PREDICTING THE END PRICES OF ONLINE AUCTIONS

Project report submitted in partial fulfillment of the requirement for the degree of Bachelor of Technology

in

Computer Science and Engineering

By

Shagun Goyal (121299)
&
Shantanu Manohar (121321)

Under the supervision of

Ms. Ramanpreet Kaur

to

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology Waknaghat, Solan-173234, Himachal Pradesh

Candidate's Declaration

I hereby declare that the work presented in this report entitled **"Predicting End Prices of Online Auctions"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from August 2015 to May 2016 under the supervision of **Ms. Ramanpreet Kaur** (Assistant Professor, Information Technology).

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Shagun Goyal-121299

Shantanu Manohar-121321

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

Ms. Ramanpreet Kaur

Assistant Professor

Information Technology

Dated:

Acknowledgement

We take this opportunity to express are profound sense of gratitude and respect to all those who helped us throughout the duration of our project.

This report acknowledges the intense driving and technical competence of all the individuals who have contributed to it. It would have been almost impossible to produce fruitful results during the working of the project without the support of those people. We extend our thanks and gratitude to our project guide **Ms. Ramanpreet Kaur** who has helped us at every step. She spent her valuable time from her busy schedule to train us and provided her active and sincere support for our daily activities.

I would like to express my heartfelt thanks to Brig. (Mr.) S.P. Ghrera, H.O.D., Computer Science and Engineering Department, Jaypee University of Information Technology, for his astute guidance, his constant encouragement and support throughout.

This report has been compiled by the sincere and active support from our guide who provided us proper guidance and direction regarding various problems. We have tried our best to summarize this report.

Signature	………………….	…………………
Name	Shagun Goyal	Shantanu Manohar
Roll No	121299	121321
Date	………………….	………………….

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

The eBay marketplace offers millions of items for sale each day. Understanding and being able to better predict the outcome of these auctions is important to buyers and sellers alike for individual profit maximization.

Online auctions have become one of the fastest growing modes of online commerce transactions. eBay has 94 million active members buying and selling goods at a staggering rate. These auctions are also producing large amount of data that can be utilized to provide services to the buyers and sellers, market research, machine learning algorithm to predict end-prices of auction items. We describe the features used, and several formulations of the price prediction problem. Using the PDA category from eBay, we show that our algorithms are extremely accurate and can result in a useful set of services for buyers and sellers in online marketplaces.

Online auctions are one the most popular methods to buy and sell items on the internet. With more than 100 million active users globally (ars of Q4 2011), eBay is the world's largest online marketplace, where practically anyone can buy and sell practically anything. The total value of goods sold on eBay was $68.6 billion, more than $2,100 every second. This kind of volume produces huge amounts of data that can be utilized to provide services to the buyers and sellers, market research, and product development.

In this analysis, we collect historical auction data from eBay and use machine learning algorithms to predict sales results of auction items. We describe the features used and formulations used for making predictions. The algorithms used can be relatively accurate and can result in a useful set of services for buyers and sellers.

Online auctions allow users to sell and buy a variety of goods, and they are now one of the most important web services. Predicting final prices on online auctions is a hard task. However, there has been much pioneering work over the past ten years. In this project an evaluation of the effectiveness of our method is also described in the project report.

For final price prediction, we find that for multiclass binary prediction decision trees models is the best for prediction. We provide a discussion on the result, as well as some insight about our particular data set and avenues for future exploration.

1. INTRODUCTION

1.1 Introduction

Online auctions have become increasingly popular in recent years, and as a consequence there is a growing body of empirical research on this topic. Online auctions are generating a new class of fine-grained data about online transactions. This data lends itself to a variety of applications and services that can be provided to both buyers and sellers in online marketplaces. We collect data from online auctions and use several classification algorithms to predict the probable-end prices of online auction items.

EBay, Yahoo! Auctions, Amazon Marketplace and other online marketplaces have become significant commercial entities and it is estimated that they will account for 25% of online ecommerce by 2005. Even today, eBay, one of the largest online marketplaces, consists of 94 million members typically offering 19 million items for sale at any given time. In 2003, $24 billion of goods were sold on eBay. Although such online marketplaces offer individuals a unique opportunity to buy and sell goods, they also offer a new source of data that can be utilized to make inferences about mass economic behavior, product marketing, market research, and provide services to the buyers and sellers participating in the online marketplaces.

Predictions are necessary as they help both the buyers and the sellers. Earlier, when the systems were incapable to predict, the sellers were uncertain about the starting bid of the product and also the price at which the product would be sold. With predictions, the seller can easily predict about the prices. The buyer has a certain budget and searching for the products which are in his budget range becomes easy.

In this project, we describe our work on a system capable of predicting the end-price of auction listings. Price prediction for auctions is a challenging task for machine learning algorithms mainly because of the large number of attributes that can vary in auction settings. Even items (of the same kind) differ in condition. The variance in shipping charges, reliability of sellers, appearance of the listing, beginning and ending times, all are factors that make it difficult to predict the price of an auction. Even if all the above variations were accounted for, there is still the uncertainly in human behavior when bidding in auctions. Auction Software Reviews reported that 15% of the auctions eBay are won in the last minute which increases the uncertainty in the end-price of a given auction.

The price prediction system is trained by using the characteristics of the seller, the item to be auctioned, the features of the auction, and historical auction data to make predictions about the outcome of an auction before it starts. We describe the features used, the various ways in which price prediction can be formulated as a machine learning problem, and the performance results of several algorithms applied to this task. These results show that we can predict the end-prices of auctions very accurately which leads to several applications that can be used to provide new services to the participants in online marketplaces.

One of the main drivers of this interest is eBay (*www.eBay.com*). On any given day, there are several million items, dispersed across thousands of categories, for sale on eBay. Specifically, we develop a dynamic forecasting system to predict the price of an ongoing auction.

By dynamic, we mean that the model can predict the price of an auction “in progress” and can update its prediction based on newly arriving information. Forecasting price in online auctions is challenging because traditional forecasting methods cannot adequately account for two features of online auction data:
(1) The unequal spacing of bids
(2) The changing dynamics of price and bidding throughout the auction.
Our dynamic forecasting model accounts for these special features by using modern functional data analysis techniques. Forecasting price in online auctions can have benefits to different auction parties. For instance, price forecasts can be used to dynamically score auctions for the same (or similar) item by their predicted price. On any given day, there are several hundred or even thousands of open auctions available, especially for very popular items such as Apple iPods or Microsoft Xboxes. Dynamic price scoring can lead to a ranking of auctions with the lowest expected price. Such a ranking could help bidders focus their time and energy on only a few select auctions, that is, those that promise the lowest price. Auction forecasting can also be beneficial to the seller or the auction house.

We consider the problem of product valuation for online auctions and present a system that automates the market research step that determines a “good” price for an item on eBay. For the high-volume reseller, automating the (intelligent) evaluation of a product offered for sale and predicting its final auction price significantly reduces human involvement, ultimately resulting in higher trading throughput and reduced labour costs for a professional eBay merchant. For the casual buyer, such a system will be helpful in determining whether to buy or not to buy an item, decreasing the time he or she will have to spend manually researching the market (or ameliorating the monetary cost of the absence of such research). There exist a large number of eBay automation tools, as well as a substantial body of research in price estimation, but the two fields appear to be largely separate. It seems that the eBay trading community does not realize the benefits that might be available to it, while computer scientists and economists are more interested in theoretical aspects of price prediction. We would like to think that our system may be the first that bridges the gap between the two.

1.2 Problem Statement

With the prevalence of the Internet and e-commerce, the online exchange market, especially the online auction market develops very fast. The activities of online auction produce a large number of transaction data. If utilized properly, these data can be of great benefit to sellers, buyers and website administrator. Typically, the final price prediction results may help sellers optimize the selling price of their items and auction attributes. At the same time, part of the information asymmetry problems may be solved for buyers. Thus, transaction time can be shortened and cost can be saved. In this project, we collect historical exchange data from EBay, an online auction website and use machine learning algorithms and traditional statistical methods to forecast the final prices of auction items. Some experiments are performed and the prediction results are discussed to verify the proposed solution.

1.3 Objectives

We are interested in 2 things:

1. **Determine whether an auction listing will result in a sale:** An auction can result in either the sale of the product or not. In this part of the report we predict the status of the product sale.

2. **Predict final sale prices for auctions:** In this, we predict the end price of a product. If a product results in a sale, then the final price would be predicted using previous trends.

1.4 Methodology

1.4.1 Data Set:

A Data Set is a collection of data. Most commonly a **data set** corresponds to the contents of a single database table, or a single statistical **data** matrix, where every column of the table represents a particular variable, and each row corresponds to a given member of the **data set** in question.

Since we did not know the working of Web Crawler, we had collected the data of products sold on Ebay.com manually.

Table 1: Attributes on which the Final Price may depend:

Feature	**Description**
Price	Final price the auction. If the listing does not result in a sale, the Price will be equal to 0.
StartingBid	Minimum bid for the auction
BidCount	Number of bids made for the auction
Title	Auction title
QuantitySold	The number of items sold in the listing. Represented by a 0 or 1.
SellerRating	Seller’s eBay rating
SellerAboutMePage	Whether the seller has an eBay About Me page
StartDate	The beginning date and time of the auction
EndDate	The ending date and time of the auction
PositiveFeedbackPercent	The percent of positive feedback (of all the feedback) received by the seller
HasPicture	Indicates the seller included a picture with the listing Represented by a 0 or 1.
SellerCountry	The country of the seller
BuyItNowPrice	The optional price to buy the item instantly
HighBidderFeedbackRating	Highest bidder’s eBay rating
ReturnsAccepted	Whether the seller accepts returns.

	Represented by a 0 or 1.
HasFreeShipping	Whether the seller provides free shipping. Represented by a 0 or 1.

From the above attributes, we have not used the following:

- Title: Does not determine the Final Price
- SellerAboutMePage: Does not determine the Final Price
- HasFreeShipping: Shipping Charges were very complex, as the charges were different for different countries.
- PositiveFeedbackPercent: All users had a feedback of 95-100 %. So this attribute is insignificant.
- SellerCountry: Insignificant
- HasPicture: All auctions had pictures. So this attribute is insignificant.
- Start Date: Insignificant
- End Date: Insignificant

1.4.2 WEKA

WEKA is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. WEKA contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes.

We had predicted the sale of items using WEKA during the 1st half of the project.

1.4.3 ANFIS

ANFIS uses a hybrid learning algorithm to tune the parameters of a Sugeno-type fuzzy inference system (FIS). The algorithm uses a combination of the least-squares and back-propagation gradient descent methods to model a training data set. ANFIS also validates models using a checking data set to test for overfitting of the training data.

We have predicted the final price of items using ANFIS during the 2nd half of the Project.

2. LITERATURE SURVEY

2.1 Previous Work on Predictions

There has been a lot of work in the Economics and Data Mining community on analyzing online auctions. Most of this work has focused on describing past auctions rather than prediction. Bajari & Hortascu develop econometric techniques to build models of bidding behavior. Lucking-Riely et al. Use data collected from eBay about auctions of collectible coins to study the factors that affect the price. Although this study is a good exploratory analysis of online auctions, it only finds correlations between attributes of the auction and the resulting price and does not aim to build predictive models.

There has been some work in price prediction of items in online markets e.g. airlines fares but not much has been done in the auction domain. The only work we are aware of that involves predicting prices in auctions was done implicitly during the Trading Agent Competition (TAC) focusing on the travel domain. TAC relies on a simulator of airline, hotel, and ticket prices and the competitors build agents to bid on these. TAC simulates prices and assumes that the supply of products (airline tickets) is unlimited. Several TAC competitors have explored a range of methods for price prediction including historical averaging, neural nets, and boosting. All of the work in this domain is performed with artificially generated data and does not use any real auction data. The work in this project is based on data collected from eBay and is aimed at predicting the prices to provide a new set of services to the buyers and sellers in online marketplaces.

2.2 Online Auction is fading Fast

The online auction is fading fast. Fewer than 15 percent of all who post EBay listings are opting for an auction-only sale these days, according to a paper published this month (PDF) by Stanford University researchers who worked with the online marketplace for their study. Most sellers instead choose to offer items at fixed, "Buy It Now" prices, making EBay more like a conventional e-commerce website than the online-auction pioneer it was in the 1990s.

"This sort of cried out from the data we saw," says Liran Einav, a Stanford economist and one of the study's four authors. "Most people view it as a big auction site, but we realized their business is much more like Amazon."

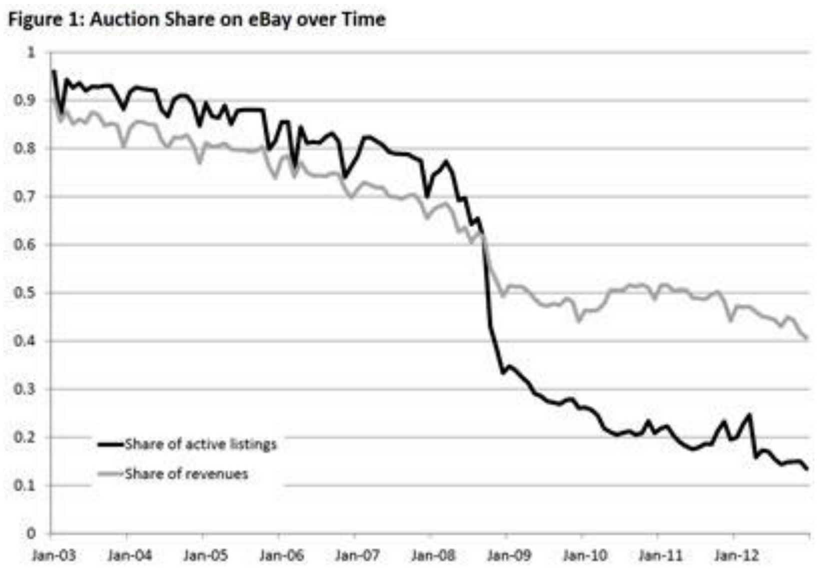


Figure 1: Auction Share on Ebay over Time

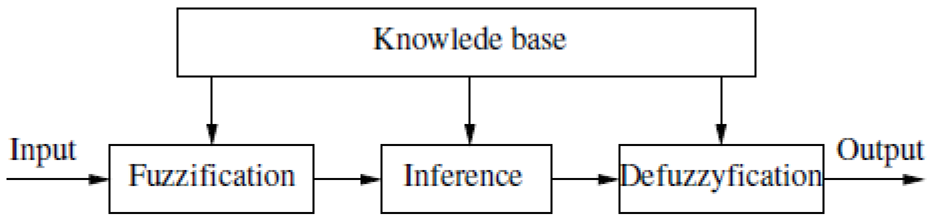
In part, the shift away from auctions stems from people preferring to focus on alternative online pastimes such as Facebook and YouTube. Einav contends that these competing diversions have drained much of the allure from online bidding, which can take days to end in a sale. "People want to do stuff quickly, and doing stuff in an ideal market setting takes time," he explains. "Some people like going to a garage sale and finding good deals, but most people hate it." This speedy shopping dynamic has accelerated on mobile devices: According to the study, EBay's mobile users reduce their browsing time by 25 percent on average, relative to those on computer.

The signature benefit of the online-auction format—arriving at efficient prices for even the most unusual items—has lost some its purpose as well, thanks to the proliferation of e-commerce venues with posted prices. Even EBay’s own archive of sales data helps users zero in on a market rate quickly, without needing to pit potential buyers against one another in an auction.

The upside of EBay’s buy-it-now economy? Buyers willing to bid are getting better deals. In recent years, auctioned items have sold at prices about 16 percent below similar items offered at fixed prices, according to the study. “The rationale for auctions becomes less about price discovery and closer to the use of couponing and other promotional strategies seen in more traditional retail markets,” the study notes. And many items still lend themselves to auctions, including collectibles, jewelry, and such rare objects as locks of Justin Bieber’s hair.

None of this is news to EBay, which for years has been pushing its business model beyond bidding. In 2008, EBay changed its search algorithm to rank results more heavily on relevance than time left in an item’s sale window. That tweak helped trigger a big drop in auctions. Still, the shift away from auctions hasn’t put a damper on results: Last year, EBay’s marketplace websites generated $6.1 billion in revenue, a 12 percent increase over 2011.

2.3 Adaptive Neuro-Fuzzy Inference System (ANFIS)



• The fuzzy inference system that we have considered is a model that maps

– input characteristics to input membership functions,

– input membership function to rules,

– Rules to a set of output characteristics,

– Output characteristics to output membership functions, and

– The output membership function to a single-valued output, or

– A decision associated with the output.

• We have only considered membership functions that have been fixed, and somewhat arbitrarily chosen.

• Also, we have only applied fuzzy inference to modelling systems whose rule structure is essentially predetermined by the user's interpretation of the characteristics of the variables in the model.

• In general the shape of the membership functions depends on parameters that can be adjusted to change the shape of the membership function.

•The parameters can be automatically adjusted depending on the data that we try to model.

Model Learning and Inference through ANFIS

• Suppose we already have a collection of input/output data and would like to build a fuzzy inference model/system that approximate the data.

• Such a model would consist of a number of membership functions and rules with adjustable

parameters similarly to that of neural networks.

• Rather than choosing the parameters associated with a given membership function arbitrarily, these parameters could be chosen so as to tailor the membership functions to the input/output data in order to account for these types of variations in the data values.

• The neuro-adaptive learning techniques provide a method for the fuzzy modelling procedure to learn information about a data set, in order to compute the membership function parameters that best allow the associated fuzzy inference system to track the given input/output data.
• Using a given input/output data set, the toolbox function anfis constructs a fuzzy inference system (FIS) whose membership function parameters are tuned (adjusted) using either a backpropagation algorithm alone, or in combination with a least squares type of method.
• This allows your fuzzy systems to learn from the data they are modelling.

FIS Structure and Parameter Adjustment

• A network-type structure similar to that of a neural network, which maps inputs through input membership functions and associated parameters, and then through output membership functions and associated parameters to outputs, can be used to interpret the input/output map.
• The parameters associated with the membership functions will change through the learning process.
• The computation of these parameters (or their adjustment) is facilitated by a gradient vector, which provides a measure of how well the fuzzy inference system is modeling the input/output data for a given set of parameters.
• Once the gradient vector is obtained, any of several optimization routines could be applied in order to adjust the parameters so as to reduce some error measure (usually defined by the sum of the squared difference between actual and desired outputs).
• anfis uses either back propagation or a combination of least squares estimation and backpropagation for membership function parameter estimation.

Familiarity Breeds Validation: Know Your Data

• The modelling approach used by anfis is similar to many system identification techniques.
• First, you hypothesize a parameterized model structure (relating inputs to membership functions to rules to outputs to membership functions, and so on).
• Next, you collect input/output data in a form that will be usable by anfis for training.
• You can then use anfis to train the FIS model to emulate the training data presented to it by

modifying the membership function parameters according to a chosen error criterion.
• In general, this type of modeling works well if the training data presented to anfis for training (estimating) membership function parameters is fully representative of the features of the data that the trained FIS is intended to model.
• This is not always the case, however. In some cases, data is collected using noisy measurements, and the training data cannot be representative of all the features of the data that will be presented to the model.
This is where model validation comes into play.

Model Validation Using Checking and Testing Data Sets
• Model validation is the process by which the input vectors from input/output data sets on which the FIS was not trained, are presented to the trained FIS model, to see how well the FIS model predicts the corresponding data set output values.
• You use a validation data set to check and control the potential for the model overfitting the data.
• When checking data is presented to anfis as well as training data, the FIS model is selected to have parameters associated with the minimum checking data model error.
• One problem with model validation for models constructed using adaptive techniques is selecting a data set that is both representative of the data the trained model is intended to emulate, yet sufficiently distinct from the training data set so as not to render the validation process trivial.
• If you have collected a large amount of data, hopefully this data contains all the necessary
representative features, so the process of selecting a data set for checking or testing purposes is made easier.
• However, if you expect to be presenting noisy measurements to your model, it is possible the training data set does not include all of the representative features you want to model.
• The basic idea behind using a checking data set for model validation is that after a certain point in the training, the model begins overfitting the training data set.
• In principle, the model error for the checking data set tends to decrease as the training takes place up to the point that overfitting begins, and then the model error for the checking data suddenly increases.

Two examples

• In the first example in the following section, two similar data sets are used for checking and training, but the checking data set is corrupted by a small amount of noise.

• This example illustrates of the use of the ANFIS Editor GUI with checking data to reduce the effect of model overfitting.

• In the second example, a training data set that is presented to anfis is sufficiently different than the applied checking data set.

Example 1: Checking Data Helps Model Validation

• By examining the checking error sequence over the training period, it is clear that the checking data set is not good for model validation purposes.

• This example illustrates the use of the ANFIS Editor GUI to compare data sets.

• Load training and checking data sets into the MATLAB workspace from the command line:

```
load fuzex1trnData.dat
load fuzex2trnData.dat
load fuzex1chkData.dat
load fuzex2chkData.dat
```

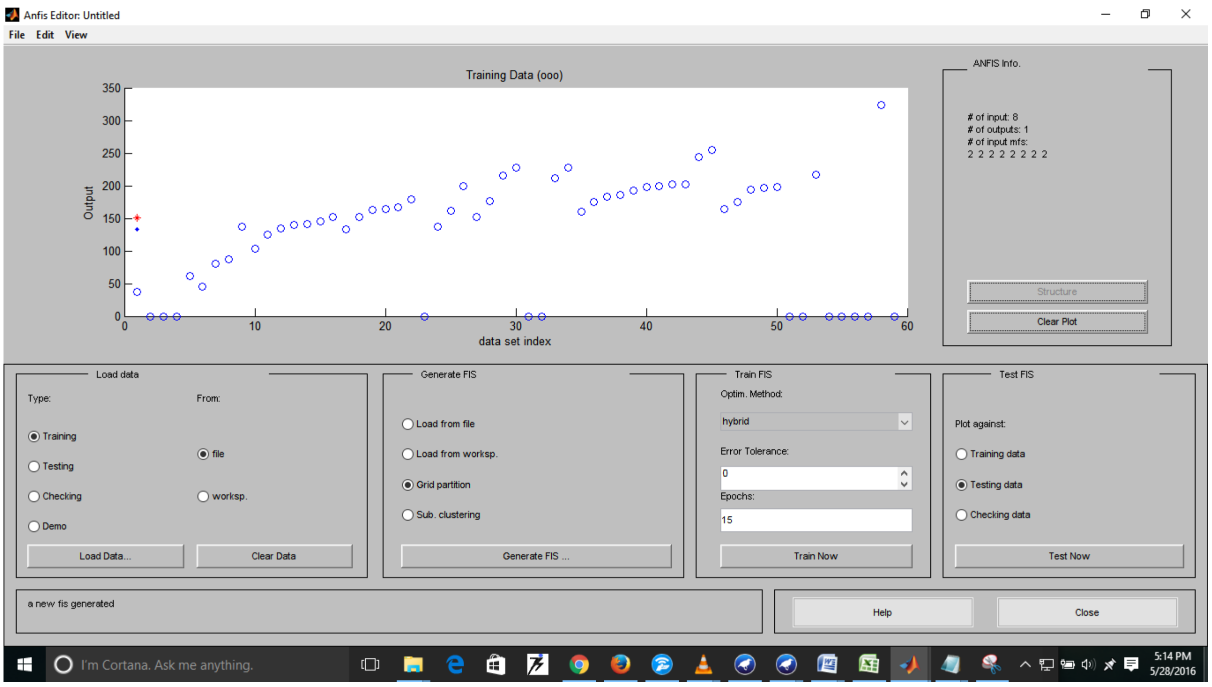
• Open the ANFIS editor GUI: anfisedit

• Load data set fuzex1trnData for training from the workspace

The training data appears in the plot in the center
of the GUI as a set of circles.

• Notice the horizontal axis is marked data set index. This index indicates the row from which that input data value was obtained (whether or not the input is a vector or a scalar).

• Load checking data fuzex1chkData from the workspace. This data appears in the GUI plot as pluses superimposed on the training data.


Anfis Editor: Untitled

ANFIS Info.

# of input: 8
# of outputs: 1
# of input mfs:
2 2 2 2 2 2 2 2

Load data

Type: [✓ Training] Testing Checking Demo

From: [✓ file] worksp.

Generate FIS

Load from file Load from worksp. [✓ Grid partition] Sub. clustering

Train FIS

Optim. Method: hybrid

Error Tolerance: 0

Epochs: 15

Test FIS

Plot against: Training data [✓ Testing data] Checking data

a new fis generated

Fig 2: GUI plot

• This data set will be used to train a fuzzy system by adjusting the membership function parameters that best model this data.

• The next step is to specify an initial fuzzy inference system for anfis to train.

• You can either initialize the FIS parameters to your own preference, or if you do not have any preference for how you want the initial membership functions to be parameterized, you can let anfis do this for you.

Automatic FIS Structure Generation with ANFIS

• Select a partition method: grid partitioning or subtractive clustering (described later). Grid partition is the default partitioning method.

• To Generate FIS choose first the number of membership functions, MFs, then

• the type of input and output membership functions.

Notice there are only two choices for the output membership function: constant and linear.

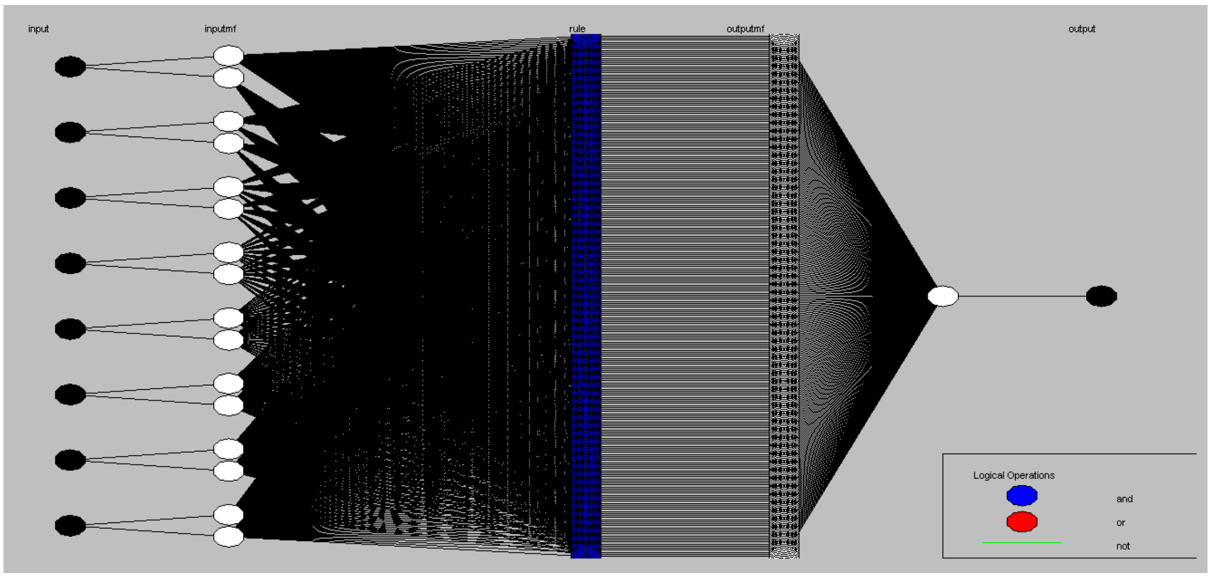


Fig 3: Structure of FIS

ANFIS Training

• The two anfis parameter optimization method options available for FIS training are hybrid (the default, mixed least squares and backpropagation) and backpropa (backpropagation).

• Error Tolerance is used to create a training stopping criterion, which is related to the error size. The training will stop after the training data error remains within this tolerance. This is best left set to 0 if you don't know how your training error is going to behave.

• Running training for 40 epochs gives the following error results:

• Notice how the checking error decreases up to a certain point in the training and then it increases.

• This increase represents the point of model overfitting.

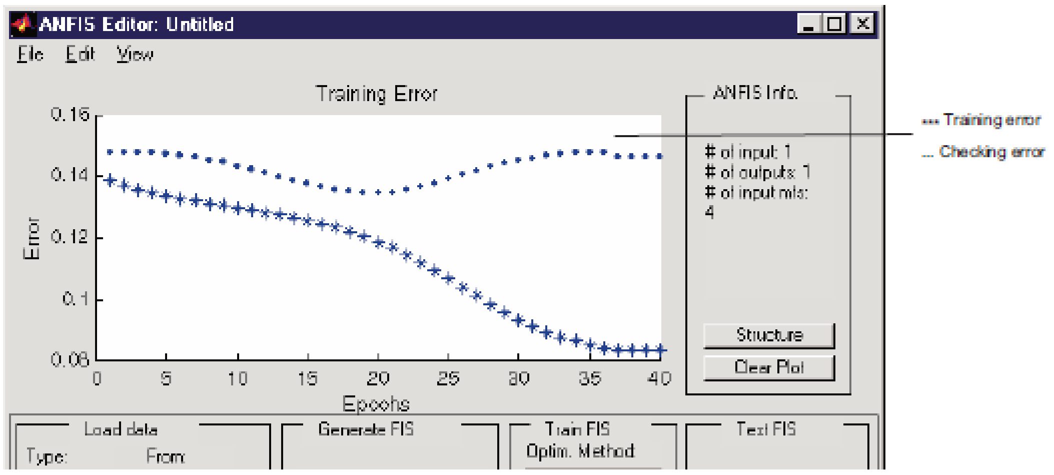
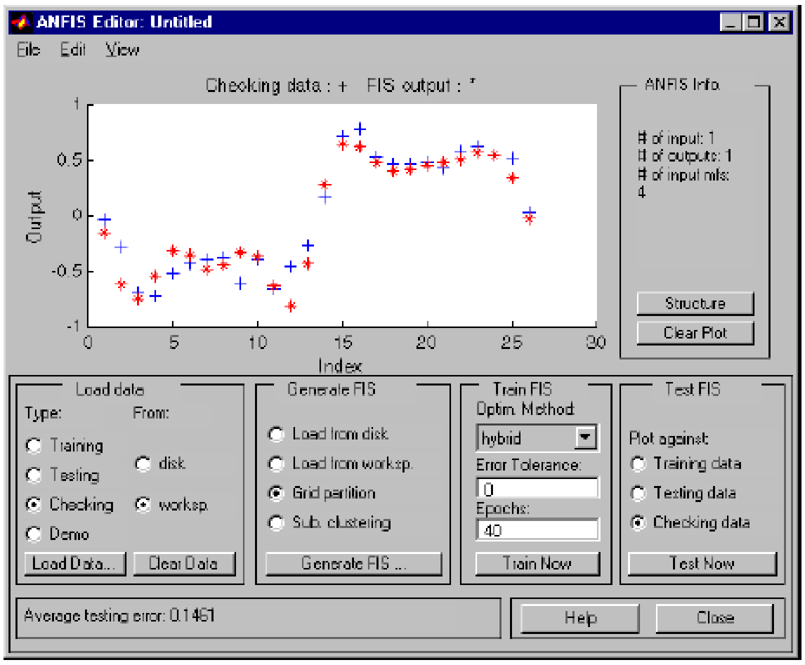


Fig 4: Overfitting

• anfis chooses the model parameters associated with the minimum checking error (just prior to this jump point).

• This is an example for which the checking data option of anfis is useful.

Testing the FIS against the checking data gives the following results.



ANFIS Info.
# of input: 1
# of outputs: 1
# of input mfs:
4

Load data
Type: Training, Testing, [✓ Checking], Demo
From: disk, [✓ worksp.]

Generate FIS
Load from disk, Load from worksp., [✓ Grid partition], Sub. clustering

Train FIS
Optim. Method: hybrid
Error Tolerance: 0
Epochs: 40

Test FIS
Plot against: Training data, Testing data, [✓ Checking data]

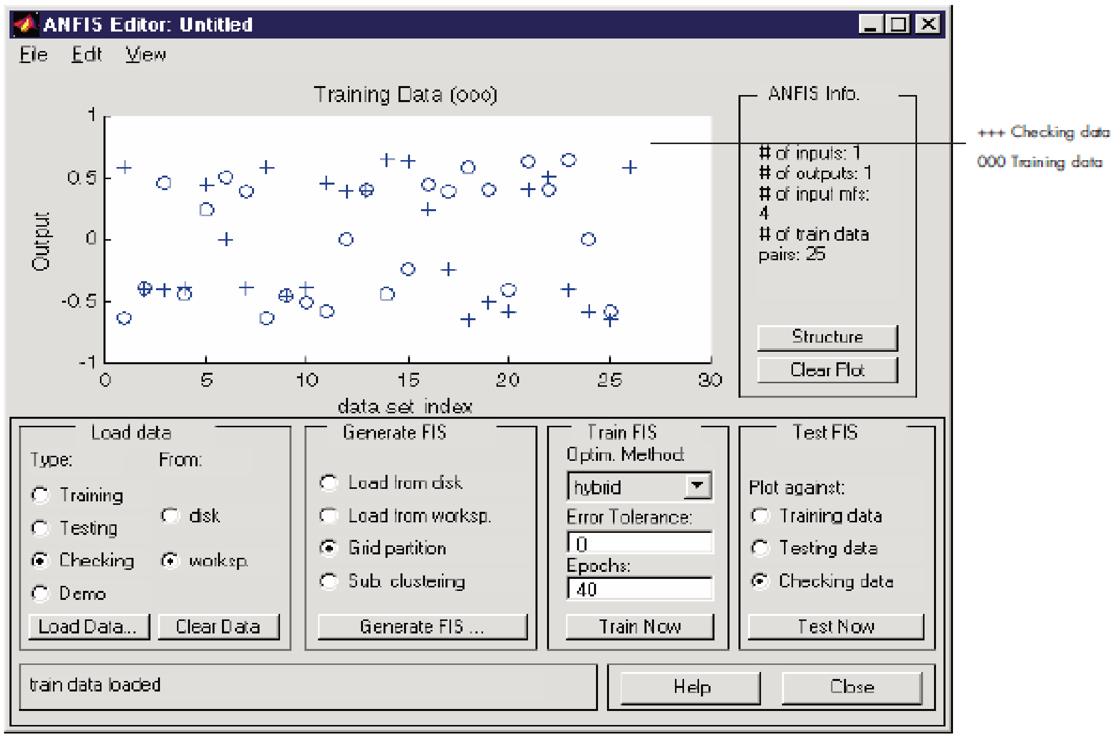
Average testing error: 0.1461

Fig 5: Testing the FIS against the checking Data

Example 2: Checking Data Does not Validate Model

After Clear Plot we load the 2nd set of training and checking data:

You should get something that looks like this.



ANFIS Info.
# of inputs: 1
# of outputs: 1
# of input mfs: 4
# of train data pairs: 25

Load data — Type: Training / Testing / [✓ Checking] / Demo; From: disk / [✓ worksp.]

Generate FIS — Load from disk / Load from worksp. / [✓ Grid partition] / Sub. clustering

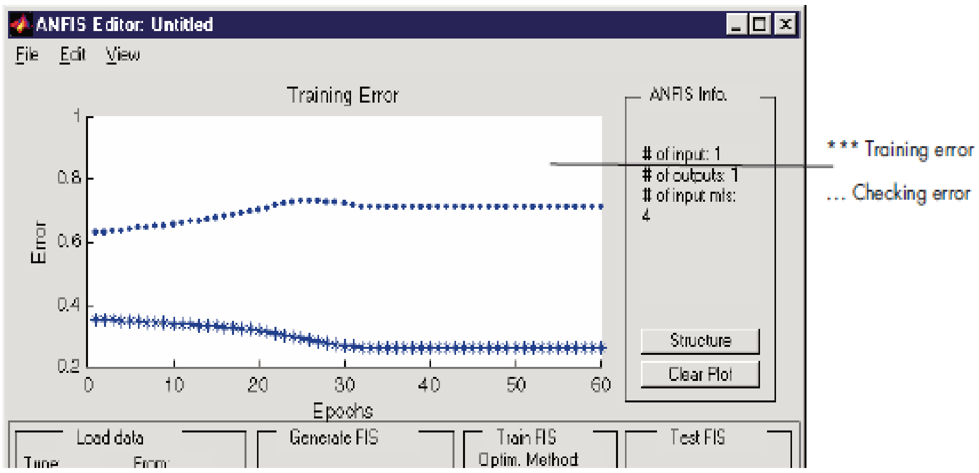
Train FIS — Optim. Method: hybrid; Error Tolerance: 0; Epochs: 40

Test FIS — Plot against: Training data / Testing data / [✓ Checking data]

train data loaded

Fig 6: Checking Data vs Training Data

- Training for 60 epochs gives the following results:
- Notice the checking error is quite large.
- It appears that the minimum checking error occurs within the first epoch.



ANFIS Info.
# of input: 1
# of outputs: 1
# of input mfs: 4

Fig 7: Training for 60 Epochs

•Recall that using the checking data option with anfis automatically sets the FIS parameters to be those associated with the minimum checking error.

• Clearly this set of membership functions would not be the best choice for modeling the training data. Whats wrong here?

• This example illustrates the problem discussed earlier wherein the checking data set presented to anfis for training was sufficiently different from the training data set.

• As a result, the trained FIS did not capture the features of this data set very well.

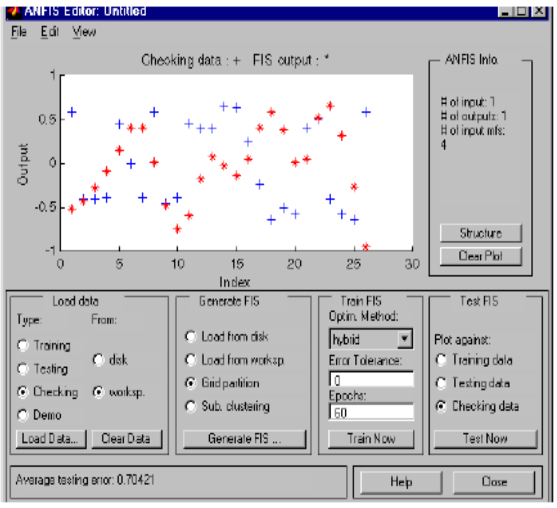
• This illustrates the importance of knowing the features of your data set well enough when you select your training and checking data.

• When this is not the case, you can analyze the checking error plots to see whether or not the checking data performed sufficiently well with the trained model.

• In this example, the checking error is sufficiently large to indicate that either more data needs to be selected for training, or you may want to modify your membership function choices (both the number of membership functions and the type).

• Otherwise the system can be retrained without the checking data, if you think the training data captures sufficiently the features you are trying to represent.

The result of approximating the data is as follows:



# of input: 1
# of outputs: 1
# of input mfs: 4

Type: Training, Testing, [✓ Checking], Demo
From: disk, [✓ worksp.]
Load from disk, Load from worksp., [✓ Grid partition], Sub. clustering
Optim. Method: hybrid
Error Tolerance: 0
Epochs: 60
Plot against: Training data, Testing data, [✓ Checking data]

Average testing error: 0.70421

Fig 8: Checking Data

2.4 WEKA—Experiences with a Java Open-Source Project

WEKA is a popular machine learning workbench with a development life of nearly two decades. This article provides an overview of the factors that we believe to be important to its success.

Rather than focusing on the software's functionality, we review aspects of project management and historical development decisions that likely had an impact on the uptake of the project.

1. Introduction

WEKA contains implementations of algorithms for classification, clustering, and association rule mining, along with graphical user interfaces and visualization utilities for data exploration and algorithm evaluation.

2. What is WEKA?

WEKA is a machine learning workbench that supports many activities of machine learning practitioners.

2.1 Basic Functionality

Here is a summary of WEKA's main features.

• **Data preprocessing**. As well as a native file format (ARFF), WEKA supports various other formats (for instance CSV, Matlab ASCII files), and database connectivity through JDBC. Data can be filtered by a large number of methods (over 75), ranging from removing particular attributes to advanced operations such as principal component analysis.

• **Classification.** One of WEKA's drawing cards is the more than 100 classification methods it contains. Classifiers are divided into "Bayesian" methods (Naive Bayes, Bayesian nets, etc.), lazy methods (nearest neighbor and variants), rule-based methods (decision tables, OneR, RIPPER), tree learners (C4.5, Naive Bayes trees, M5), function-based learners (linear regression, SVMs, Gaussian processes), and miscellaneous methods. Furthermore, WEKA includes meta-classifiers like bagging, boosting, stacking; multiple instance classifiers; and interfaces for classifiers implemented in Groovy and Jython.

• **Clustering.** Unsupervised learning is supported by several clustering schemes, including EM- based mixture models, k-means, and various hierarchical clustering algorithms. Though not as many methods are available as for classification, most of the classic algorithms are included.

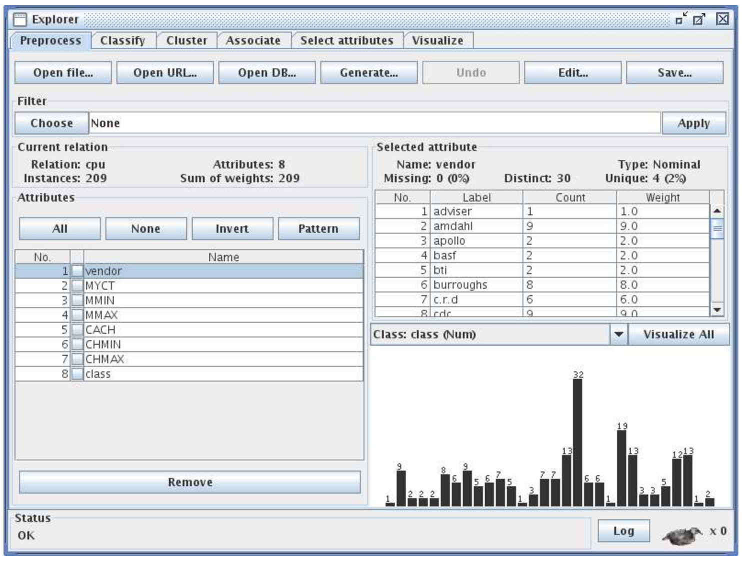
• **Attribute selection.** The set of attributes used is essential for classification performance. Various selection criteria and search methods are available.

• **Data visualization.** Data can be inspected visually by plotting attribute values against the class, or against other attribute values. Classifier output can be compared to training data in order to detect outliers and observe classifier characteristics and decision boundaries. For specific methods there are specialized tools for visualization, such as a tree viewer for any method that produces classification trees, a Bayes network viewer with automatic layout, and a dendrogram viewer for hierarchical clustering. WEKA also includes support for association rule mining, comparing classifiers, data set generation, facilities for annotated documentation generation for source code, distribution estimation, and data conversion.

2.2 Graphical User Interfaces

WEKA's functionality can be accessed through various graphical user interfaces, principally the Explorer, and Experimenter interfaces shown in Figure 9, but also the Knowledge Flow interface. The most popular interface, the Explorer, allows quick exploration of data and supports all the main items mentioned above—data loading and filtering, classification, clustering, attribute selection and various forms of visualization—in an interactive fashion.

The Experimenter is a tool for setting up machine learning experiments that evaluate classification and regression methods. It allows easy comparison of performance, and can tabulate summaries in ways that are easy to incorporate into publications. Experiments can be set up to run in parallel over different computers in a network so that multiple repetitions of cross validation (the default method of performance analysis) can be distributed over multiple machines.



Filter: None

Current relation
Relation: cpu   Attributes: 8
Instances: 209   Sum of weights: 209

Attributes

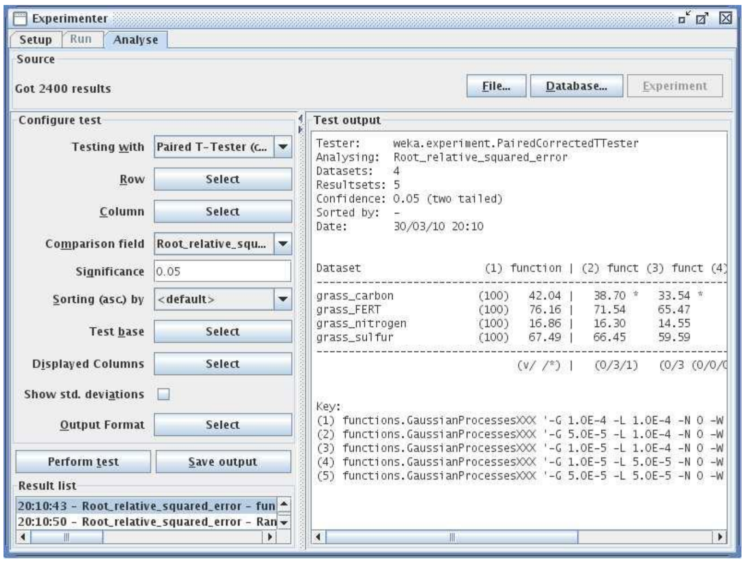
No.	Name
1	vendor
2	MYCT
3	MMIN
4	MMAX
5	CACH
6	CHMIN
7	CHMAX
8	class

Selected attribute
Name: vendor   Type: Nominal
Missing: 0 (0%)   Distinct: 30   Unique: 4 (2%)

No.	Label	Count	Weight
1	adviser	1	1.0
2	amdahl	9	9.0
3	apollo	2	2.0
4	basf	2	2.0
5	bti	2	2.0
6	burroughs	8	8.0
7	c.r.d	6	6.0
8	cdc	9	9.0

Class: class (Num)

Status
OK



Source
Got 2400 results

Configure test
Testing with: Paired T-Tester (c...
Comparison field: Root_relative_squ...
Significance: 0.05
Sorting (asc.) by: <default>
Show std. deviations: ☐

Result list
20:10:43 - Root_relative_squared_error - fun
20:10:50 - Root_relative_squared_error - Ran

Test output

```
Tester:     weka.experiment.PairedCorrectedTTester
Analysing:  Root_relative_squared_error
Datasets:   4
Resultsets: 5
Confidence: 0.05 (two tailed)
Sorted by:  -
Date:       30/03/10 20:10


Dataset                   (1) function | (2) funct (3) funct (4)
------------------------------------------------------------------
grass_carbon            (100)   42.04 |   38.70 *   33.54 *
grass_FERT              (100)   76.16 |   71.54     65.47
grass_nitrogen          (100)   16.86 |   16.30     14.55
grass_sulfur            (100)   67.49 |   66.45     59.59
------------------------------------------------------------------
                                (v/ /*) |   (0/3/1)   (0/3 (0/0/0


Key:
(1) functions.GaussianProcessesXXX '-G 1.0E-4 -L 1.0E-4 -N 0 -W
(2) functions.GaussianProcessesXXX '-G 5.0E-5 -L 1.0E-4 -N 0 -W
(3) functions.GaussianProcessesXXX '-G 1.0E-5 -L 1.0E-4 -N 0 -W
(4) functions.GaussianProcessesXXX '-G 1.0E-5 -L 5.0E-5 -N 0 -W
(5) functions.GaussianProcessesXXX '-G 5.0E-5 -L 5.0E-5 -N 0 -W
```

Figure 9: The Explorer and Experimenter interfaces.

The Knowledge Flow interface is a Java Beans application that allows the same kind of data exploration, processing and visualization as the Explorer (along with some extras), but in a workflow-oriented system. The user can define a workflow specifying how data is loaded, pre-processed, evaluated and visualized, which can be repeated multiple times. This makes it easy to optimize the workflow by tweaking parameters of algorithms, or to apply it to other data sources. In the Explorer, on the other hand, the individual steps must be invoked manually, one at a time. This is a rather tedious process, and is prone to errors such as omitting pre-processing steps.

WEKA also includes some specialized graphical interfaces, such as a Bayes network editor that focuses on Bayes network learning and inference, an SQL viewer for interaction with databases, and an ARFF data file viewer and editor.

All functionality and some more specialized functions can be accessed from a command line interface, so WEKA can be used without a windowing system.

2.3 Extending WEKA

One of WEKA's major strengths is that it is easily extended with customized or new classifiers, clusterers, attribute selection methods, and other components. For instance, all that is needed to add a new classifier is a class that derives from the Classifier class and implements the buildClassifier method for learning, and a classifyInstance method for testing/predicting the value for a data point. The code fragment in Figure 2 shows a minimal implementation of a classifier that returns the mean or mode of the class in the training set (double values are used to store indices of nominal attribute values).

Any new class is picked up by the graphical user interfaces through Java introspection: no further coding is needed to deploy it from WEKA's graphical user interfaces. This makes it easy to evaluate how new algorithms perform ompared to any of the existing ones, whichexplainsWEKA's popularity among machine learning researchers.

Besides being easy to extend, WEKA includes a wide range of support for adding functionality to basic implementations. For instance, a classifier can have various optional settings by implementing a pre-defined interface for option handling. Each option can be documented using a tool-tip text method, which is picked up by the help dialogs of the graphical user interfaces. Classes typically implement methods described in papers, and this provenance metadata can be captured in a method that is used to generate documentation. Some methods only apply to certain kinds of data, such as numeric class

values or discrete attribute values. A 'capabilities' mechanism allows classes to identify what kind of data is acceptable by any method, and the graphical user interfaces incorporate this by making methods available only if they are able to process the data at hand.

2.5 Naive Bayes Classification Algorithm

Naive Bayes is a simple probabilistic classifier based on applying Bayes' theorem (or Bayes's rule) with strong independence (naive) assumptions.

Explanation of Bayes's rule:

Bayes's rule: $P(H|E) = \frac{P(E|H)*P(H)}{P(E)}$

The basic idea of Bayes's rule is that the outcome of a hypothesis or an event (H) can be predicted based on some evidences (E) that can be observed. From Bayes's rule, we have

(1) A priori probability of H or P(H): This is the probability of an event before the evidence is observed.

(2) A posterior probability of H or P(H | E): This is the probability of an event after the evidence is observed.

Example 1: To predict the chance or the probability of raining, we usually use some evidences such as the amount of dark cloud in the area.

Let H be the event of raining and E be the evidence of dark cloud, then we have

$$\mathrm{P(raining \mid dark\ cloud)} = \frac{P(dark\ cloud \mid raining)\ x\ P(raining)}{\mathrm{P(dark\ cloud)}}$$

–P (dark cloud | raining) is the probability that there is dark cloud when it rains. Of course, “dark cloud” could occur in many other events such as overcast day or forest fire, but we only consider “dark cloud” in the context of event “raining”. This probability can be obtained from historical data recorded by some meteorologists.
–P (raining) is the priori probability of raining. This probability can be obtained from statistical record, for example, the number of rainy days throughout a year.
–P (dark cloud) is the probability of the evidence “dark cloud” occurring. Again, this can be obtained from the statistical records, but the evidence is not usually well recorded compared to the main event. Therefore, sometimes the full evidence, i.e., P(dark cloud), is hard to obtain.

Explanation of Naive Bayes:

As you can see from Example 1, we can predict an outcome of some events by observing some evidences. Generally, it is “better” to have more than one evidence to support the prediction of an event. Typically, the more evidences we can gather, the better the classification accuracy can be obtained. However, the evidence must relate to the event (must make sense). For example, if you add an evidence of “earthquake” to Example 1, the above model might yield worse performance. This is since “raining” is not related to the evidence of “earthquake”, i.e., if there is an earthquake, it doesn't mean that it will rain.

Suppose we have more than one evidence for building our NB model, we could run into a problem of dependencies, i.e., some evidence may depend on one or more of other evidences. For example, the evidence “dark cloud” directly depends on the evidence “high humidity”. However, including dependencies into the model will make it very complicated. This is because one evidence could depend on many other evidences. To make our life easier, we make an assumption that all evidences are independent of each other (this is why we call the model “naive”).

Bayes's rule for multiple evidences::

$$\mathrm{P(H \mid E1, E2, \ldots, En)} = \mathrm{P}\frac{\mathrm{(\,E1, E2, \ldots, En \mid H)\ x\ P(H)}}{\mathrm{P(E1, E2, \ldots, En)}}$$

With the independence assumption, we can rewrite the Bayes's rule as follows:

$$P(H \mid E1, E2, \ldots, En) = \frac{P(\,E1 \mid H)\ x\ P(\,E2 \mid H)\ x \ldots P(\,En \mid H)\ x\ P(H)}{\mathrm{P(E1, E2, \ldots, En)}}$$

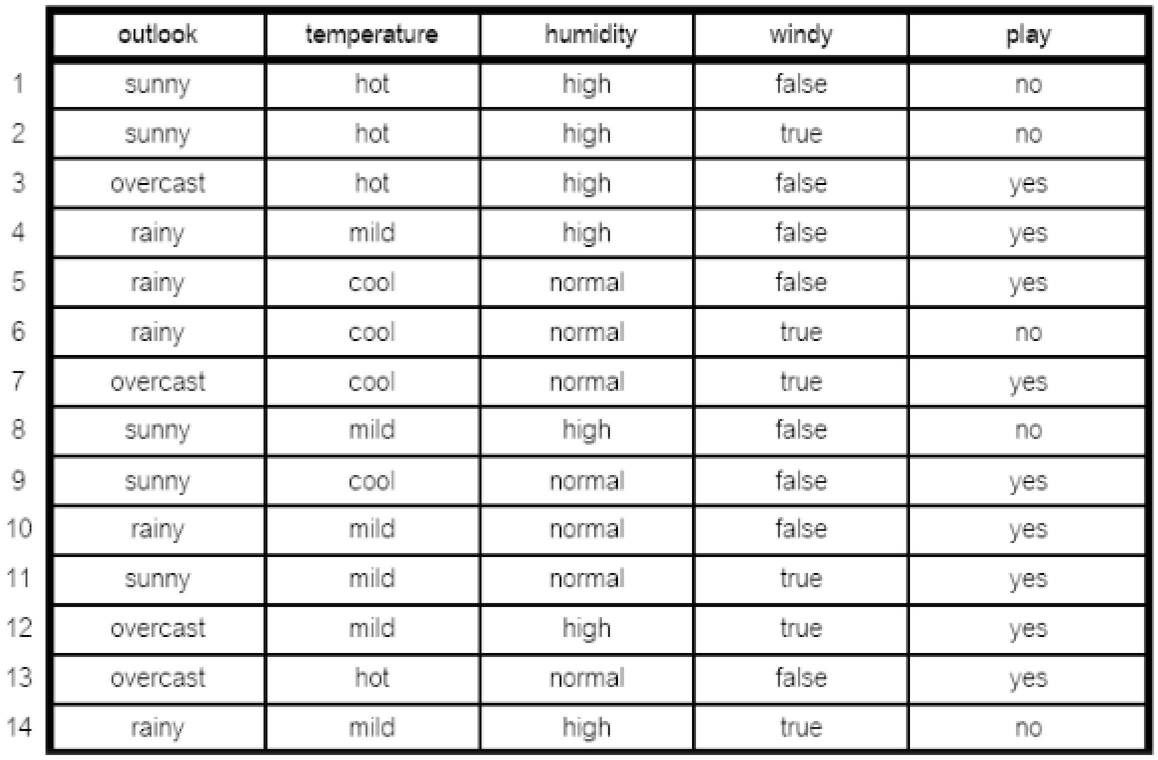
Example 2:

From Example 1, we could have the following NB model for raining,

$$P(raining \mid dark\ cloud, wind\ speed, humidity) = \frac{P(dark\ cloud \mid raining)\ x\ P(wind\ speed \mid raining)\ x\ P(humidity \mid raining)\ x\ P(raining)}{\mathrm{P(dark\ cloud, wind\ speed, humidity)}}$$

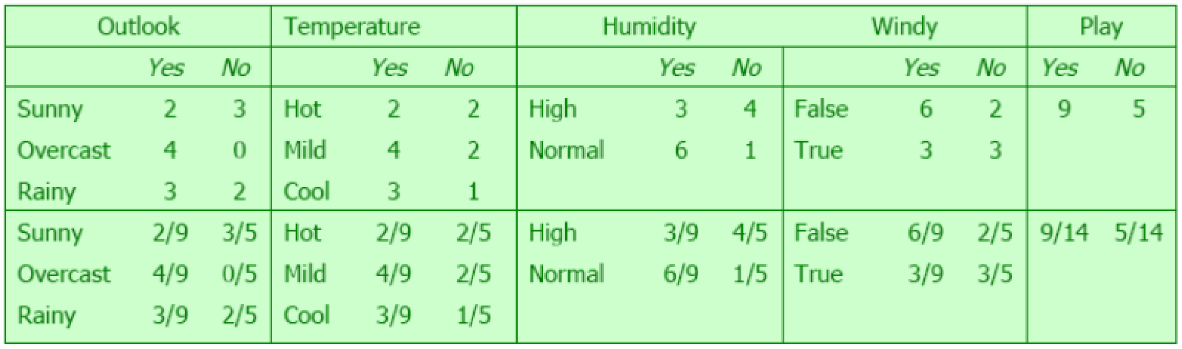
Example 3:

To understand how to build an NB model, let's use the example from our lecture slides (Chapter 3 of the data mining book). Given the weather data set for predicting play condition. There are 14 instances (or examples) and 5 attributes. All attributes are nominal.



	outlook	temperature	humidity	windy	play
1	sunny	hot	high	false	no
2	sunny	hot	high	true	no
3	overcast	hot	high	false	yes
4	rainy	mild	high	false	yes
5	rainy	cool	normal	false	yes
6	rainy	cool	normal	true	no
7	overcast	cool	normal	true	yes
8	sunny	mild	high	false	no
9	sunny	cool	normal	false	yes
10	rainy	mild	normal	false	yes
11	sunny	mild	normal	true	yes
12	overcast	mild	high	true	yes
13	overcast	hot	normal	false	yes
14	rainy	mild	high	true	no

We need to build the NB model from the given above data set. The result are shown below:

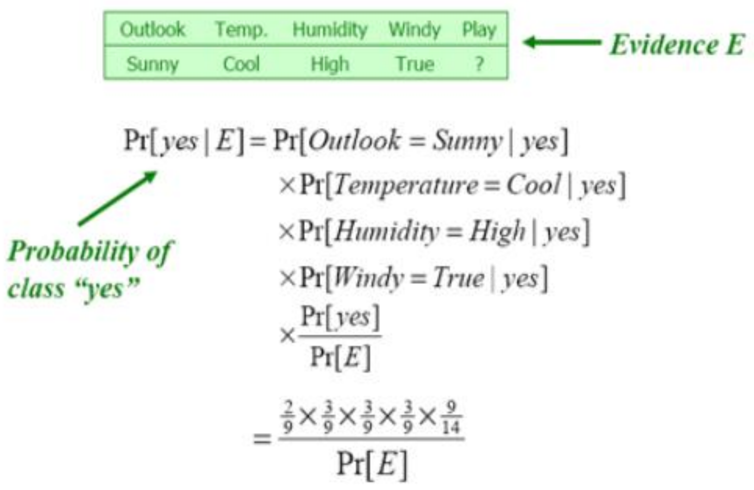


Outlook			Temperature			Humidity			Windy			Play	
	Yes	*No*		*Yes*	*No*		*Yes*	*No*		*Yes*	*No*	*Yes*	*No*
Sunny	2	3	Hot	2	2	High	3	4	False	6	2	9	5
Overcast	4	0	Mild	4	2	Normal	6	1	True	3	3		
Rainy	3	2	Cool	3	1								
Sunny	2/9	3/5	Hot	2/9	2/5	High	3/9	4/5	False	6/9	2/5	9/14	5/14
Overcast	4/9	0/5	Mild	4/9	2/5	Normal	6/9	1/5	True	3/9	3/5		
Rainy	3/9	2/5	Cool	3/9	1/5								

The top part of the table contains the frequency of different evidences. For example, there are 2 instances (examples) from the data set showing (outlook=sunny) when (play=yes). Once you have finished counting all frequency, we need to build the NB model by calculating all P(E | H) and P(H). For example, P(outlook=sunny | play=yes) = 2/9

P(play=yes) = 9/14

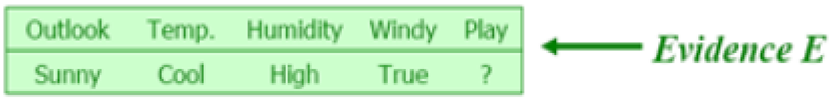
Once we have the NB model, we can use it to predict the event "play" based on different set of evidences. For example, if we observe (outlook=sunny), (temperature=cool), (humidity=high) and (windy=true), then we can estimate the posterior probability as follows:



Outlook	Temp.	Humidity	Windy	Play
Sunny	Cool	High	True	?

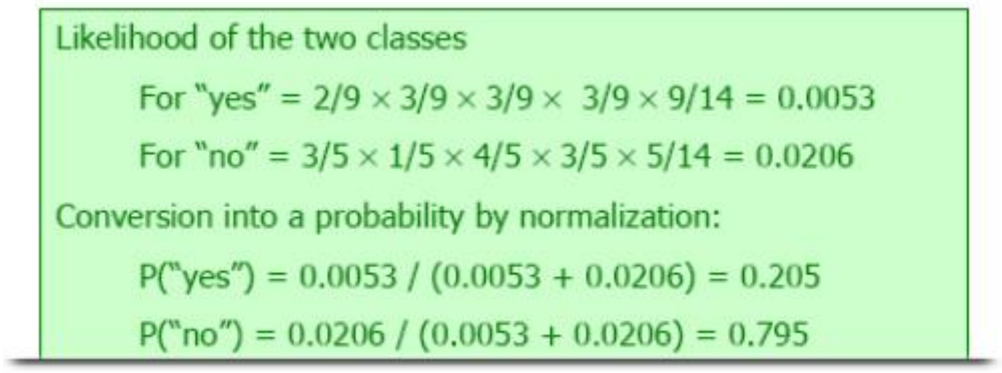
$$
\begin{aligned}
Pr[yes \mid E] = {} & Pr[Outlook = Sunny \mid yes] \\
& \times Pr[Temperature = Cool \mid yes] \\
& \times Pr[Humidity = High \mid yes] \\
& \times Pr[Windy = True \mid yes] \\
& \times \frac{Pr[yes]}{Pr[E]} \\
= {} & \frac{\frac{2}{9} \times \frac{3}{9} \times \frac{3}{9} \times \frac{3}{9} \times \frac{9}{14}}{Pr[E]}
\end{aligned}
$$

Evidence E
Probability of class "yes"



Outlook	Temp.	Humidity	Windy	Play
Sunny	Cool	High	True	?

We can ignore Pr(E) because we only need to "relatively" compare the value to other class. Therefore we have the following results:



Likelihood of the two classes

For "yes" = 2/9 × 3/9 × 3/9 × 3/9 × 9/14 = 0.0053

For "no" = 3/5 × 1/5 × 4/5 × 3/5 × 5/14 = 0.0206

Conversion into a probability by normalization:

P("yes") = 0.0053 / (0.0053 + 0.0206) = 0.205

P("no") = 0.0206 / (0.0053 + 0.0206) = 0.795

2.6 J48 Decision Tree Analysis using WEKA

There exist a number of prominent Machine Learning algorithms used in modern computing applications. These algorithms have been engineered over the last several decades, and many are open-source and available for public usage and modification. A common application for these algorithms often involves decision-based classification and adaptive learning over a training set. The decision tree is a popular utility for implementing such tactics. A decision tree is a decision-modelling tool that graphically displays the classification process of a given input for given output class labels.

Weka is an open-source Java application produced by the University of Waikato in New Zealand. This software bundle features an interface through which many of the aforementioned algorithms (including decision trees) can be utilized on preformatted data sets. Using this interface, several test-domains were experimented with to gain insight on the effectiveness of different methods of pruning an algorithmically induced decision tree.

Pruning decision trees is a fundamental step in optimizing the computational efficiency as well as classification accuracy of such a model. Applying pruning methods to a tree usually results in reducing the size of the tree (or the number of nodes) to avoid unnecessary complexity, and to avoid over-fitting of the data set when classifying new data. For example, when classifying an example in a decision tree and reaching a certain node, there are two possible outcomes: positive and negative. Imagine that running a testing set through this node shows that 95% of the examples are evaluated as positive. The assumption is made that this test node can be replaced entirely with a positive classifier. Depending on the percentage of total error caused by pruning, however, the noisy data incurred may not be deemed negligible depending on the tolerance from the data set (and/or a sufficiently dominant consistency of classification by the test node). Such a replacement is illustrated in Figure 10. A smaller sub tree can be inserted instead of replacing a test node with a class label to prune a decision tree as well. In reference to Figure 1, this would involve replacing a node such as ‘t2’ with a smaller test node, like ‘t5’. In either case, the resulting tree is smaller and ideally more efficient than the pre-pruned tree. The process of pruning traditionally begins from the bottom of the tree (at the child leaves), and propagates upwards.

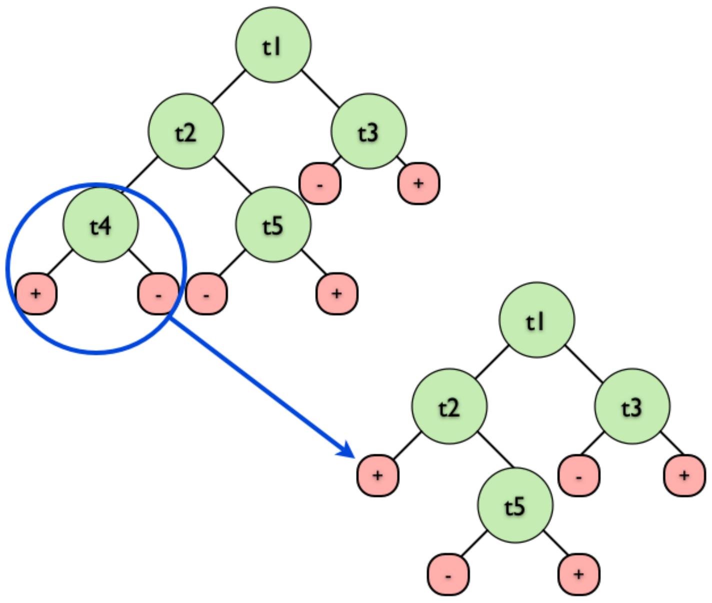


Figure 10: A decision tree's test node (t4) is replaced in entirety with a positive

classification (assuming a binary class domain).

Pruning Methods:

Two pruning methods: the post-pruning method, and the online pruning method. Each method has strengths and weaknesses, yet their largest differences concern the timing of implementation, and the parameters they consider for node removal. In either case, a decision tree must be in the process of being induced, or completely created for pruning to occur (depending on the method). The overlying principle of pruning is to compare the amount of error that a decision tree would suffer before and after each possible prune, and to then decide accordingly to maximally avoid error.

Post-pruning is implemented on a fully induced decision tree, and sifts through to remove statistically insignificant nodes. Working from the bottom up, the probability (or relative frequency) of sibling leaf nodes will be compared, and any overwhelming dominance of a certain leaf node will result a pruning of that node in one of several ways. The error estimate of each child node is calculated and used to derive the total error of the parent node. The parent node is then pruned according to the relative frequencies of the child nodes, and this replacement node's error is compared with that of the old parent node (which was influenced by the child nodes' error). This comparison will dictate whether or not pruning is advantageous at a given node.

Online-pruning differs from post-pruning in that it operates on the decision tree while is being induced. To create the decision tree, contemporary algorithms divide the data set on the attributes that provide the most information gain about the class label; this dividing factor serves as the deciding attribute for each test node. Whenever a split is made which yields a child leaf that represents less than a minimum number of examples from the data set, the parent node and its children are compressed into a single node. This process continues throughout the creation of the entire tree. Figure 11 exemplifies how at a given split (marked at ‘t4’), the minimum number of data points needed to represent a class label is not met. Weakness of the positive class (in this example) leads to the negative classifier replacing the parent node ‘t4’; this takes place as the tree is populated algorithmically.

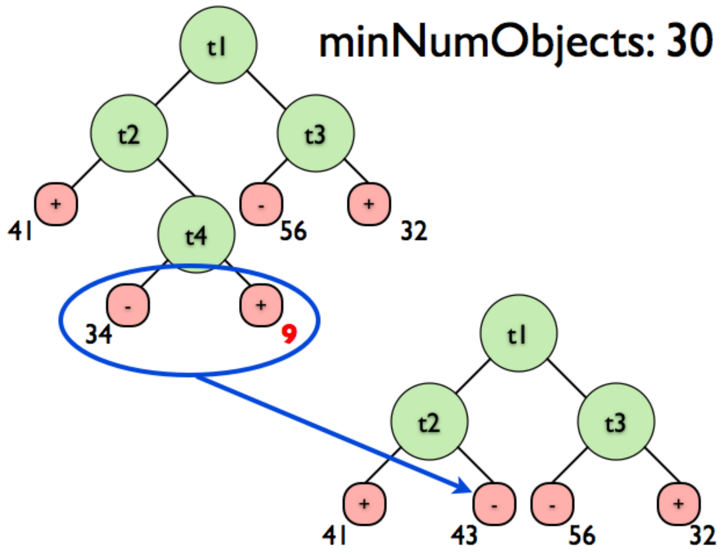


Figure 11: Replacing a test node that meets the requirements for online pruning due to a leaf with less than the minNumObjects classified items.

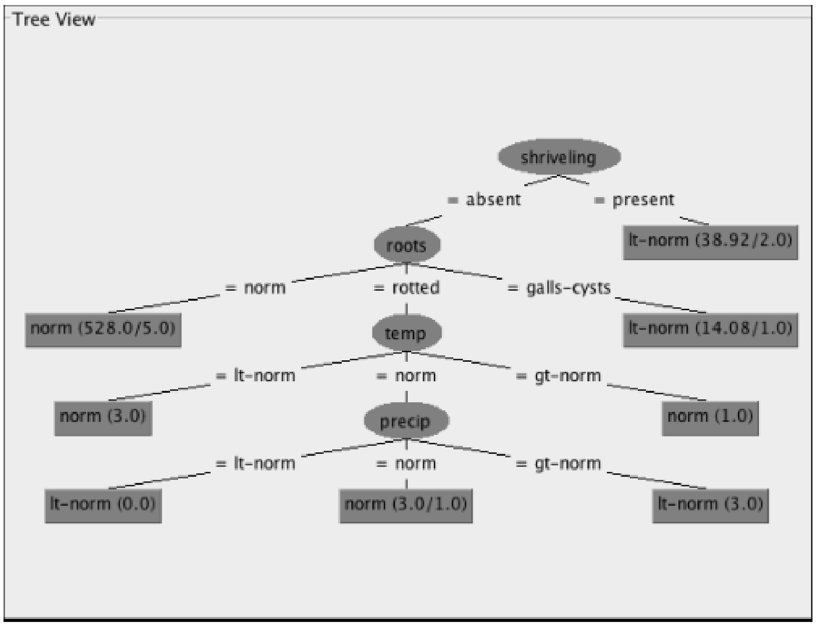
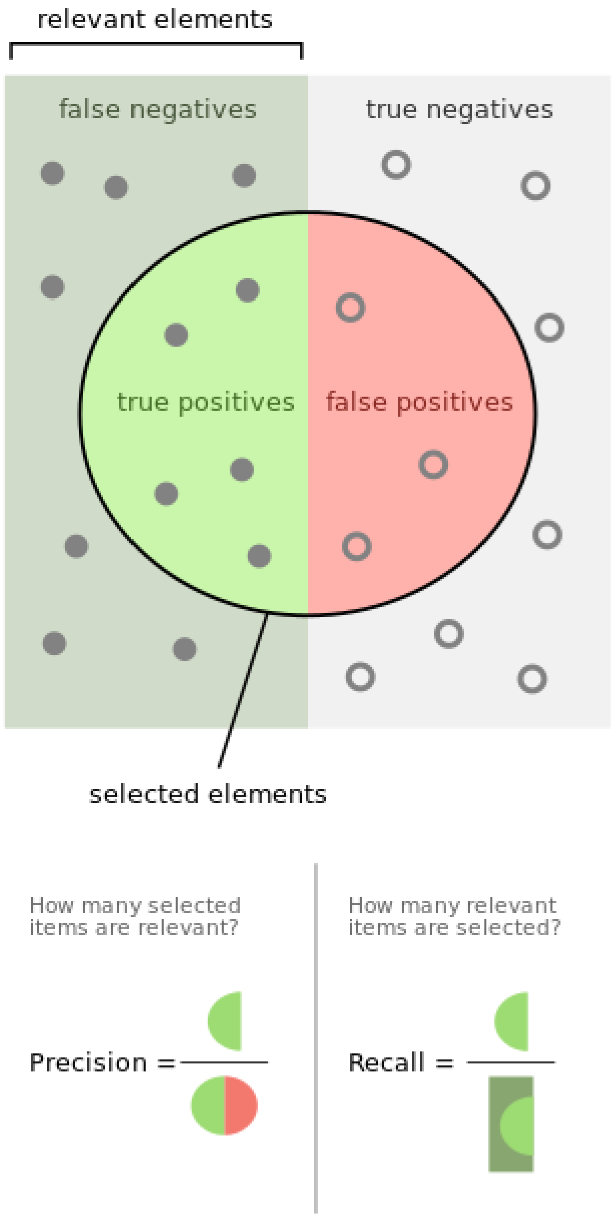


Figure 12: A tree visualization from one of the sample domains in Weka.

2.7 Precision and Recall



In pattern recognition and information retrieval with binary classification, precision (also called positive predictive value) is the fraction of retrieved instances that are relevant, while recall (also known as sensitivity) is the fraction of relevant instances that are retrieved. Both precision and recall are therefore based on an understanding and measure of relevance. Suppose a computer program for recognizing dogs in scenes from a video identifies 7 dogs in a scene containing 9 dogs and some cats. If 4 of the identifications are correct, but 3 are actually cats, the program's precision is 4/7 while its recall is 4/9. When a search engine returns 30 pages only 20 of which were relevant while failing to return 40 additional relevant pages, its precision is 20/30 = 2/3 while its recall is 20/60 = 1/3.

So, in this case, precision is "how useful the search results are", and recall is "how complete the results are".

In simple terms, high precision means that an algorithm returned substantially more relevant results than irrelevant, while high recall means that an algorithm returned most of the relevant results.

Precision and recall are then defined as:

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

Recall in this context is also referred to as the true positive rate or sensitivity, and precision is also referred to as positive predictive value (PPV); other related measures used in classification include true negative rate and accuracy. True negative rate is also called specificity.

$$\text{True negative rate} = \frac{tn}{tn + fp}$$

$$\text{Accuracy} = \frac{tp + tn}{tp + tn + fp + fn}$$

F-measure

A measure that combines precision and recall is the harmonic mean of precision and recall, the traditional F-measure or balanced F-score:

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

3. SYSTEM DEVELOPMENT

Existing System:

There is no method to determine the end prices of online auctions (on eBay). Each day, many items are put on auctions and many are sold. Our project aims at predicting the price of these products depending upon the attributes which are filled by the user regarding the products. From these attributes, we determine derived attributes in order to precisely predict the prices.

3.1 System Implemented:

In our project we determine:

- **Determine whether an auction listing will result in a sale:** An auction can result in either the sale of the product or not. In this part of the report we predict the status of the product sale.
- **Predict final sale prices for auctions:** In this, we predict the end price of a product. If a product results in a sale, then the final price would be predicted using previous trends.

Analysis:

In order to predict the end price of online auctions we would need to create data entries of the already sold products. After the creation of dataset we would need to work on WEKA and ANFIS and compare their results. We require the system with the following:

<u>MATLAB (ANFIS):</u>

OS: Windows/ MAC OS X/ Linux

RAM: 2GB or higher

No specific graphics card required.

Processor: Any Intel x86-64 processor

WEKA:

The following matrix shows what Java version is necessary to run a specific Weka version. It also specifies when a changeover happened, listing date and **Subversion** revision (the number prefixed with *r*).

		Java				
		1.4	**1.5**	**1.6**	**1.7**	**1.8**
WEKA	**<3.4.0**	X	X	X	X	X
	3.4.x	X	X	X	X	X
	3.5.x	3.5.0-3.5.2	>3.5.2 r2892, 20/02/2006	X	X	X
	3.6.x		X	X	X	X
	3.7.x		3.7.0	>3.7.0 r5678, 25/06/2009	>3.7.13	X

Table 2: WEKA Requirements

Note that Java 5.0 and later in combination with Linux/Gnome has problems with the default Look'n'Feel. A workaround for the problem was introduced with version 3.4.5/3.5.0. For more information refer to the **Troubleshooting Wiki** article. From Weka 3.6.5/3.7.4 Mac OS X users will need Java for Mac OS X 10.6 Update 3 (java 1.6.0_22) or later.

Flow Chart:

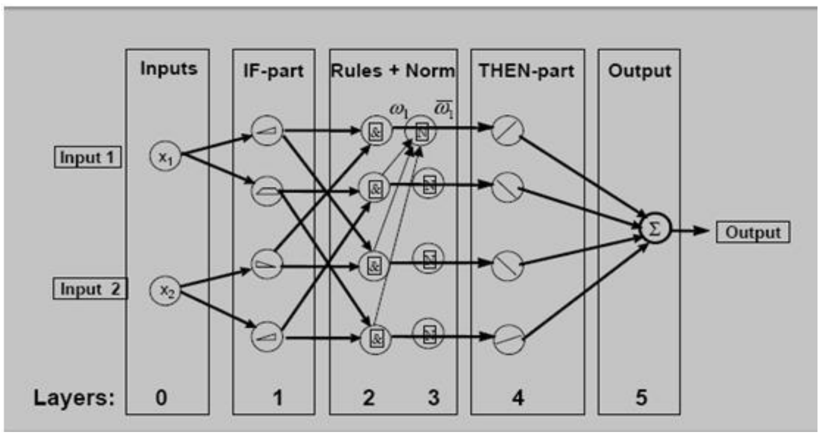


Fig 13: ANFIS Flow Chart

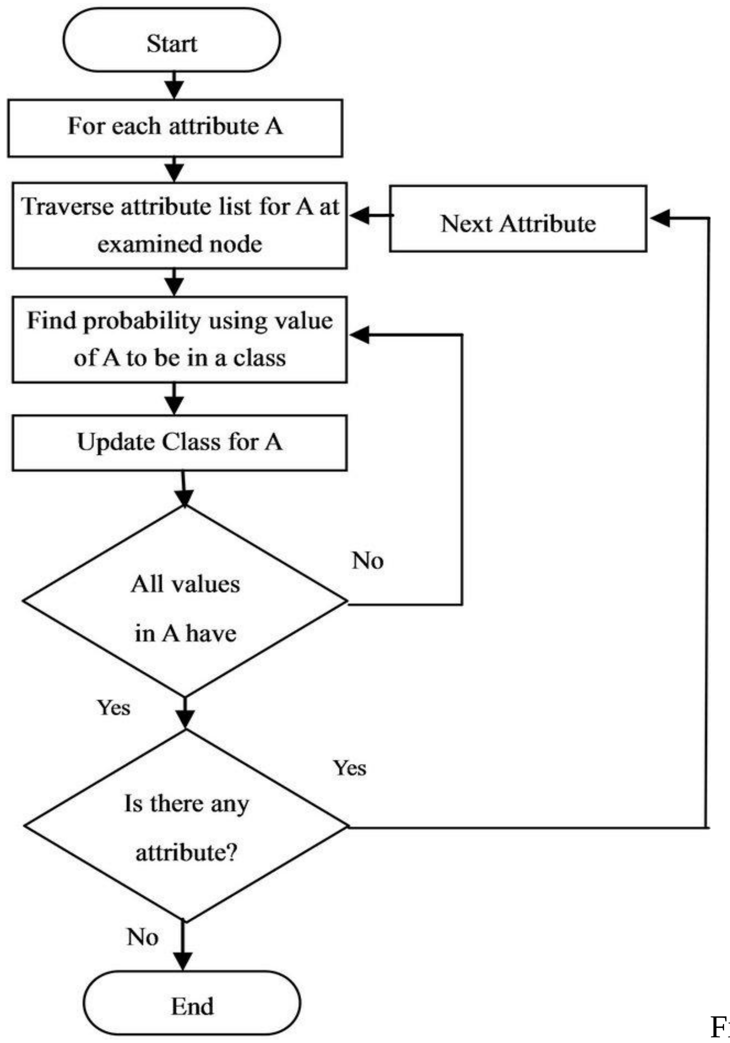


Fig 14: NaiveBayes Flow Chart

3.2 Data Set Used

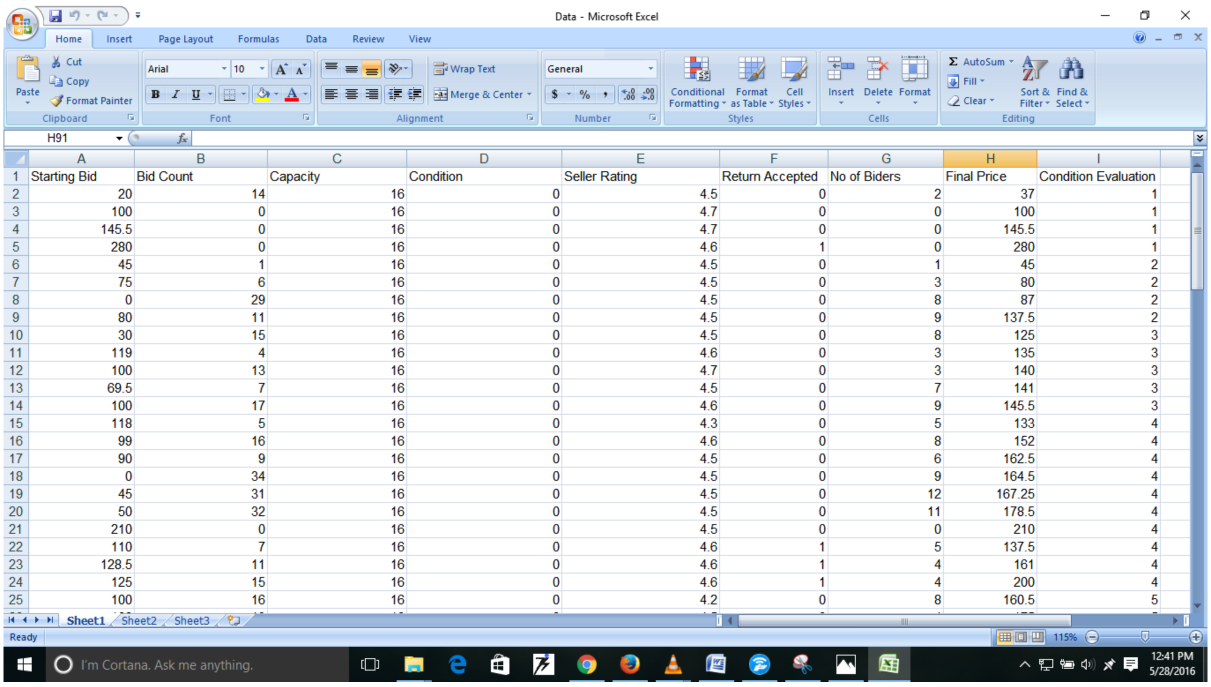
In this part of development we study the previous trends and try to find out the attributes.

Following are the attributes:

1. Starting Bid: It is the minimum bid for the auction.
2. Bid Count: It is the number of bids made for the auction.
3. Capacity: It is the memory size of iPhone 5s.
4. Condition: 0→ USED, 1→NEW
5. Seller Rating: Sellers eBay Rating
6. Return Accepted: Whether the seller accepts the returns. It is represented by 0 or 1.
7. No: of bidders: It is the number of bidders involved during the auction of an item.
8. Condition Evaluation: Attribute defining the condition of the product.
   1→ iPhone not working.
   2→ Two or more components of the iPhone are not working.
   3→One component of Iphone not working

4→Iphone having scratches
5→Iphone in new condition

9. Final Price: End price of Iphone 5s at which it is auctioned.



	A	B	C	D	E	F	G	H	I
1	Starting Bid	Bid Count	Capacity	Condition	Seller Rating	Return Accepted	No of Biders	Final Price	Condition Evaluation
2	20	14	16	0	4.5	0	2	37	1
3	100	0	16	0	4.7	0	0	100	1
4	145.5	0	16	0	4.7	0	0	145.5	1
5	280	0	16	0	4.6	1	0	280	1
6	45	1	16	0	4.5	0	1	45	2
7	75	6	16	0	4.5	0	3	80	2
8	0	29	16	0	4.5	0	8	87	2
9	80	11	16	0	4.5	0	9	137.5	2
10	30	15	16	0	4.5	0	8	125	3
11	119	4	16	0	4.6	0	3	135	3
12	100	13	16	0	4.7	0	3	140	3
13	69.5	7	16	0	4.5	0	7	141	3
14	100	17	16	0	4.6	0	9	145.5	3
15	118	5	16	0	4.3	0	5	133	4
16	99	16	16	0	4.6	0	8	152	4
17	90	9	16	0	4.5	0	6	162.5	4
18	0	34	16	0	4.5	0	9	164.5	4
19	45	31	16	0	4.5	0	12	167.25	4
20	50	32	16	0	4.5	0	11	178.5	4
21	210	0	16	0	4.5	0	0	210	4
22	110	7	16	0	4.6	1	5	137.5	4
23	128.5	11	16	0	4.6	1	4	161	4
24	125	15	16	0	4.6	1	4	200	4
25	100	16	16	0	4.2	0	8	160.5	5

Fig 15: Data Set Used

3.3 Performance Parameters

1. *WEKA-Precision, Recall And FMeasure*

$$\text{Precision} = \frac{tp}{tp + fp}$$

$$\text{Recall} = \frac{tp}{tp + fn}$$

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Where: Tp-true positive Fp-false positive Fn-false negative

2. *Errors:*

An 'error' is a deviation from accuracy or correctness. A 'mistake' is an error caused by a fault: the fault being misjudgment, carelessness, or forgetfulness.

Types of Errors:

2.1 *Relative Error or Fractional Error*

It is defined as the ratio of the error and the specified magnitude of the quantity. Mathematically we write as,

$$Relative\ Error\ =\ \frac{dA}{A}$$

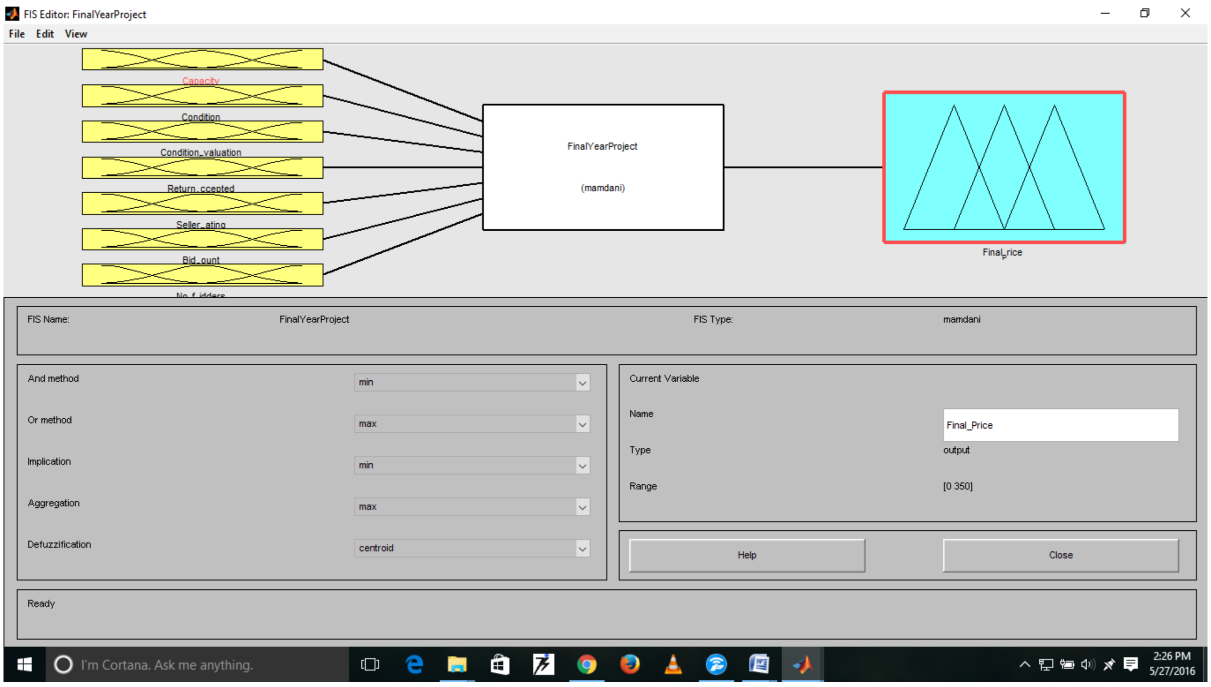
Where dA is the error and A is the magnitude.

2.2 *Root Mean Square Deviation*

The root-mean-square deviation (RMSD) or root-mean-square error (RMSE) is a frequently used measure of the differences between values (sample and population values) predicted by a model or an estimator and the values actually observed. The RMSD represents the sample standard deviation of the differences between predicted values and observed values. These individual differences are called residuals when the calculations are performed over the data sample that was used for estimation, and are called *prediction errors* when computed out-of-sample.

$$\text{RMSE}= \sqrt{\frac{1}{n}\sum_{j=1}^{n}(predicted\ value - actual\ value)^2}$$

3.4 Snapshots of System Designed



FIS Name: FinalYearProject

FIS Type: mamdani

And method: min

Or method: max

Implication: min

Aggregation: max

Defuzzification: centroid

Current Variable

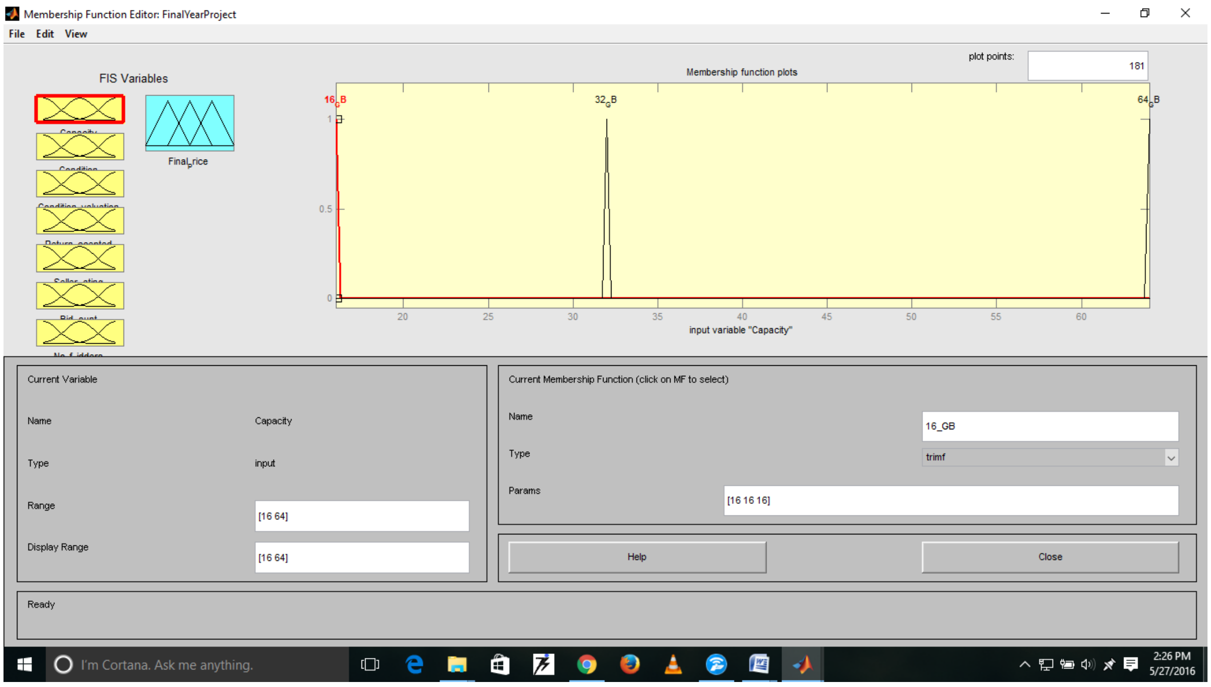
Name: Final_Price

Type: output

Range: [0 350]

Ready

Fig 16: Fuzzy Toolbox



plot points: 181

Current Variable

Name: Capacity

Type: input

Range: [16 64]

Display Range: [16 64]

Current Membership Function (click on MF to select)

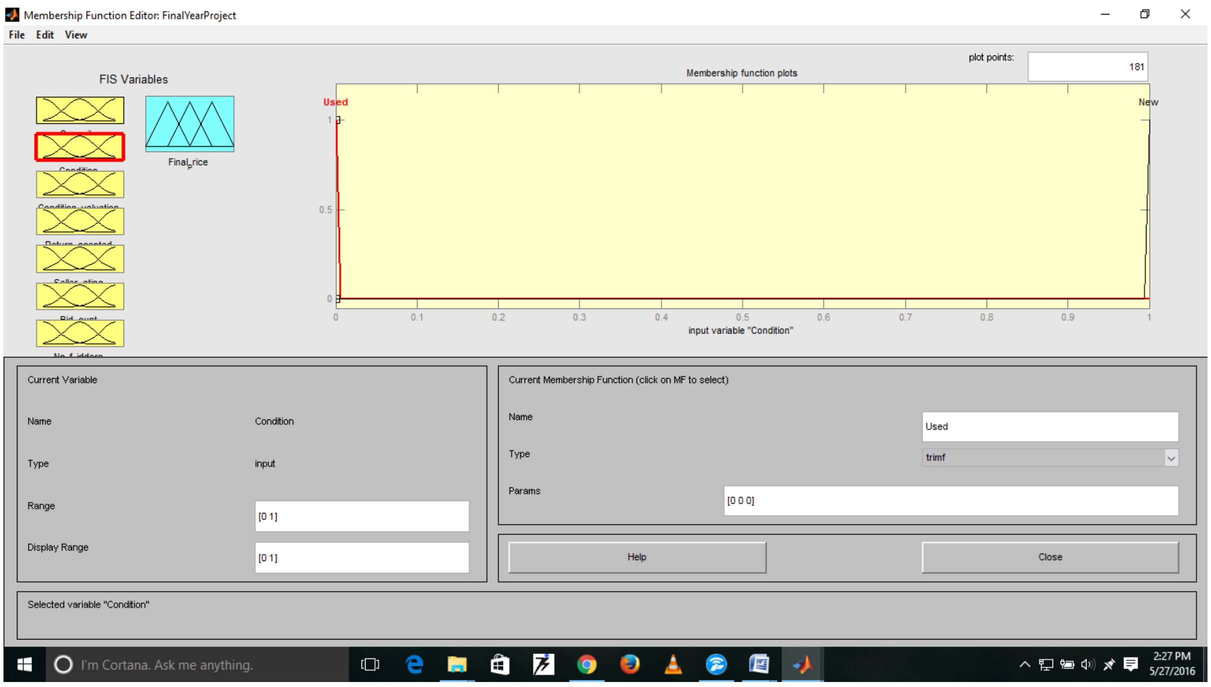
Name: 16_GB

Type: trimf

Params: [16 16 16]

Ready

Fig 17: Capacity


Membership Function Editor: FinalYearProject

Current Variable

Name: Condition

Type: input

Range: [0 1]

Display Range: [0 1]

Current Membership Function (click on MF to select)

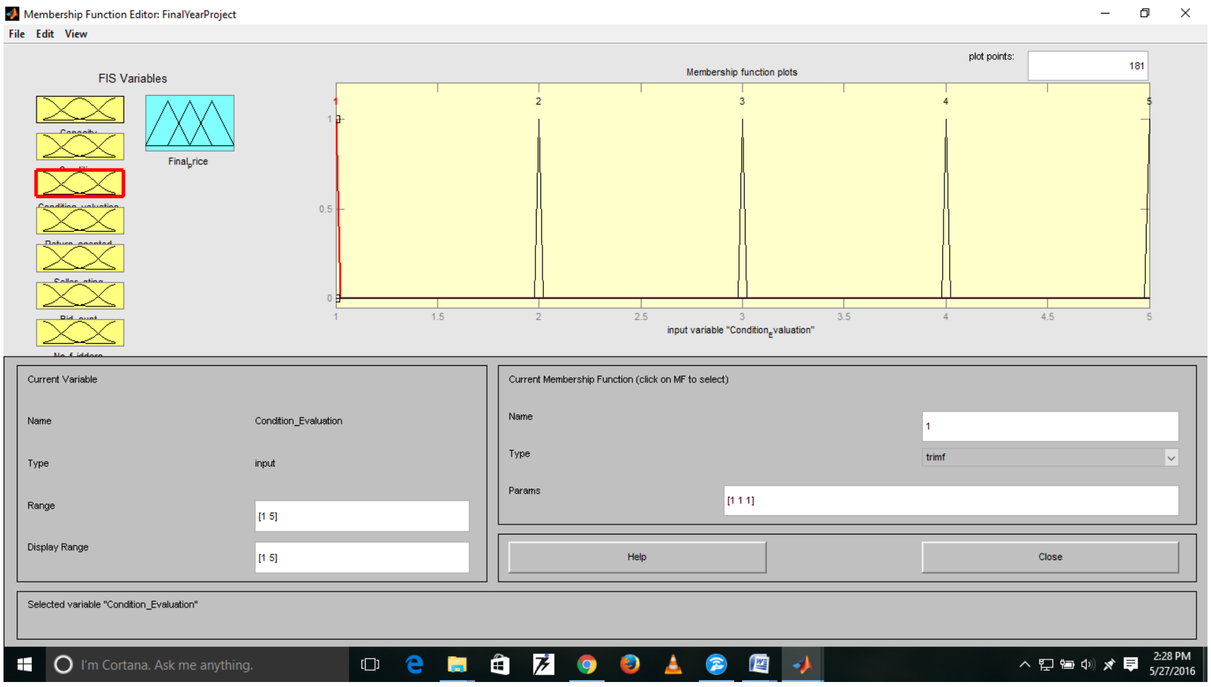
Name: Used

Type: trimf

Params: [0 0 0]

Selected variable "Condition"

Fig 18: Condition


Membership Function Editor: FinalYearProject

Current Variable

Name: Condition_Evaluation

Type: input

Range: [1 5]

Display Range: [1 5]

Current Membership Function (click on MF to select)

Name: 1

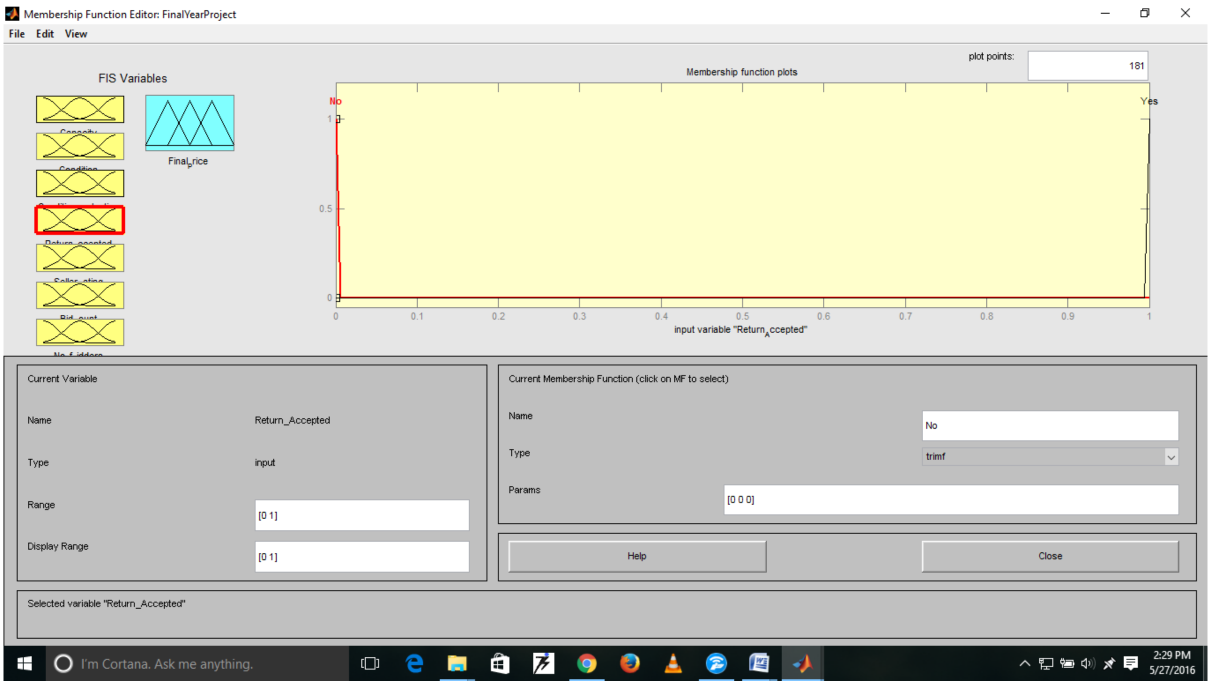
Type: trimf

Params: [1 1 1]

Selected variable "Condition_Evaluation"

File
Edit
View
FIS Variables
Final_rice
Membership function plots
plot points:
181
input variable "Condition_Evaluation"
Help
Close

Fig 19: Condition Evaluation


Membership Function Editor: FinalYearProject
File Edit View
FIS Variables
Final_price
Membership function plots
No
Yes
input variable "Return_Accepted"
Help
Close
I'm Cortana. Ask me anything.
2:29 PM
5/27/2016

plot points: 181

Current Variable

Name: Return_Accepted

Type: input

Range: [0 1]

Display Range: [0 1]

Current Membership Function (click on MF to select)

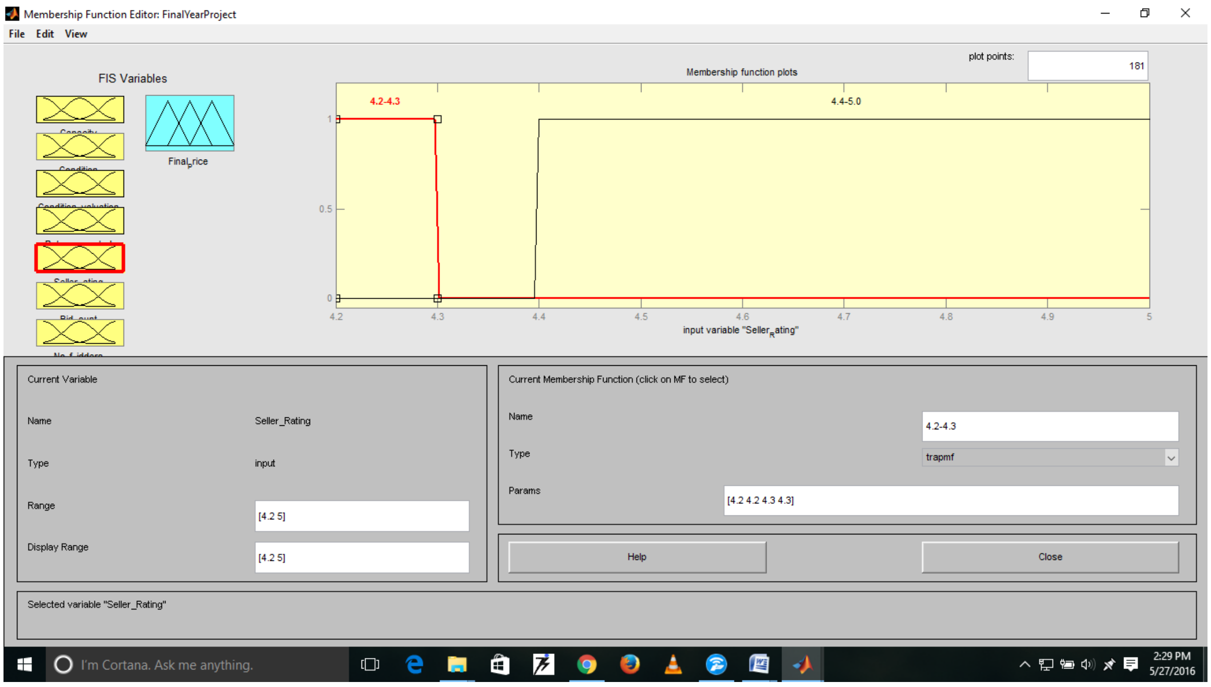
Name: No

Type: trimf

Params: [0 0 0]

Selected variable "Return_Accepted"

Fig 20: Return Accepted


Membership Function Editor: FinalYearProject
File Edit View
FIS Variables
Final_price
Membership function plots
4.2-4.3
4.4-5.0
input variable "Seller_Rating"
Help
Close
I'm Cortana. Ask me anything.
2:29 PM
5/27/2016

plot points: 181

Current Variable

Name: Seller_Rating

Type: input

Range: [4.2 5]

Display Range: [4.2 5]

Current Membership Function (click on MF to select)

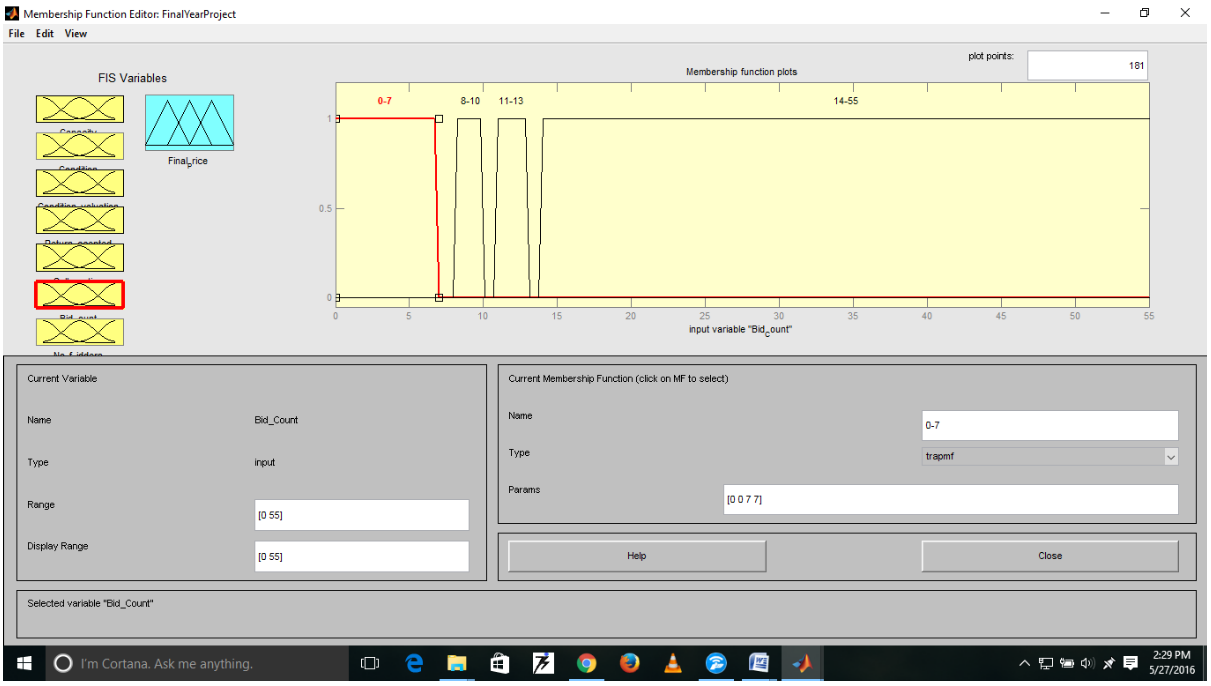
Name: 4.2-4.3

Type: trapmf

Params: [4.2 4.2 4.3 4.3]

Selected variable "Seller_Rating"

Fig 21: Seller Rating


Membership Function Editor: FinalYearProject
File
Edit
View
FIS Variables
Final_price
Membership function plots
plot points:
181
0-7
8-10
11-13
14-55
input variable "Bid_count"
Help
Close
I'm Cortana. Ask me anything.
2:29 PM
5/27/2016

Current Variable

Name Bid_Count

Type input

Range [0 55]

Display Range [0 55]

Current Membership Function (click on MF to select)

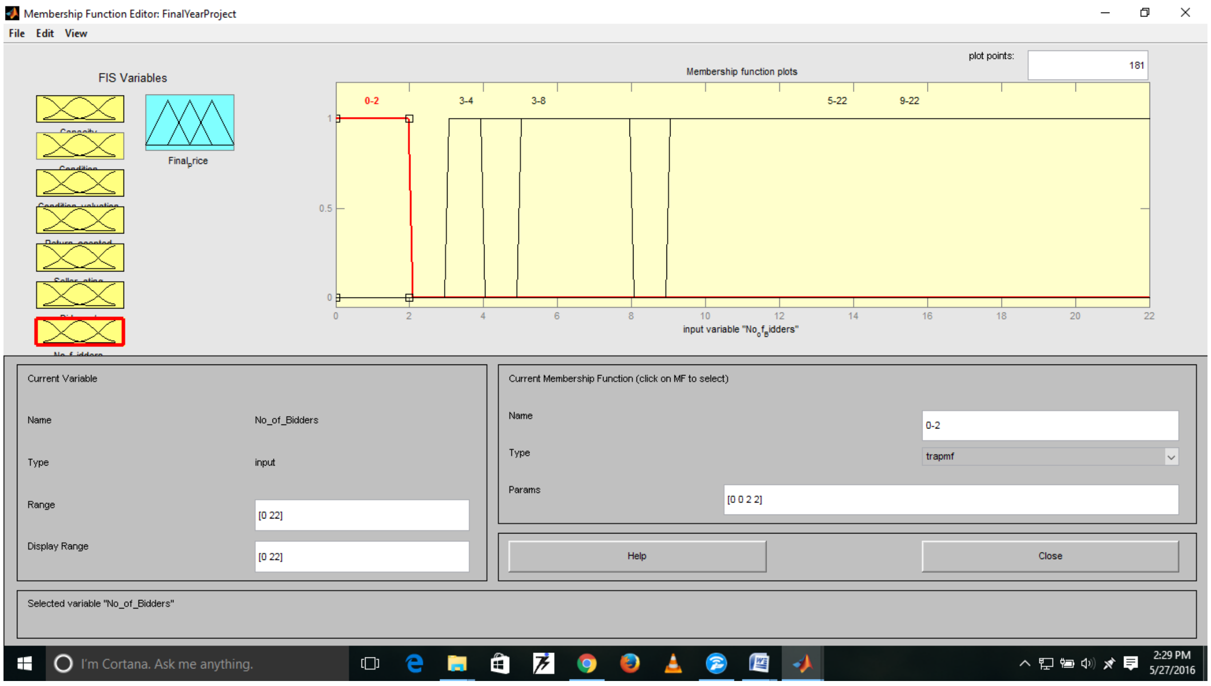
Name 0-7

Type trapmf

Params [0 0 7 7]

Selected variable "Bid_Count"

Fig 22: Bid Count


Membership Function Editor: FinalYearProject
File
Edit
View
FIS Variables
Final_price
Membership function plots
plot points:
181
0-2
3-4
3-8
5-22
9-22
input variable "No_of_bidders"
Help
Close
I'm Cortana. Ask me anything.
2:29 PM
5/27/2016

Current Variable

Name No_of_Bidders

Type input

Range [0 22]

Display Range [0 22]

Current Membership Function (click on MF to select)

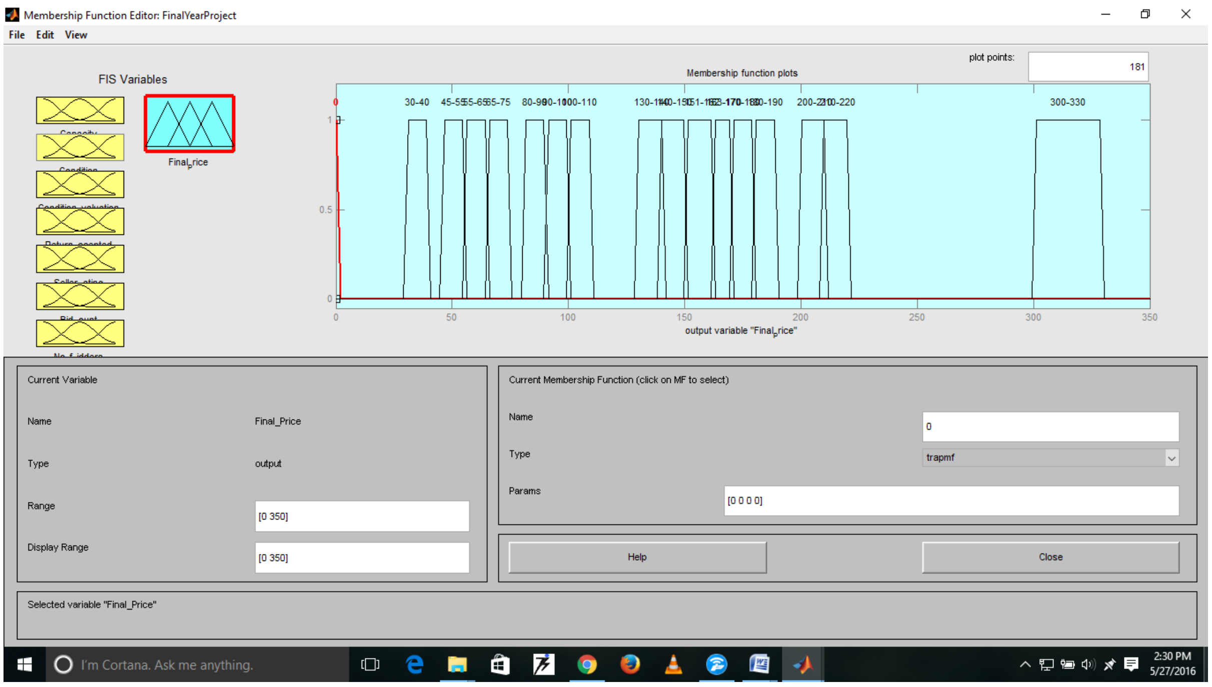
Name 0-2

Type trapmf

Params [0 0 2 2]

Selected variable "No_of_Bidders"

Fig 23: No of Bidders



plot points: 181

Current Variable

Name Final_Price

Type output

Range [0 350]

Display Range [0 350]

Current Membership Function (click on MF to select)

Name 0

Type trapmf

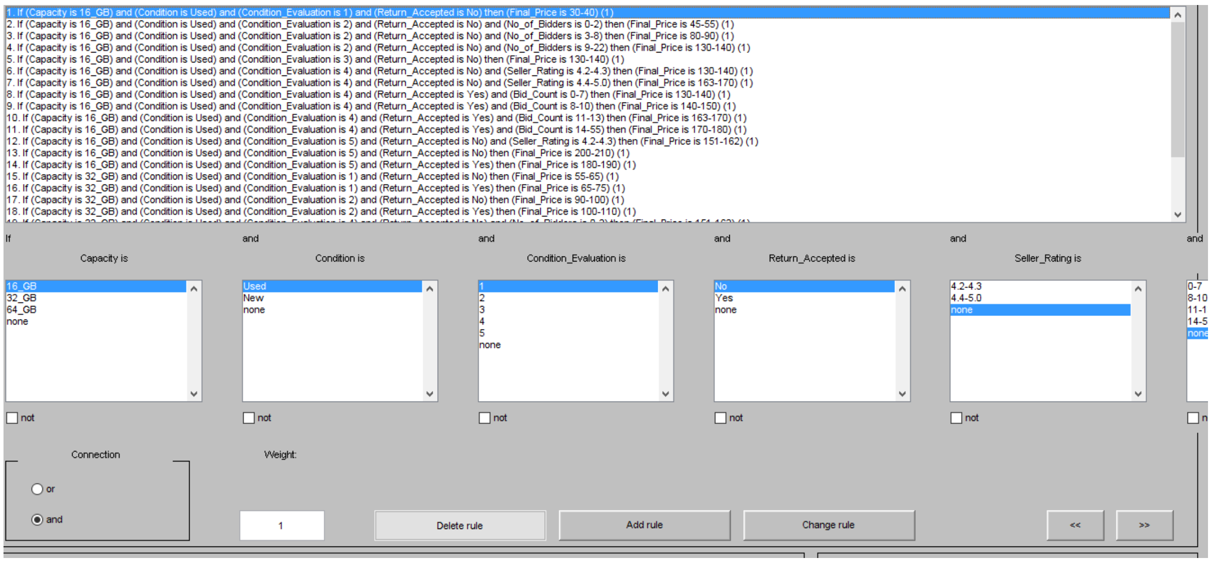
Params [0 0 0 0]

Selected variable "Final_Price"

Fig 24: Final Price

Implementation:

The image shown below describes the rules created from the attributes above.

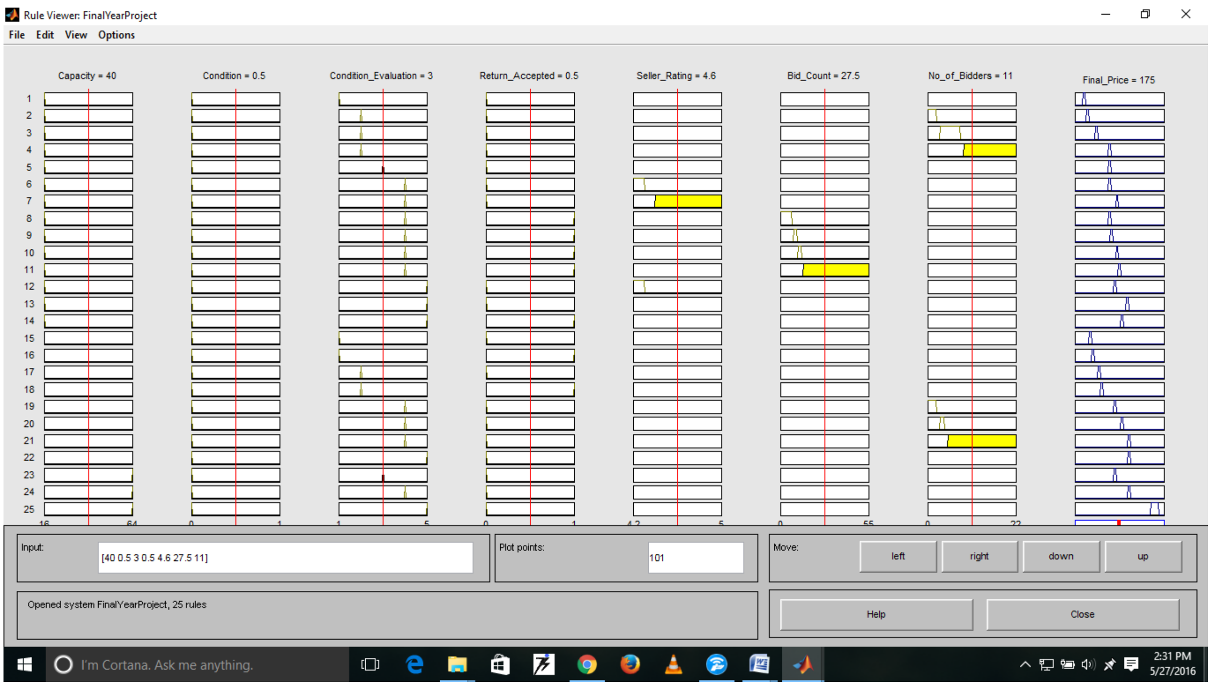

1. If (Capacity is 16_GB) and (Condition is Used) and (Condition_Evaluation is 1) and (Return_Accepted is No) then (Final_Price is 30-40) (1)
2. If (Capacity is 16_GB) and (Condition is Used) and (Condition_Evaluation is 2) and (Return_Accepted is No) and (No_of_Bidders is 0-2) then (Final_Price is 45-55) (1)
3. If (Capacity is 16_GB) and (Condition is Used) and (Condition_Evaluation is 2) and (Return_Accepted is No) and (No_of_Bidders is 3-8) then (Final_Price is 80-90) (1)
4. If (Capacity is 16_GB) and (Condition is Used) and (Condition_Evaluation is 2) and (Return_Accepted is No) and (No_of_Bidders is 9-22) then (Final_Price is 130-140) (1)
5. If (Capacity is 16_GB) and (Condition is Used) and (Condition_Evaluation is 3) and (Return_Accepted is No) then (Final_Price is 130-140) (1)
6. If (Capacity is 16_GB) and (Condition is Used) and (Condition_Evaluation is 4) and (Return_Accepted is No) and (Seller_Rating is 4.2-4.3) then (Final_Price is 130-140) (1)
7. If (Capacity is 16_GB) and (Condition is Used) and (Condition_Evaluation is 4) and (Return_Accepted is No) and (Seller_Rating is 4.4-5.0) then (Final_Price is 163-170) (1)
8. If (Capacity is 16_GB) and (Condition is Used) and (Condition_Evaluation is 4) and (Return_Accepted is Yes) and (Bid_Count is 0-7) then (Final_Price is 130-140) (1)
9. If (Capacity is 16_GB) and (Condition is Used) and (Condition_Evaluation is 4) and (Return_Accepted is Yes) and (Bid_Count is 8-10) then (Final_Price is 140-150) (1)
10. If (Capacity is 16_GB) and (Condition is Used) and (Condition_Evaluation is 4) and (Return_Accepted is Yes) and (Bid_Count is 11-13) then (Final_Price is 163-170) (1)
11. If (Capacity is 16_GB) and (Condition is Used) and (Condition_Evaluation is 4) and (Return_Accepted is Yes) and (Bid_Count is 14-55) then (Final_Price is 170-180) (1)
12. If (Capacity is 16_GB) and (Condition is Used) and (Condition_Evaluation is 5) and (Return_Accepted is No) and (Seller_Rating is 4.2-4.3) then (Final_Price is 151-162) (1)
13. If (Capacity is 16_GB) and (Condition is Used) and (Condition_Evaluation is 5) and (Return_Accepted is No) then (Final_Price is 200-210) (1)
14. If (Capacity is 16_GB) and (Condition is Used) and (Condition_Evaluation is 5) and (Return_Accepted is Yes) then (Final_Price is 180-190) (1)
15. If (Capacity is 32_GB) and (Condition is Used) and (Condition_Evaluation is 1) and (Return_Accepted is No) then (Final_Price is 55-65) (1)
16. If (Capacity is 32_GB) and (Condition is Used) and (Condition_Evaluation is 1) and (Return_Accepted is Yes) then (Final_Price is 65-75) (1)
17. If (Capacity is 32_GB) and (Condition is Used) and (Condition_Evaluation is 2) and (Return_Accepted is No) then (Final_Price is 90-100) (1)
18. If (Capacity is 32_GB) and (Condition is Used) and (Condition_Evaluation is 2) and (Return_Accepted is Yes) then (Final_Price is 100-110) (1)

If	and	and	and	and	and
Capacity is	Condition is	Condition_Evaluation is	Return_Accepted is	Seller_Rating is	
16_GB	Used	1	No	4.2-4.3	0-7
32_GB	New	2	Yes	4.4-5.0	8-10
64_GB	none	3	none	none	11-1
none		4			14-5
		5			none
		none			
not	not	not	not	not	n

Connection: or [✓ and]

Weight: 1

Fig 25: Rules Created



Capacity = 40 Condition = 0.5 Condition_Evaluation = 3 Return_Accepted = 0.5 Seller_Rating = 4.6 Bid_Count = 27.5 No_of_Bidders = 11 Final_Price = 175

Input: [40 0.5 3 0.5 4.6 27.5 11]

Plot points: 101

Opened system FinalYearProject, 25 rules

Fig 26: Predicting Final Price

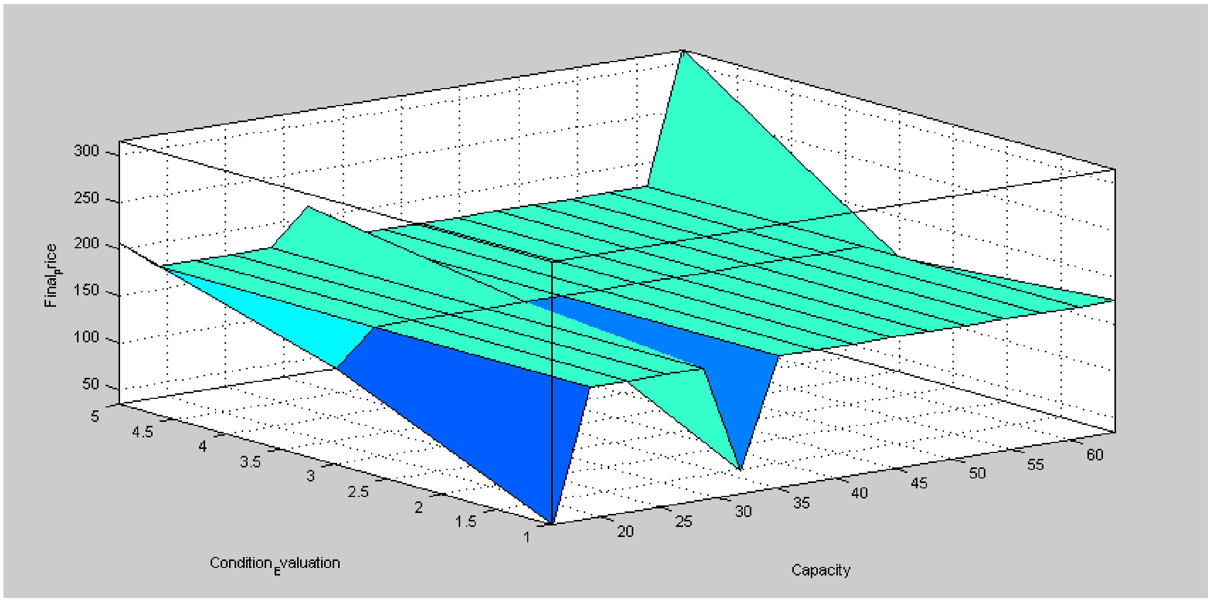


Fig 27: Capacity vs Condition Evaluation Surface

4. PERFORMANCE ANALYSIS

4.1 Definition

Performance Analysis is a specialist discipline involving systematic observations to enhance **performance** and improve decision making, primarily delivered through the provision of objective statistical (Data **Analysis**) and visual feedback (Video **Analysis**).

We distinguish three basic steps in the performance analysis process: data collection, data transformation, and data visualization. *Data collection* is the process by which data about program performance are obtained from an executing program. Data are normally collected in a file, either during or after execution, although in some situations it may be presented to the user in real time.

The raw data produced by profiles, counters, or traces are rarely in the form required to answer performance questions. Hence, *data transformations* are applied, often with the goal of reducing total data volume. Transformations can be used to determine mean values or other higher-order statistics or to extract profile and counter data from traces.

Parallel performance data are inherently multidimensional, consisting of execution times, communication costs, and so on, for multiple program components, on different processors, and for different problem sizes. Although data reduction techniques can be used in some situations to compress performance data to scalar values, it is often necessary to be able to explore the raw multidimensional data. As is well known in computational science and engineering, this process can benefit enormously from the use of *data visualization* techniques.

4.2 Outputs in WEKA

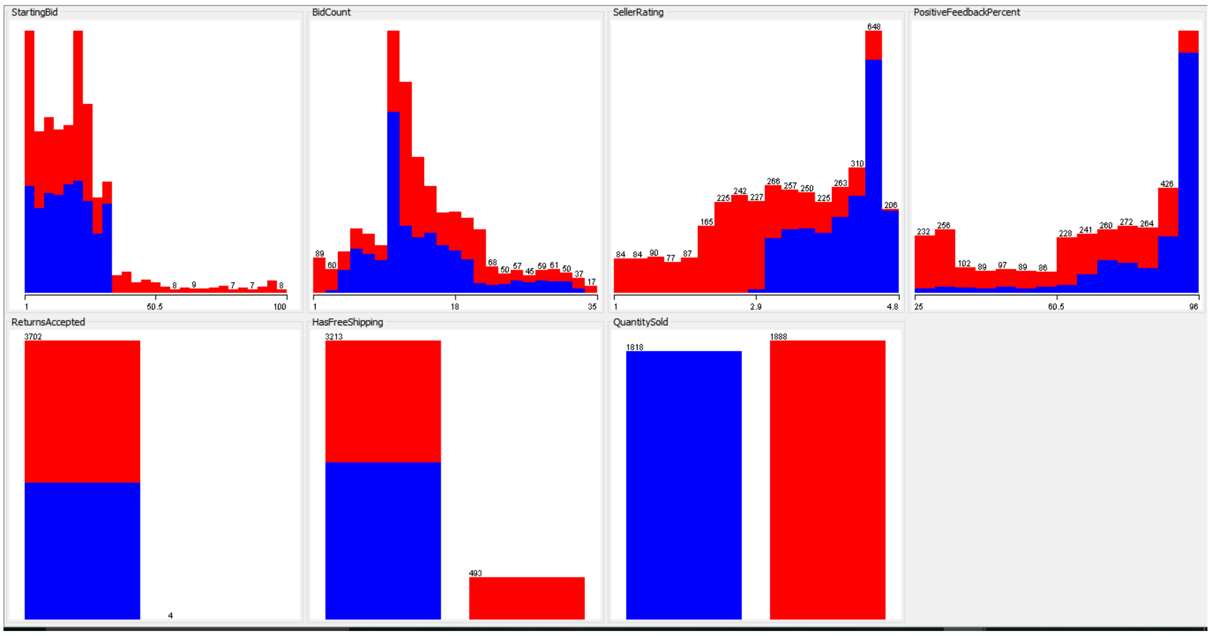


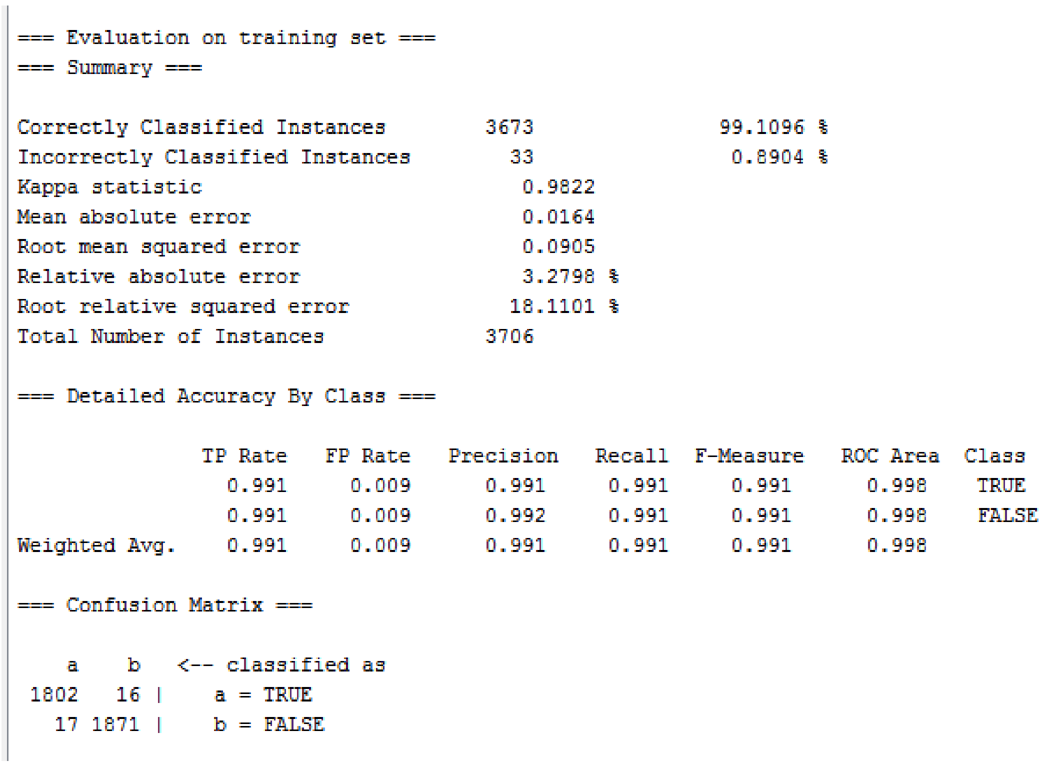
Fig 28: Graphs showing attributes which determine the Final Price

Output using Naïve Bayes

```
Time taken to build model: 0.02 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances        3424               92.3907 %
Incorrectly Classified Instances       282                7.6093 %
Kappa statistic                          0.8476
Mean absolute error                      0.0997
Root mean squared error                  0.2257
Relative absolute error                 19.9517 %
Root relative squared error             45.1412 %
Total Number of Instances             3706

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.893     0.047      0.949     0.893     0.92       0.985    TRUE
                 0.953     0.107      0.903     0.953     0.927      0.985    FALSE
Weighted Avg.    0.924     0.077      0.925     0.924     0.924      0.985

=== Confusion Matrix ===

    a    b   <-- classified as
 1624  194 |    a = TRUE
   88 1800 |    b = FALSE
```

Precision: 0.925   Recall: 0.924   F-Measure: 0.924

Output using J48 Tree



```
=== Evaluation on training set ===
=== Summary ===

Correctly Classified Instances        3673               99.1096 %
Incorrectly Classified Instances        33                0.8904 %
Kappa statistic                          0.9822
Mean absolute error                      0.0164
Root mean squared error                  0.0905
Relative absolute error                  3.2798 %
Root relative squared error             18.1101 %
Total Number of Instances             3706

=== Detailed Accuracy By Class ===

               TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
                 0.991     0.009     0.991     0.991     0.991      0.998    TRUE
                 0.991     0.009     0.992     0.991     0.991      0.998    FALSE
Weighted Avg.    0.991     0.009     0.991     0.991     0.991      0.998

=== Confusion Matrix ===

    a    b   <-- classified as
 1802   16 |    a = TRUE
   17 1871 |    b = FALSE
```

Precision: 0.991 Recall: 0.991 F-Measure: 0.991

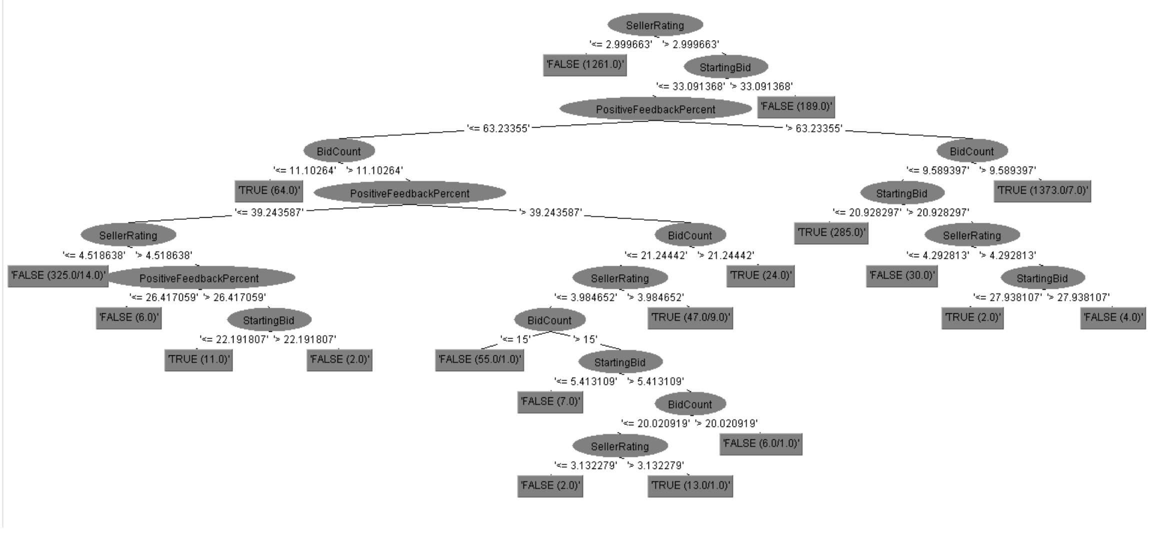


Fig 29 : J48 Decision Tree

Comparison of NaïveBayes and J48 Decision Tree

As the value of Recall, Precision and F-Measure are higher for J48 decision tree, it is a better classification algorithm than NaiveBayes.

4.3 Outputs in FUZZY TOOLBOX

Capacity	*Condition*	*Condition Evaluation*	*Return Accepted*	*Seller Rating*	*Bid Count*	*No of Bidders*	*Predicted Final Price*	*Actual Final Price*	*Error(in %)*
16	0	1	0	4.7	12	6	35	42	16.67
32	0	5	0	4.7	8	2	214	202	-5.9
64	0	3	0	4.4	18	5	158	161	1.86

Table 3: Outputs in Fuzzy Toolbox

Example 1: If

Capacity = NAN Condition = 0 Condition Evaluation = NAN Return Accepted = 0

Seller Rating = 4.6 Bid Count = 27 No: of bidders = 11

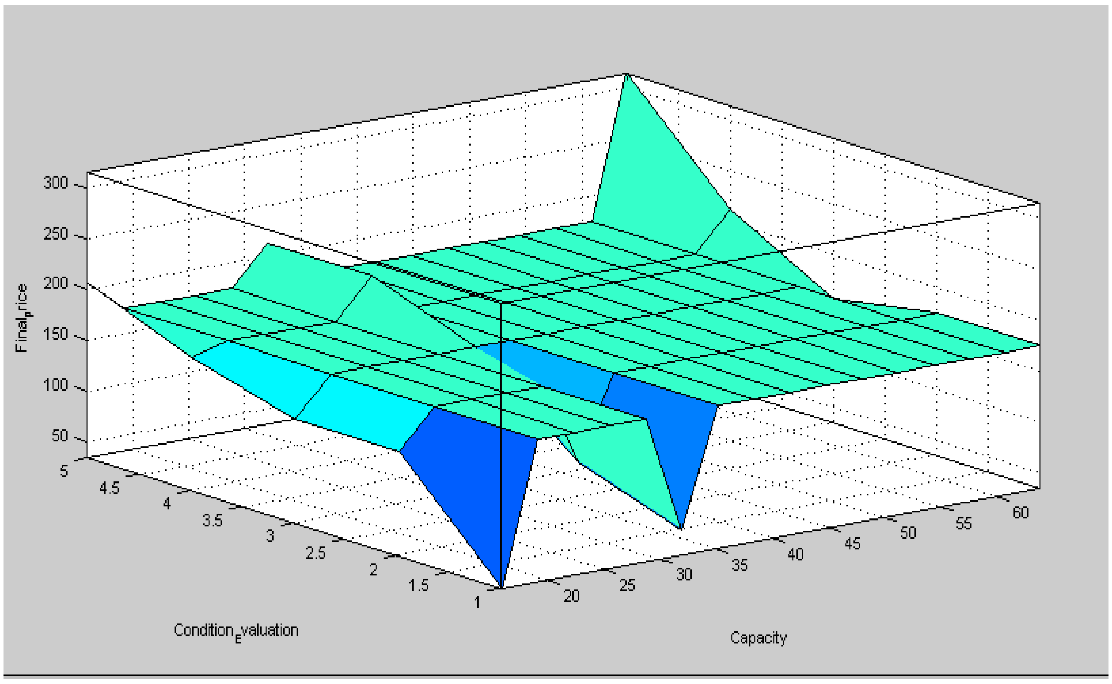


Fig 30: Capacity Vs Condition Evaluation Surface

Example 2: If Capacity = 16 GB Condition = 0 Condition Evaluation = NAN

Return Accepted = NA Seller Rating = 4.6 Bid Count = 27 No: of bidders = 11

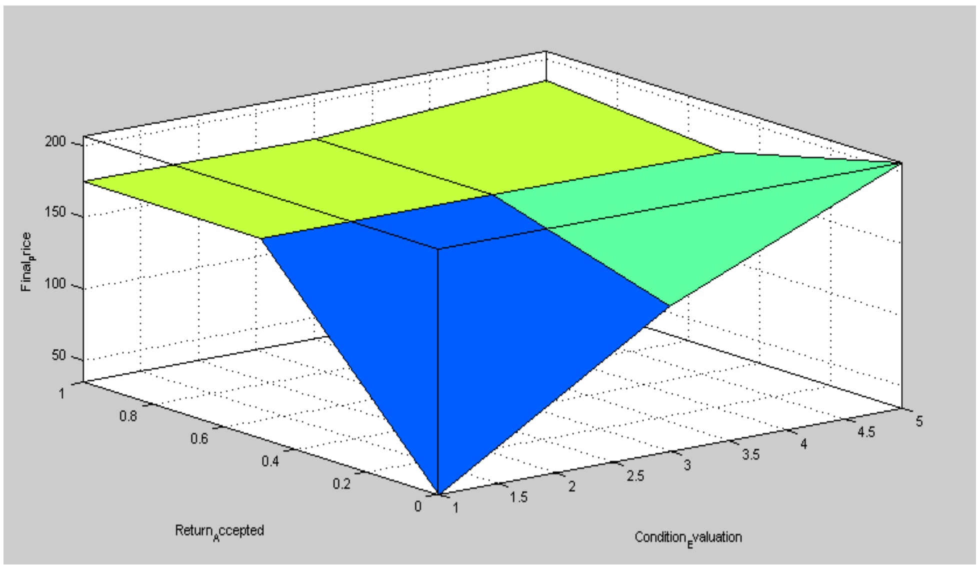


Fig 31: Condition Evaluation vs Return Accepted Surface

4.4 Outputs in ANFIS

Starting Bid : 2 Bid Count : 2 Capacity : 2 Condition : 2 Seller Rating : 2

Return Accepted : 2 No of Bidders : 3 Condition Evaluation : 4

No of Membership Functions	*MF Type*	*No of Epochs*	*Error*
2 2 2 2 2 2 3 4	trimf	5	Epoch 5:error= 0.055804
2 2 2 2 2 2 3 4	trimf	10	Epoch 10:error= 0.05227
2 2 2 2 2 2 3 4	trimf	15	Epoch 15:error= 0.048108
2 2 2 2 2 2 3 4	trapmf	5	Epoch 5:error= 3.8907
2 2 2 2 2 2 3 4	gaussmf	5	Epoch 5:error= 0.08112
2 2 2 2 2 2 3 4	gaussmf	10	Epoch 10:error= 0.076243
2 2 2 2 2 2 3 4	gaussmf	15	Epoch 15:error= 0.073867

Table 4: Outputs in ANFIS

Clearly, from the above, Triangular MF is better.

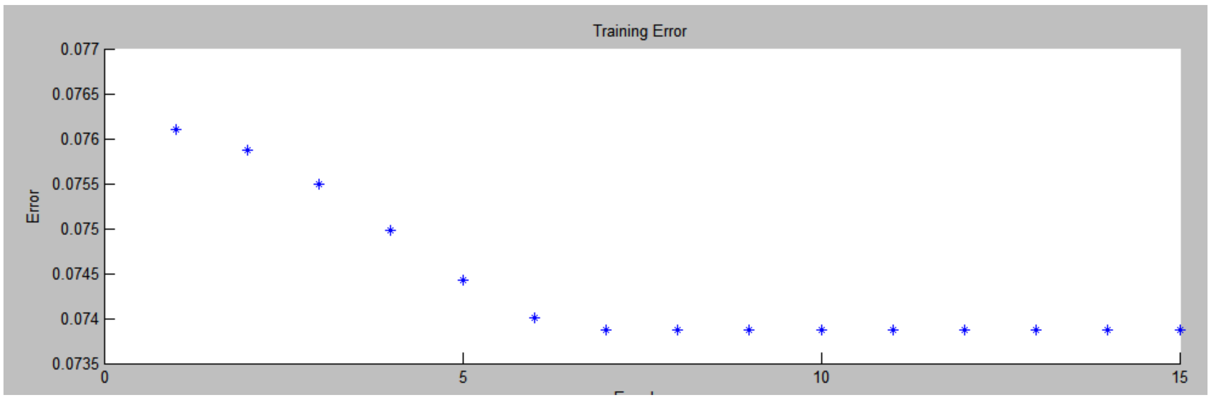


Fig 32: Training Error

Example:

For:

Starting Bid : 40 Bid Count : 17 Capacity : 16 Condition : 0 Seller Rating : 4.5

Return Accepted : 0 No of Bidders : 10 Condition Evaluation : 4

Predicted Price: $153   Auctual Price(1 hr still left for auction to be completed): $132.5

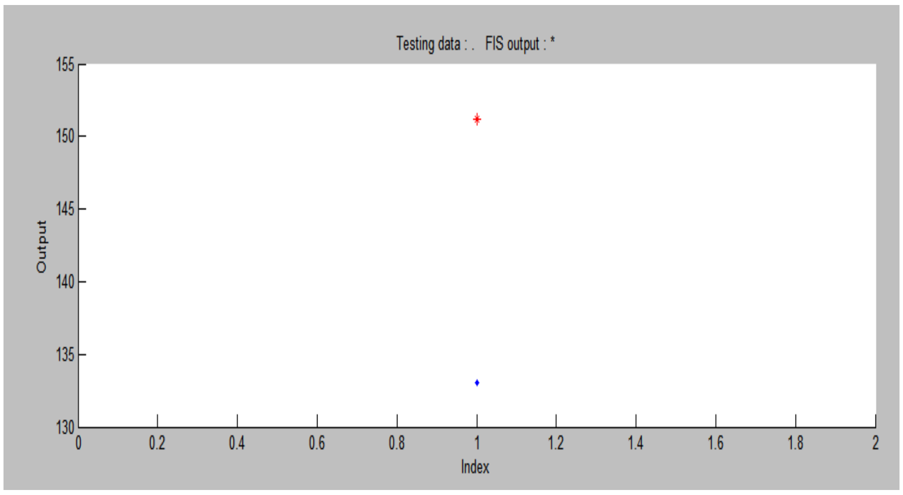


Fig 33: Example 1

5. CONCLUSION AND APPLICATIONS

Conclusion and Future Work:

We described our work on a system capable of predicting the end-price of online auctions. The system requires the information provided by the seller of an item and uses machine learning algorithms to accurately predict the end-price. We find that among a variety of problem formulations, posing price prediction as a series of binary classification problems is best suited for this task. There are several ways to extend the applicability of our approach and try alternative methods. In this paper, we use PDAs because they can be described and compared using "hard" features/specifications (e.g memory size, speed, screen type, operating system). In contrast, "soft" products such as clothing items don't have the same kinds of attributes that can be used to compare different kinds of items. Features such as size, material and color do exist but they are not the kind of attributes that "define" the style of the product. To apply the algorithms in that context, we can use ideas described in some earlier work to first extract product attributes from free-text descriptions of products available online (in stores or auction websites), and then use these attributes as part of the learning process. This would extend the applicability of our approach to "soft" products such as apparel, fashion items, antiques, and collectibles.

In this project, we only used data from auctions that were about the same item. We encoded the context by using temporal features that described past auctions that were "similar" to the one that was being studied. Another direction that we intend to follow is to use data about auctions that are not related to the current item. This is similar to work done in machine learning from learning with unlabeled data where the unlabeled data implicitly provides background knowledge and correlations between attributes that are not directly related, but useful for the classification task. Since there is data available for auctions in general which can be collected fairly cheaply, it would be valuable to study and develop techniques that can learn general patterns about auctions to make inferences about specific items and auctions.

Applications

The ability to predict the ending price of online auction items lends itself to a variety of applications. In this section, we briefly describe some concrete applications that we have developed.

Price Insurance: Knowing the end-price before an auction starts provides an opportunity for a third-party to offer price insurance to sellers. The insurer, knowing the likely ending price for any auction listing before it starts, can charge a premium to insure that the item will sell for at least the insured price. If the item sells for less than the insured price, the seller is reimbursed for the difference (between the insured price and the selling price) by the insurer. We have done some simulations using the price prediction algorithms described in this paper and have found that this insurance service would be profitable given the accuracy of the price prediction algorithms. We are currently in the process of doing detailed experiments and simulations with the price insurance algorithms.

Listing Optimizer: The model of the end-price based on the input attributes of the auction can also be used to help sellers optimize the selling price of their items. When the seller enters their personal information and the item they want to sell in an auction, our service would give suggestions for the auction attributes (such as starting time, starting bid, use of photos, reserve price, words to describe the item, etc.) that would maximize the end-price.

There are several other applications that can be enabled by the price prediction techniques described in this paper. While we have not provided an exhaustive list of applications, we believe that having access to the likely end-price of auction items opens up a large variety of services that can be offered to both buyers and sellers in online auctions.

REFERENCES

- Data from: http://www.ebay.com/sch/i.html?_from=R40&_sacat=0&LH_Auction=1&_nkw=iphone+5s&_sop=1
- WEKA: http://www.cs.waikato.ac.nz/ml/weka/
- MATLAB: http://in.mathworks.com/products/matlab/
- Youtube Videos:
  - https://www.youtube.com/watch?v=Wm-UghoeYFE
  - https://www.youtube.com/watch?v=nArwT9FnCC8
  - https://www.youtube.com/watch?v=m7kpIBGEdkI
  - https://www.youtube.com/watch?v=gd5HwYYOz2U
  - https://www.youtube.com/watch?v=bPrTeUAS6_I
- Research Papers:
  - https://www.cs.uic.edu/~shatz/papers/seke10b.pdf
  - http://cs229.stanford.edu/proj2013/dnicholson_rparanjpe_finalpaper_references_corrected.pdf
  - http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.190.4766&rep=rep1&type=pdf
  - http://www.ijert.org/view-pdf/7911/a-study-on-different-methods-and-algorithms-to-predict-end-prices-i-n-online-auctions
  - http://jaygrossman.com/post/2013/06/10/Predicting-eBay-Auction-Sales-with-Machine-Learning.aspx
  - http://www.aaai.org/ocs/index.php/IAAI/IAAI09/paper/viewFile/264/1029
  - https://www.cs.uic.edu/~shatz/papers/seke10b.pdf
- Software's downloaded from:
  - http://www.cs.waikato.ac.nz/ml/weka/downloading.html
  - http://in.mathworks.com/downloads/
- Neuro Fuzzy- https://en.wikipedia.org/wiki/Neuro-fuzzy