Exploring Data Mining in Used Car Data

University of Nevada, Reno
Department of Computer Science and Engineering
CS 425/625: Project Assignment 3

Team #8:
Patrick Austin, Mile Cilic, Eric Stutzman, Guang Xu

Instructors: Sergiu Dascalu, Devrin Lee

Advisors: Stephen Williams, Lei Yang

12 November 2017

Table of Contents

Abstract

Techniques for efficient analysis and machine learning with large datasets have become a major research area in computer science in the last twenty years. Such advances in data mining technology may enable businesses to target advertising to specific groups and maximize the effectiveness of a marketing budget by speaking directly to interested customers. However, data mining algorithms often perform poorly on very large, multidimensional datasets, and it can be difficult to extract results into useful, human-readable information.

In this project we present an approach to enable targeted marketing solutions directed at used car sales. Our solution is based on training a predictive model on a 50GB used car transactions history dataset made available to us by the analytics company Marketing Evolution. In this document we detail a design for a system that will build a model on a server and enable clients to query the model for prediction results.

Introduction

In the "Exploring Data Mining in Used Car Data" project, Team 8 intends to use state-of-the-art data mining techniques together with an exclusive data set to discover novel and valuable information about used car sales. By building a training model on a large dataset and using the model to make predictions about future sales, Team 8 intends to enable predictive, targeted advertising approaches to make efficient use of an available marketing budget.

Working in collaboration with the firm Marketing Evolution and their advisor at Marketing Evolution, Stephen Williams, the team has been granted access to a 50GB dataset documenting used car transactions in the United States over the last several years. This dataset contains attributes such as extensive geographic, demographic, and financial information, as well as details about the car models being purchased and sold.

Using data mining algorithms and strategies learned from our faculty advisor Dr. Lei Yang, we hope to use a decision tree induction approach on the data set in order to train a predictive model. This model will then allow users to enter available demographic or financial details about a customer as input, and receive a recommendation of cars that customer is likely to be interested in as output. This model could be used to target personalized advertisements to a customer, or could even be packaged as an app to provide support to used car dealers on the ground.

In order to accomplish these goals, Team 8 intends to build a back end application in Python which will conduct the decision tree induction on the data, and a front end mobile application which can be used to query the decision tree for predictive results. Our intent is for a client user on the ground such as a car salesman or advertiser that can input information about his or her clients and obtain predictions in a timely and convenient fashion.

In this design document we focus on detailing the structure of the client and server components of the project, which we will then go on to prototype and implement. The server will perform the training and generate the model, and clients will be able to query the server for prediction results that make use of the model via the app. To support these operations, we also detail a simple login system to maintain client privacy and the security of the dataset and our training model. We also detail design for the client systems that will allow the client to visualize the model and prediction data to maximize the human-readability of the information the server provides, as well as a simple payment system for the client application supported by Apple Pay for iOS.

High-level and Medium-level Design

System Level Diagram - Context Model

In Figure 1, we show a high-level context model for the system that outlines the modules needed for system functionality at a high level of abstraction. The user authentication system governs user access to the system. The client and server user interface systems allow for human interaction with the system; the client, working on the iOS app, can request predictions from the server among other services via GUI where the server GUI provides diagnostic information, logging, and appropriate admin actions. The model training system is used to train a model on the used car dataset, while the prediction system uses that model to make a prediction. These are both done on the server side. On the client side, the model query system allows the user to create a request for a prediction on the app, which is then sent to the server for processing. The visualization system is used to gather, calculate, and provide information to the GUI about the model and its statistical properties.

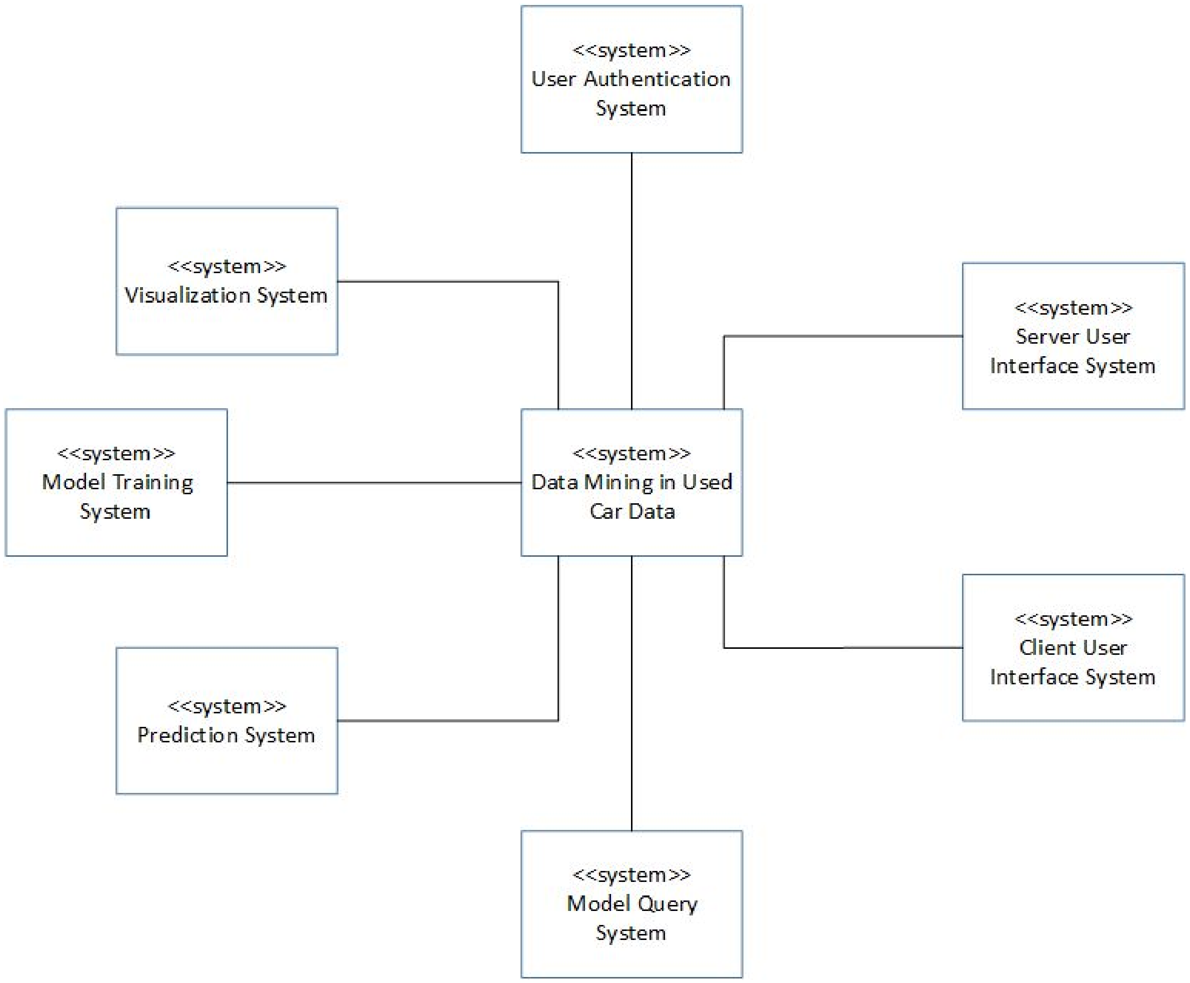

<<system>> User Authentication System
<<system>> Visualization System
<<system>> Server User Interface System
<<system>> Model Training System
<<system>> Data Mining in Used Car Data
<<system>> Client User Interface System
<<system>> Prediction System
<<system>> Model Query System

Figure 1: A context diagram for the used car data mining system.

To better visualize our system level diagram, Figure 2 illustrates the relationship between components.

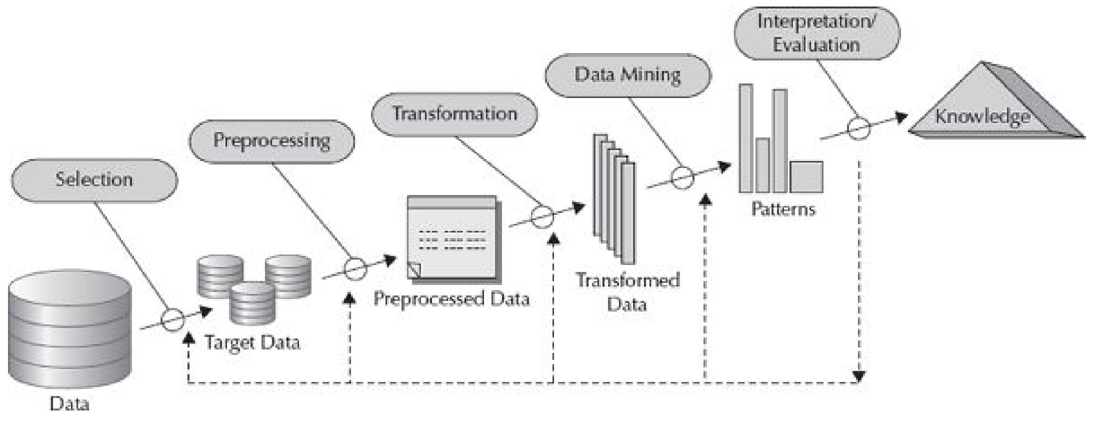


Figure 2: A diagram illustrating a data mining process. The data mining steps shown here are built into the structure of the used car data mining system. (Fayyad, 1996)

Selection, Preprocessing, Transformation, and Data Mining steps take place in the Model Training System on the server side.

After the User Authentication is successful, the Model Query System is used to call the Prediction System, and the prediction can then be visualized by the Visualization System. These actions are supported by the Client and Server User Interface Modules. All of these actions and systems fall in the Interpretation/Evaluation stage of the graph.

Class Diagram and Descriptions

In Figure 3 we show our full class diagram, illustrating classes in the system, their component data and relevant methods, and the proposed interactions between classes in the system.

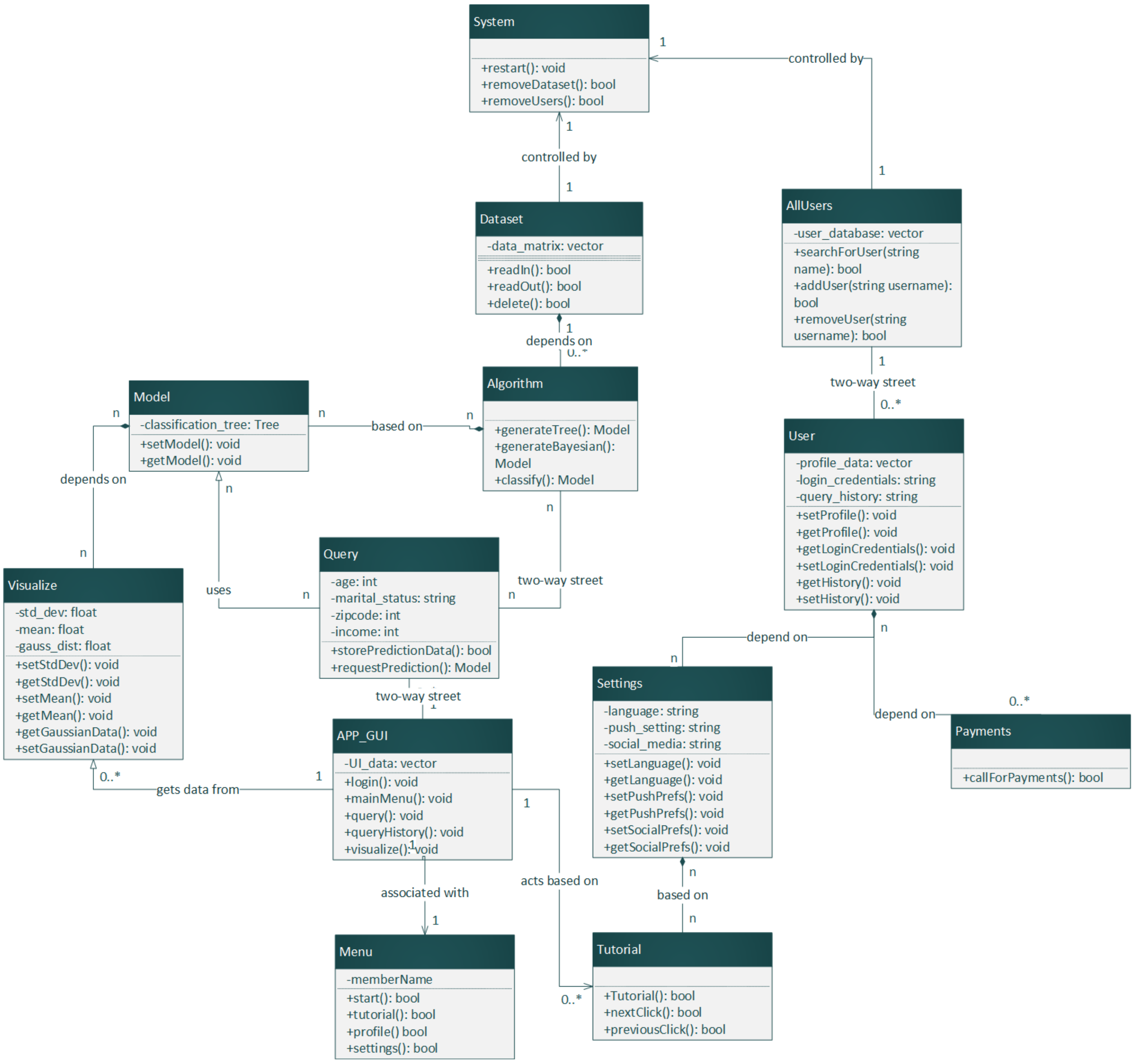


Figure 3: The full class diagram for the used car data mining system.

Here we provide more detailed class descriptions for the classes shown in Figure 3:

System - The system class is used to manage system and administrator level actions such as restarting the system, removing users, or removing the dataset from operation.

- restart()
  - allows for the system restart in case of malfunction
  - returns true if successful, false otherwise
- removeDataset()
- removeUsers()

Dataset - The dataset class reads and stores the used car dataset so the system can apply data mining algorithms.

- Car data matrix

- readIn()
  - reads the data from the used car dataset for processing
  - returns true if successful, false otherwise
- readOut()
  - writes data to the used car dataset if modification is necessary
  - returns true if successful, false otherwise
- delete()
  - delete the stored used car dataset
  - returns true if successful, false otherwise

Algorithm - The algorithm class contains methods that are used to process the used car dataset. This includes algorithms used to generate models, as well as algorithms called for the prediction task.
● generateTree() ○ takes the used car dataset as input, and outputs a trained model that can be used to predict future instances ○ returns trained model ● generateBayesian() ○ takes the used car dataset as input, and outputs a naive Bayes classifier model that can be used to predict future instances ○ returns trained model ● classify() ○ given a valid query from a client and a trained model, use the trained model to make a prediction about the used car that is likely to be desired ○ returns classification data

Model - The model class contains the trained model that is the result of performing a data mining algorithm on the used car dataset. The model will be used in prediction tasks.
● Classification tree
● setModel() ● getModel()

User - The user class is a structure containing data about a single user such as their login credentials and query history. Class methods can be used to set and get this data.
● Profile data ● Login credentials ● Query history
● setProfile() ● getProfile() ● getLoginCred() ● setLoginCred() ● setHistory() ● getHistory()

AllUsers - The AllUsers class is the structure containing all authenticated users in the system. This class is searched when user login is initiated, and users can be added or removed as need occurs for users to access or no longer access the system.
● User database
● searchForUser(string name) ○ takes a string and searches for the user name that matches. Used when searching for a matching user on attempted login ○ returns the user, if found, and a value denoting not found otherwise ● addUser(string username) ○ creates a new user in the database ○ returns true if successful, false otherwise

Menu - The menu class provides the client operations necessary to navigate to different areas of the user interface.
● start() ○ initializes a new prediction by making appropriate UI and class calls ○ returns true if successful, false otherwise. ● tutorial() ○ initializes the tutorial by making appropriate UI and class calls ○ returns true if successful, false otherwise ● profile() ○ initializes user profile option screen by making appropriate UI and class calls ○ returns true if successful, false otherwise ● settings() ○ initializes the settings screen by making appropriate UI and class calls ○ returns true if successful, false otherwise.

Settings - The settings class handles changes and storage of user settings such as display language, social media preferences, and push preferences for phone.
● Language preferences ● Push setting preferences ● Social media preferences
● setLanguage() ● getLanguage() ● setPushPrefs() ● getPushPrefs() ● setSocialPrefs() ● getSocialPrefs()

Query - The query class handles the case where a user requests a prediction from the server. It stores the user-supplied data about the prediction, handles the server call, and returns the result.
● Age ● Marital status ● Zipcode ● Income
● storePredictionData() ○ saves user input from prediction setup, i.e. the provided data about details such as income, zip code, etc., taken from GUI entry ○ returns true if operation successful ● requestPrediction() ○ calls the server and requests a prediction using the trained model using the stored prediction data ○ returns the data about the prediction for use in display, classification, etc.

Visualize - The visualize class handles calculation and preparation for display of advanced statistical information about a prediction.

- Standard deviation
- Mean
- Gaussian distribution data

- setStdDev()
  - takes provided prediction data from the server and stores it for use in the advanced statistical information display screen.
- getStdDev()
- setMean()
- getMean()
- setGaussianData()
- getGaussianData()
- calculateGaussian()
  - prepares a graph showing probability distribution information for display by the GUI
  - returns values needed for GUI distribution display

Tutorial - The tutorial class is used to handle the operation of the user tutorial, which proceeds through a series of fixed screens that teach a client about app functionality.

- Tutorial()
  - starts the tutorial mode and loads necessary components
  - returns true if successful, false otherwise
- nextClick()
  - goes to the next stage of the tutorial
  - returns true if successful, false otherwise
- previousClick()
  - goes to the previous stage of the tutorial
  - returns true if successful, false otherwise

APP_GUI - The APP_GUI class handles and maintains user interaction with the client-side GUI, i.e. moving between menus and displaying appropriate information.
• UI image data
• login()   ○ obtains username and password from user, verifying credentials   ○ returns true if user auth was successful, false otherwise • mainMenu()   ○ makes appropriate GUI and class calls to generate the main menu and prepare it for user input   ○ returns true if successful, false otherwise • query()   ○ initializes process of collecting prediction input data from user such as demographics via appropriate UI and class calls   ○ sends query to server to obtain prediction   ○ returns true if successful, false otherwise • queryHistory()   ○ displays past queries and results for user browsing   ○ returns true if successful, false otherwise • visualize()   ○ displays predictions in a graphical manner as well as advanced statistical information about the prediction via appropriate GUI and class calls   ○ returns true if successful, false otherwise

Payments - The payments class handles interaction with the Apple Pay service used to support subscription to the app's service.
• callForPayment()   ○ calls Apple Pay to request, authenticate, verify, etc. a valid credit card payment   ○ returns true if Apple Pay process completes successfully

Data structures:

In addition to using common data structures like stack, queue, and list in our implementation, we note that the classification approaches we intend to use are typically tree-based, so trees are likely to be one of our main data structures and likely to be used to support the trained model itself.

In terms of database tables, we intend to employ 2 primary database objects. The first of these stores user login information. The structure of an entry in this table (an object of the user class, stored in the All_Users class) should contain the following, with strings and lists being used as appropriate:

User ID	Username	Login Credential	Saved Preferences	Query History

The second database table is used for storage of our initial used car dataset. As this dataset is proprietary and very large, containing over 280 attributes, only a small selection of these are shown in this example:

Customer ID	ZIP Code	Children	Marital Status	Age	Car Model

Detailed Design

Server Side Query

Figure 4 shows a system state diagram for handling a query for a prediction on the server side. That is to say, this is the process followed when the client sends data to the server about an individual and asks for a prediction about what car the user is likely to purchase. The server waits for such a request, receives it, and then processes it as shown below.

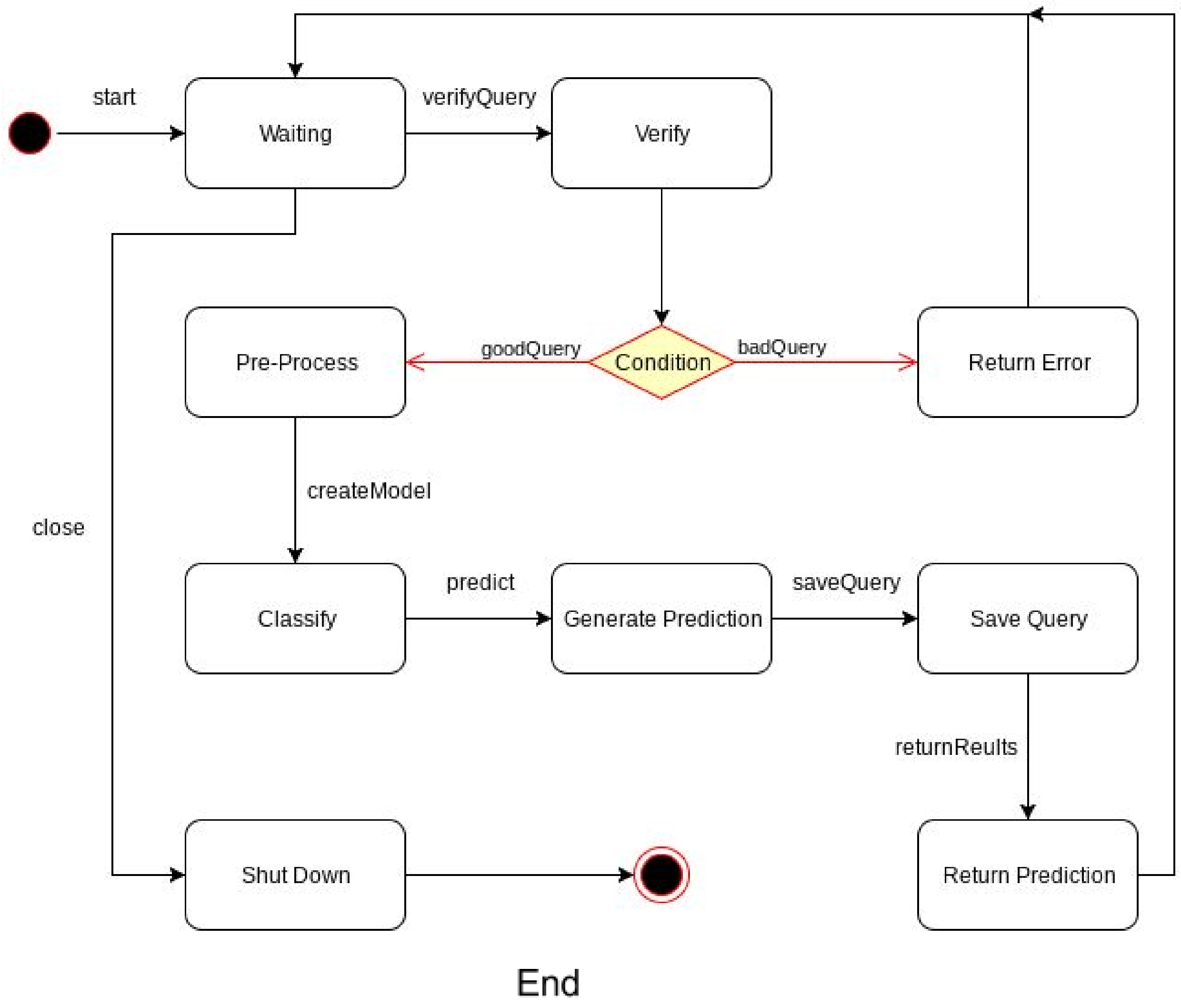


Figure 4: The system state diagram for a server side query in the used car data mining system.

Admin GUI

In Figure 5 we illustrate an activity diagram for the admin GUI, used for diagnostics and testing of the used car data mining system on the server side. This can be used to display a variety of diagnostic information and take admin-end actions like removing users from the system. The system is robust to failure conditions such as intermittent network failure or unsuccessful login.

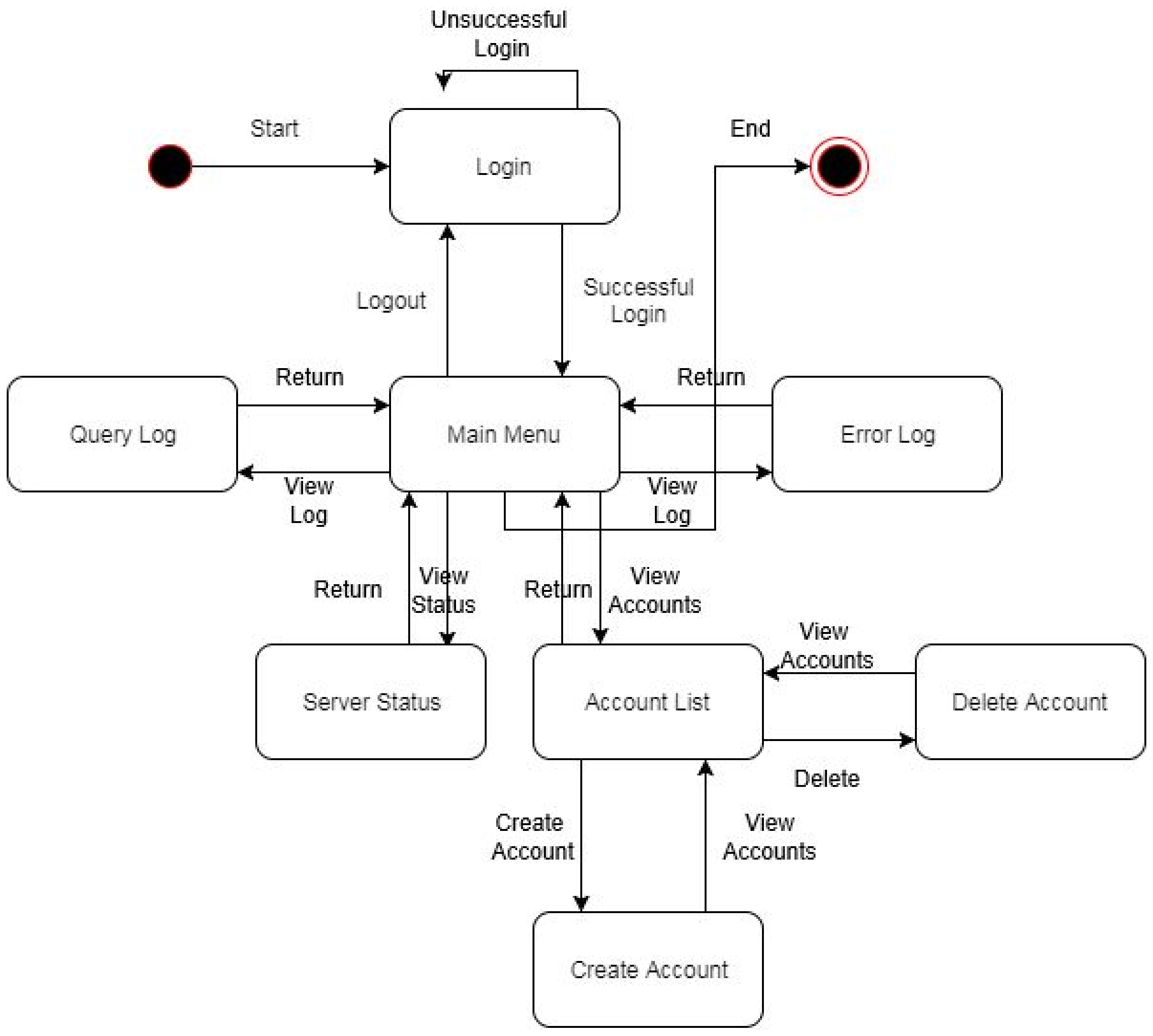


Figure 5: The activity diagram for the Admin-side GUI for the used car data mining system.

Client Side Query

Figure 6 shows an activity diagram for the process the system enters when initiating a query from the client side. This process allows the client to enter information about an individual and then send that information to the server for a prediction about the car the user is likely to purchase. The system is robust to failure conditions such as intermittent network failure or unsuccessful login.

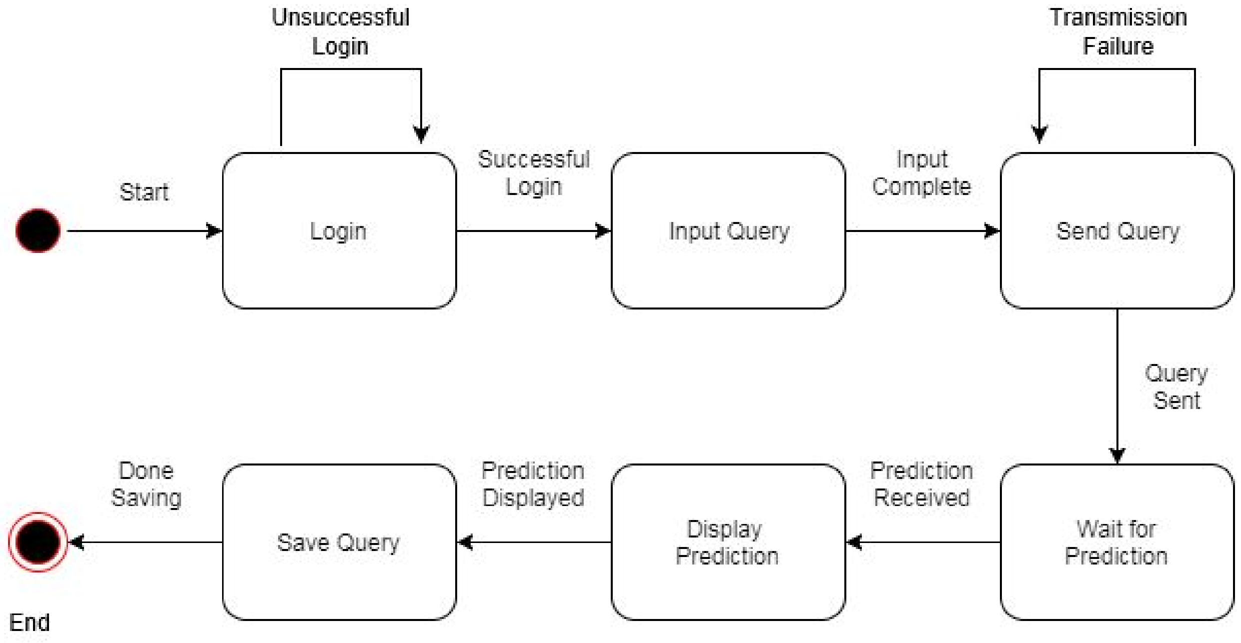


Figure 6: The activity diagram for a client-side query for the used car data mining system.

APP GUI

Figure 7 shows an activity diagram for the GUI on the application or client side of the system. This GUI allows for menu navigation, access to, and movement between the major features of the client application. The system is robust to failure conditions such as intermittent network failure or unsuccessful login.

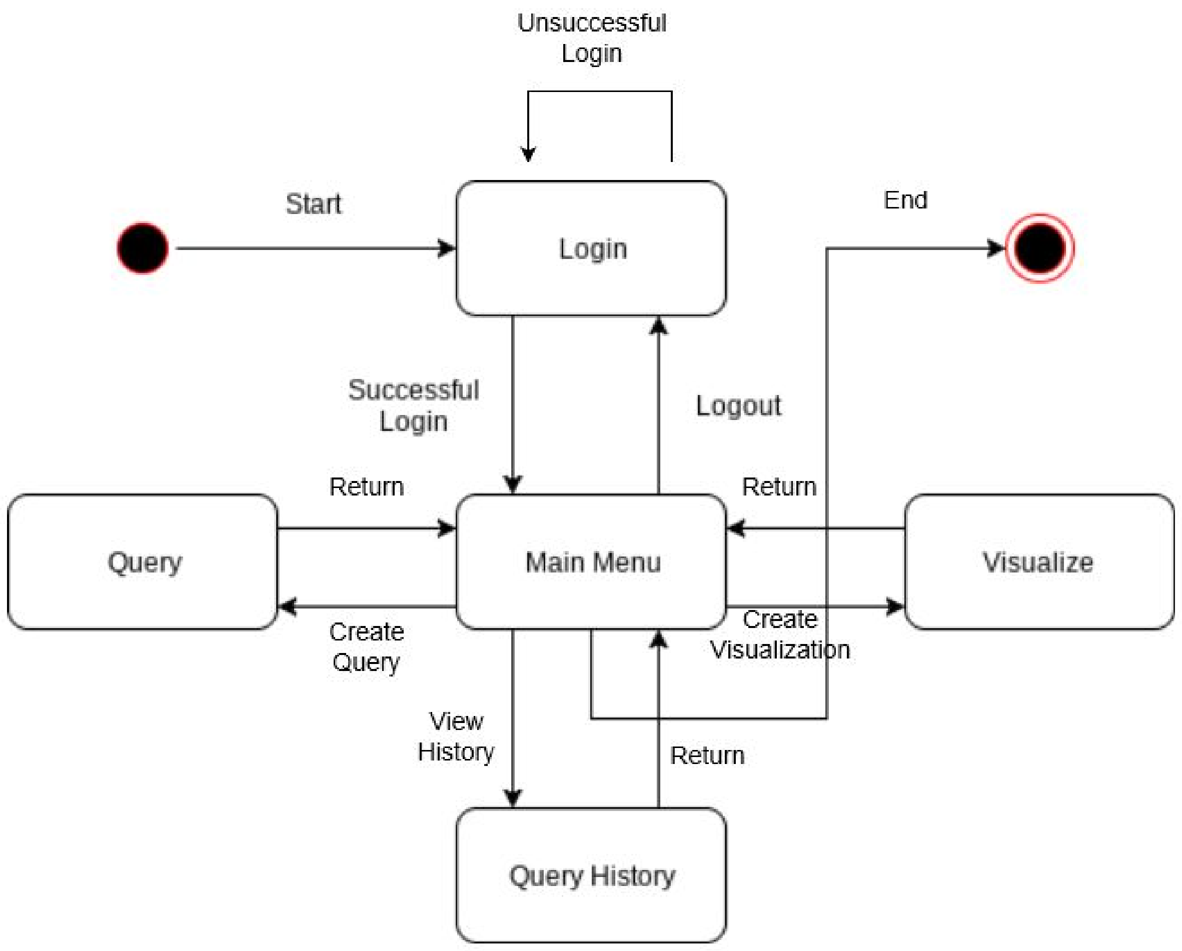


Figure 7: The activity diagram for the application GUI in the used car data mining system.

Initial Hardware Design

Figure 8 shows a simple diagram illustrating the relationship between hardware components in the used car data mining system. We then briefly discuss each of the components in the diagram.

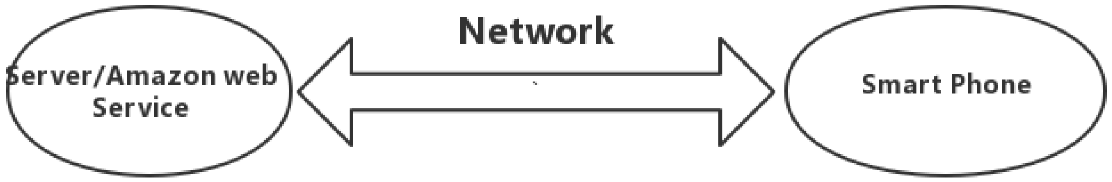


Figure 8: The relationship between hardware components in the used car data mining system.

Server/Amazon Web Service:
This will be the back end of our application; the dataset will be stored here as well as the user login and profile information. The actual data mining training and classification process will take place on the server side. One such server is shown in Figure 9.

Smartphone:
This will serve as the frontend of our application. The user will query and interact with the server side and receive prediction results using their smartphones. Summary and visualization of the results will be displayed here. One such smartphone is shown in Figure 10.

Figure 9: An example of a datacenter with servers supporting a cloud computing platform. This project will use AWS as a cloud platform.

Figure 10: An example of a common smartphone. The client side of the project will support iOS and Android, and should have minimal performance requirements for use on most any modern phone.

User Interface Design

In Figures 11-22 we provide mockups of various UI features for the application or client-side of the used car data mining system. These mockups show a variety of core functionalities we intend to include, reflecting the design shown above.

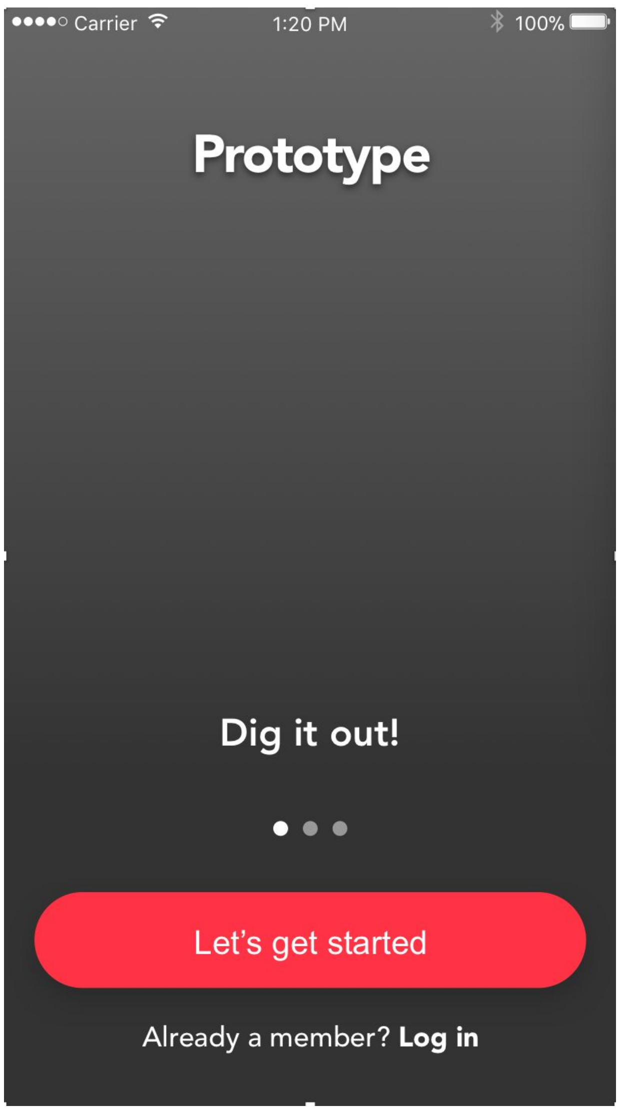


Figure 11: This is the welcome screen when the application is opened, on this screen you can choose to login by clicking the login button or register by clicking the "Let's get started" button. You can slide the "Dig it out!" slogan to see more information about the app.

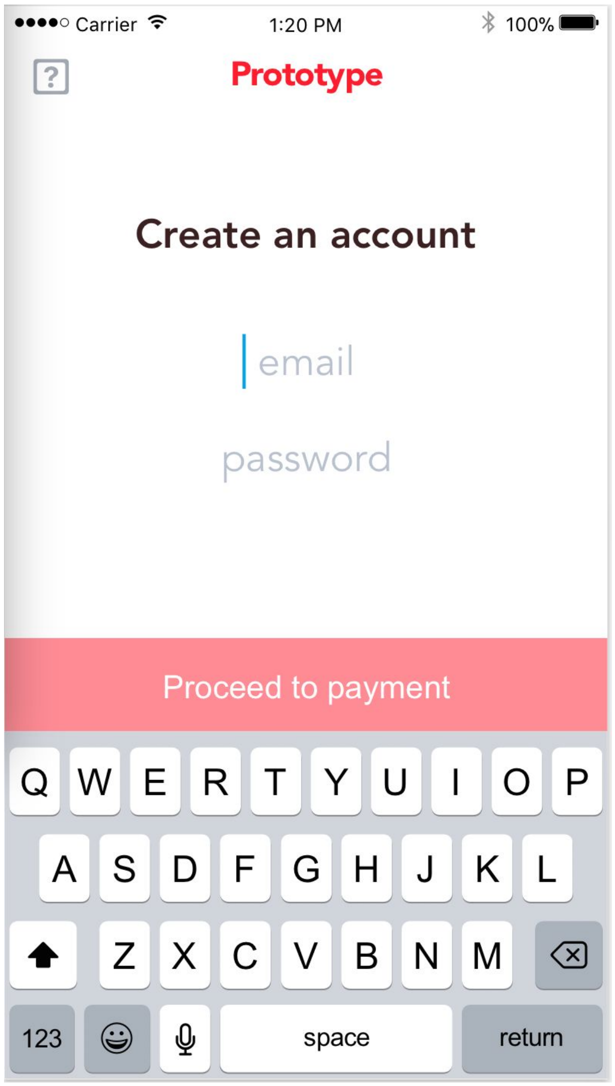


Figure 12: This is a mockup for the register screen. In this mockup the user must type in a valid email address and a qualified password in order to proceed to the payment stage. The "proceed to payment" button will remain grey (not available) until the user does so. Clicking the question mark button in the left-top side will enable the tutorial mode which will take the user to go through the basic functions.

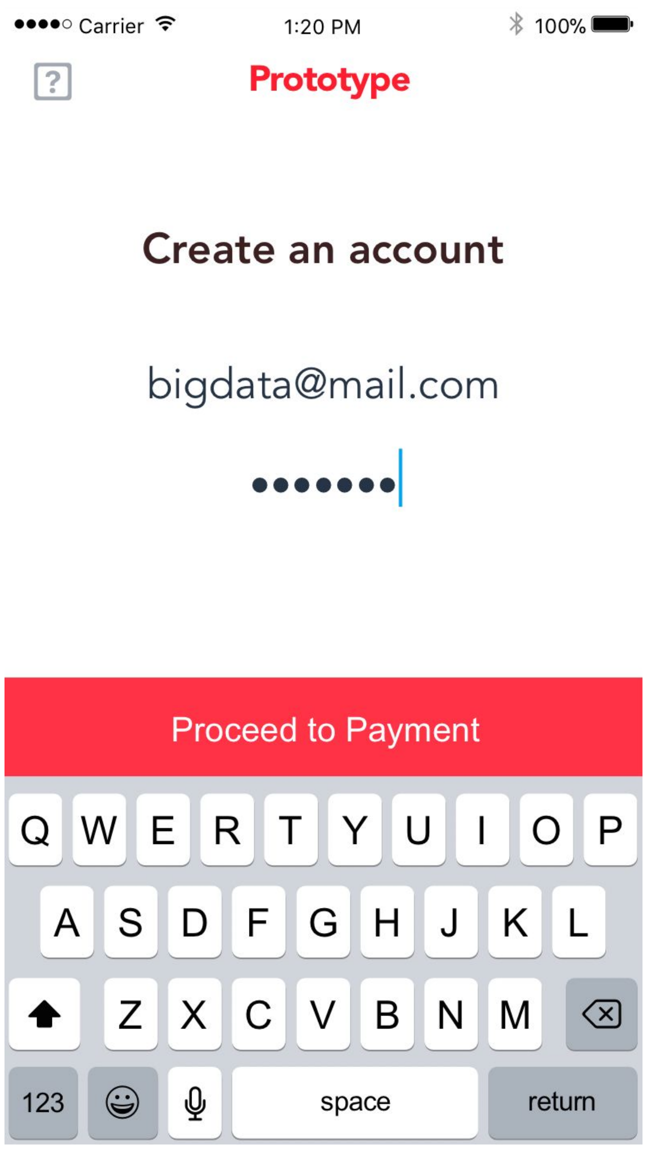
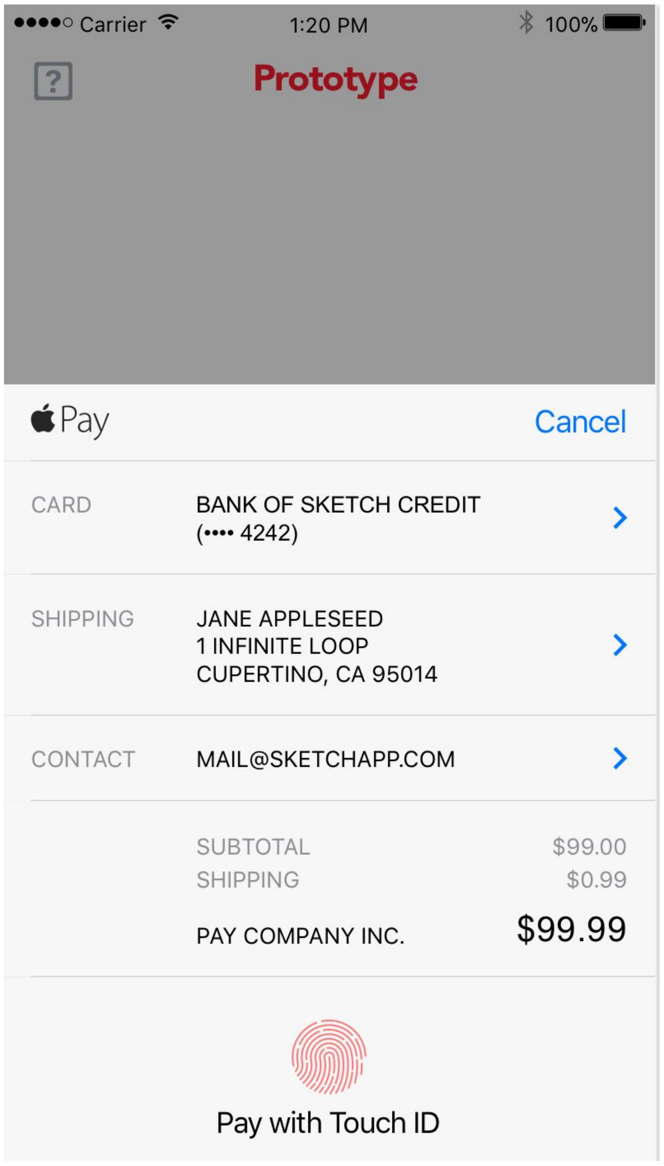


Figure 13: In the registration screen, we assume the user has now typed in a valid email and qualified password. The password will be hidden for added security, and the "proceed to payment" button will now become available.



CARD BANK OF SKETCH CREDIT (•••• 4242)

SHIPPING JANE APPLESEED
1 INFINITE LOOP
CUPERTINO, CA 95014

CONTACT MAIL@SKETCHAPP.COM

SUBTOTAL	$99.00
SHIPPING	$0.99
PAY COMPANY INC.	$99.99

Figure 14. This is a mockup of the payment screen; we can use Apple Pay as our payment collecting tool for its security and convenience. In this mockup we show a subscription payment system. Shown here, the user pays $99.99 for one year of usage. Security and recurrent payment are handled on Apple's end via the Apple Pay interface.

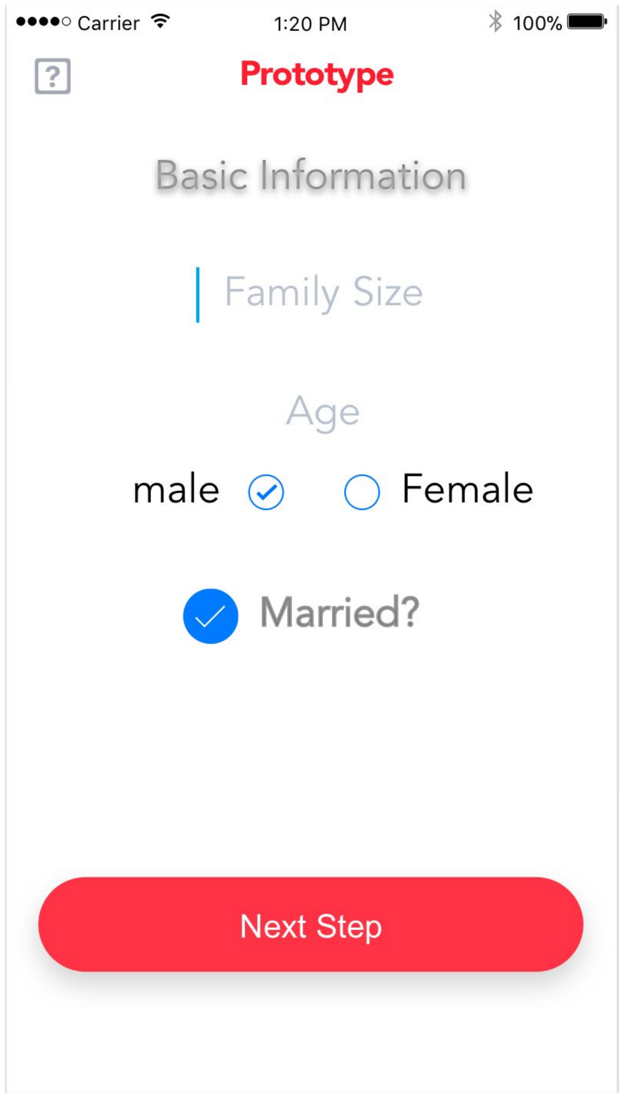


Figure 15: This is a mockup of the data input screen for a user performing a query. The user can input demographic information in this phase. After the user inputs family size and age in the corresponding fields, the “Next” button can be used to proceed.

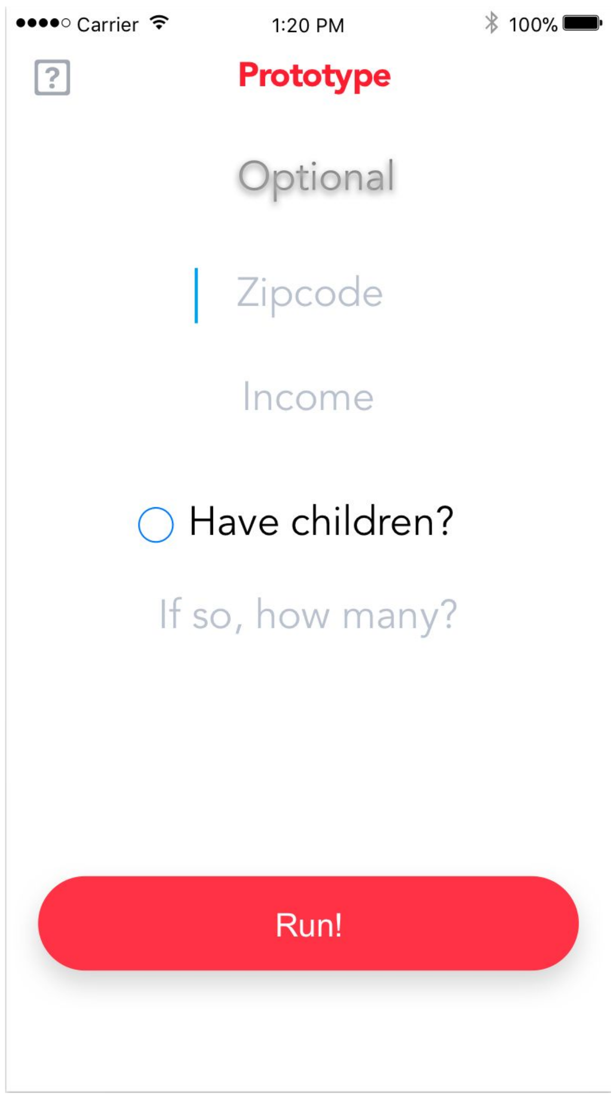


Figure 16. On the subsequent query screen the user can input additional optional information; note that the user may or may not enter these and they could be partially filled or blank, but the accuracy of the prediction will improve when more information is provided by the user.. By clicking the “Run” button, the application will send the information to the backend for prediction.

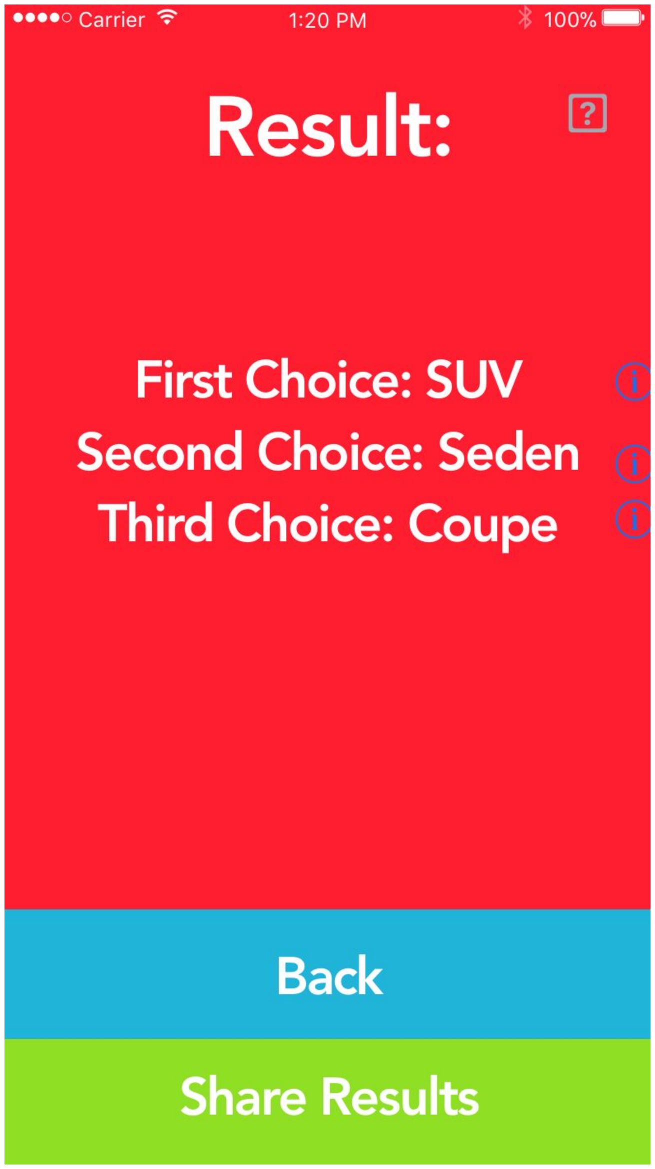


Figure 17: This is a mockup of the result page after the application receives the prediction results from the backend. In this mockup the algorithm gives the three most possible outcomes with ranking, the user can click the information icons next to the results for easy access to more details about the vehicle in question. There is also a question mark icon in the right-top side, which will take the user to the advanced result mode, which shows visualization and statistical data about the result. The "Back" button will take the user to the main menu and the "Share Results" button will provide some options for the user to share their results using social apps.

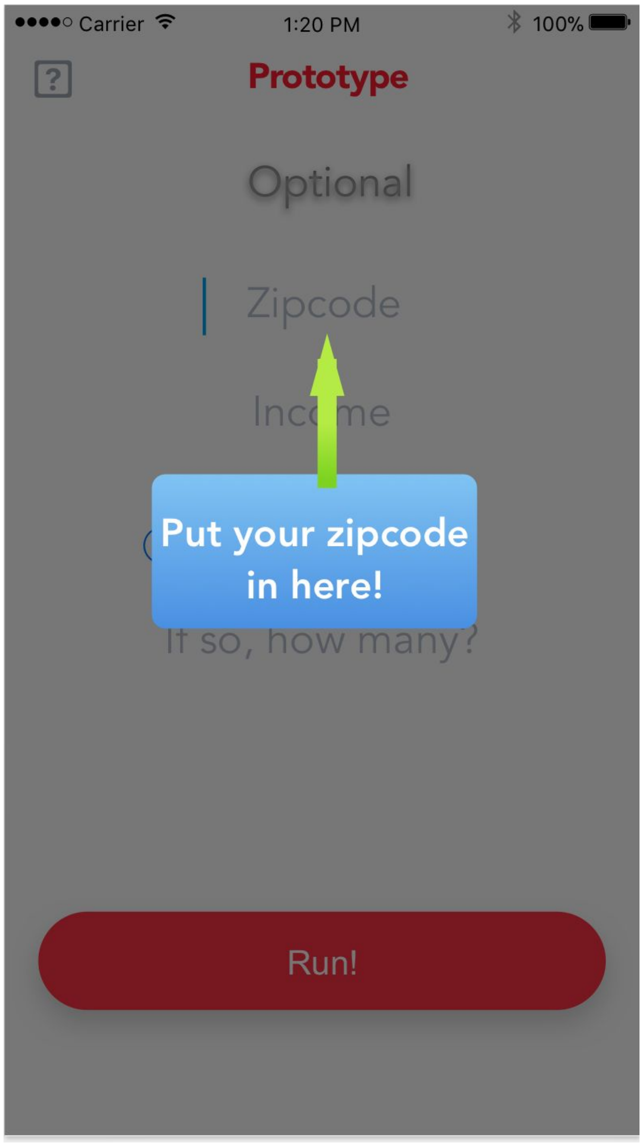
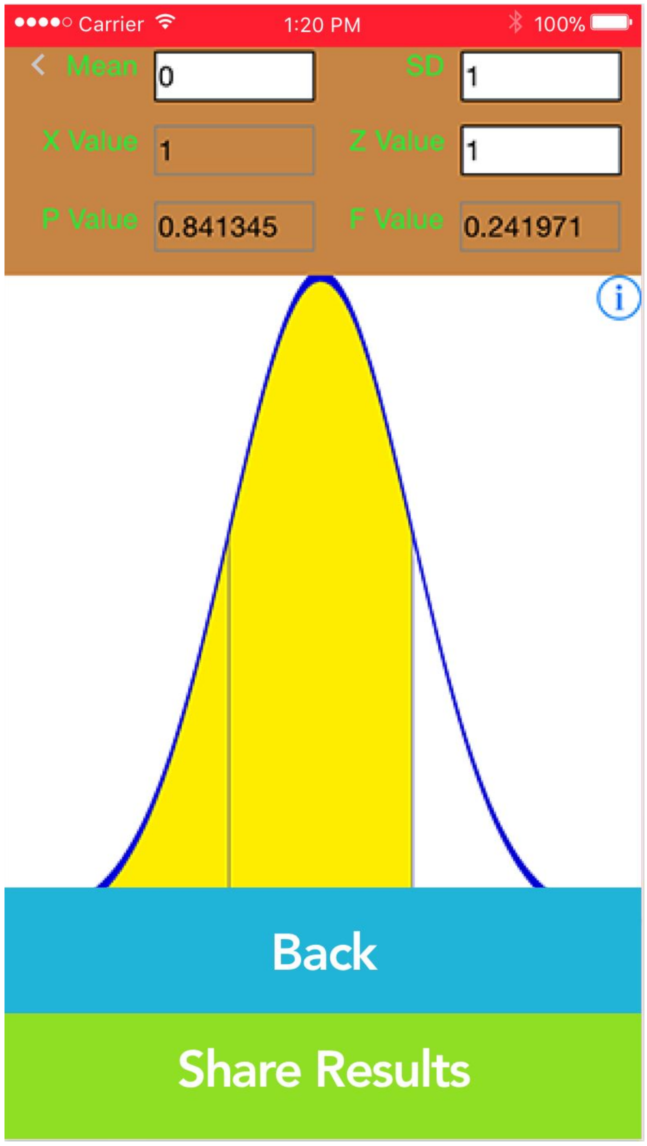
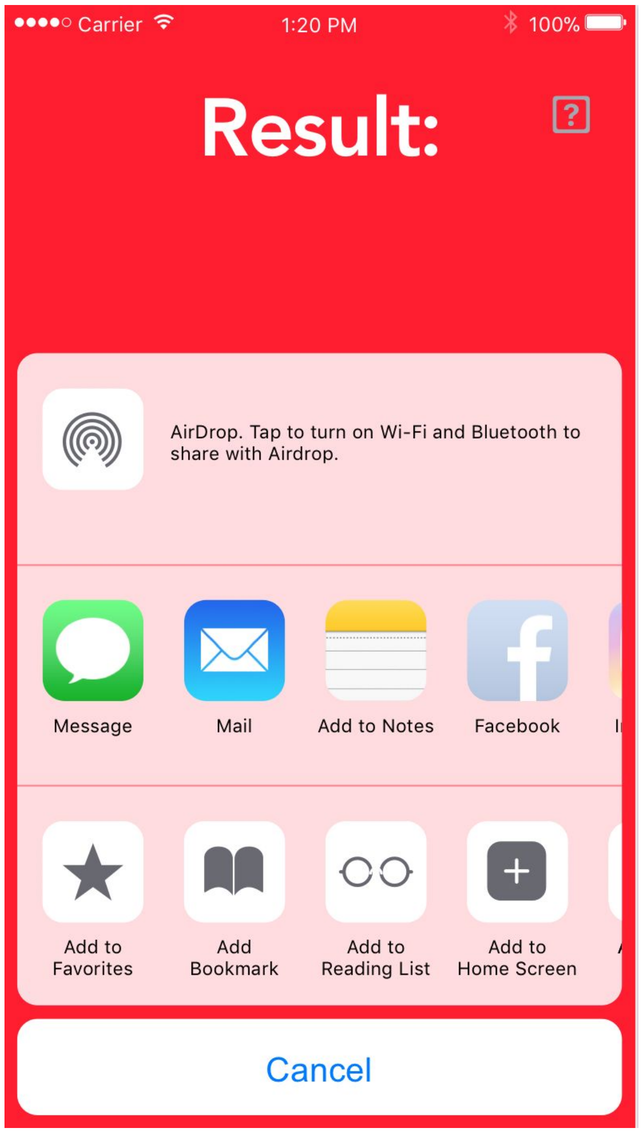


Figure 18: This is a mockup of the tutorial mode. In tutorial mode, nothing is clickable; the tutorial moves one stage forward if the user taps the left side of the screen, and moves to the previous stage if the user taps the right side of the screen.



Mean 0
SD 1
X Value 1
Z Value 1
P Value 0.841345
F Value 0.241971

Figure 19: This is a mockup of the advanced results screen, which gives additional statistical information about the prediction. The user can choose to return to the main menu or share the results by clicking the corresponding button.



AirDrop. Tap to turn on Wi-Fi and Bluetooth to share with Airdrop.

Figure 20: This image shows the share menu which appears after a user clicks the “Share Results” button.

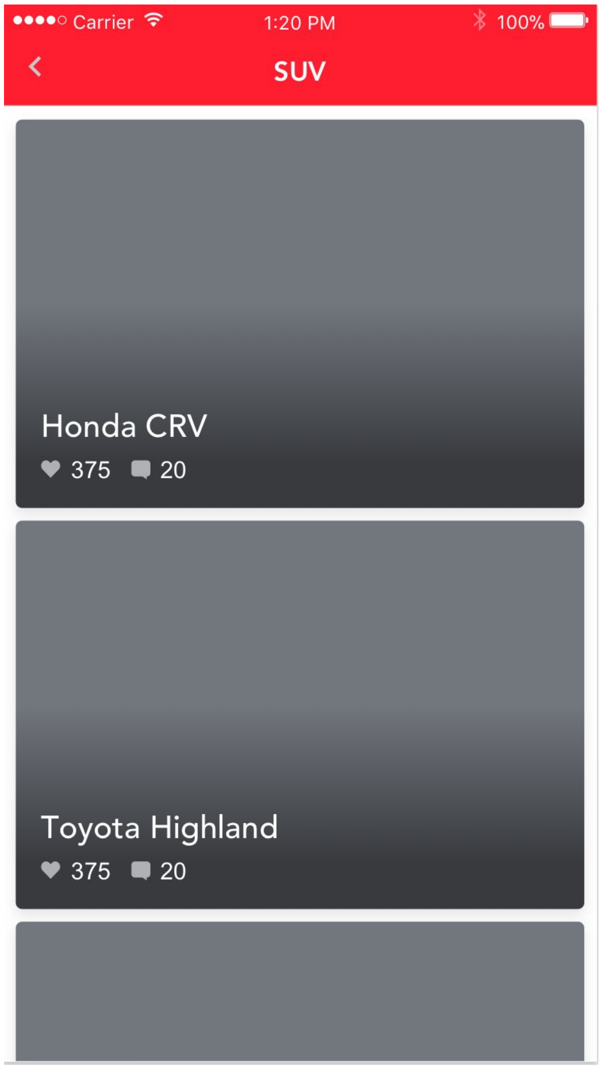


Figure 21: This image shows a mockup of the car information screen which appears if a user interacts with one of the cars recommended in the prediction results. In an implementation these boxes might show images of the car model in question, perhaps taken from a blue book website, with access to comments about the car or other social media details.

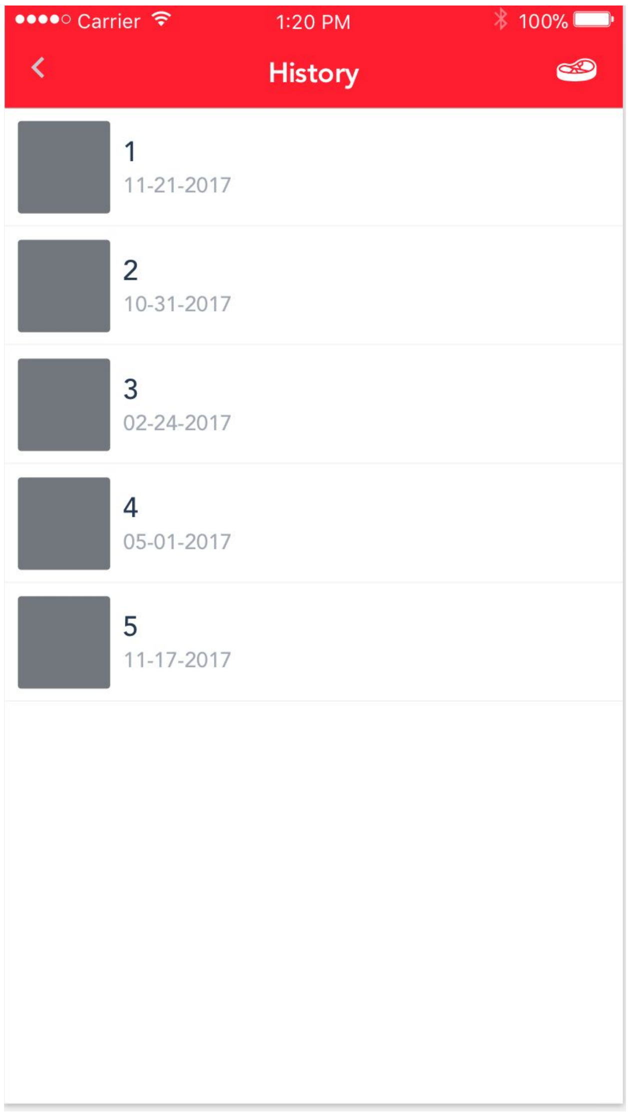


Figure 22: This is a mockup of the history feature of the application. Here the user can view all the past results for queries the user has previously run. If you click one of them, it will take you to the result page above. You can also click the back button in the top-left corner. The icon in the top-right corner is the edit button, which can be used to delete chosen records as necessary.

Glossary Updates

The glossary has been revised and updated from the previous deliverable with 5 additional entries, for a total of 25.

Accuracy- The ratio at which a predictive model such as a decision tree gives correct results when compared to actual outcomes.

Amazon Web Services (AWS)- AWS is a cloud computing platform offered by Amazon, and is widely used for enterprise computing and hosting needs. We intend to use AWS to host the server component of the project, as it will allow us to bring superior hardware resources to bear on the computational challenge of training the classification model.

Attribute- A property of an object or entry in a dataset, such as age or gender in a dataset of people. Corresponds to the columns in a tabular data set.

Classification- A type of data mining task where a predictive model is generated from a training data set. The model is then used to make predictions about new data.

Clustering- A type of data mining task where data entries are mapped to space, so that entries can be compared and classified in terms of their distance to one another.

Continuous variable- A type of entry in a data table whose values are represented by real numbers, such as temperature, height, or weight.

Data mining- The process of automatically discovering useful information or knowledge using large data repositories.

Decision tree- A technique for classification in which the available data is used to build a tree that branches based on the data attributes. By following this tree for a new data entry, the tree will predict some attribute of interest about that entry. For example, the tree may branch on the basis of the income of the individual being considered. For our purposes, at the end the tree will predict the car or type of car the user is likely to purchase.

Dimensionality- Refers to the number of attributes in a dataset. High degrees of dimensionality often make data mining algorithms more computationally expensive and less accurate. Several techniques to reduce the dimensionality of a data set exist.

Discrete variable- A type of entry in a data table whose possible values are finite or countably infinite. These can include binary attributes, i.e. yes or no, as well as counts and categories.

Gain ratio- A value used during decision tree induction. The gain ratio reflects whether and how much accuracy the tree would gain by splitting a node using one of the attributes.

Induction- Refers to the process by which a decision tree is built. Decision tree induction is a recursive, greedy process where splits are made so that the gain ratio is optimized.

Machine learning- Refers to the domain of computer science focused on enabling computers to discover or learn solutions to problems that have not been explicitly programmed. Many data mining tasks are also machine learning tasks, including the classification task we intend to implement for this project.

Neural network- An approach to data mining and machine learning that attempts to simulate the learning of new information by simulating a group of objects that behave like neurons in the human brain, forming connections amongst each other and firing off outputs in response to learned stimuli.

Overfitting- A phenomenon observed in predictive data mining tasks where a model that is trained too closely on a training set loses accuracy when applied to unseen test data. Overfitting should be avoided, and pruning is one approach to doing so for a decision tree.

Performance- We use performance to refer to both the time and memory needs of our data mining algorithm as well as the success rate of our decision tree against real world data (see accuracy).

Preprocessing- Refers to the process of 'cleaning up' messy, real-world data to prepare it for processing by a data mining. For example, values may need to be normalized, outliers may need to be excluded, and missing values may need to be handled.

Pruning- Used to reduce the overfitting of a decision tree by removing branches of the tree that hurt the tree's accuracy when used on test data. A variety of pruning approaches exist.

Record- An individual or entry in a dataset, such as one person in a dataset of people. Corresponds to the rows in a tabular data set.

Return on Investment (ROI)- ROI is a business term used to measure the efficiency of an investment. For our purposes we use ROI mainly from an advertising and marketing perspective, to reflect the efficiency of a marketing plan which will hopefully be improved by using our data mining approach. Marketing Evolution, our industry sponsor, is a company focused on creating marketing plans that maximize ROI.

Singular Value Decomposition (SVD)- SVD is a linear algebra technique which is commonly used in data mining to reduce the dimensionality of data by creating new attributes that are composites of the original attributes. Since many data mining algorithms perform better when dimensionality is reduced, SVD is one technique we will explore when preparing the used car dataset for processing.

Sampling- A preprocessing approach which reduces the number of records to be considered by randomly selecting a sample of the original data set. Can be used to achieve similar results at lower computational cost, so long as the sample is representative of the original data.

Similarity- A numerical measure of the degree to which two objects are alike- for example, the distance between two objects in some space. Similarity is used in clustering approaches to data mining.

Training- The process of building a predictive model using pre-existing, available data. In this project we will train our model on the 50GB dataset, for use on new data entries.

Visualization- The display of information as a graphic or table. Visualization is used to make relationships in data easier for humans to understand and analyze.

Contributions of Team Members

Figure 23 shows a table of primary team member tasks and time worked on each task.

Team Member	Task/Contribution	Time Worked
Patrick	Proofreading, editing	2 hours
	Abstract	30 minutes
	Introduction	1 hour
	Glossary Updates	30 minutes
	Contributions of Team Members	30 minutes
Mile	High-level and Medium-level Design	4 hours
Eric	Detailed Design	3 hours
Guang	Initial Hardware Design	45 minutes
	User Interface Design	2 hours

Figure 23: Table of group member contributions.

In addition to time spent working on individual sections, the team conducted several meetings to coordinate, discuss, and ensure the quality of the document. The team also met with a member of the teaching staff, Devrin Lee, to discuss questions and concerns. These meetings lasted approximately 5 hours in total.

References:

Fayyad, Usama, Gregory Piatetsky-Shapiro, and Padhraic Smyth. "From data mining to knowledge discovery in databases." *AI magazine* 17.3 (1996): 37.