value has been obtained it can then be converted into odds ratio, see equation 4.5.

$$LogitDifference(x = c, x = d) = g(c) - g(d) = ld \tag{4.4}$$

$$OddsRatio(x = c, x = d) = e^{ld} \tag{4.5}$$

The trick with the logistic regression model is that in many cases it isn't necessary to calculate the logit difference. If the model variables have been properly coded then the desired information can usually be got by simply looking at the model coefficients. As an example, consider a hypothetical situation where we have developed a model that relates smoking to the development of cancer. Our hypothetical model might look something like that shown in equation 4.6.

$$g(Smoker) = 0.3 + 0.69Smoker \tag{4.6}$$

If we code our smoking variable as being equal to 1 if someone smokes and 0 if they don't then the calculation of the logit difference is simply equal to the *Smoker* coefficient (4.7).

$$\begin{aligned} g(Smoker=1) - g(Smoker=0) &= 0.3 + 0.69(1) - (0.3 + 0.69(0)) \quad (4.7) \\ &= 0.69 \\ OddsRatio(Smoker) &= e^{0.69} \\ OddsRatio(Smoker) &= 2 \end{aligned}$$

$$(4.8)$$

As can be seen above in 4.6 we need not have bothered calculating the logit difference and instead just used the model coefficient. This is also true for continuous and multi-value nominal variables if they are coded correctly (Chapter 4, Hosmer and Lemeshow (2000)). Once we have the odds ratio the relationship between input variable and the class variable is clear. We have focused most of our discussion on examples with only a single input variable for simplicity sake but the above observations are also true in multi-variable problems. In the next section we discus how information derived from the logistic regression model can be used to provide convincing explanations.

4.1.2 Creating a Local Case-base

A very important element of the approach that we have taken is the generation of a local case-base. A key characteristic of the local case-base that we generate is that we want to ensure that it contains cases from both sides of the decision boundary. We use a simple iterative algorithm to do this:

1. We begin with an ordered list of the cases in the case-base based on their similarity to the Query Case and an empty local case-base. The similarities of each case are calculated using the *k-NN* algorithm.

2. We then add copies of cases to the local case-base iteratively until we have at least Q cases of each class type and the similarity of the Nth+1 case and the Nth case of the ordered list are not equal. The last clause of our stopping condition ensures that if there are ties in similarities between cases that all the cases are included.

To make this process clearer consider Figure 4.2. On the left hand side we have an ordered list of cases and on the right an empty local case-base. The different class values

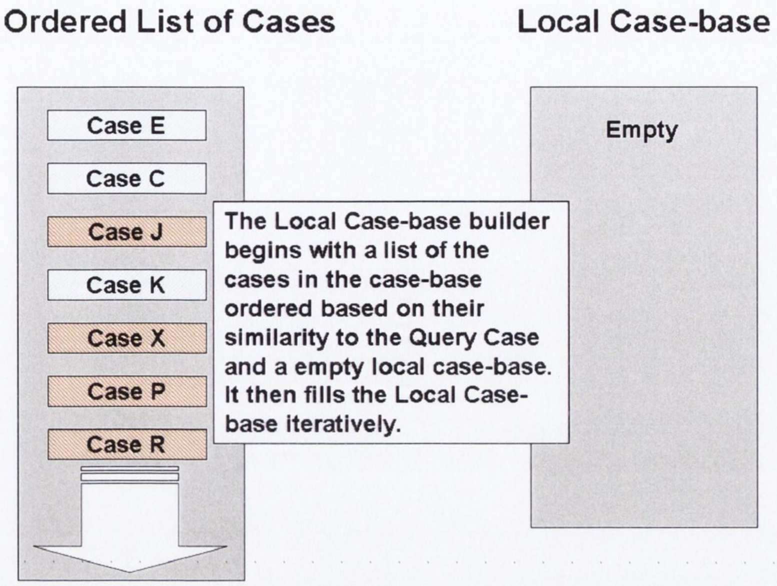

Ordered List of Cases
Local Case-base
Case E
Case C
Case J
Case K
Case X
Case P
Case R
Empty
The Local Case-base builder begins with a list of the cases in the case-base ordered based on their similarity to the Query Case and a empty local case-base. It then fills the Local Case-base iteratively.

Figure 4.2: Building a local Case-base

of each case are represented by their different colour schemes, blue strips for one class and red for the other. We then start to add cases iteratively as can be seen in Figure 4.3.

In this example we have set the parameter Q to be two. Eventually we end up with the situation in Figure 4.4. Here we can see that our local case-base has been filled. In this particular case we ended up with 3 cases from each class but there are different reasons for the inclusion of extra case of each class. In the case of the blue class there is an extra case included because Case K was reached before two cases of the red class were found. The extra red case, P, was included because it actually had the same similarity score as case X. In this situation we have no way of distinguishing between each case and so they are both included. Finally Case R isn't included since it has a lower similarity score and the quota for each class type has been met.

This approach ensures that we have cases from both sides of the decision boundary. As an example of how the local case-base might look in the feature space consider Figures 4.5.a and 4.5.b. In Figure 4.5.a we can see the decision boundary and how the cases are distributed around it. The three nearest neighbours used to make the original recommendation are surrounded by a dotted circle. We wish to expand around these cases so that we have a case-base with a good representation of the deciding factors in class values. To do this we use the iterative algorithm described earlier and the resulting local case-base

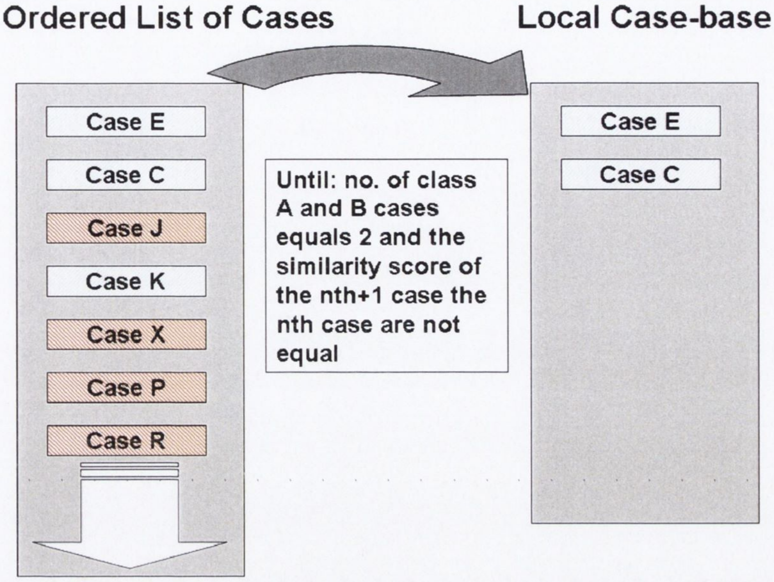

Ordered List of Cases
Local Case-base
Case E
Case C
Case J
Case K
Case X
Case P
Case R
Until: no. of class A and B cases equals 2 and the similarity score of the nth+1 case the nth case are not equal
Case E
Case C

Figure 4.3: The local Case-base is filled iteratively

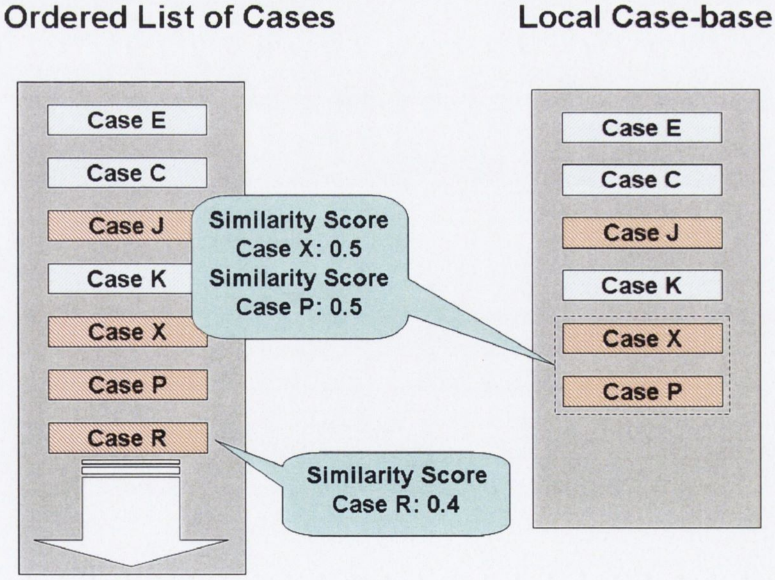
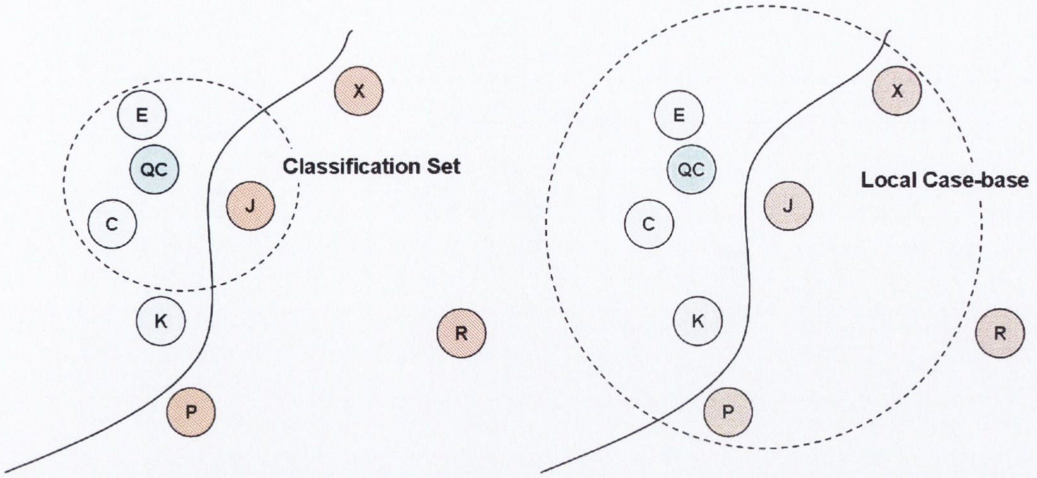


Figure 4.4: The final Local Case-base



a. The Nearest Neighbours b. The Local Case-base

Figure 4.5: The distribution of cases and the local case-base

can be seen in Figure 4.5.b. This algorithm requires that one parameter, Q, is set. In Section 4.1.5 we will discuss how the best value for Q can be chosen.

4.1.3 Finding a-Fortiori Cases

In selecting a case to present to the user as an explanation we would like to form as strong an argument as possible. In the framework we have developed this is done by selecting cases that form *a fortiori* arguments. This means finding cases that are more marginal, that lie nearer the decision boundary. Previously this has been done by designing special similarity measures that encode domain knowledge about the relationship between feature-values and class-values (Doyle et al. 2004). Using the local Logistic Regression model we can generate *a fortiori* arguments dynamically and without any prior domain knowledge. As discussed in Section 4.1.1 Logistic Regression models allow us to generate a probability for a given set of inputs, a case, being a certain class. In the Explanation Case retrieval process we can then use this to find an explanation case that is nearer the decision boundary and so a more convincing argument. We consider each of the cases in our localised case-base as a candidate case for inclusion in the explanation. By passing each of our candidate explanation cases through our local logistic model using Equation 4.1 we can generate a probability for each of being a particular class. A case that is nearer the decision boundary and of the same class as our CBR system has predicted will have a more marginal probability and so this should be the case we select. However, finding the most marginal case is not always the best policy if the selected case is so different to the

Query Case that it seems irrelevant. This was noted by Doyle, Cunningham, Bridge and Rahman (2004) and they adjusted the explanation similarity measures appropriately. In our case we can ensure that the selected case is reasonably similar to the Query Case as it selected from the Local Case-base.

Table 4.1: The Query Case and the Candidate Explanation Cases

Features	Query Case	Nearest Neighbour 1	Nearest Neighbour 2	Nearest Neighbour 3
Weight	88	82	79	76
Duration	120	120	120	120
Gender	Male	Male	Male	Male
Meal	Full	Full	Full	Full
Units	5.2	5.0	7.2	4.6
BAC	Under	Under	Under	Under
Probability	0.98	0.97	0.89	0.96

To make this process a little clearer we will discuss it in relation to an example which again is taken from the BAC domain. In Table 4.1 we can see a Query Case, its predicted classification and three candidate explanation cases which are in fact the Nearest Neighbours used to classify it. In order to select a case to use as an Explanation Case we first run each of the cases (including the Query Case) through our local logistic regression model. This gives us the set of probabilities that can also be seen in Table 4.1 which were calculated using Equation 4.1. The logistic regression model is built on the entire local case-base and the parameters of the model are estimated by minimising an error function using standard approaches as we describe in Section 5.2.3. We can see that *Nearest Neighbour 2* has the lowest probability and so is the case nearest the decision boundary. We then select *Nearest Neighbour 2* as our Explanation Case as can be seen in 4.2. Although the case that we have selected forms a better argument than if we had selected the nearest neighbour it does contain feature-value differences that may make it seem quite different and irrelevant to the present Query Case. The case we have selected does however form a better argument since: *"although the Explanation Case had consumed more units of alcohol and weighed less, they were under the limit so it seems reasonable that our Query Case should be too"*. However such an argument is made in a implicit fashion and is dependent on the user having the same domain knowledge.

We can make this argument more explicit to the end user by explaining the effects

Table 4.2: The Query Case and Selected Explanation Case

Features	Query Case	Explanation Case
Weight	88	79
Duration	120	120
Gender	Male	Male
Meal	Full	Full
Units	5.2	7.2
BAC	Under	Under

of the feature differences between the Query Case and Explanation Case. In the next Section we will explain how the influence of the feature differences that exist between the Query and Explanation Case can be explained using information extracted from the Local Logistic Regression model.

4.1.4 Describing Feature-Value Differences

As we stated in Chapter 3 the success of a case-based explanation lies in ensuring that the case presented to the end user seems relevant and the argument posed by the presented case is clear. In order to ensure this, we need to explain to the end-user the effect of any feature-value differences that might exist between the Query and the Explanation Case. To do this using only the knowledge that is stored in the case-base we again use the Logistic Regression model. One of the great strengths of the Logistic Regression model is that it can extract information about the influences of feature-values on the class value in terms of odds ratios. Using Equations 4.4 and 4.5 from Section 4.1.1 we can substitute each of the feature differences into the equations individually and get an odds ratio for each. Using the odds ratio we can then determine the effect of the change. As discussed in section 4.1.1 an odds ratio greater than 1 means that a feature difference makes an event more likely and vice versa. Looking at each feature difference in turn we can then make lists of features differences that make the classification more likely and those that have the opposite effect.

As an example of this process we will extend the example used in Section 4.1.3. As can be seen in Figure 4.3 there are feature-value differences in terms of weight of the two subjects and in terms of the number of units consumed by each. Using the Logistic Regression model we can determine the effects that feature-value differences would have

Table 4.3: The Odds Ratios of Associated with each Feature-value Difference

Features	Query Case	Explanation Case	Odds Ratio
Weight	88	79	2.3
Duration	120	120	1
Gender	Male	Male	1
Meal	Full	Full	1
Units	5.2	7.2	3.1
BAC	Under	Under	

if they were substituted into the Query Case. On the right hand side of Table 4.3 we can see the odds ratio produced for each feature by the Logistic Regression model. For the features `Duration`, `Gender` and `Meal` the odds ratio is 1 since there are no feature-value differences between the two cases for those features. Both `Weight` and `Units` have a odds ratio greater than 1 meaning the differences that exists in these features have the effect of making the Query Case more likely to be under the limit than the Explanation Case. By simply looking through each of the odds ratios produced we can quite easily make lists of the feature differences that make a classification more likely and those that have the opposite effect. These lists can then be used with simple text templates to form discursive texts describing the effects of feature-value differences to the end user. Examples of the texts produced can be seen in Section 4.2.

4.1.5 Providing Confidence Measures

Another key facet of the framework that we have developed is that it can produce confidence measures so that users can be alerted to when there is a suspicion that the system is making a mistake. The Logistic Regression model can produce a probability of a case belonging to a particular class and this can be used to determine the confidence we should have in a recommendation. As shown in Table 4.1 we can pass the Query Case through the model and this gives us a probability of the recommendation being correct. If the probability is below a certain threshold we can alert the user that we have low confidence in that recommendation. However this then requires us to define a threshold of confidence and this is not a straightforward issue.

A key issue in providing any confidence measure is ensuring that we don't give false confidence in the system's recommendations. When the system alerts the user that it is confident in a recommendation it is important that the recommendation is in fact correct. If the system is falsely confident the user will quickly lose faith in the system and the whole purpose of providing confidence measures is undermined. Conversely bringing too many correct recommendations into question is damaging also. Constantly supplying users with recommendations attached with a cavet expressing uncertainty about that recommendation is also bound to damage their confidence in the system. There often is a trade-off between these two conflicting desires and a threshold level where there is a suitable balance must be chosen.

We can characterize our wish for accurate confidence as being our Confident Correct Rate (CCR) as defined in Equation 4.9. Likewise we can encapsulate our need to minimise pessimism in the Not Confident Correct Rate ($NCCR$) as defined in Equation 4.10.

$$CCR = \frac{CC}{CC + CI} \tag{4.9}$$

$$NCCR = \frac{NCC}{NCC + NCI} \tag{4.10}$$

Where CC is the number of times the measure is confident and the system is correct and CI is the number of times measure is confident and the system is incorrect. Likewise NCC is the number of times the measure is not confident and the system is correct and NCI is the number of times the system is not confident and is right to be so. To make the definition of these parameters a little clearer we have displayed them in the form of truth table in Table 4.4.

Table 4.4: A Truth Table Defining the Equation Parameters

	Incorrect	**Correct**
Confident	CI	CC
Not Confident	NCI	NCC

Clearly we would like to maximise the Confident Correct Rate while minimising the Not Confident Correct Rate. In reality this isn't ever possible as to do so would in fact mean discovering an improved classifier. If it is possible to accurately predict when the system is making a mistake while also ensuring that we are not attaching uncertainty to correct recommendations then when confidence is low we could just reverse our recommendation. Improvements in one criterion will come at the cost of the other. One way to set a threshold is to choose the level that balances the trade-off. This can be found by finding the threshold that minimises Equation 4.11. However, the issue isn't often that simple and in some domains there may be more of a cost associated with one criterion over another.

$$\textit{Trade-Off} = \frac{NCCR}{CCR} \tag{4.11}$$

In the spam domain the cost of incorrectly classifying a mail as spam is extremely high. This has lead researchers to investigate the use of confidence measures as a means of sorting spam into definite spam and probable spam categories (Delaney et al. 2005a). The probable spam folder contains mail which is classified as spam but for which there is low confidence. The user can then occasionally check the probable folder to see if any mail has been misclassified. Ideally in such a system the number of mails in the definite spam folder is maximised while the number in the probable folder is minimised so as to reduce the load on the user. However there would be a heavy penalty to pay for misclassifying a mail as definite spam and such a mistake could not be tolerated. Reducing the number of misclassified definite spam is equivalent to maximising the Confident Correct Rate and this can only be done at the cost of the Not Confident Correct Rate. However, in the spam domain this balance is swung in favour of the minimising the Confident Correct Rate as such mistakes can prove far more costly than on the other criterion.

A further complication is that confidence assessment mechanisms often involve parameters which must be set. In our own confidence system we must chose the number of each class value we would like to have included in our local case-base. Choosing the parameter that maximises performance while also choosing the threshold that satisfies the criteria of the domain for the CC and NCC rates can be a confusing process. Here, the threshold refers to the level of confidence that must exist before a recommendation is labelled as being confident. For instance, a level of 90% might be chosen as the threshold. Looking at table 4.1 the probabilities attached to each of the cases mean that if they were to be recommendations Nearest Neighbour 1 and 3 would be labelled confident while Nearest Neighbour 2 would be labelled as Not Confident.

The characteristics of the factors involved in this problem have led us to investigate a concept similar to Receiver Operating Characteristics (ROC) analysis as means of examining the factors involved (Flach et al. 2003). ROC Analysis originated from signal detection theory, as a model of how well a receiver is able to detect a signal in the presence of noise. Its key feature is the distinction between hit rate (or True Positive Rate)

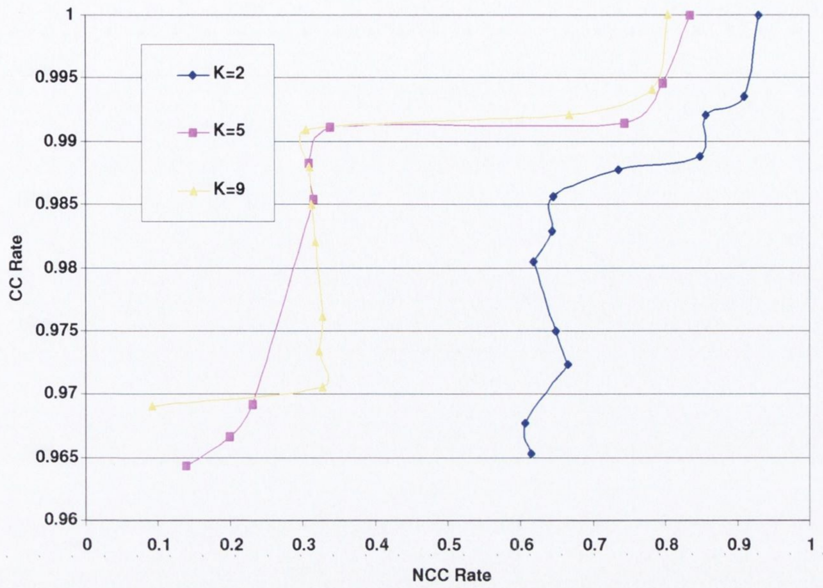


Figure 4.6: The Characteristic Confidence Curve for the UCI Spam Case-base

and false alarm rate (or False Positive rate) as two separate performance measures. It has been introduced to machine learning relatively recently, in response to classification tasks with varying class distributions or misclassification costs. A large part of ROC analysis is the plotting of ROC curves. ROC curves plot the True Positive Rate against the False Positive Rate. Each point on a ROC curve represents a different classifier. The characteristics of the performance of a classifier can be plotted as the parameters of that classifier are changed. ROC curves serve as an excellent tool for visualising the trade-offs involved and assessing the true performance of different classifiers over a range of criteria.

The measures that we have defined, CCR and NCCR, are not dissimilar to the True Positive Rate and the False Negtive Rate and so it was natural to extend the idea of ROC curves to the confidence domain (Nugent, Cunningham and Doyle 2005). To investigate the various parameter options for a particular domain we can plot Characteristic Confidence Curves which are simply the CCR against NCCR. An example of one can be seen in Figure 4.6. Here we can see three different curves which represent different choices of K, the number of each class to be included in the local case-base. The different points along each curve represent different choices of threshold value. It is quite easy using the characteristic curves to find the scheme that best suits the requirements of the domain and investigate the various trade-offs. In the absence of any preference for one criterion over

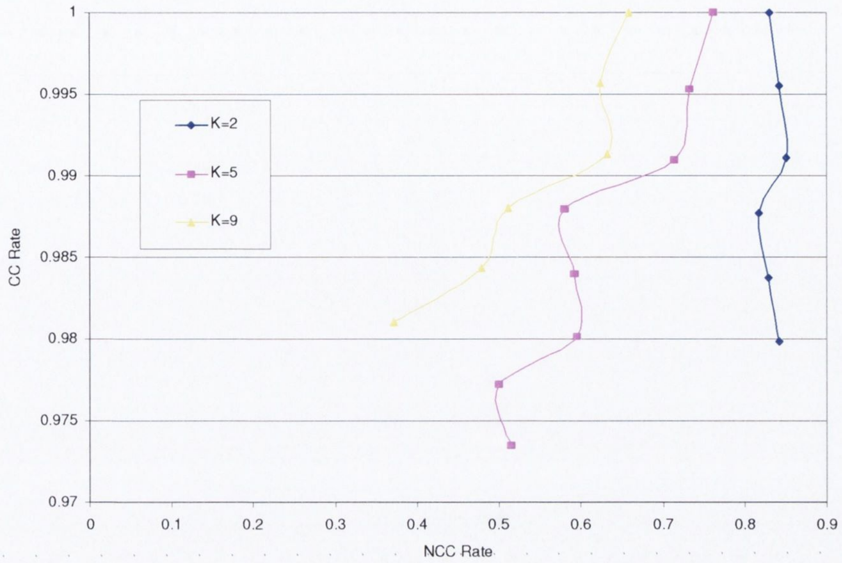


Figure 4.7: The Characteristic Confidence Curve for the E-Clinic Case-base

another the point that lies nearest the top left hand corner maximises the trade-off and is the best solution. However, which criterion to favour may not always be a straightforward issue and the Characteristic Curves provide a neat way of investigating the options available. It is also possible to eliminate certain schemes as being definitely worse than another just as in ROC curves. If the curve of one scheme lies entirely inside another then it is definitely worse than that scheme as under no circumstances does it out perform the other.

In the case of Figure 4.6 the schemes where K equals 5 and 9 are definitely better than the K equal 2 scheme under all possible criteria. The K equal 5 and 9 schemes are better than each other under different possible criteria which are reflected in the manner in which their curves weave over one another. In the absence of any domain requirements the point that lies closet the top left corner lies on the K equal 9 curve so this parameter along with threshold value represented by that point should be the options that we chose.

Another example can be seen in Figure 4.7. In this case the case-base is the e-clinic. Here the scheme with K equals 9 is clearly superior to the other two options.

Using the Logistics Regression model's probabilistic qualities and the methodology we have described it is possible to provide users with an insight into the performance of confidence measures over a range of threshold values and parameters settings. Using this information the user can then decide on settings that best meet their requirements. This can be useful when it isn't known clearly in advance what the costs of different trade-offs are. However, in some applications, such as spam filtering, these costs are well defined. In such cases a more robust system of deciding threshold and parameter settings chould be used such as Decision Theory (Lewis 1995, Amati and Crestani 1999).

In the next Section we will demonstrate how we have combined confidence measures we generate along with discursive texts and our selected Explanation Case to form useful convincing explanations.

4.2 Explanations Produced

We have seen how the explanation framework we have developed can produced a lot of information that can then be presented to the user. As we highlighted in Section 3.4.1 generating effective explanations is a user dependent task. Using the Logistic Regression model it is possible to generate a lot of information about the exact effects of feature differences. However, in the explanations that we produce we are primarily concerned with demonstrating how the framework can be used to produce simple and effective explanations that can be used across many domains. An example of the explanation produced by the framework can be seen in Table 4.5. This is again an example taken from the BAC domain. The Query and the Explanation Cases are presented along with a discursive text and a measure of confidence in the system's recommendation. In this case the confidence in the explanation is high; however this will not always be the case. When there is doubt in the system's recommendation we can use the framework to help provide the user with extra information that might help them in determining the correct classification. In the next Section we will describe the simple measures we take to help assist the user in low confidence situations.

4.2.1 Low Confidence

When confidence is low in a recommendation we would like to assist the end-user as much as possible in deteremining what the correct classification is. By presenting the user with similar cases that lie either side of the decision boundary we can help them make a more informed decision. This kind of approach has previously been introduced by Leake et al. (2001) who used *bracketing cases* to help delineate the limits of the problem

Table 4.5: Sample Explanation

	Query Case	**Explanation Case**
Weight (kgs)	57.0	79.0
Duration (mins)	240.0	240.0
Gender	Male	Male
Meal	Full	Full
Amount (Units)	12.6	9.6
BAC		Over

The prediction for the individual in the Quey Case is: **Over the limit**

The confidence that this prediction is correct is: **high**

Discursive Text:

In support of this prediction we have the person presented by the Explanation Case who was also Over the limit. `Weight` being lighter and `Amount` being bigger have the effect of making the Query individual more likely to be Over the limit than the Explanation individual.

being considered. When confidence is low we can adopt a similar approach: presenting the user with cases from either side of the decision boundary and using discursive texts to explain how the feature values affected the different classifications. Again using the logistic regression model in the same way that it was used to find *a fortiori* arguments, it is possible to find the nearest case on the opposite side of the decision boundary. We can see an example of the kind of recommendation that we can produce in Table 4.6. Here we can see the Query Case flanked either side by the Explanation Case and the Counter Example. By looking at the feature-value differences across the case and text accompanying them the user may be able to determine what the correct recommendation is.

4.3 Conclusion

We have presented an explanation framework that provides interpretable CBR explanations that does not require specialist domain knowledge and maintains CBR's advantages in terms of lazy learning approach and its applicability to *weak-theory* domains. The explanations produced by the framework contain three important elements which all address

Table 4.6: Sample Explanation with Counter Example

	Explanation	**Query**	**Counter**
	Case	**Case**	**Example**
Weight (kgs)	52.0	53.0	73.0
Duration (mins)	270.0	330.0	210.0
Gender	Male	Female	Male
Meal	Lunch	Lunch	Lunch
Amount (Units)	9.1	10.4	9.0
BAC	Over		Under

The prediction for the individual in the Query Case is: **Over the limit**

The confidence that this prediction is correct is: **low**

Discursive Text:

In support of this prediction we have the person represented by the Explanation Case who was also Over the limit. `Gender` being `Female` and `Amount` being bigger have the effect of making the Query individual more likely to be Over the limit than the Explanation individual. However, `Weight` being heavier and `Duration` being longer have the effect of making the Query individual less likely to be Over the limit than the Explanation individual.

As there is low confidence in the prediction we also have a counter example of someone who is similar but Under the limit for you to inspect.

`Duration` being longer has the effect of making the Query individual more likely to be Under the limit than the counter example. However, `Weight` being lighter, `Gender` being `Female` and `Amount` being bigger have the effect of making the Query individual less likely to be Under the limit than the counter example.

short-comings that existed in the traditional CBR-light approach to explanations:

- It selects cases that form *a fortiori* arguments in an automated way and without domain knowledge,
- It provides discursive texts describing the effects of differences in feature-values between the Query Case and the Explanation Case,
- It provides a measure of confidence in the system's recommendation to alert the user as to when there is doubt about a system decision.

.

In the next Section we will describe how the framework was implemented.

Chapter 5

Implementation

In this Chapter we describe how the Explanation Framework was implemented and the technologies used. The Explanation Framework was developed as part of the Knowledge Discovery Project[1] which was funded by Science Foundation Ireland [2]. The Knowledge Discovery Project encompasses projects on many different aspects of machine learning. It quickly became apparent that the development of a unified workbench which defined some of the basic components of machine learning would greatly improve the usability of techniques and code developed as part of project. This led the author to initiate the developement of *Fionn* — a workbench for the development of machine learning tools with a special emphasis on CBR systems (Doyle et al. 2005). The Explanation Framework was developed using *Fionn* and integrated into it so that it provided a generic CBR explanation component.

Case-bases need to be described in and stored in some useful form. *Fionn* is built upon a CBML base which describes how case-bases should be defined using XML. In the following sections we will describe CBML and the *Fionn* workbench itself. We will then describe the various components we developed and integrated into *Fionn* thus providing it with a generic Case-based Explanation Component.

5.1 CBML

Case-base reasoning is dependent on the retrieval and reuse of cases so it is important that the cases are represented and stored in a way that supports the manipulation of cases and

[1] The Knowledge Discovery Project `https://www.cs.tcd.ie/research_groups/mlg/kdp/`

[2] Science Foundation Ireland `http://www.sfi.ie`

their underlying stucture. One prominent Case Representation format was created by the INRECA group in the form of CASUEL (Manago et al. 1994). CAUSEL was based on ASCII files and used Extended Backus Naur Form (EBNF) representation format (Wirth 1977). The current standard for marking up structured, knowledge-rich data is XML, the eXtensible Mark-up Language. XML is a description language that supports meta-data descriptions for particular domains and these meta-descriptions allow applications to interpret data marked-up according to this format. A representation language based on XML has many advantages: interoperability, ease of reuse, as well as the application-independent exchange of data over existing network protocols. Most importantly it allows the developer access to the entire XML tool-set. This tool-set includes fast, reliable document parsers, e.g. SAX and DOM, validating documents e.g. DTDS and XML-Schema, and document transformers, e.g. XSLT. The earliest work in the CBR community on an XML-based case representation language was the introduction of CBML by Hayes et al. (1998). They called this language CBML (CBMLv1).

CBML was further developed and the current version is CBMLv3[3] (Coyle et al. 2003, 2004). The XML community recognised the limitations of the early DTD model and developed an alternative which allows for structural and type validation called XML-Schema. The maturity of XML-Schema led to the redevelopment of CBML and now the description of CBML is stored in an XML-Schema document — the **CBML Schema**. Only documents that follow this schema exactly can be considered valid CBML.

The use of CBML ensures that case-bases are well defined and documentated. CBML provided the solid base on which *Fionn* was developed.

5.2 *Fionn* — The Machine Learning Framework

Fionn was developed on the 1.4 Java platform[4]. The purpose of *Fionn* was to provide abstract descriptions of much of the common functionality encountered in machine-learning in terms of interfaces and basic objects. By providing this basic structure *Fionn* provides the means by which code can be developed in a reusable way. The *Fionn* Framework contains three important elements that form the bedrock of the framework;

case object The simplist and most fundamental element of any case-based system is a

[3] The CBML web page is located at `http://www.cs.tcd.ie/research_groups/mlg/CBML/`

[4] The Java homepage `http://java.sun.com/`

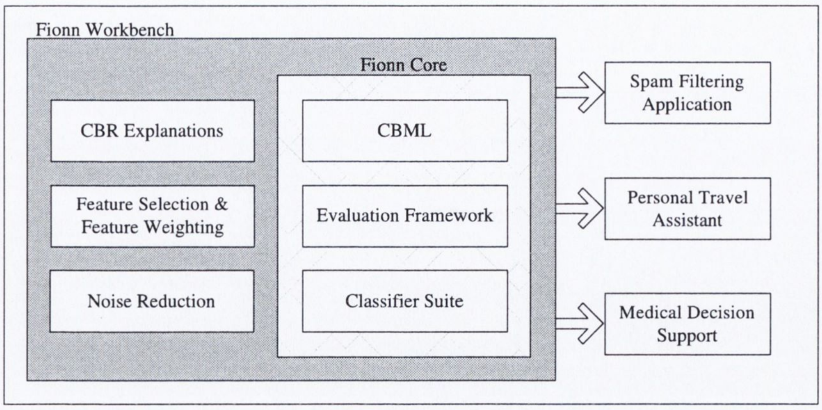


Figure 5.1: The *Fionn* Workbench

case.

casebase object The case-base object represents a list of cases and allows them to be manipulated in a defined way. It is is populated from the CBML XML description of a case-base. All other components within the *Fionn* framework are designed to interface with this defined abstraction of a case-base.

classifier interface This is an abstract description of the functionality of a machine learning classifier. It defines in an abstract non-alogorithm specific manner how a classifier should be built and how it can be called on to perform classficiations. For instance, to build any classifier the `build` method must be called and a `casebase` object passed in. Each classifier that is added to the classifier suite within *Fionn* must meet the criteria defined by the classifier interface. Other componets within the *Fionn* Framework can then be designed to operate on `classifier` objects and this means that components can be designed in a non-specific manner which allows greater flexibility and usability. For instance *Fionn* contains feature selection and evaluation components which can then be used on any classifier types. The suite of classifiers developed for *Fionn* include *k-NN*, Support Vector Machines, Neural Nets, Logistic Regression, Naïve Bayes and Ensemble based approaches. *Fionn* also contains regression tools such Linear Regression and specially adapted Neural Network and *k-NN* models.

Using these basic compoents, higher order functionality has been provided such as an Evaluation Suite, Feature Selection and Weighting, and Noise Reduction components.

There are four applications which have benn in developed using the *Fionn* Framework:

The Personal Travel Assistant An application developed to assist users in finding online flights that best meet their individual travel requirements and perferences.

Spam Filtering Application Delany and Cunningham are working on a spam filtering application called ECUE (E-mail Classification Using Examples) that dynamically adapts to the changing nature of spam e-mails (Cunningham, Nowlan, Delany and Haahr 2003, Delany et al. 2005). Because of the volume of spam e-mail and to its evolving nature their application uses several case-base maintenance techniques that remove noisy and redundant cases (Delany and Cunningham 2004a). Their application uses many features of the *Fionn* framework including the k-NN classifiers and evaluation framework.

Medical Decision Support Doyle and Cunningham have developed a decision support application that was used in the medical domain. This application was designed to support a doctor in making the decision to admit or discharge a child with bronchiolitis to hospital for observation. The system makes a prediction and backs it up with a compelling explanation based on *a fortiori* arguments (Doyle et al. 2004). Using the *Fionn* framework, they were able to concentrate on developing explanations while reusing existing feature selection, noise reduction and evaluation components.

A Generic Case-based Explanation System As described in this thesis.

Part of the classifier suite in the *Fionn* core is an efficient implementation of a *k-NN* retrieval algorithm and a Logistic Regression classifier. These two *Fionn* components provide key aspects of the functionality of the Explanation Framework. In the next two Sections we will describe each of these components in turn.

5.2.1 Case Retrieval Nets

The standard k-NN algorithm calculates similarity on a case-by-case basis. This approach is quite inefficient in domains where there is feature-value redundancy and/or missing features in cases, e.g. in the spam filtering domain (Cunningham, Nowlan, Delany and Haahr 2003, Delany and Cunningham 2004a). A Case Retrieval Net (CRN) is a structured net constructed from the case base to retrieve similar cases to a presented problem-case

(Lenz et al. 1998). CRNs are also able to deal with ambiguous terms, can handle partial cases and are reasonably scalable. They are made up of a number of components:

Case Nodes These represent the cases in the case-base.

Information Entities (IEs) These represent feature-value pairs within cases.

Relevance Arcs These link case nodes with the IEs that represent them. They have weights relating to the level of importance between the connected IE and the case (i.e. the weight of that IE's feature).

Similarity Arcs These interconnect IEs that refer to the same features. These have weights relating to the similarity between the values of connected IEs. The weight of a similarity arc is analogous to the local similarity between the feature-values of the two IEs they connect.

The idea behind the CRN architecture is that if a target case is connected to the net via a set of relevance arcs, and activated, this activation will spread across the net. As activation spreads along arcs in the net, it is attenuated by their weights. Each of the other case nodes will accumulate an activation score relating to its similarity to the target case. The case nodes with the highest activation represent the most similar cases to the target case.

Figure 5.2 illustrates the structure of a case retrieval net using data from the travel case base (Lenz 1993). In this figure each case node is associated with a set of information entities using relevance arcs. We can see that many offers shares information entities. For example, *Offer 20219* and *Offer 500122* have a number of features in common and this is reflected arcs linking each of these offers to information entities such as *Crete*, *Matala* and *swimming*. The advantages of CRNs are that local similarity calculations are only performed once and that missing feature values are ignored. By caching the local similarities between individual cases and reducing feature-value redundancies speed ups in operation are achieved.

5.2.2 *Fionn*'s *k-NN* Implementation

Fionn's *k-NN* implementation uses a combination of case-by-case similarity calculations and CRN optimisations. There is a significant cost associated with initialising a CRN. As such there is a trade-off between time spent initialising the net and faster retrieval times.

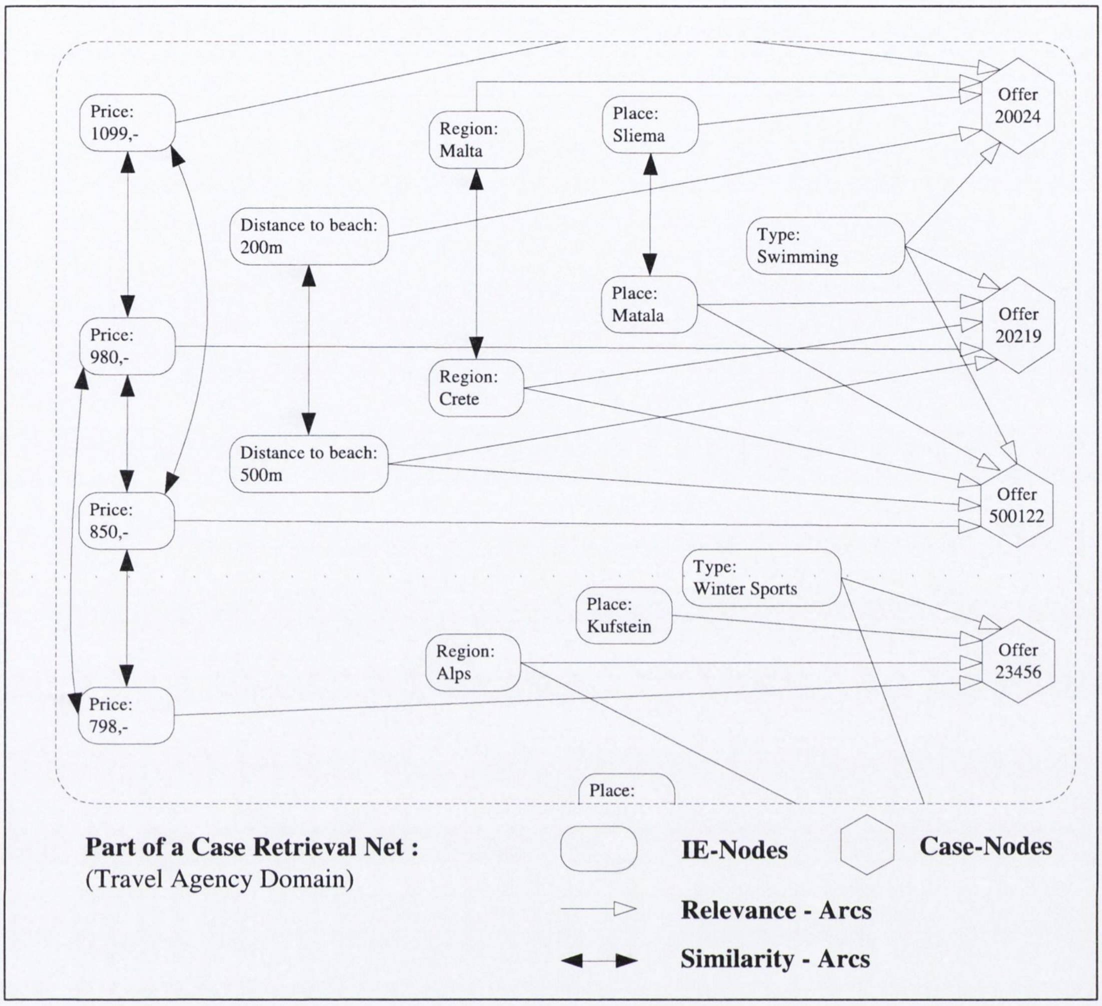


Figure 5.2: The case retrieval net structure (from Lenz and Burkhard 1996)

The CRN-derived speed-ups are counter-productive if there is little or no redundancy in feature values in the case-base. Therefore the speed-ups should not be used unless there are features in the case-base which will benefit from the net structure.

Fionn needs to determine which features will benefit from inclusion in the CRN structure. Features that are not included in the CRN structure will have their local similarities calculated on a case-by-case basis. Our *k-NN* implementation has a *net-initiator* function which assesses the features in a domain in order to determine which ones will benefit from inclusion. It reads the case structure document and automatically leaves out `double` and `string` feature types as they are unlikely to have much feature-value redundancy. `symbol` and `boolean` types will most likely benefit and so they are flagged for inclusion. The `integer` feature type may also be flagged depending on the estimated feature-value redundancy — if the range of an `integer` feature is less than half the size of the case-base it is flagged for inclusion.

Fionn then loads the cases one-by-one into the k-NN structure and adds IEs to the net structure as needed. On presentation of a target case, activation is spread to the case nodes through the CRN. The relevance and similarity arcs in the CRN structure are kept in place between retrievals, therefore subsequent retrievals should be quicker. This is because local similarities are effectively cached in the similarity arcs. Next, *Fionn* calculates the similarity contributions of the non-included features. It goes through these features and calculates the local similarities directly between the target case and each case in the case-memory, adding the similarity values to that case's global similarity. On subsequent retrievals, the local similarity values of non-included features will have to be recalculated.

5.2.3 *Fionn*'s Logistic Regression Implementation

Fionn contains a Logistic Regression classifier that is in keping with its `classifier` interface. A Logistic regression model is defined by the set of parameters that form the logit function (Equation 4.1.1). Building a logistic regression model involves fitting these parameters to the data and involves a directed search through the parameter space. *Fionn*'s implementation of the logistic regression model does this using a globally convergent Newton's Method adapted from numerical recipes in C (le Cessie and van Houwelingen 1992). This alogorithm had originally been ported into Java for use in Weka's machine learning framework [5] but we then adapted and modified the Java version of the algorithm exstensively so that it fitted into the *Fionn* Framework.

Once the model has been built the logsitic regression system is set to act as classifier. As a classifier the logistic regression model will just return a simple class label, howevever we are more interested in the probabilistic qualities of the model. In the next Section we outline how the *Fionn* Framework and the two classifiers we have described are used in the Explanation Framework.

[5] The Weka homepage `http://www.cs.waikato.ac.nz/ml/weka/`

5.3 The Explanation Framework in *Fionn*

We will now briefly outline how the functionality of the Explanation Framework we described in the previous chapter is implemented using *Fionn*. The code which performs the entire explanation process is the `explanationBuilder` object. As can be seen in Figure 5.3 this object contains within it the case-base used to generate explanations and utilises a number of other objects which encapsulate the functionality of various aspects of the Framework that we have described. We will now discuss each of these objects and the funcationality that they provide in turn;

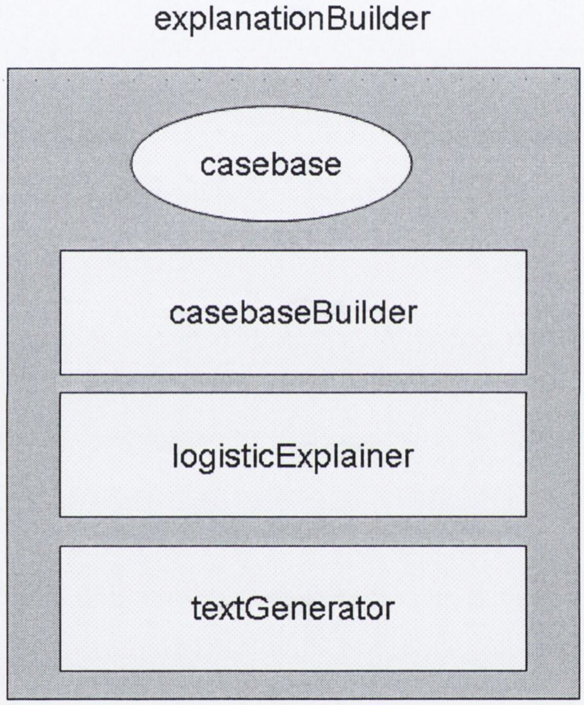


Figure 5.3: The `explanationBuilder` object

`casebaseBuilder:` this carries out the function of producing a local case-base as defined in Section 4.1.2. It is built using the case-base that is to be used to generate explanations and an `integer` defining the number of cases of each class that must be included in each local case-base. To generate a local case-base the `buildCasebase` method is called and a `case` representing the Query Case is passed in. *Fionn*'s implementation of the *k-NN* algorithm is then used to order the list of cases stored in the `casebaseBuilder`'s case-base based on similarity to the Query Point passed in.

A new empty `casebase` is created and is filled with references to `cases` based on the local case-base algorithm we have defined (Section 4.1.2). Once filled, the local case-base that has been created is then returned.

`logisticExplainer`: provides the means through which explanation cases can be selected and feature differences between cases described. The `logisticExplainer` object is intended to be built on local case-bases produced by the `casebaseBuilder`. The `logsiticExplainer` object extends *Fionn*'s logistic regression classifier and so contains all of its functionality but adds two important methods;

`getExplanationCases`: this method takes in a `string` indicating the class of the Query Case for which a explanation case is being selected. The `casebase` is then searched by the `logisticExplainer` for an Explanation Case. It does this using the probabilistic qualties of the underlying logistic regression model as as described in Section 4.1.3. The best case is selected and returned.

`getOddsRatio`: This method provides the means through which the effects of differences can be quantified. As described in Section 4.1.4, coefficients of logistic regression model can be used to calculate the odds ratio of different feature-values. The `getExplanationCases` method takes in two cases, the Query Case and the Explanation Case, and returns an array of `double` objects containing the odds ratio for each feature difference between the Explanation Case and Query Case. The method uses the underlying coefficients representing the Logistic Regression Model and Equations 4.4 and 4.5 from Section 4.1.1 to calculate the effects if a value from the Query Case was substituted into the Explanation Case. In this way the odds ratio is calculated from each feature and returned in the array of `doubles`.

The extra functionality contained within the `logisticExplainer` object allows the underlying logistic regression model to be used in the explanation process in an easy accessible way.

`textGenerator`: handles the generation of discursive texts given the information extracted from logistic regression model. Importantly, it itself also takes in two objects in its constructor; a `featureDifferenceMapper` object and a `featureMapper` object. These objects are used to generate the correct term to describe the differences in feature-values and to generate more user friendly names for the feature names. By default the `featureDifferenceMapper` objects just returns the `string` values *bigger* or *smaller* values and the `featureMapper`, the normal feature names. However, these objects can be extended. We developed an extended version of this object for work on the BAC case-base so that more natural and user friendly language relevant to the domain can be used in the discursive text. Using these objects, the `dialogGenerator` can generate text specific to certain domains or else the defaults can be used.

The `textGenerator:` conatins one important method, `getDiscursiveText`. This method takes in the Query Case, a list of explanation cases and a reference to a `logisticExplainer` object. The list of explanation cases either contains one or two cases. In cases where the system has low confidence in a prediction a extra case, the counter example is also supplied. The `logisticExplainer` is used to get the odds ratios for each feature difference that there may be in the explanation cases. Using these a discursive text is generated and returned in the form of a `string`

`explanation:` is a simple wrapper object which contains the generated explanation text as well as the explanation case(s). This is is the object returned by `explanationBuilder` each time it is called to provide an explanation.

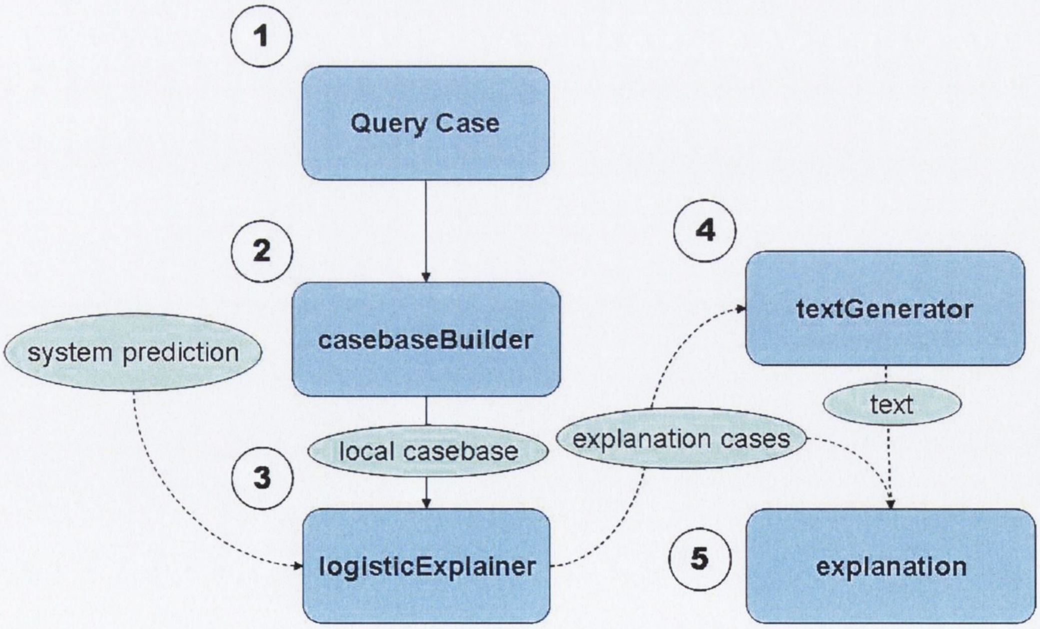


Figure 5.4: The flow of execution

The explanationBuilder uses each of these objects to produce the case-based explanations. The `getExplanation` object is called with a Query Case being passed in and each of the objects that we have described is then used to generate the explanation. The flow of excecution undertaken by the `explanationBuilder` can be broken down into a number of distinct stages as can be seen in Figure 5.4. We will now describe each of these stages;

1. First a case is passed into the underlying prediction system and a classification is made. This case and the recommendation assigned to it are then used to seed the explanation process.

2. The `buildCasebase` method of the `casebaseBuilder` object is then called with the Query Case being passed in and a local case-base is produced.

3. The local case-base is then used to build a `logisticExplainer` object. A method called `getExplanationCases` of this object is then used to find the cases to use as the basis of a case-based explanation. These cases are then passed to the `textGenerator` along with a reference to the `logisticExplainer` object itself.

4. The `textGenerator` uses its reference to the `logisticExplainer` to find the effects of any feature differences that exist between the Query Case and Explanation Case(s) using the `getOddsRatio` method. It then uses the `featureDifferenceMapper` and `featureMapper` objects to generate discursive text describing the effects of the differences

5. Finally the explanation case(s) are combined with the generated text and encapsulated in a `explanation` object.

This process is carried out each time a recommendation is requested. It is important to note that in this implementation no assumption is made about the underlying classifier for which explanations are being provided. The implementation allows for CBR explanations to be provided for any of the classifiers in the *Fionn* suite.

5.4 Summary

We have describe *Fionn* the machine learning Framework we have developed along with how it was used to implement our Explanation Framework. In the next Chapter we describe how we evaluated our Explanation Framework

Chapter 6

Evaluation

The concept of an explanation is intuitively understood by most people, we use it effortlessly throughout our everyday lives. However the vast array of ways and contexts in which explanations are used means that the concept of an explanation is an extremely difficult one to concretely define. What might be considered a valid and good explanation in one context might have no meaning in another. The slippery nature of the concept of an explanation of course impacts on the way in which explanations can be evaluated. There is no one fixed criterion on which all explanations can be judged. Explanations can only really be evaluated in terms of how effectively they meet the objectives that necessitated their creation.

However within the machine learning community, explanations are usually used within a restricted range of contexts and with clear objectives. As highlighted in Section 3.4.1, Sormo et al. defined five goals that explanations in CBR might serve:

- Transparency
- Justification
- Relevance
- Conceptualization
- Learning

The primary role that the framework for explanations we have developed serves is in justification, we want to reassure the user of the system's recommendation and instil confidence in them in that system. Although, as we have explained in Section 1, CBR explanations have great potential to deliver on this issue, presenting the nearest case in its rawest form may not be enough. In Chapters 4 and 5 we discussed how we developed our approach to providing more convincing explanations. We begin this Chapter with a discussion of the merits of our approach for tackling issues in CBR explanations. Following this we describe a User Evaluation of our framework designed to establish whether our approach does indeed improve user confidence and we discuss our results.

Although the primary objective of our Explanation Framework was to provide explanations that improve user confidence, the design of the Framework also has potential in terms of providing assessments of confidence and as a recommendation strategy. Using the probabilistic qualities of the Local Logistic Regression model it is possible to produce a probability of the Query Case belonging to one class or another. In some domains this might prove a more powerful and effective recommendation strategy. In Section 6.3 we evaluate our Local Logistic Regression model as a recommendation strategy in comparison with *k-NN* and global logistic models. In Section 6.4 we evaluate the effectiveness of our systems ability to provide confidence measures. Finally we end this chapter with some concluding remarks in Section 6.5.

6.1 The Merits of Our Approach

One of the principle shortcomings of the traditional knowledge-light approach to explanations is its dependence on the users ability to appreciate and understand the similarity between the query and the explanation cases (Sormo and Cassens 2004, McSherry 2003, Nugent and Cunningham 2005). This is a key concern when the goal of the explanation is to generate user confidence in the CBR system. We developed an approach to tackling this problem which utilises locally built models to extract local information and build convincing explanations. There are a number of advantages to the approach we have taken and we will talk about each of these in turn.

6.1.1 Ability to Deal with Complex or Incomplete Data

One of the great strengths of Case-based Reasoning is the ability to deal with weak theory problems and to learn incrementally. Many weak theory domains are non-linear in nature and are best suited to being described on an instance by instance basis by local models. Previous attempts have used global models to provide users with supplemental information

(McSherry 2003). However, such models may be unsuitable in some complex weak-theory domains and may produce unsatisfactory explanations that don't truly reflect the case-base (Nugent and Cunningham 2005). Our approach sidesteps this issue by building an explanation around the point of interest and so capturing the interactions of the feature-values in that area of the feature space.

Our approach also supports incremental learning as it applies a lazy approach to building explanations. Information is only gathered as the explanation is needed and so always uses the most up-to-date information to generate its explanations. This approach reflects the lazy and incremental learning qualities of CBR which are considered some of the methodology's strengths (Aha 1997). This means that the explanations generated reflect more truly the knowledge captured within the case base. However it is worth noting that if the case-base lacks coverage in certain areas of the feature space then this can be reflected in the explanations produced and they may seem counter intuitive. For instance consider example 6.1.

In this example the case-base in that area of the feature space doesn't adequately represent the problem. In this area of the case-base the feature `duration` is heavily correlated with `units` and so a larger `duration` value is seen as evidence in favour of being over the limit when the reality is the opposite. In some cases this may alert expert users to system failures and deficiencies in the case-base. However, it may well either cause confusion or go unnoticed amongst non-expert users.

6.1.2 Little Expert Knowledge Needed:

One of the great strengths of our approach is from the Knowledge Discovery perspective (Nugent, Doyle and Cunningham to appear 2005). We are able to extract information about the interactions of feature-values on the recommendation task without any prior knowledge of the domain. This means that our Explanation Framework can easily be used in many different domains and without lengthy consultations with experts from those areas. Table 6.2 shows an example from the BAC domain. It is clear that the feature information is in line with our intuitive understanding of the problem. This is all achieved without the intervention of domain experts. However our Framework isnt limited to purely producing explanations in the BAC domain.

We can quickly generate explanations for different domains without any prior knowledge as can be seen in Table 6.3. This is an example that was built on the UCI Credit-card

Table 6.1: Incomplete Case Base

	Query Case	**Explanation Case**
Weight (kgs)	88	63
Duration (mins)	90	120
Gender	Male	Male
Meal	Full	Full
Amount (Units)	5.2	5.2
BAC		Under

The prediction for the individual in the Quey Case is: **Under the limit**

The confidence that this prediction is correct is: **high**

Discursive Text:

In support of this prediction we have the person presented by the Explanation Case who was also Under the limit. `Weight` being heavier and `Duration` being shorter have the effect of making the Query individual more likely to be Under the limit than the Explanation individual.

case-base. The problem involves using information about an individual's status to assess whether they should be given a loan or not. In the BAC domain we used a small amount of linguistic knowledge to help describe the feature differences in a more natural way. For example, instead of describing differences in weight as being larger or smaller the text templates used the terms lighter and heavier. However, in this example we have inserted no extra knowledge of any sort but are still left with an interpretable and useful explanation.

We have also been able to provide explanations which use *a fortiori* arguments without consulting a domain expert and constructing explanation-specific similarity measures. Using the probabilistic qualities of the Logistic Regression Model we can find the most marginal cases without knowledge as to what the effects of different features are on the recommendation.

6.1.3 Confidence Measures

Researchers have recently highlighted the dangers of seeming to be confident in a recommendation when it is false (Sormo and Cassens 2004, Cheetham 2000). This can seriously damage the long term confidence of users in a system. Explanations can have an impor-

Table 6.2: An Example Containing Discovered Domain Knowledge

	Query Case	**Explanation Case**
Weight (kgs)	60	73
Duration (mins)	60	140
Gender	Female	Male
Meal	Full	Lunch
Amount (Units)	2.6	5.7
BAC		Under

The prediction for the individual in the Quey Case is: **Under the limit**

The confidence that this prediction is correct is: **high**

Discursive Text:

In support of this prediction we have the person presented by the Explanation Case who was also Under the limit. `Meal` being Full and `Amount` being smaller have the effect of making the Query individual more likely to be Under the limit than the Explanation individual. Although confidence in the prediction is high it is worth noting `Weight` being lighter, `Duration` being shorter and `Gender` being Female have the effect of making the Query individual less likely to be Under the limit than the Explanation individual.

tant role in highlighting system failures and as we saw in Section 6.1.1 our system can highlight deficiencies in the case-base. Our approach also provides us which a meaure of confidence in a given recommendation which can then supply to the end user.

6.2 User Evaluation

Although our Framework addresses the perceived deficiencies in CBR explanations we needed to establish whether the explanations produced by our Framework were in fact effective. To do so we carried out a user evaluation. In designing the user trial there were three principle questions we wished to address;

1. Do people find the explanations understandable and useful?

2. Do the explanations increase users' confidence in the case-based system?

3. Can the explanations alert users to when the system might be in error?

Table 6.3: Credit Card Data Set Example

	Query Case	**Explanation Case**
Checking Status	0-	0-
Duration	6.0	21.0
Credit History	critical/other existing credit	critical/other existing credit
Purpose	radio/tv	new car
Credit Amount	1169.0	571.0
Savings Status	no known savings	100-
Installment Commitment	7+	7+
Personal Status	4.0	4.0
Residence Since	single male	male single
Property Magnitude	real state	real estate
Age	67.0	65.0
Other Payment Plans	none	none
Housing	own	own
Existing Credits	2.0	2.0
Job	skilled	skilled
Number of Dependents	1.0	1.0
Own Telephone	Yes	none
Foreign Worker	Yes	Yes
Grant Loan?		Granted a Loan

The recommendation for the individual in the Quey Case is: **Granted a Loan**

The confidence that this recommendation is correct is: **high**

Discursive Text:

In support of this recommendation we have the person represented by the Explanation Case who was also Granted a Loan. `Duration` being smaller, `Purpose` being radio/tv, `Savings Status` being no known savings and `Own Telephone?` being yes have the effect of making the Query individual more likely to be Granted a Loan than the Explanation individual. Although confidence in the recommendation is high it is worth noting `Credit Amount` being bigger and `Age` being bigger have the effect of making the Query individual less likely to be Granted a Loan than the Explanation individual

The case-base on which the trial was carried out was the blood alcohol case-base as descibed in previous Sections. We built a simple Nearest Neighbour algorithm on the data set and applied our framework to providing explanations of it's recommendations.

In the trial, subjects were given a questionnaire in which they were shown three different forms of explanation;

- **The Framework Explanation:** This is an explantion that includes the selected *a fortiori* explantion case, a discursive text and a measure of confidence as seen in table 4.5.
- **Case-based Explanation:** In this form of explanation the subject is just shown the selected *a fortiori* case as evidence in favour of the recommendation.
- **No Explanation:** The user is just presented with the feature-values of the query and the systems prediction.

The trial subjects were shown four examples of each type of explanation which were selected at random and asked two questions after each example shown;

1. Do you think the prediction is correct?
2. How would you rate this Explanation?

Below each question the trial subject had five options to select from. In question one the options were; No, Maybe No, Don't Know, Maybe Yes and Yes. In question two the options were; Poor; Fair; Okay, Good and Very Good.

To assess the use of explanations in terms of alerting users to when the system might be in error, one of the four examples shown of each explanation type was a mis-classification. Examples of each type of explanation shown to the users and the structure of the questionnaire can be seen in Appendix A. The Framework explanations were structured as outlined in Chapter 4. In the case of the incorrect Framework explanation the confidence in the recommendation was labelled as being *low*. In cases of low confidence we presented users with additional counter examples as outlined in Section 4.2.1 and as can be seen in Appendix A.4.

Twelve people from a number of different backgrounds took part in the evaluation. In the next section we will discuss how we analysised the data and we then move on to the results of our analysis.

6.2.1 Analysis of Data:

The first step in analysing the user responses was to encode them so that they could be interpreted. To do this we translated the reponses into a simple numeric scale. The responses clearly have a natural ordering and we coded them into values between 1 and 5. Scores that indicated positive responses in favour of a explanation scheme were given higher numeric scores such as 5. In Question One, the responses No to Yes were coded as being 1 to 5 respectively. Likewise in Question Two the responses Poor to Very Good were coded as 1 to 5. In the case of recommendations that were incorrect the encoding scheme was reversed with Yes to No being coded 1 to 5 respectively. There was no reversal of the ratings of the responses to Question Two when the system was incorrect. Once we had encoded the users reponses we were then able to analyse the data.

In analysing the data from our user evaluation we would like to establish whether one explanation technique is statistically better or worse than another. Our evaluation was designed so that each survey participant rated examples of each type of explanation. This allows us to consider the average rating a participant gives one explanation scheme linked with the average rating that they give another scheme. This is often referred to as a *paired result* and this is an important characteristic as it allows us to use a *Student's paired t-test* to compare our different explanation schemes. The *Student's paired t-test* is a parametric test that can be used to determine whether the differences scores for one scheme and another are significant. It is a test designed for use on data with small sample sizes and assumes that underlying data is normally distributed. The ratings scores of each explanation scheme follow normal distributions which is evident in Figures 6.4 and 6.3. Our small sample size and the distribution of data mean that the *Student's paired t-test* is ideally suited for analysing the results of the user evaluation.

The *Student's paired t-test* can be calculated in the following way. Let x_i and y_i be the average user rating for `user i` using on explanation scheme A and B respectively. We use the following equation to calculate a value for the paired t:

$$t = \frac{(\overline{x} - \overline{y})\sqrt{n}}{s}$$

where $\overline{x}$ is the overall average rating of scheme A and $\overline{y}$ is the average rating of strategy B, n is the number of users (12) and s is the standard deviation of the differences in ratings between scheme A and scheme B, i.e.

$$s = \sqrt{\frac{\sum_{i=1}^{n} (x_i - y_i)^2}{n-1}}$$

For the paired t-test the null hypothesis supposes that the difference between the two samples is 0. If there is enough doubt in the null hypothesis we can reject it and say that one strategy is better than the other. In practice the *t-value* returned allows us to determine with a level of certainity whether the difference that exists between one scheme and another is significant.

6.2.2 User Trial Results:

Question One: ***Do you think this prediction is correct?***

The first question was designed to test how each of the explanation schemes affected users' confidence in the system's recommendations. We can see the average ratings for each scheme when the systems recommendations were correct in Figure 6.1. It is clear that the explanations given by the framework instil greater confidence in the system than either of the other two schemes. The survey participants answered *Yes* 88% with just four answers being anything other than yes. Three people answered *Maybe Yes*, one *Don't Know* and there were no negative answers. This result is statistically significant as can be seen in Table 6.4. When the system is correct the users were far more confident in recommendations that were accompanied with the full framework explanation than when simply supplied with an explanation case.

Table 6.4: Summary of statistics for the difference in ratings in Question One for the Framework and Case-based explanations when the system is correct.

Statistic	Value
Number Of Samples (n)	12
Degrees Of Freedom (d)	11
Sample Mean of Framework Ratings ($\overline{x}$)	4.86
Sample Mean of Case-based Ratings ($\overline{y}$)	3.72
Student's t (t)	4.16
99 Percentile Student's t Distribution ($t_{.99}$)	3.106

We also found case-based explanations to be an improvement on providing no explanation at all although the improvement is more modest than that of the Framework explanation (see Table 6.5). The average rating for the case-base explanation is 3.7 while

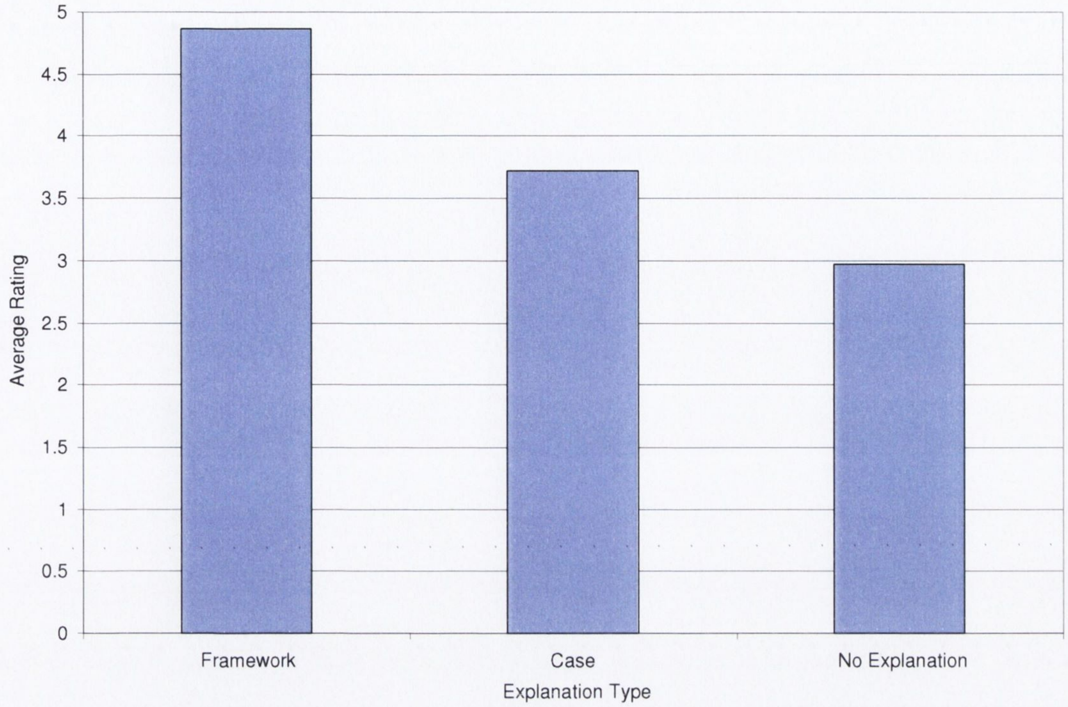


Figure 6.1: The ratings in Question One for each explanation scheme when the System was correct

supplying no explanation has an average rating score of 2.97. These ratings translate into responses somewhere between *Don't Know* and*Maybe Yes* for the case-based approach and of *Don't know* when no explanation is supplied. Both these response patterns reflect that there is an element of doubt in users of the system when supplied with either no explanation or a case-based explanation. Although the average rating for the case-base approach is higher than supplying no explanation this difference is significant at a lower level. Simply supplying an explanation case instils more confidence than not supplying any explanation however a large element of doubt still persists. This further reinforces the value of the framework explanation and justifies the concerns expressed by researchers about simple case-based explanations (Section 3.4.1).

We also examined the user's responses when the system had made an incorrect classification and the results can be seen in Figure 6.2. Each schemes' average rating drops down to around 3, the *Don't Know* mark. In the case of the Framework explanations this is a significant drop as can be seen in Table 6.6. When the system is incorrect users seem to be unsure of the system. In fact there is no significant difference in the performance of

Table 6.5: Summary of statistics for the difference in rating in Question One of Case-based explanation and no explanation when the System is correct.

Statistic	Value
Sample Mean of Case-based Ratings ($\overline{x}$)	3.72
Sample Mean of Ratings for no explanation ($\overline{y}$)	2.97
Student's t (t)	2.58
99 Percentile Student's t Distribution ($t_{.99}$)	3.106
95 Percentile Student's t Distribution ($t_{.95}$)	2.201

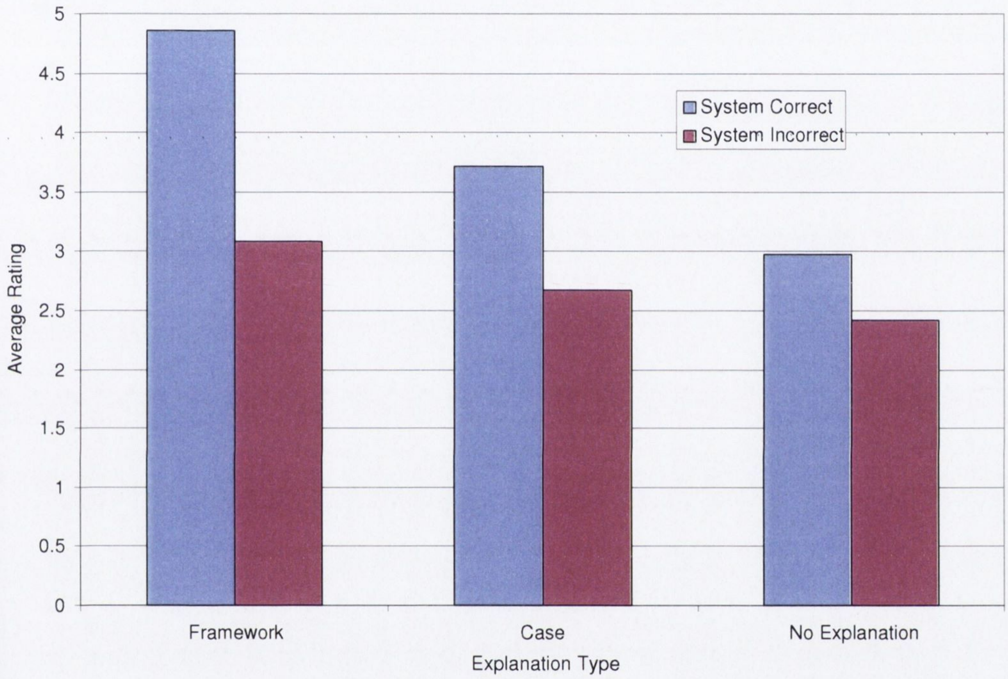


Figure 6.2: The ratings for each explanation scheme in Question One when both System was correct and incorrect.

either algorithm when the system is incorrect (Table 6.7). This may seem to be a positive result, as at least users don't seem to believe that the system is definitely right. However, ideally we would like users to realise that the system is incorrect as a *Don't Know* reponses reflect uncertainity and confusion.

The level of confusion that exists can clearly be seen if we look at the distribution of responses behind the the average ratings. In Figure 6.3 we can see the distribution of responses for each explanation scheme when the system is correct. We can see that the

Table 6.6: Summary Statistics for the ratings for Framework explanations when system correct and when the system is incorrect.

Statistic	Value
Sample Mean when System was correct ($\overline{x}$)	4.96
Sample Mean when System was incorrect ($\overline{y}$)	3.08
Student's t (t)	5.93
99 Percentile Student's t Distribution ($t_{.99}$)	3.106

Table 6.7: Summary of statistics for Question One of Framework and Case-based explanations when the System was incorrect.

Statistic	Value
Sample Mean of Framework Ratings ($\overline{x}$)	3.08
Sample Mean of Case-based Ratings ($\overline{y}$)	2.66
Student's t (t)	0.86
99 Percentile Student's t Distribution ($t_{.99}$)	3.106
80 Percentile Student's t Distribution ($t_{.80}$)	1.36

average rating for the framework explanation reflects the mean of a very tightly pointed distribution centered on *Yes* response. This reflects certainty of the users and the confidence in the system that has been instilled in them. Conversely, the distibutions of the other two explanation schemes are far flatter reflecting the uncertainty and lack of confidence that exists when a case-based or no explanation is used.

The graph of frequencies of responses when the system is incorrect reveals a very different user response pattern (Figure 6.4). Although no one responded *Yes* in the case of the explanations produced by the framework there is far less certainty in the users' responses. When the system was incorrect the distribution for each of the schemes is quite flat and all centred around the *Don't Know*; clearly there is a high degree of variance in the responses. It is encouraging that no users thought that the system's prediction was definitely correct in the case of the framework explanation and it is clear that they heeded the warning about low confidence. However, it is also clear that users weren't sure if the system was correct or incorrect, a level of doubt remained that was indistinguishable from that experienced under the other two schemes.

It is clear that the explanations that are produced by the framework are effective at instilling confidence in users and that they represent a considerable improvement on simple case-based explanations. However, when the system is incorrect they proved to be no more

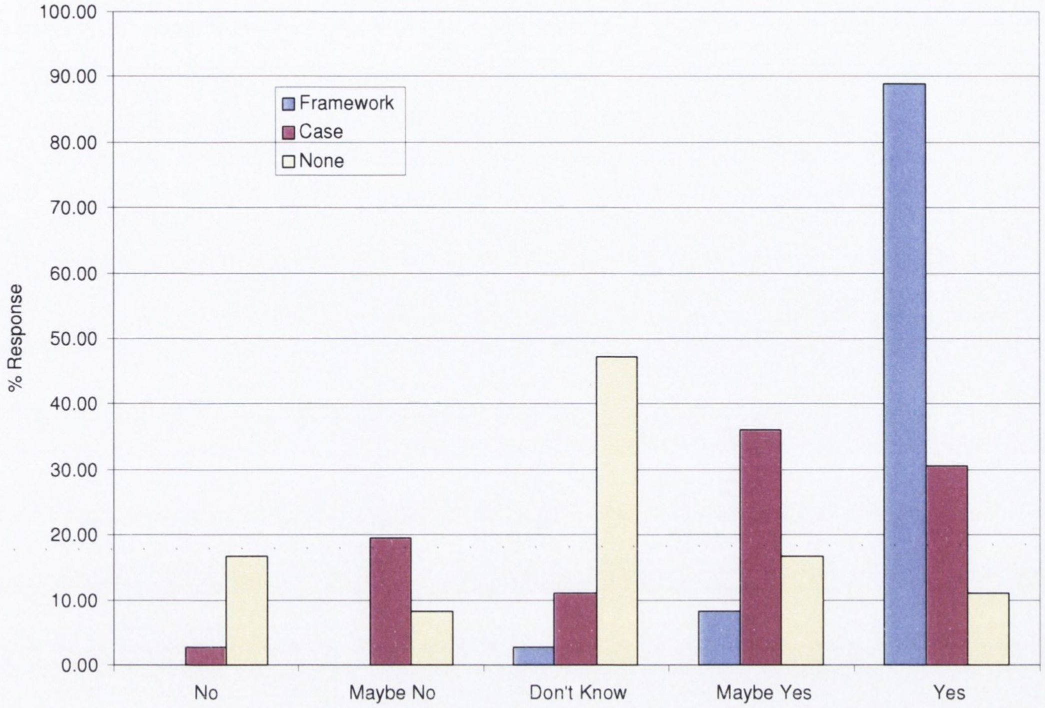


Figure 6.3: The distribution of user responses in Question One when the system predictions were correct

effective than any other explanation scheme in limiting confusion and alerting the user as to the correct prediction.

Question Two: *How would you rate this explanation?*

In question two we were trying to determine how satisfactory people found the explanations. We can see the average rating value given to each explanation scheme in Figure 6.5. Clearly people found the framework explanations to be far more satisfying then the other two schemes and generally the rating level for the framework explanation was quite high with an average rating of 4.2 when the system was correct. This equates to most users rating the explanations as being somewhere between *Good* and *Very Good*. The average ratings for the case-based explanation and no explanation when the system was correct are 2.3 and 1.05 respectively. This places the responses for the case-based approach between *Fair* and *Okay* while the ratings for no explanation are understandably firmly set in the *Poor* categorey. The difference in rating between the framework and case-based explanation is strongly statistically significant indicating the greater user satisfaction in

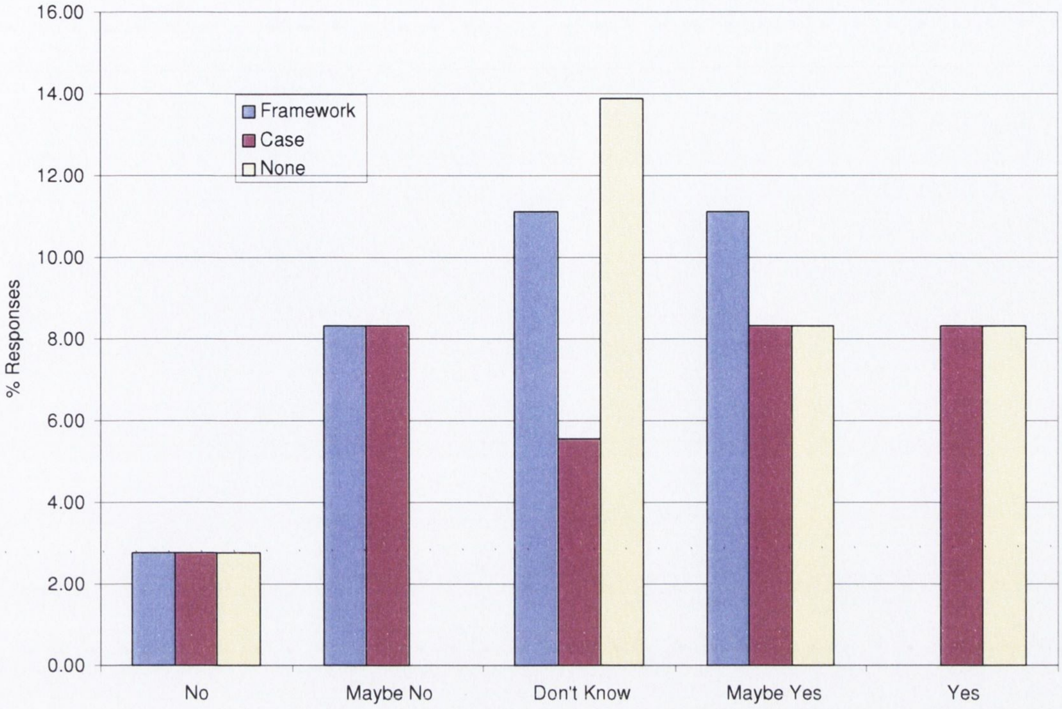


Figure 6.4: The distribution of user responses in Question One when the system predictions were wrong

the Framework Explanation as can be seen in Table 6.8. A Students t-value of 3.106 is required to ensure that there is 99% confidence that there is a significant difference. The Students t-value is twice that needed for 99% confidence so we can say that the results are strongly statistically significant.

Table 6.8: Summary of statistics for the difference in ratings in Question Two for the framework and case-based explanations.

Statistic	Value
Sample Mean of Framework ($\overline{x}$)	4.22
Sample Mean of Case-based ($\overline{y}$)	2.83
Student's t (t)	6.62
99 Percentile Student's t Distribution ($t_{.99}$)	3.106

We also examined the situation when the system made an incorrect recommendation and the ratings for each explanation can be seen in Figure 6.6. There is a drop in the rating for both the framework and case-based explanations while the rating for no explanation has

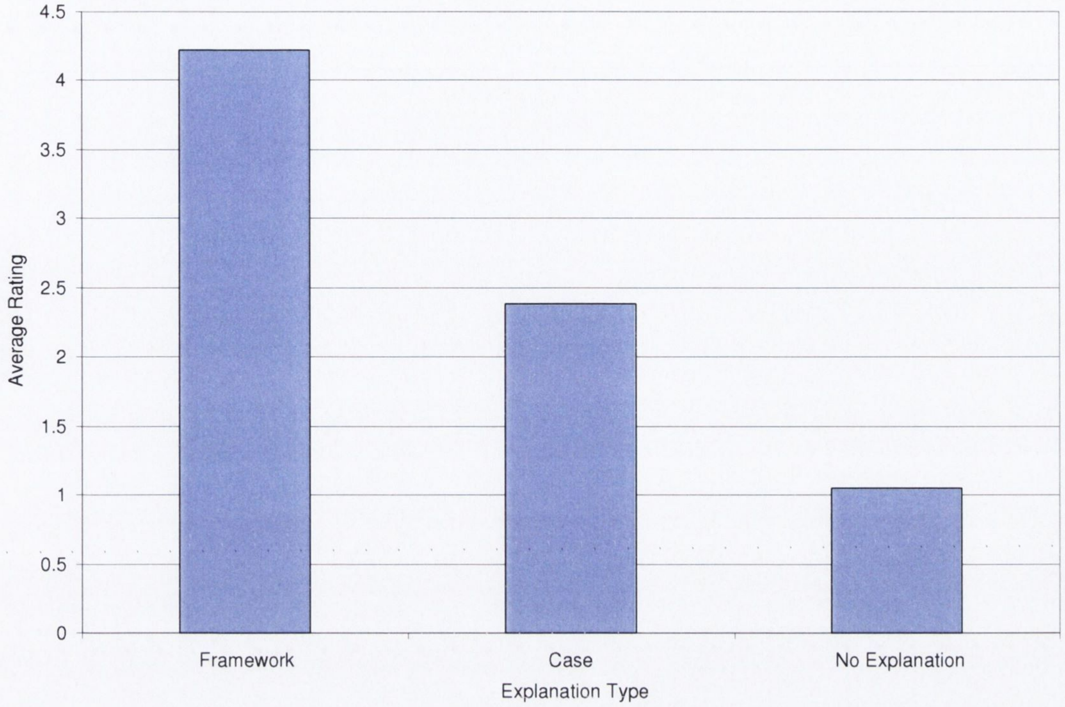


Figure 6.5: The average rating Scores for Question Two of the explanations produced by each different Scheme

naturally stayed the same. In the case of the framework explanation the ratings dropped from being between *Good* and *Very Good* to *Good* and *Okay*. This drop is significant but not strongly so as can be seen in Table 6.9. As we saw in the analysis of the results from question one, the Framework Explanations proved unable in this domain to alert users that the prediction was definitely incorrect. However, user satisfaction in the Framework Explanations has dropped but remained reasonably high. This is despite the increased cognitive load associated with explanations produced when confidence in a prediction is low. As described in Section 4.2.1 users are presented with counter examples when the confidence is low. This leads to longer explanations as can be seen in Table 4.6.

There is also a drop in level of satisfaction in the case-based approach when the system is incorrect. This may reflect the perceived lack of relevance of the explanation case retrieved. The retrieved case in this circumstance is a poor explanation case reflecting the lack of coverage in that point of the case-base. The average rating for the case-based explanation drops from being *Fair* to *Okay* to being *Poor* to *Fair*. However, it is clear that the level of satisfaction in the case-based approach when the selected explanation

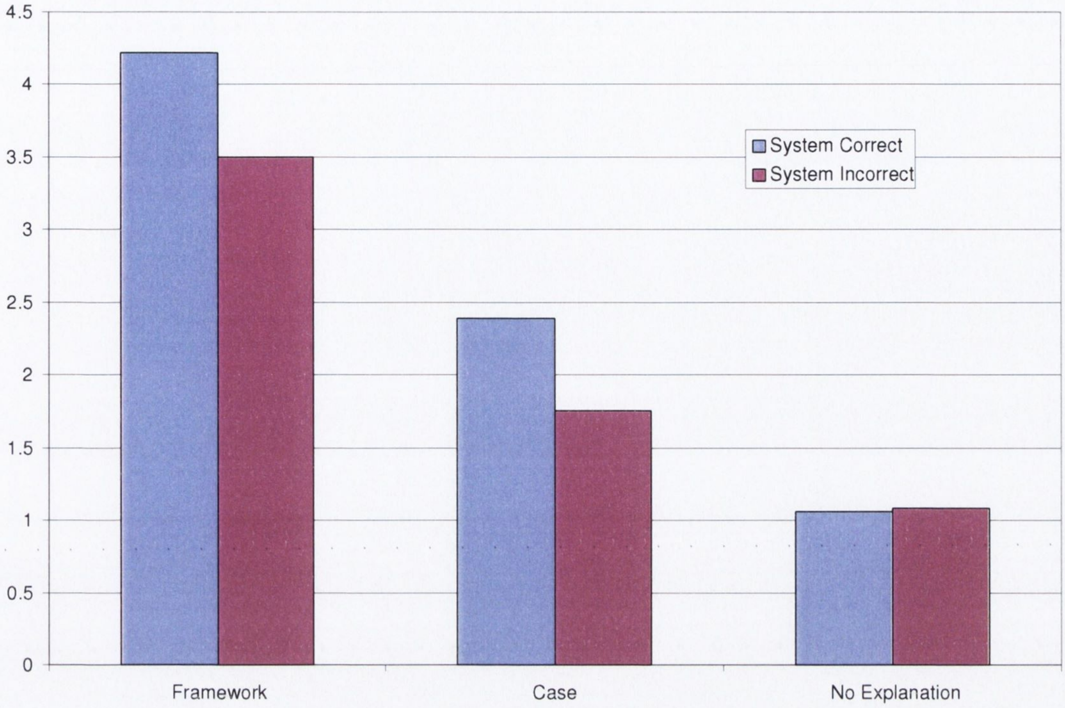


Figure 6.6: The average rating scores for Question Two of the explanations produced by each different Scheme when both the system was correct and incorrect

cases are good and the system is correct is already very poor to begin with. The level of satisfaction in the framework explanations is still greatly superior and significantly so (Table 6.10).

Table 6.9: Summary of statistics of the difference in ratings in Question Two of the Framework explanations when the System was correct and incorrect.

Statistic	Value
Sample Mean when System correct($\overline{x}$)	4.22
Sample Mean when System incorrect($\overline{y}$)	3.5
Student's t (t)	1.96
99 Percentile Student's t Distribution ($t_{.99}$)	3.106
90 Percentile Student's t Distribution ($t_{.90}$)	1.79

It is clear that users show far greater satisfication in the framework explanations in comparison with the the case-based approach even when the system has made an incorrect prediction 6.10. It would appear that satifaction in the framework explanations is reasonably robust even under the extra cognitive load experienced when there is low confidence in the system's prediction. Satisfaction levels in the case-based approach are unacceptably low in both circumstances and this again reinforces the need for the explanatory texts supplied with the framework explanations.

Table 6.10: Summary of statistics of the difference in ratings in Question Two of the Framework and Case-base explanations when the system was incorrect.

Statistic	Value
Sample Mean of Framework ($\overline{x}$)	3.5
Sample Mean of Case-based ($\overline{y}$)	1.75
Student's t (t)	4.47
99 Percentile Student's t Distribution ($t_{.99}$)	3.106

6.2.3 Discusion of Results

The evaluation that we have carried out has demonstrated that there is much value in the explanations produced by the framework. Simple case-based explanations have their short-comings which is evident in the user ratings of such explanations. Both in terms of increasing user confidence and in the level of satisfaction expressed by users, simple case-based explanations performed poorly. The case-based explanations only proved to be marginally effective in increasing user confidence in the system and as can be seen in Figure 6.3 much confusion still remained. Likewise the level of satisfaction expressed by users in the explanations was quite low being at best between *Fair* and *Okay*.

Conversely, the framework explanations proved to extremely effective at instilling confidence as is evident in the strong *yes* response expressed by users when presented by framework explanations (Figure 6.3). User satisfaction ratings of the explanations were also far higher. It is clear that generating discursive texts explaining the effects of feature-value differences greatly improves upon simple case-based explanations.

However, in terms of our goal of alerting users to when the system is making a mistake, the framework explanations only proved a limited success. There was a marked difference in the users' responses to the different explanations produced when system was correct and incorrect. Although encouragingly users heeded the confidence warning and no one responded that the system was definitely correct it is clear that users were none the wiser as to what the correct recommendation should be as is evident in Figure 6.4. Perhaps this reflects their lack of domain expertise as satisfaction levels in the framework explanations still remained high. These results may indicate that it may be extremely difficult to maintain user confidence in a system that makes mistakes.

6.3 Local Logistic Regression as a Classifier

As we acknowledged in Nugent, Cunningham and Doyle (2005) providing explanations can maintain confidence but if the underlying system is inaccurate confidence will be lost regardless of these efforts. A fundamental requirement to maintaining confidence is that the underlying recommendation system is sufficiently accurate. We have used the Local Logistic Regression Model as an aid to providing explanations but it is also possible to use the locally produced model to provide recommendations. Using the probabilistic model of the local feature space it is possible to produce of probability of the Query Case belonging to one class or another. In some domains this might prove a more powerful and effective classification strategy than that normally employed by *k-NN*. Likewise, the localised logistic regression might prove more effective than use a global logistic regression model. The local approach might be able to approximate patterns in the data locally that are lost when the logistic model is applied to the data in its entirety. In this Section we examine the effectiveness of out localised logistic regression model as a classification strategy. We evaluated it on a number of different case-bases and compared the results with those of the *k-NN* and Logistic Regression strategies. In the next Section we will describe the experimental setup before going on to discuss our results.

6.3.1 Experimental Setup

In order to evaluate the three different recommendation schemes we user Leave-one-out cross validation on each scheme for each 13 UCI case-bases. Leave-one-out cross-validation is an evaluation method used to assess the likely real world performance of a system on unseen data. Using leave-one-out cross-validation, all the cases in the case-base except one are used to construct the classification model. The remaining case that has been excluded is then passed into the constructed model and a recommendation for that case is produced. This process is repeated for each case in the case-base leaving us with a classification for each.

Both *k-NN* and Local Logistic Regression models have parameters that must be set. In the case of *k-NN*model it is K; the number of neighbours used to determine the class.

In the localised logistic regression model we must choose the number of each class we want to guarantee will be present in the local case-base. We evaluated the performance of both schemes using different parameter settings in the range from 1 to 15 in both cases. We selected the parameter on which the best performance was recorded and used it as representative of the best performance for that classification scheme. In the next section we present and discuss the results of our evaluation.

6.3.2 Experimental Results

The results of the evaluation carried out can be seen in Table 6.11. For each of case-bases used in the evaluation we have recorded the accuracy of each classifier. In the case of the *k-NN* and Local Logistic Regression schemes the accuracy score is accompanied by the parameter setting used in brackets after the accuracy score. The results show that in all but three cases the Local Logsitic Regression model is as accurate or more accurate than simple *k-NN*. In just four cases the Local approach is the best recommendation strategy. The poor performance of the Local appraoch is extemely surprising when compared to that of the global approach.

Table 6.11: The Accuracies of the Different Classification Schemes

Case-base	Global Logistic Regression	*K-NN*	Local Logistic Regression
DNAp	86.79	85.85 (2)	**89.62** (6)
ionosphere	**89.17**	88.89 (2)	86.89 (9)
e-clinic2	98.66	98.66 (2)	**99.33** (3)
diabetes	**77.73**	73.7 (4)	75.78 (10)
vote	**93.79**	92.64 (2)	92.64 (10)
German Credit	**75.10**	74.7 (10)	69.60 (2)
spam	93.75	89.75 (2)	**95.75** (6)
BAC	83.67	82.65 (2)	**86.73** (3)
e-clinic	**96.32**	86.96 (6)	94.98 (13)
Breast Cancer	70.40	**76.90** (4)	69.68 (6)
Bronchiolitis	66.87	**71.08** (6)	66.87 (11)
Heart	**82.96**	**82.96** (10)	77.41 (11)
Liver	**68.41**	67.54 (2)	68.12 (4)

It is worth nothing that the Local and Global Logistic Regression schemes are equivalent if the local case-base is grown to be so large that it encompasses the entire case-base. This may seem to be moot point but it means the local scheme is guaranteed to be as good as the global scheme. The restricted range of values we tried for defining our local case-base means that the results presented may not reflect the best possible accuracy. To investigate this further we extended the range used on two case-bases; `e-clinic` and `BAC`.

Figures 6.7 and 6.8 depict the change in accuracy of the Local Logistic Regression approach as the number of cases of each class to be included in the local case-base is increased. As we can see in Figure 6.7 the localised approach quickly reaches an optimal value using extremely small local case-bases. As the local case-base size increases the accuracy drops before slowly increasing again until it levels out at 83.67 which is the equivalent to the global model.

However, in the case of the `e-clinic` case-base a very different form of behaviour is observed. The model very slowly reaches an optimal performance and then very slowly drops back to that of the global model. The optimal performance is reached with the local case-base needing to contain at least 56 of each class value. This represents a far larger local case-base than that used in BAC case-base.

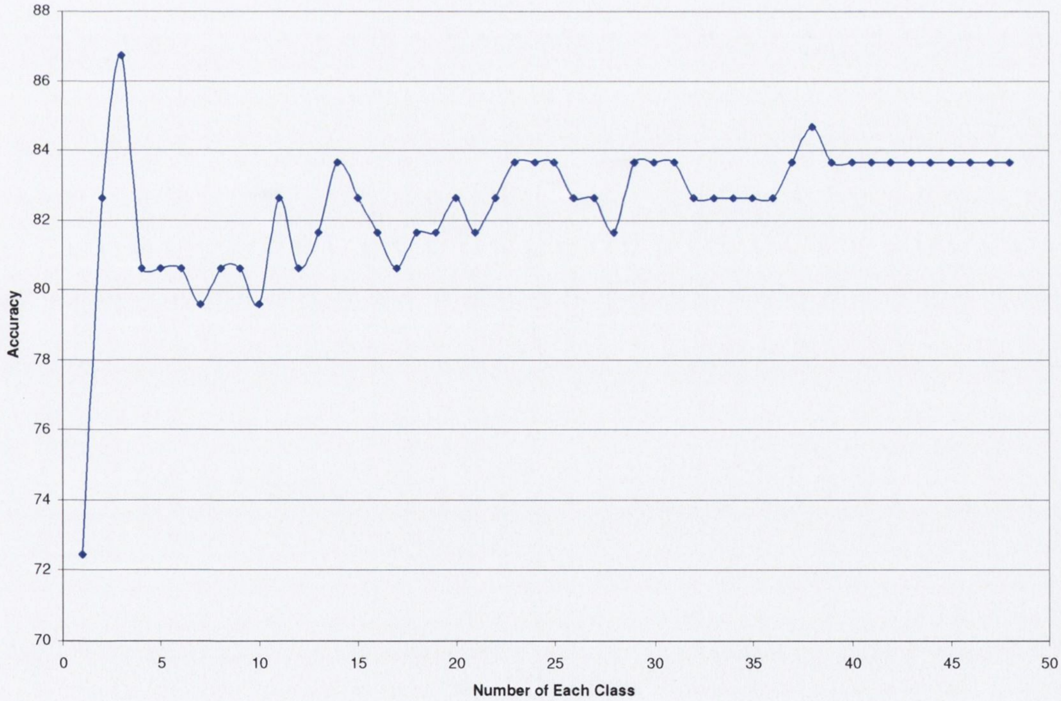


Figure 6.7: The change in accuracy for different parameter settings using Local Logistic Regressionon the `BAC` case-base

It appears that, for the Local Logistic Model, the optimal size of local case-base is very

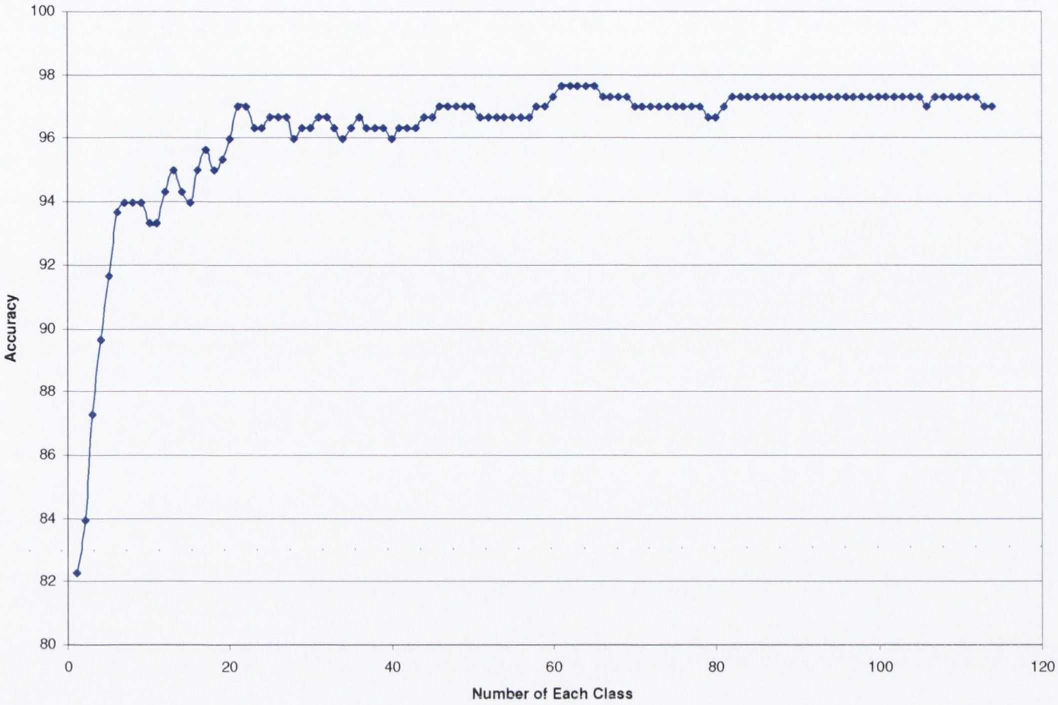


Figure 6.8: The change in accuracy for different parameter settings using Local Logistic Regression on the `e-clinic` case-base

much case-base dependent and that it can be much larger than those we had used in our evaluation. With this in mind we repeated our evaluation this time extending the range of values tested for the Local Logistic Regression Model until the performance of the model was equivalent to that of the globel model. The results of this second evaluation can be seen in Table 6.12. After extending the range of parameter values tested it is clear that Local Logistic Regression model is indeed as strong as the global model and that in many cases it is stronger than *k-NN*. However there is a need for a parameter setting excercise to determine how large the local neighbourhood should be.

6.3.3 Discussion of Results

It is clear that Local Logistic Regression shows some promise as a classification strategy. It performs as well as the other schemes in many of the case-bases we tested and outperformed the other recommendation schemes on some case-bases. For case-bases such as `Liver`, `German Credit`, `BAC` and `Spam` it represents a substantial improvement over the other recommendation schemes with improvements of a least 2% over the rival schemes. In

Table 6.12: The Accuracies for Local Logistic with Extended Parameter Range

Case-base	Global Logistic Regression	*K-NN*	Local Logistic Regression
DNAp	86.79	85.85 (2)	**89.62** (6)
ionosphere	89.17	88.89 (2)	**90.31** (25)
e-clinic2	98.66	98.66 (2)	**99.33** (3)
diabetes	77.73	73.7 (4)	**77.99** (117)
vote	93.79	92.64 (2)	**94.25**(128)
German Credit	75.10	74.7 (10)	**77.1** (109)
spam	93.75	89.75 (2)	**95.75** (6)
BAC	83.67	82.65 (2)	**86.73** (3)
e-clinic	96.32	86.96 (6)	**97.56** (65)
Breast Cancer	70.40	**76.90** (4)	75.45(72)
Bronchiolitis	66.87	**71.08** (6)	69.88 (37)
Heart	82.96	82.96 (10)	**84.44** (109)
Liver	68.41	67.54 (2)	**74.48** (38)

all cases the Local logistic regression approach outperforms the global approach but in many case-bases such `diabetes` and `vote` the improvements are quite modest and not particularly significant. In two case-bases, `Breast Cancer` and `Bronchiolitis`, the *k-NN* outperformed the Local Logistic Regression approach. Although the Local Logistic model is considerably better than the global approach on these case-bases the *k-NN* is the overall best recommendation strategy being at least 1% better than the Local Logistic model in both cases.

In the course of our evaluation it also became apparent that the Local Logistic Regression Model's performance was very sensitive to the size of local case-base used. The sensitivity encountered means that, although the performance of the model is very strong, parameter selection must be performed carefully.

6.4 Confidence Measure Evaluation

In this section we examine the potential of the framework to produce an estimate of confidence. As we stated in Section 4.1.5 designing confidence mechanisms requires choosing between many different possible parameter settings so as to maximise performance on a domain dependent set of criteria. In the absence of specific domain problems we have examined the performance of our framework on a number of different case-bases using criteria we ourselves defined. We used the criteria to investigate the flexibility and potential of the confidence measures that we can produce. In the next section we will describe the experimental setup that we used to perform this investigation before going on to discuss the results

6.4.1 Experimental Setup

Previously in Section 4.1.5 we defined two criteria that could be used to assess the performance of our confidence measures. These are the Confident Correct Rate (CCR) as defined in Equation 6.1 and the Not Confident Correct Rate ($NCCR$) as defined in Equation 6.2.

$$CCR = \frac{CC}{CC + CI} \tag{6.1}$$

$$NCCR = \frac{NCC}{NCC + NCI} \tag{6.2}$$

Where CC is the number of times the measure is confident and the system is correct and CI is the number of times the measure is confident and the system is incorrect. Likewise NCC is the number of times the measure is not confident and the system is correct and NCI is the number of times the system is not confident and right to be so.

Our mechanism for assessing confidence requires two parameters that need to be set; K, the number cases of each class type value that is required in order to stop the local case-base building process and the threshold value. As we have discussed, changing these parameters can affect performance and they must be choosen based on domain requirements. To investigate the performance of our confidence measure we decided to fix the value of K for each case-base and investigate the performance of the mechanism at different threshold levels. We chose that K value for each case-base on which the maximum performance in balancing the two criteria was achieved. We performed leave-one-out cross-validations on each case-base recording the probablilty predicted by our confidence measure for each case. This allowed us to investigate the performance of the mechanism at different threshold values.

Leave-one-out cross-validation is an evaluation method used to assess the likely real world performance of a system on unseen data. Using leave-one-out cross-validation, all the cases in the case-base except one are used to construct our *k-NN* model and confidence mechanism too. The remaining case that has been excluded is then passed into the contructed models and their performance on this unseen case is recorded. This process is repeated for each case in the case-base leaving us with probability value for each.

We used these criteria to investigate the performance of our confidence mechanism in two different scenarios;

1. When no particular emphasis was put on one criterion over another
2. When we wished to eliminate all assessments of high confidence on incorrect recommendations.

In first situation we found the threshold that best balanced the two criteria based on Equation 4.11. In the second case situation we found the threshold value which minimised the number of incorrect predictions given confident ratings. In the next Section we will discuss the results of our investigation.

6.4.2 Results of Investigations

The results for each case-base under the first scenario can be seen in Tables 6.13 and 6.14. In Table 6.13 there is a column outlining each combination of confidence prediction and whether the prediction was correct or not. The left most column is the accuracy of the *k-NN* model on each case-base. Ideally, in this situation we would like to minimise the values in the *Confident Incorrect* and *Not Confident Correct* columns while maximising the performance in the other two columns, *Confident Correct* and *Not Confident Incorrect*. These two desires are captured by our criteria shown in equations 6.1 and 6.2 and these can be seen in Table 6.14.

The Confident Correct Rate represents the percentage of times that we make a prediction of confidence and are right to do so. This rate is reasonably high for most case-bases; being in the late 90's for 7 of the case-bases and in the 75 to 90 range for the remaining 6 case-bases. However, the Confident Correct Rate cannot be looked at in isolation as, although we might be predicting confidence accurately, we would also like to minimise the number of correct predictions which get a not *confident rating.* We would like to achieve high *Confident Correct Rates* while also attaining low *Not Confident Correct Rates.*

This is achieved with some degree of success on four of the case-bases. On the `DNAp`, `e-clinic2`, `spam` and `e-clinic` case-bases *Confident Correct Rates* in the high 90's are combined with *Not Confident Correct Rates* below 0.3. However, it it is clear that the confidence mechanism isn't always sucessful. Notably for the `Diabetes`, `Breast-Cancer`,

Table 6.13: Confidence Results for Scenario One

Case-base	k-NN Accuracy	Confident Correct	Confident Incorrect	Not Confident Correct	Not Confident Incorrect
DNAp	85.85	88	7	3	8
ionosphere	88.89	279	16	33	23
e-clinic2	98.66	295	1	0	3
diabetes	73.70	79	27	487	175
vote	92.64	381	13	22	19
German Credit	74.70	485	103	262	150
spam	89.75	357	13	2	28
BAC	82.65	73	5	8	12
e-clinic	86.96	250	5	10	34
Breast Cancer	76.90	98	29	115	35
Bronchiolitis	71.08	30	4	88	44
Liver	67.54	180	57	53	55
Heart	82.96	192	29	32	17

`Bronchiolitis`, `Liver` and `Heart` case-bases the performance of the mechanism isn't great with lower *Confident Correct Rates* being combined with high *Not Confident Correct Rates*.

We can look at the trade-off by examining the results of Equation 6.3 from Section 4.1.5 for each case-base. As we stated earlier we would like to minimise this value.

$$\textit{Trade-Off} = \frac{NCCR}{CCR} \tag{6.3}$$

In Table 6.14 we can see the Trade-off results for each case-base. As expected the values for the `DNAp`, `e-clinic2`, `spam` and `e-clinic` case-bases are quite low while the values for the `Diabetes, Breast-Cancer, Bronchiolitis, Liver and Heart` case-bases are much higher. Comparing the accuarcy results for *k-NN* on each case-base with the trade-off value for that case-base there would appear to be a clear relation. The confidence mechanism performs far better on case-bases which appear to have a higher level of competence reflected in the higher accuracy result achieved by the *k-NN*. This trend can be clearly seen in Figure 6.9. There is quite a strong linear trend in the data which is clearly visible and reflected by a Pearson's correlation figure of 0.63. It appears that the performance of our confidence mechanism is dependent on the competence of the case-base and the higher the level of competence the better the performance.

In the second situation we examined the performance of our confidence mechanism when we wished to minimise incorrect predictions that we were confident about. As

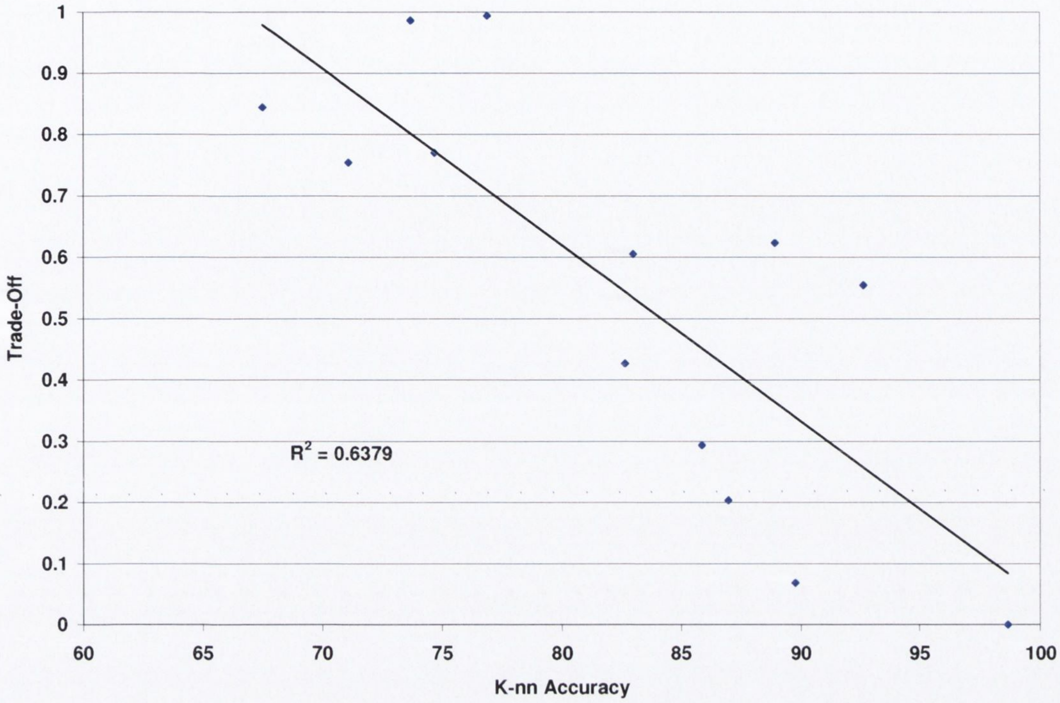


Figure 6.9: Graph of the Balance of Rate against *k-nn*'s accuracy for each case-base

Table 6.14: Confidence Rates for Scenario One

Case-base	k-NN Accuracy	Confident Correct Rate	Not Confident Correct Rate	Trade Off
DNAp	85.85	0.93	0.27	0.29
ionosphere	88.89	0.95	0.59	0.62
e-clinic2	98.66	0.99	0.0	0.0
diabetes	73.7	0.75	0.74	0.98
vote	92.64	0.97	0.54	0.55
German Credit	74.7	0.83	0.64	0.77
spam	89.75	0.96	0.07	0.06
BAC	82.65	0.94	0.4	0.42
e-clinic	86.96	0.94	0.19	0.20
Breast Cancer	76.9	0.77	0.77	0.99
Bronchiolitis	71.08	0.88	0.67	0.76
Liver	67.54	0.75	0.44	0.65
Heart	82.96	0.87	0.65	0.75

outlined in Section 4.1.5 this involves setting the confidence level threshold so that this can be minimised. The results can be seen in Tables 6.15 and 6.16. In this case we are trying to minimise the *Confident Correct* column at the cost of the making more *Not Confident* predictions and this can be clearly seen as the values in the *Confident Incorrect* column are far lower then those in Table 6.13. The mistakes in confidence haven't been entirely eliminated in many of the case-bases as this proved impossible without defaulting to predicting *Not Confident* all of the time. The decreased number of *Confident Incorrect* predictions come at the cost of reduced *Confident Correct* predictions and an increase in the number of cases being given *Not Confident* ratings. This clearly reflects the trade-off that is involved in choosing threshold values.

This trade-off is particularly apparent in a number of case-bases such as in `Spam` and `Vote`. In the case of the Spam case-base the number of *Confident Incorrect* predictions is reduced from 13 to 0 and in Vote from 13 to 6. However the improvement in both cases comes at a cost. In the case of the Vote case-base 33.3% of predictions are now labelled as *Not Confident* whereas previously 9.4% were. In the Spam case-base the number of predictions given *Not Confident* labels jumps from 7.5% to 65.5%.

Table 6.15: Confidence Results for Scenario Two

Case-base	k-NN Accuracy	Confident Correct	Confident Incorrect	Not Confident Correct	Not Confident Incorrect
DNAp	85.85	62	0	29	15
ionosphere	88.89	167	7	145	32
e-clinic2	98.66	286	0	9	4
diabetes	73.7	39	22	527	180
vote	92.64	284	6	119	26
German Credit	74.7	58	3	689	250
spam	89.75	138	0	221	41
BAC	82.65	67	3	14	14
e-clinic	86.96	228	1	32	38
Breast Cancer	76.9	54	13	159	51
Bronchiolitis	71.08	4	2	114	46
Liver	67.54	66	20	167	92
Heart	82.96	44	8	180	36

The confidence rates for Scenario Two can be seen in 6.16. The *Not Confident Correct Rate* is increased in all case-bases in comparison with those in Scenario One reflecting the increased number of cases that are given *Not Confident* ratings. In all but 3 of the case-bases the *Not Confident Correct Rate* is over 0.60 meaning that over 60% of cases that are classified as being not confident predictions are in fact correct predictions. In the case of the `ionosphere`, `vote` and `spam` case-bases over 80% of cases are being mislabelled as being *Not Confident* predictions.

There are modest increases in the *Confident Correct Rates* in most case-bases however not in all. In the `Diabetes` and `Bronchiolitis` case-bases the *Confident Correct Rate* decreases. This is as a result of the decreased number of predictions given a confident label. Although the number of *Confident Incorrect* predictions has decreased the number that are still made represent a higher proportion of the number of predictions labelled as being confident.

Although on case-bases such as `Spam` the decrease in the number of *Confident Incorrrect* predictions is reduced, it is at the cost of quite a large increase in *Not Confident Correct Rate* this is not a entirely general trend. In the `BAC`, `DNAp` and `e-clinic` case-bases the reduction in *Confident Incorrect* predictions comes at a much more modest increase in the *Not Confident Correct Rate.* It would appear that the cost in reducing the *Confident Incorrect* predictions varies and is dependent on the characteristics of the particular case-base.

Table 6.16: The Confidence Rates for Scenario Two

Case-base	k-NN Accuracy	Confident Correct Rate	Not Confident Correct Rate
DNAp	85.85	1.0	0.66
ionosphere	88.89	0.96	0.81
e-clinic2	98.66	1.0	0.69
diabetes	73.7	0.64	0.75
vote	92.64	0.98	0.82
German Credit	74.7	0.95	0.73
spam	89.75	1.0	0.84
BAC	82.65	0.96	0.5
e-clinic	86.96	0.99	0.5
Breast Cancer	76.9	0.80	0.46
Bronchiolitis	71.08	0.66	0.71
Liver	67.54	0.77	0.64
Heart	82.96	0.85	0.82

6.4.3 Discussion of Results

It appears that our confidence mechanism has the potential to work reasonably well but performance is very much dependent on the underlying case-base and the requirements of the domain. In terms of maintaining a high *Confident Correct Rate* while maintaining a low *Not Confident Correct Rate* the performance of our mechanism is very much correlated to the competence of the underlying case-base. The ideal situation of a perfect *Confident Correct Rate* combined with a zero *Not Confident Correct Rate* is unattainable and decisions must be made on a domain by domain basis on how to best balance the trade-off that exits between these two conflicting criteria. This is clearly demonstrated in the varying degree of cost incurred in terms of increased *Not Confident Correct Rate* when we tried to minimise *Confident Incorrect* predictions. This reinforces the value of the methodology we proposed for investigating the factors involved in designing confidence mechanisms.

The difficulty in creating good confidence measures has been well documented and our results reflect this same difficulty (Delaney et al. 2005b, Cheetham and Price 2004). It is worth noting that there is a subtle but important difference in the objective that we have investigated compared with that of predicting confidence in the spam domain. We have tried to predict confidence prioritising accuracy on both classes equally. In the spam domain this same symmetry doesn't exist and inaccurately predicting confidence on spam that has been classified as real mail isn't as costly as predicting confidence on real mail that has been classified as spam.

In conclusion, we have shown a mechanism for assessing classifier confidence that is quite effective when the classifier has high accuracy (greater than 90%). However, the need for confidence assessments is greater in classifiers that are less accurate so that the impact of errors can be ameliorated.

6.5 Conclusions

We have developed an Explanation Framework that has many favourable characteristics;

- It applies a lazy approach in keeping with the strengths of the CBR methodology
- Little Expert Knowledge Needed
- Predicting Confidence

We have evaluated the performance of our framework and found that it successfully addresses the short-comings of traditional case-based explanations. High satisfaction ratings among users of the explanations produced by the framework were recorded. In terms of instilling confidence, the framework also proved successful, however in cases of low confidence doubt and confusion still remained. Given the difficulties in accurately predicting confidence this result reinforces the sentiment expressed in our previous work that long term user confidence can only really be maintained if the underlying system is accurate (Nugent, Cunningham and Doyle 2005).

We have also shown that Localised Logistic Regression is a extremely effective recommendation strategy. However, careful attention must be paid to choice of parameter dictating the size of local case-base as the performance is quite sensitive to this. The difference in performance of our localised approach compared with global logistic regression further reinforces that idea that there are local feature interactions which can only be captured by a local approach.

Chapter 7

Conclusions and Future Work

As we discussed in Chapter 1 it has been apparent since the early days of Expert Systems in the 1970s that there has been a need for Artificial Intelligence technologies to somehow explain and justify the recommendations that they make. It is a problem that persists to this day. People are still understandably often suspicious of such systems and reluctant to accept their recommendations without question. The provision of explanations by such systems should help alleviate any suspicion the user might have and instil confidence in them in the system's prediction. However, in practice, the explanations provided by such systems are in terms of rules which are often not usefully interpretable and have proved unsuccessful in reassuring users.

CBR represents an alternative approach which has inherent advantages in terms of transparency and user acceptance as discussed in Chapter 3. However, the traditional approach in CBR-light applications of simply supplying the nearest neighbour has been found to have shortcomings in terms of providing satisfactory explanations. In this thesis we have described the Explanation Framework we have developed to address these issues. The primary issues we addressed were;

- The selection of *a fortiori* cases as explanation cases without use of domain knowledge,
- Explaining the details and relevance of retrieved cases without domain knowledge.
- Providing measures of confidence.

The Framework we developed uses a localised solution to these problems which is in keeping with the CBR philosophy and uses Logistic Regression models to extract information useful to the explanation process. This approach supports continued incremental learning and eliminates the need for expert domain knowledge. We have also demonstrated that Case-based explanations can be used for non-CBR systems and our current implementation is not restircted to such systems.

The approach we have taken also has potential in terms of assessing confidence and also as a classifier mechanism. We have also evaluated our Framework in terms of these two objectives. First, however, we will discuss the merits of our Framework in terms of addressing its primary purpose, providing improved CBR explanations that instil users with confidence.

7.1 Improving Case-based Explanations

The Explanation Framework we have developed addresses the shortcoming of the traditional CBR-light explanations of not making the relationships between the feature values in the Query Case and Explanation Case explicit. We performed a user evaluation of the Framework using the BAC data and found results to be largely in favour of the explanations produced by our framework.

The Evaluation we carried out highlighted the level of confusion and uncertainty that can exist when the system supplies no explanation. Our evaluations also showed that although Case-based explanations are an improvement on no explanation some uncertainty still remained.

Conversely, the framework explanations proved to be extremely effective at instilling confidence. Users expressed a strong conviction that the systems prediction was correct when presented with framework explanations and user ratings of those explanations were very high. It is clear that generating discursive texts explaining the effects of feature-value differences greatly improves upon simple case-based explanations.

However, in terms of our goal of alerting users to when the system is making a mistake, the framework explanations only proved a limited success. Although it is encouraging that users heeded the confidence warning and no one responded that the system was definitely correct it is clear that users were none the wiser as to what the correct recommendation should be. These results indicate that it may be extremely difficult to maintain user confidence in a system that makes mistakes.

The primary role of the explanations we provide is in justification as we want to reassure the user of the systems predictions. In trying to achieve this, our explanations also touch on other goals as define by Cassens (2004) such as transparency, relevance and learning. In providing a set of goals Sormo and Cassens were defining a spectrum on which explanations could be placed. Most importantly they highlighted that the purpose for which an explanation is intended and the user are of primary importance in defining an explanation scheme. Providing explanations is a domain dependent task and each individual explanation needs to be addressed with the targeted audience of the explanations in mind. The framework explanations that we have developed will not be universally acceptable as in some domains the requirements of the user may not be met buy such explanations. In some cases the style of explanations produced by the framework may need to be adjusted to meet the demands of the domain they are operating in. For example, we added a small amount of linguistic information for the BAC domain. The explanations provided by the framework as we have implemented it are also limited to binary classification tasks.

However, we have demonstrated that, in a domain were traditional CBR explanations are acceptable, the framework explanations provided a much more effective solution. The framework addresses many of the general shortcomings of the traditional CBR explanation approach for instilling confidence and does so in a flexible manner.

7.2 Assessing Confidence

Our Framework also has potential in terms of assessing confidence in recommendations. Providing confidence measures is very much a domain dependent task with many conflicting criteria needing to be balanced depending on the demands of that domain. We have described how the use of ROC-like curves can be used to assess in a clear visual manner the effects of different model parameters on the performance of the confidence assessment mechanism. In this way the optimal performance can quickly be identified.

We investigated the effectiveness of our Framework at assessing confidence on 13 UCI case-bases and in two different scenarios. We found that our confidence mechanism has the potential to work reasonably well but the performance is very much dependent on the underlying case-base and the requirements of the domain. Our mechanism proved to be quite effective when the classifier has high accuracy (greater than 90%). However, the need for confidence assessments is greater in classifiers that are less accurate so that the impact of errors can be ameliorated. To do so accurately when there is low case-base competence appears to be quite difficult. We are not the first to discover this limitation and it is a problem which as been remarked upon by other researchers (Delaney et al. 2005b, Cheetham and Price 2004).

7.3 Classification using Local Logistic Regression

The use of Local Logistic Regression Models proved to be quite effective as a classification mechanism. We evaluated this scheme on 13 UCI case-bases and compared it with global Logistic Regression and the *k-NN* model. The Local Logistic Regression approach proved to be more accurate than the other two classification schemes on many of the case-bases. We found the local logistic approach to be more accurate than the global approach on all the case-bases that we have tested. However, we also found the performance of the classification scheme to be very sensitive to parameter selection which must be carried out carefully.

7.4 Future Work

7.4.1 Local Logistic Regression as a Classifier

The primary focus of this Thesis has been on producing useful and interpretable explanations however we have also demonstrated that Local Logistic Regression is effective as a classification method. In the future we would like to investigate this further. We would like to develop more sophisticated ways of defining a local case-base that are automated, parameter free and based on the characteristics of the case-base being used.

7.4.2 Further work on Case-Based Explanations

Many of the case-bases that we have presented in this thesis have relatively small numbers of features. This may not always be the case and we would like to investigate how the information that we can extract from our Local Logistic Regression model could be used to filter out irrelevant features and select the most useful to present to the user.

Bibliography

Aamodt, A. and Plaza, E.: 1994, Case-based reasoning: Foundational issues, methodological variations, and system approaches., *AI Commun.* **7**(1), 39–59.

Aha, D. W.: 1997, Special issue on lazy learning, *Artificial Intelligence Review* **11**, 7–10.

Amati, G. and Crestani, F.: 1999, Probabilistic learning for selective dissemination of information., *Inf. Process. Manage.* **35**(5), 633–654.

Andrews, R., Diederich, J. and Tickle, A.: 1995, A survey and critique of techniques for extracting rules from trained artificial neural networks, *knowledge based systems* **8**, 187–202.

Armengol, E., Palaudàries, A. and Plaza, E.: 2001, Individual prognosis of diabetes long-term risks: A CBR approach, *Methods of Information in Medicine* **40**, 46–51.

Atkeson, C. G. and Moore, A. W.and Schaal, S.: 1997, Locally weighted learning, *Artificial Intelligence Review* **11**, 11–73.

Bergmann, R.: 1993, Integrating abstraction, explanation-based learning from multiple examples and hierarchical clustering with a performance component for planning, *in* E. Plaza (ed.), *Proceedings of the ECML-93 Workshop on Integrated Learning Architectures (ILA-93)*, Vienna, Austria.

Breiman, L., Friedman, J., Olshen, R. and Stone, C.: 1984, *Classification and Regression Trees*, Belmont, CA: Wadsworth.

Cassens, J.: 2004, Knowing what to explain and when, *1st Workshop on Case-Based Explanation (Proceedings of the ECCBR 2004 workshops)*, pp. 97–104.

Cestnik, B., Kononenko, I. and Bratko, I.: 1987, Assistant 86: a knowledge-elicitation tool for sophisticated users, *Progress in Machine Learning*, Sigma, Wilmslow, England, pp. 31–45.

Cheetham, W.: 2000, Case-based reasoning with confidence., *in* E. Blanzieri and L. Portinale (eds), *Advances in Case-Based Reasoning, 5th European Workshop, EWCBR 2000, Trento, Italy, September 6-9, 2000, Proceedings*, Vol. 1898 of *Lecture Notes in Computer Science*, Springer, pp. 15–25.

Cheetham, W. and Price, J.: 2004, Measures of solution accuracy in case-based reasoning systems, *in* P. Funk and P. A. G. Calero (eds), *Advances in Case-Based Reasoning, 7th. European Conference on Case-Based Reasoning (ECCBR 2004)*, Vol. 3155 of *Lecture Notes in Computer Science*, Springer, pp. 106–118.

Clancy, W. and Bock, C.: 1982, Mrs/neomycin: Representing metacontrol in predicate calculus, *Report No. HPP-82-3*, pp. 343–348.

Coyle, L., Doyle, D. and Cunningham, P.: 2004, Representing similarity for CBR in XML, *in* P. A. G. Calero and P. Funk (eds), *Advances in Case-Based Reasoning, 7th European Conference on Case-Based Reasoning, ECCBR 2004*, Vol. 3155, Springer, Madrid, Spain, pp. 119–127.

Coyle, L., Hayes, C. and Cunningham, P.: 2003, Representing cases for CBR in XML, *Expert Update* **6**(2), 7–13.

Craven, M. W. and Shavlik, J. W.: 1996, Extracting tree-structured representations of trained networks, *in* D. S. Touretzky, M. C. Mozer and M. E. Hasselmo (eds), *Advances in Neural Information Processing Systems*, Vol. 8, The MIT Press, pp. 24–30.
URL: *citeseer.ist.psu.edu/craven96extracting.html*

Craw, S., Wiratunga, N. and Rowe, R.: 1998, Case-based design for tablet formulation., *in* B. Smyth and P. Cunningham (eds), *Advances in Case-Based Reasoning, 4th European Workshop, EWCBR-98, Dublin, Ireland, September 1998, Proceedings*, Springer, pp. 358–369.

Cunningham, P.: 1998, Cbr: strengths and weaknesses, *in* A. del Pobil, J. Mira and M. Ali (eds), *Proceedings of 11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, number 1416 in *LNAI*, Springer, p. 517523.

Cunningham, P., Doyle, D. and Loughrey, J.: 2003, An evaluation of the usefulness of case-based explanation, *in* K. D. Ashley and D. G. Bridge (eds), *Case-Based Reasoning Research and Development, 5th International Conference on Case-Based Reasoning (ICCBR 2003)*, Vol. 2689 of *Lecture Notes in Computer Science*, Springer, pp. 122–130.

Cunningham, P., Nowlan, N., Delany, S. J. and Haahr, M.: 2003, A case-based approach to spam filtering that can track concept drift, *In The ICCBR'03 Workshop on Long-Lived CBR Systems, Trondheim, Norway, June 2003. Available: http://www.cs.tcd.ie/publications/tech-reports/reports.03/TCD-CS-2003-16.pdf.*

Davis, R.: 1982, Expert systems: Where are we? and where do we go from here?, *The AI Magazine* **3**(2), 3–22.

Delaney, S.-J., Cunningham, P. and Doyle, D.: 2005a, Generating estimates of classification confidence for a case-based spam filter, *Proceedings of the 6th International Conference on Case-Based Reasoning, Chicago*, Springer-Verlag, pp. 507–530.

Delaney, S.-J., Cunningham, P. and Doyle, D.: 2005b, Generating estimates of classification confidence for a case-based spam filter, *Proceedings of the 6th International Conference on Case-Based Reasoning, Chicago*, Springer-Verlag, pp. 507–530.

Delany, S. J. and Cunningham, P.: 2004a, An analysis of case-base editing in a spam filtering system., *in* P. Funk and P. A. González-Calero (eds), *Advances in Case-Based Reasoning, 7th European Conference, ECCBR 2004, Madrid, Spain, August 30 - September 2, 2004, Proceedings*, Vol. 3155 of *Lecture Notes in Computer Science*, Springer, pp. 128–141.

Delany, S. J. and Cunningham, P.: 2004b, An analysis of case-based editing in a spam filtering system, *in* P. Funk and P. González-Calero (eds), *7th European Conference on Case-Based Reasoning (ECCBR 2004)*, Vol. 3155 of *LNAI*, Springer, pp. 128–141.

Delany, S. J., Cunningham, P., Tsymbal, A. and Coyle, L.: 2005, A case-based technique for tracking concept drift in spam filtering, *Knowledge-Based Systems* **18**(4–5), 187–195.

Domingos, P.: 1998, Knowledge discovery via multiple models., *Intell. Data Anal.* **2**(1-4), 187–202.

Doyle, D.: 2005, *A Knowledge-Light Mechanism for Explanation in Case-Based Reasoning*, PhD thesis, Dublin University, Trinity College.

Doyle, D., Cunningham, P., Bridge, D. and Rahman, Y.: 2004, Explanation orientated retrieval, *in* P. Funk and P. A. G. Calero (eds), *Advances in Case-Based Reasoning, 7th. European Conference on Case-Based Reasoning (ECCBR 2004)*, Vol. 3155 of *Lecture Notes in Computer Science*, Springer, pp. 157–168.

Doyle, D., Loughrey, J., Nugent, C., Coyle, L. and Cunningham, P.: 2005, Fionn: A framework for developing CBR systems, *Expert Update* **8**(1), 11–14.

Flach, P., Blockeel, H., Ferri, C., Hernandez-Orallo, J. and Struyf, J.: 2003, *Decision support for data mining: introduction to ROC analysis and its application*, Kluwer Academic Publishers, pp. 81–90.

Gentner, D.: 1983, Structure-mapping: A theoretical framework for analogy, *Cognitive Psychology* **7**, 155–170.

Gentner, D., Holyoak, K. and Kokinov, B.: 2001, *The Analogical Mind: Perspectives from Cognitive Science*, MIT Press, Cambridge.

Gentner, D., Loewenstein, J. and Thompson, L.: 2003, Learning and transfer: A general role for analogical encoding, *Journal of Educational Psychology* **95**(2), 393–408.

Goel, A. K., Kolodner, J. L., Pearce, M., Billington, R. and Zimring, C.: 1991, Towards a case-based tool for aiding conceptual design problem solving, *Proc. of the Case-Based Reasoning Workshop*, Washington, D.C., pp. 109–120.

Hayes, C., Cunningham, P. and Doyle, M.: 1998, Distributed CBR using XML, *in* W. Wilke and J. Schumacher (eds), *Proceedings of the KI-98 Workshop on Intelligent Systems and Electronic Commerce.*

Holyoak, K. and Thagard, P.: 1989, Analogical mapping by constraint satisfaction, *Cognitive Science* **13**(3), 295–355.

Hosmer, D. and Lemeshow, S.: 2000, *Applied Logistic Regression*, 2nd edn, Wiley.

Jackson, P.: 1986, *Introduction to expert systems*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Kononenko, I., Bratko, I. and Roskar, E.: 1984, Experiments in automatic learning of medical diagnostic rules, *Technical report*, Jozef Stefan Institute, Ljubljana, Yugoslavia.

le Cessie, S. and van Houwelingen, J.: 1992, Ridge estimators in logistic regression, *Applied Statistics* **41**(14), 191–201.

Leake, D.: 1992, *Evaluating Explanations: A Content Theory*, Lawrence Erlbaum, Hillsdale, NJ.

Leake, D.: 1996, *Case-Based Reasoning: Experiences, Lessons and Future Directions*, AAAI/MIT Press.

Leake, D., Birnbaum, L., Hammond, K., Marlow, C. and Yang, H.: 2001, An integrated interface for proactive, experience-based design support, *Proceedings of the 2001 International Conference on Intelligent User Interfaces*, pp. 101–108.

Lenz, M.: 1993, Cabata - a hybrid case-based reasoning system, *in* M. Lenz, B. Bartsch-Spörl, H.-D. Burkhard and S. Wess (eds), *In Proceed. of the First European Conference on Case- Based Reasoning. SEKI Report SR-93-12. University of Keiserslautern*, pp. 204–209.

Lenz, M., Auriol, E. and Manago, M.: 1998, *Diagnosis and Decision Support*, Springer, pp. 51–90.

Lenz, M. and Burkhard, H.-D.: 1996, Case retrieval nets: Basic ideas and extensions, *in* G. Gorz and S. Holldobler (eds), *KI-96: Advances in Artificial Intelligence*, number 1137 in *Lecture Notes in Artificial Intelligence*, Springer Verlag, pp. 227–239.

Lewis, D. D.: 1995, Evaluating and optimizing autonomous text classification systems., *in* E. A. Fox, P. Ingwersen and R. Fidel (eds), *SIGIR*, ACM Press, pp. 246–254.

Majchrzak, A. and Gasser, L.: 1991, On using artificial intelligence to integrate the design of organixational and process change in us manufacturing, *AI and Society* **5**, 321–338.

Manago, M., Bergmann, R., Conruyt, N., Traphöner, R., Pasley, J., Renard, J. L., Maurer, F., Wess, S., Althoff, K.-D. and Dumont, S.: 1994, Casuel: A common case representation language.

McSherry, D.: 2003, Explanation in case-based reasoning: an evidential approach, *8th UK Workshop on Case-Based Reasoning*, pp. 47–55.

McSherry, D.: 2004, Explaining the pros and cons of conclusions in cbr, *in* P. A. G. Calero and P. Funk (eds), *Advances in Case-Based Reasoning, 7th European Conference,* *ECCBR 2004 Madrid, Spain, August 30th through Sep 2nd, 2004*, Vol. 3155 of *LNAI*, Springer, pp. 317–330.

Mitchell, T. M.: 1997, *Machine Learning*, McGraw-Hill, chapter 3, pp. 52–80.

Nugent, C. and Cunningham, P.: 2005, A case-based explanation system for black box systems, *Artificial Intelligence Review* p. to appear.

Nugent, C., Cunningham, P. and Doyle, D.: 2005, The best way to instil confidence is by being right; an evaluation of the effectiveness of case-based explanations in providing user confidence, *in* H. Muñoz-Avila and F. Ricci (eds), *6th International Conference on Case-Based Reasoning*, Vol. 3620 of *LNAI*, Springer, pp. 368–381.

Nugent, C., Doyle, D. and Cunningham, P.: to appear 2005, *CBR in Knowledge Discovery and Data Mining*, Wiley, chapter Gaining Insight through Case-Based Explanation.

Quinlan, J.: 1992, Learning with continuous classes, *in* Adams and Sterling (eds), *AI92*, World Scientific, pp. 343–348.

Quinlan, J.: 1993, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Francisco.

Quinlan, J. R.: 1986, Induction of decision trees, *Machine Learning* **1**, 81–106.

Richter, M. M.: 1995, The knowledge contained in similarity measures. Invited talk at ICCBR95, http://www.cbr-web.org/documents/Richtericcbr95remarks.html.

Richter, M. M.: 1998, Introduction, *in* M. Lenz, B. Bartsch-Spörl, H.-D. Burkhard and S. Wess (eds), *Case-Based Reasoning Technology, From Foundations to Applications*, Springer, pp. 1–16.

Riesbeck, C. and Schank, R.: 1989, *Inside Case-Based Reasoning, Erlbaum*, Erlbaum.

Salzberg, S.: 1991, Distance metrics for instance-bsed learning., *ISMIS*, pp. 399–408.

Schank, R.: 1986, *Explanation Patterns: Understanding Mechanically and Creatively*, Lawrence Erlbaum, Hillsdale, NJ.

Schank, R. and Abelson, R.: 1975, Scripts, plans, and knowledge, *Proceedings of the Fourth International Joint Conference on Artificial Intelligence*, IJCAI, Tbilisi, USSR.

Schank, R. C.: 1999, *Dynamic Memory Revisited*, Cambridge University Press.

Shortliffe, E. H.: 1976, *Computer Based Medical Consultations: MYCIN*, American Elsevier, New York (1976.

Smyth, B. and McKenna, E.: 2001, Competence models and the maintenance problem, *Computational Intelligence* **17**(2), 235–249.

Sormo, F. and Cassens, J.: 2004, Explanation goals in case-based reasoning, *1st Workshop on Case-Based Explanation (Proceedings of the ECCBR 2004 workshops)*, pp. 165–174.

Stahl, A.: 2001, Learning feature weights from case order feedback., *in* D. W. Aha and I. Watson (eds), *Case-Based Reasoning Research and Development, 4th International Conference on Case-Based Reasoning, ICCBR 2001, Vancouver, BC, Canada, July 30 - August 2, 2001, Proceedings*, Springer, pp. 502–516.

Stahl, A.: 2002, Defining similarity measures: Top-down vs. bottom-up., *in* S. Craw and A. D. Preece (eds), *Advances in Case-Based Reasoning, 6th European Conference, ECCBR 2002 Aberdeen, Scotland, UK, September 4-7, 2002, Proceedings*, Springer, pp. 406–420.

Stahl, A. and Gabel, T.: 2003, Using evolution programs to learn local similarity measures., *in* K. D. Ashley and D. G. Bridge (eds), *Case-Based Reasoning Research and Development, 5th International Conference on Case-Based Reasoning, ICCBR 2003, Trondheim, Norway, June 23-26, 2003, Proceedings*, Springer, pp. 537–551.

Swartout, W.: 1983, Xplain: A system for creating and explaining expert consulting systems, *Artificial Intelligence* **21**, 285 – 325.

Swartout, W. R. and Moore, J. D.: 1993, Explanation in second generation expert systems, pp. 543–585.

Thagard, P.: 1989, Explanatory coherence, *The Behavioral and Brain Sciences* **12**(3), 435–502.

Tickle, A.and Andrews, R., Golea, R. and Diederich, J.: 1998, The truth will come to light: Directions and challenges in extracting rules from trained neural networks, *IEEE Transactions on Neural Networks* **9**, 1057–1068.

Veloso, M. M. and Carbonell, J. G.: 1994, Case-based reasoning in PRODIGY, *in* R. S. Michalski and G. Teccuci (eds), *Machine Learning: A Multistrategy Approach, Volume IV*, Morgan Kaufmann, pp. 523–548.

Watson, I.: 1997, *Applying Case-based Reasoning: Techniques for Enterprise Systems*, Morgan Kaufmann.

Watson, I. and Gardingen, D.: 1985, Meta level knowledge, *in* F. HayesRoth, D. A. Waterman and D. B. Lenat (eds), *Rule- Based Systems*, Addison-Wesley, pp. 507–530.

Wettschereck, D., Aha, D. W. and Mohri, T.: 1997, A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms, *Artif. Intell. Rev.* **11**(1-5), 273–314.

Wick, M. R. and Thompson, W. B.: 1992, Reconstructive expert system explanation, *Artif. Intell.* **54**(1-2), 33–70.

Wirth, N.: 1977, What can we do about the unnecessary diversity of notation for syntactic definitions?, *Commun. ACM* **20**(11), 822–823.

Wittgenstein, L.: 1943, *Philosophical Investigations*, Blackhall.

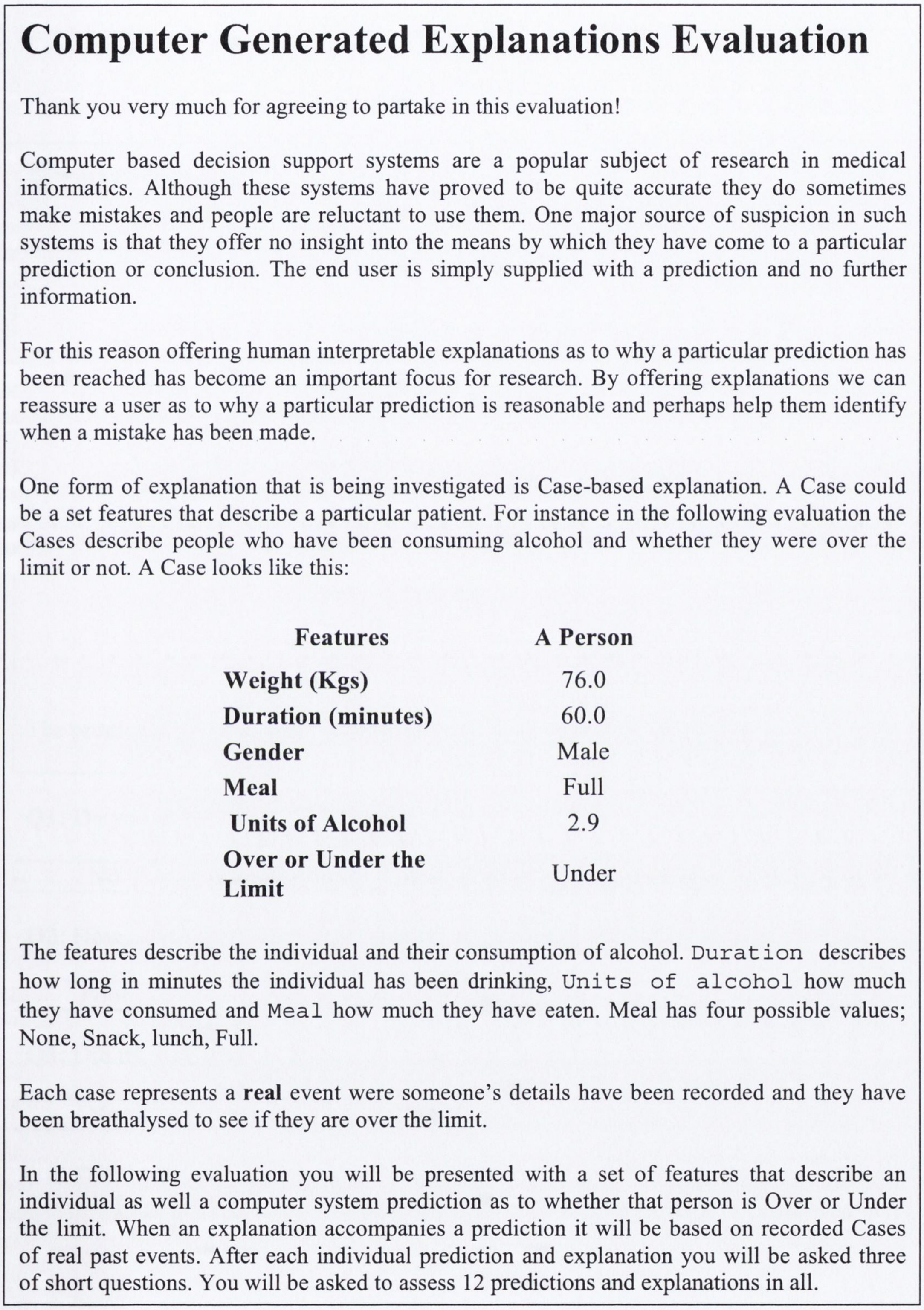
Zhang, J. and Yang, Y.: 2004, Probabilistic score estimation with piecewise logistic regression, *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, ACM Press, New York, NY, USA, p. 115.

Zhou, Z.-H. and Jiang, Y.: 2003, The truth will come to light: Directions and challenges in extracting rules from trained neural networks, *IEEE Transactions on Neural Networks* **7**(1), 37–42.

Appendix A

User Evaluation

This appendix contains examples of the explanations presented to users in the evaluation as well as the coversheet. Question three was dropped from the evaluation as the results were found to be inconclusive



Computer Generated Explanations Evaluation

Thank you very much for agreeing to partake in this evaluation!

Computer based decision support systems are a popular subject of research in medical informatics. Although these systems have proved to be quite accurate they do sometimes make mistakes and people are reluctant to use them. One major source of suspicion in such systems is that they offer no insight into the means by which they have come to a particular prediction or conclusion. The end user is simply supplied with a prediction and no further information.

For this reason offering human interpretable explanations as to why a particular prediction has been reached has become an important focus for research. By offering explanations we can reassure a user as to why a particular prediction is reasonable and perhaps help them identify when a mistake has been made.

One form of explanation that is being investigated is Case-based explanation. A Case could be a set features that describe a particular patient. For instance in the following evaluation the Cases describe people who have been consuming alcohol and whether they were over the limit or not. A Case looks like this:

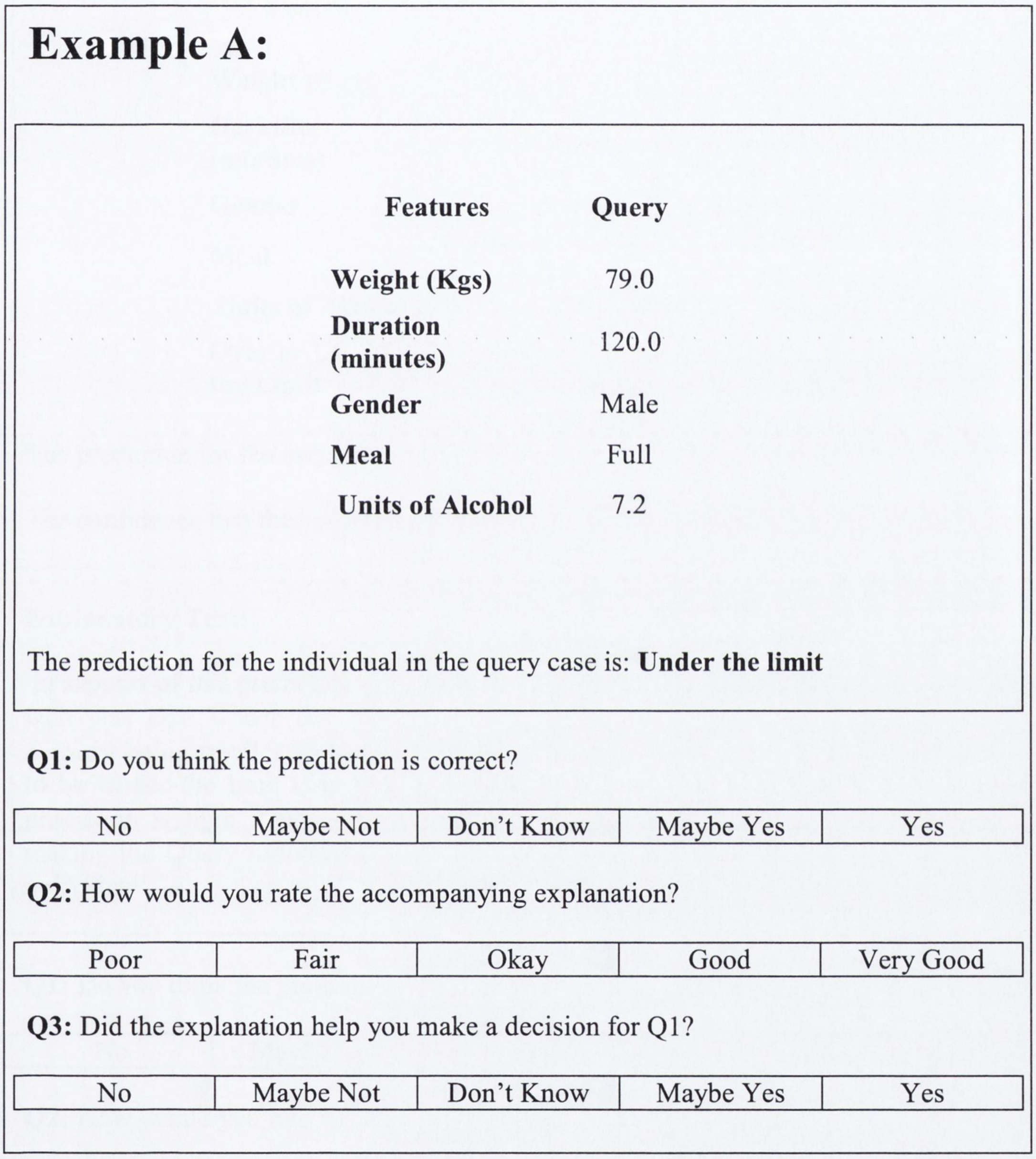
Features	**A Person**
Weight (Kgs)	76.0
Duration (minutes)	60.0
Gender	Male
Meal	Full
Units of Alcohol	2.9
Over or Under the Limit	Under

The features describe the individual and their consumption of alcohol. `Duration` describes how long in minutes the individual has been drinking, `Units of alcohol` how much they have consumed and `Meal` how much they have eaten. Meal has four possible values; None, Snack, lunch, Full.

Each case represents a **real** event were someone's details have been recorded and they have been breathalysed to see if they are over the limit.

In the following evaluation you will be presented with a set of features that describe an individual as well a computer system prediction as to whether that person is Over or Under the limit. When an explanation accompanies a prediction it will be based on recorded Cases of real past events. After each individual prediction and explanation you will be asked three of short questions. You will be asked to assess 12 predictions and explanations in all.

Figure A.1: Cover Sheet of Questionaire



Example A:

Features	Query
Weight (Kgs)	79.0
Duration (minutes)	120.0
Gender	Male
Meal	Full
Units of Alcohol	7.2

The prediction for the individual in the query case is: **Under the limit**

Q1: Do you think the prediction is correct?

No	Maybe Not	Don't Know	Maybe Yes	Yes

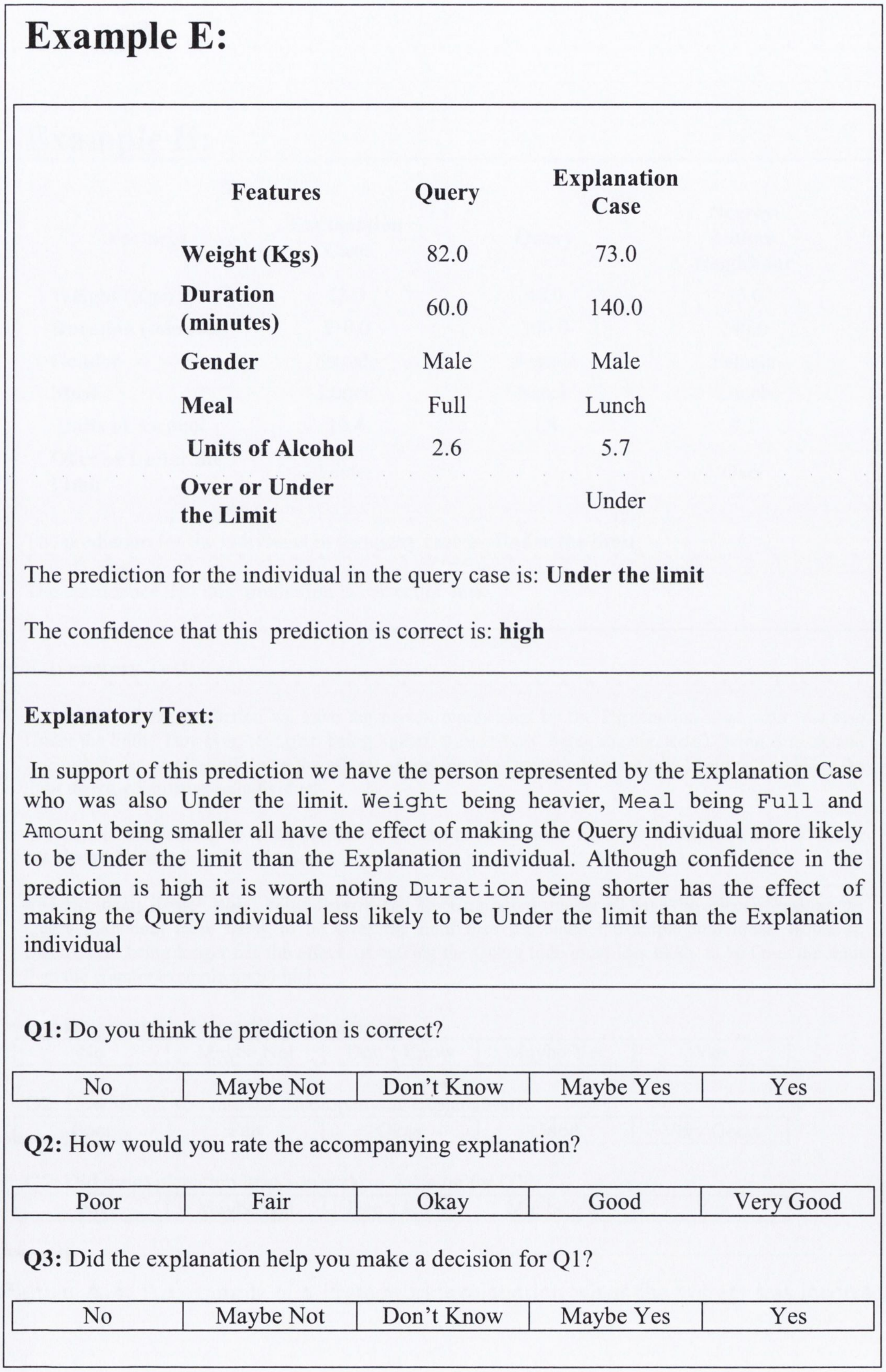
Q2: How would you rate the accompanying explanation?

Poor	Fair	Okay	Good	Very Good

Q3: Did the explanation help you make a decision for Q1?

No	Maybe Not	Don't Know	Maybe Yes	Yes

Figure A.2: An example of when no explanation was presented



Example E:

Features	Query	Explanation Case
Weight (Kgs)	82.0	73.0
Duration (minutes)	60.0	140.0
Gender	Male	Male
Meal	Full	Lunch
Units of Alcohol	2.6	5.7
Over or Under the Limit		Under

The prediction for the individual in the query case is: **Under the limit**

The confidence that this prediction is correct is: **high**

Explanatory Text:

In support of this prediction we have the person represented by the Explanation Case who was also Under the limit. `Weight` being heavier, `Meal` being `Full` and `Amount` being smaller all have the effect of making the Query individual more likely to be Under the limit than the Explanation individual. Although confidence in the prediction is high it is worth noting `Duration` being shorter has the effect of making the Query individual less likely to be Under the limit than the Explanation individual

Q1: Do you think the prediction is correct?

No	Maybe Not	Don't Know	Maybe Yes	Yes

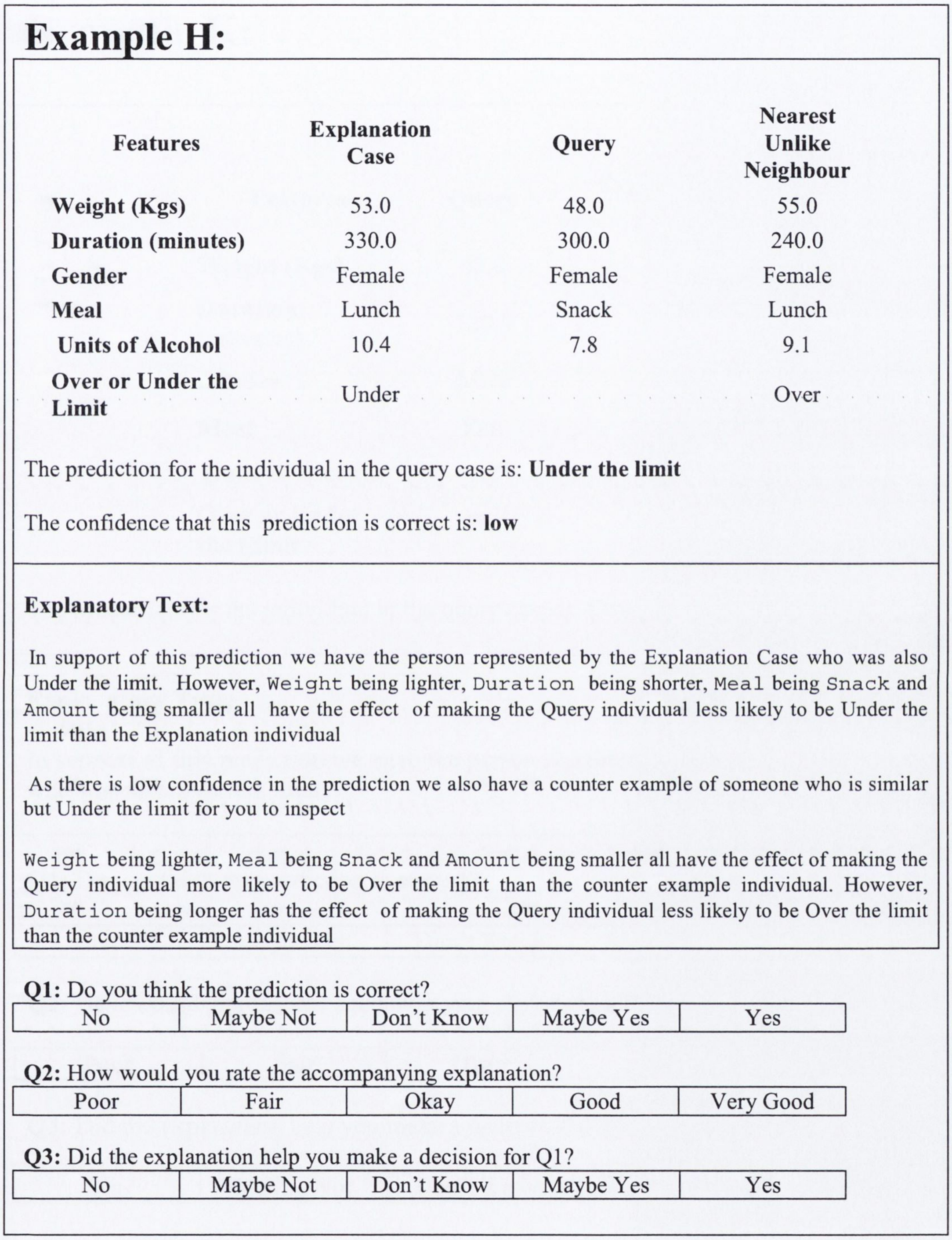
Q2: How would you rate the accompanying explanation?

Poor	Fair	Okay	Good	Very Good

Q3: Did the explanation help you make a decision for Q1?

No	Maybe Not	Don't Know	Maybe Yes	Yes

Figure A.3: An example of a Framework explanation when the system was correct



Example H:

Features	Explanation Case	Query	Nearest Unlike Neighbour
Weight (Kgs)	53.0	48.0	55.0
Duration (minutes)	330.0	300.0	240.0
Gender	Female	Female	Female
Meal	Lunch	Snack	Lunch
Units of Alcohol	10.4	7.8	9.1
Over or Under the Limit	Under		Over

The prediction for the individual in the query case is: **Under the limit**

The confidence that this prediction is correct is: **low**

Explanatory Text:

In support of this prediction we have the person represented by the Explanation Case who was also Under the limit. However, `Weight` being lighter, `Duration` being shorter, `Meal` being `Snack` and `Amount` being smaller all have the effect of making the Query individual less likely to be Under the limit than the Explanation individual

As there is low confidence in the prediction we also have a counter example of someone who is similar but Under the limit for you to inspect

`Weight` being lighter, `Meal` being `Snack` and `Amount` being smaller all have the effect of making the Query individual more likely to be Over the limit than the counter example individual. However, `Duration` being longer has the effect of making the Query individual less likely to be Over the limit than the counter example individual

Q1: Do you think the prediction is correct?

No	Maybe Not	Don't Know	Maybe Yes	Yes

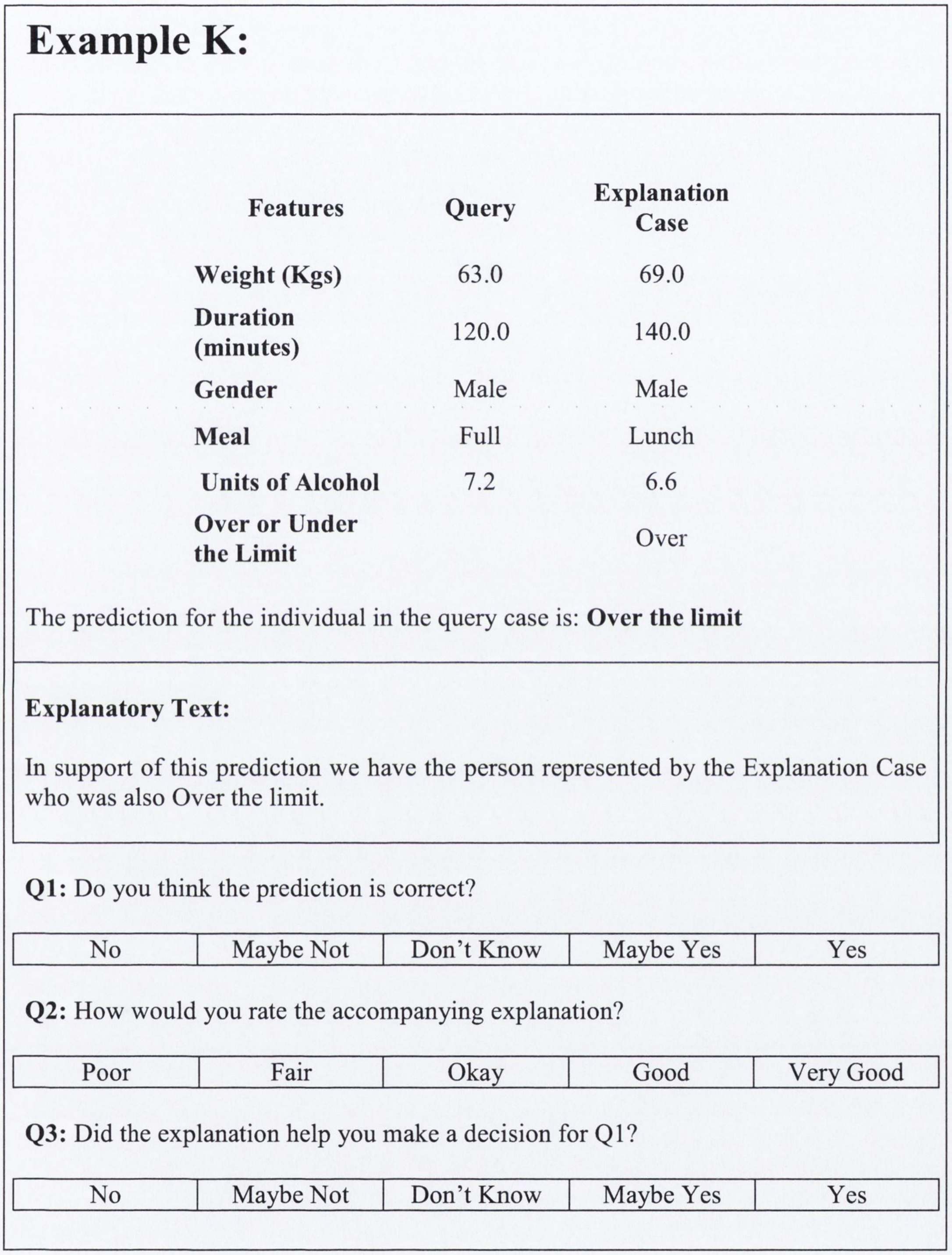
Q2: How would you rate the accompanying explanation?

Poor	Fair	Okay	Good	Very Good

Q3: Did the explanation help you make a decision for Q1?

No	Maybe Not	Don't Know	Maybe Yes	Yes

Figure A.4: An example of a Framework explanation when the system was incorrect



Example K:

Features	Query	Explanation Case
Weight (Kgs)	63.0	69.0
Duration (minutes)	120.0	140.0
Gender	Male	Male
Meal	Full	Lunch
Units of Alcohol	7.2	6.6
Over or Under the Limit		Over

The prediction for the individual in the query case is: **Over the limit**

Explanatory Text:

In support of this prediction we have the person represented by the Explanation Case who was also Over the limit.

Q1: Do you think the prediction is correct?

No	Maybe Not	Don't Know	Maybe Yes	Yes

Q2: How would you rate the accompanying explanation?

Poor	Fair	Okay	Good	Very Good

Q3: Did the explanation help you make a decision for Q1?

No	Maybe Not	Don't Know	Maybe Yes	Yes

Figure A.5: An example of a case-based explanation