A

REPORT

ON

The Kaggle Project
Acquire Valued Shoppers

By

Annu Sharma	36731206
Srinivas Gubbala	21317376
Venkata Trived Menta	87371010
Keke Zhai	98697900
Kishan Rao	68568951

Instructor: Prof. Daisy Zhe Wang

Course: Introduction to Data Science

14th, December 2015

PROBLEM STATEMENT

A number of consumer brands often offer discounts to attract customers to their products. The focus of these brands however, are the customers who come back for a repeat purchase after this initial incentive. Through this project, we try to understand and model this repeat purchase pattern in customers.

The customer transaction data (around 22GB of basket level transactions) provided did not make available any relevant features that could be used towards model building or prediction. Hence, we had to creatively engineer features in an iterative manner coupled with prediction, in order to arrive at the final set of most predictive set of features. The goal was to identify the most predictive features and subsequently the right tuned, most performing models (ensemble of models) that could be tested out on the Kaggle provided test data with reasonable accuracy.

WORK DISTRIBUTION

Module	**Contributors**
Exploratory Data Analysis	Venkata Trived Menta, Annu Sharma, Srinivas Gubbala, Keke Zhai
Data Cleaning / Reduction	Venkata Trived Menta
Feature Engineering	Srinivas Gubbala
Prediction	Annu Sharma
Data Visualization	Keke Zhai, Venkata Trived Menta, Annu Sharma
Feature Visualization	Kishan Rao

LITERATURE SURVEY AND NOVELTY

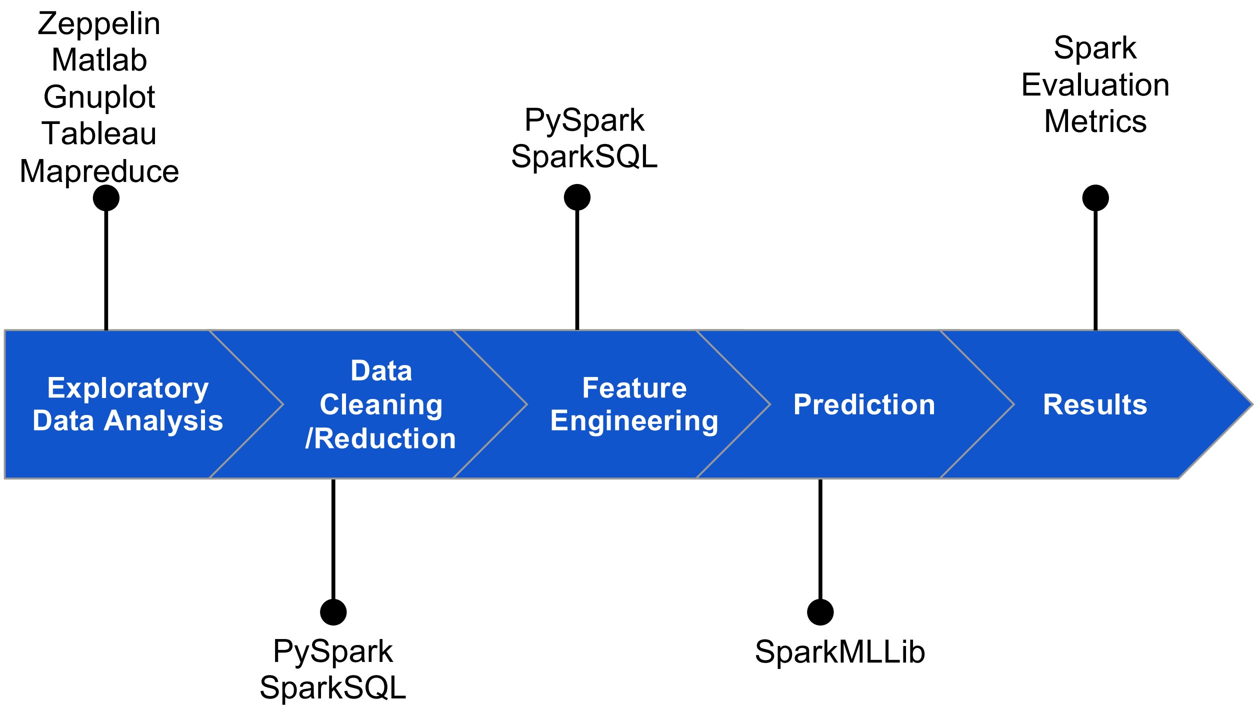
The problem was hosted on Kaggle on the 10th April 2014 and ran through till 14th July 2014. It saw participation from over 1165 participants with the winning team attaining a Kaggle score of 62.72. The problem was majorly approached using Vowpal Wabbit's Machine Learning library. From the discussions on the Kaggle forum, it could be inferred that a majority of the winners deployed Quantile Regression giving a score of around 53-54 along with a bunch of other ensemble models.

An interesting observation from the performance evaluations of previous participants was that the increase in score obtained from using Quantile Regression alone and it coupled with numerous other ensemble models was not significant. This indicated that quality of features generated was a major factor deciding the final score.

We approached Feature Engineering and Prediction in a novel manner, iteratively generating features and testing both our simple - Logistic Regression and Support Vector Machines as well as the Decision Tree based ensemble models - Gradient Boosted and Random Forest. With every iteration, we introduced increasingly detailed features. Prior to prediction, the newly created set of features passed through Univariate Feature Selection to ensure that models did get over fitted to the training data.

A number of novel (novel to the competition) ensembling methods to harness the performance of our best performing models were employed including taking a Majority Voting from Random Forest, Gradient Boost and Decision Tree model predictions.

DATA SCIENCE PIPELINE & TECHNOLOGIES



We can look into each of these stages and the associated work:

EXPLORATORY ANALYSIS

This step of the data science pipeline played a critical role in helping us get a clearer view of how the data was organized and gave us valuable insights, based on which contributed ideas for both the data cleaning task as well as feature engineering. Visualizing the dataset posed a challenge as all fields were anonymized and values were large ranged and continuous. We had to employ *MapReduce* coupled with other visualizing tools (listed below) to get meaningful and interpretable graphs. Some of the tools used for this purpose were:

- **Zeppelin** which is a web based notebook that enables interactive data analytics and has inbuilt support for Spark.
- **Matlab** which is a high performing language that integrates computation, visualization and programming in an easy to use environment.
- **GNU-Plot** which is a plotting engine having support for many third party applications.
- **Tableau** which is a interactive data visualization application.

A lot of interesting observations were made on the characteristics of the raw data including it having -

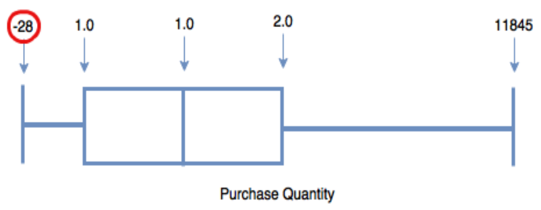
- More than 340 million transactions.
- Over 300k customers.
- As many as 2030 unique companies.
- 1503 unique brands.
- 210 unique categories of products.
- 134 unique chains.
- Transactions records were available for 03-05-2012 through 07-21-2013
- All fields in the dataset were anonymized

  Example: Company ID range - [1036030 - 10688819686]

  Customer ID range - [86246 - 4853598737]

  Category ID range - [201 - 9909]

We calculated *Summary Statistics* for the fields which were relevant to features derivation, them being - purchase amount and purchase quantity for the basket level transactions.



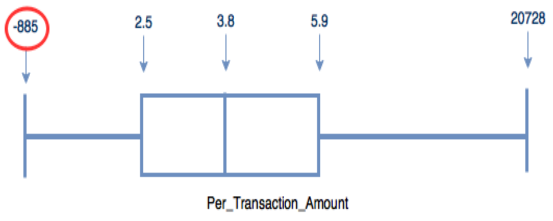


Figure. 1.1: Box and Whisker Plots for basket level transaction amount(left) and purchase quantity(right)

Remarks: We inferred that transaction amounts had negative values if the transaction is a for a returned product. Also, transactions where the purchase quantity is negative are irrelevant transactions and need to be cleaned.

Moreover, the data we had in hand portrayed a year worth of customer behavior and creative visualizations helped us identify insightful patterns in the data. Some of which are depicted below.

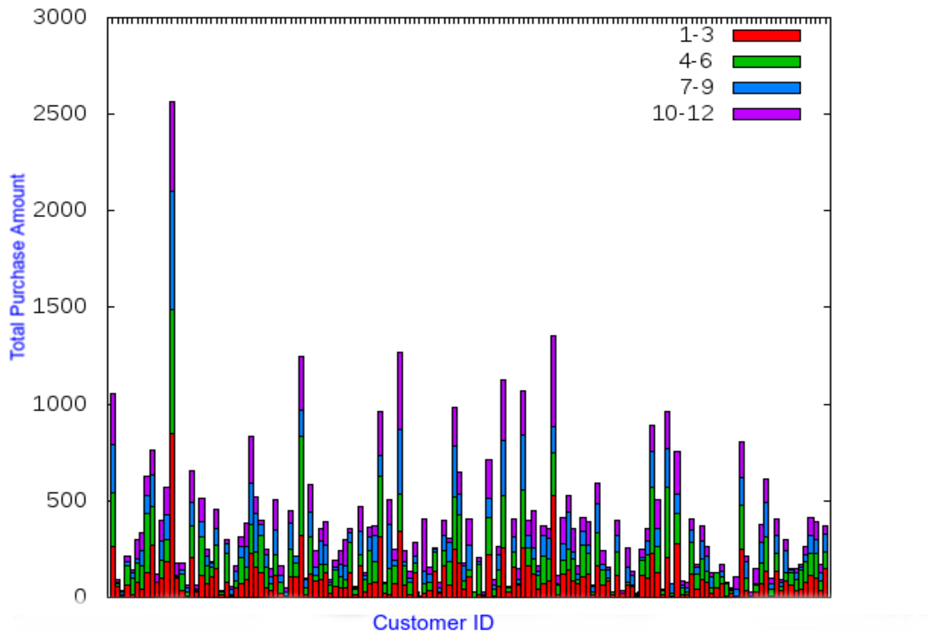


Figure. 1.2: Purchase amount (in dollars) of customers in each season

Remarks: From the stacked plot we observed that expenditures are highest in the months 1-3 and 10-12 which could be the optimum period for offers.

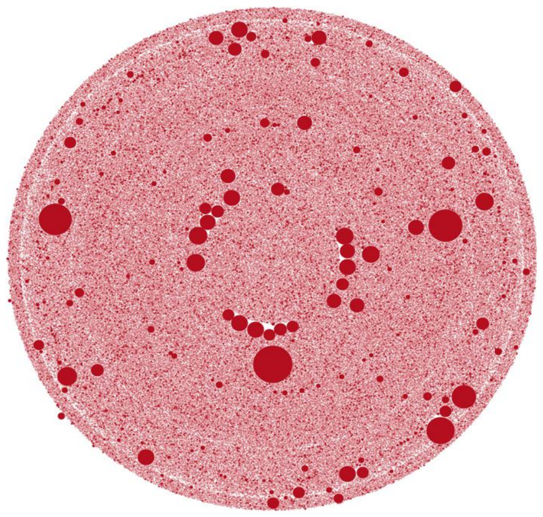


Figure. 1.3: Frequency of transactions by individual customers

Remarks: Each customer is represented by a circle and size of circle represents the customer transaction frequency relative to others.

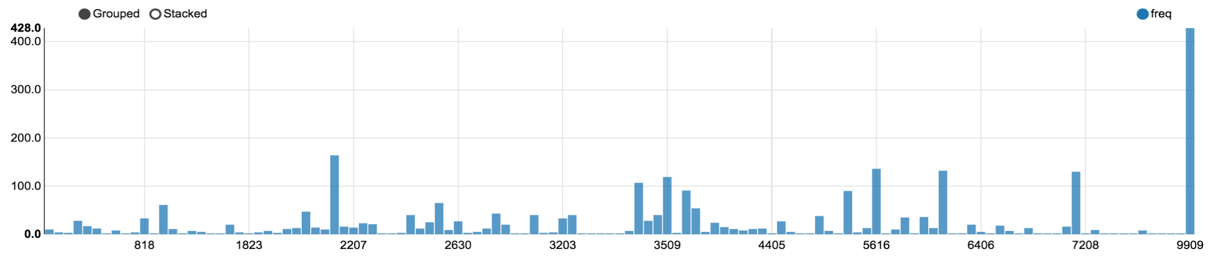


Figure. 1.4: Bar Plot of frequency of transactions in categories

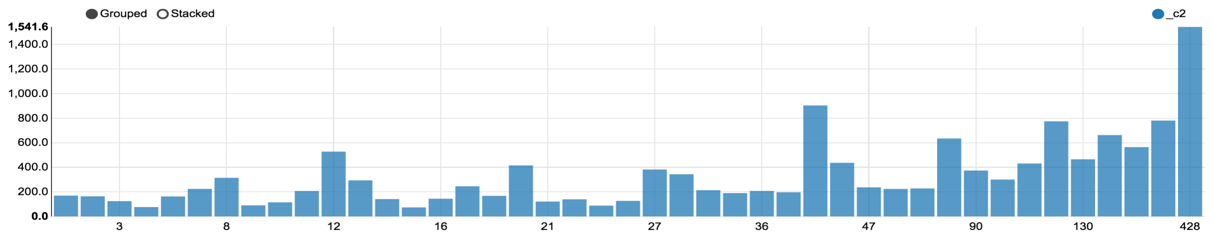


Figure. 1.5: Bar Plot of gross amount plotted against category frequency

Remarks: Fig 1.5 shows that category frequency is not linearly proportional to gross amount which indicates that the data is fairly realistically distributed and rules out outliers in our prediction in these fields.

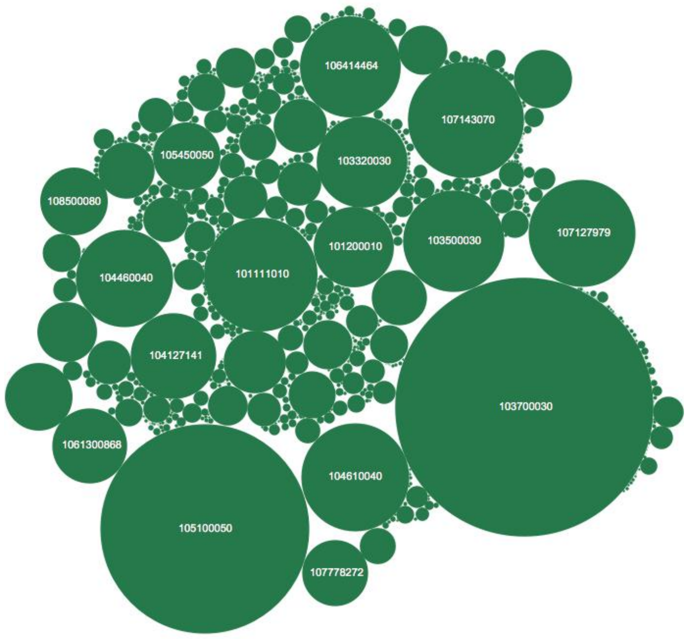


Figure. 1.6: Frequency of transactions in companies

Remarks: Each company is represented as a circle, value inside the circle is Company ID and size of circle represents transaction frequency of the company relative to others.

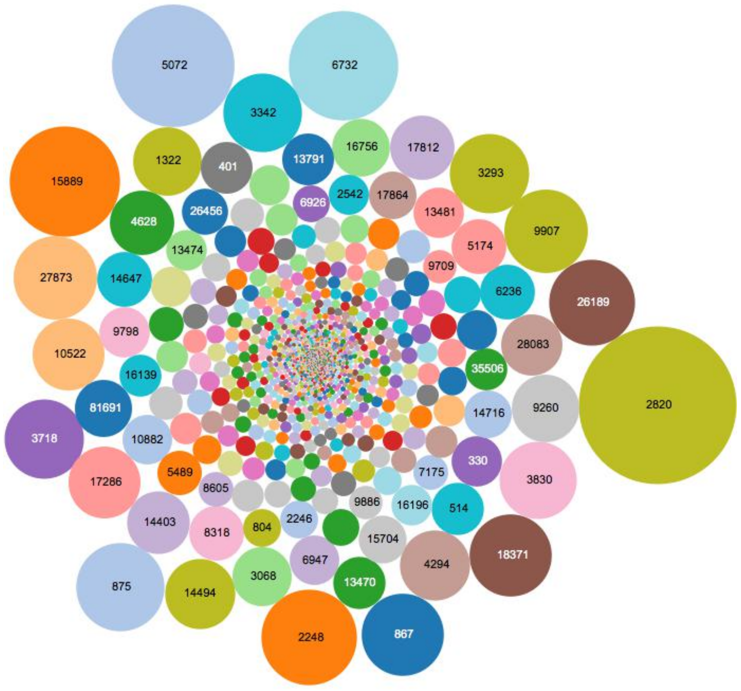


Figure. 1.7: Frequency of transactions in brands

Remarks: Each brand is represented as a circle, value inside the circle is the brand id and the size of circle represents transaction frequency in the specific brand relative to others.

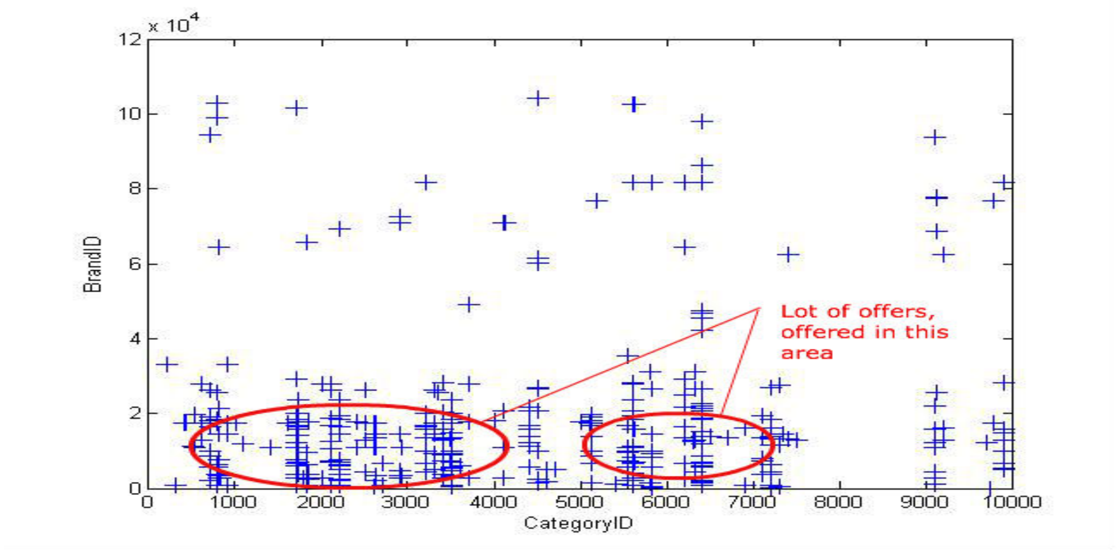


Figure. 1.8: Mid range grossing brands in each category

Remarks: We selected mid range grossing brands based on number of transactions of a brand for a particular category. We mapped these entries to the offers file and observed that a lot of offers were made from these mid range brands.

More visualizations can be found in the github repository link specified in the end.

DATA CLEANING

The main objective of the cleaning task was to filter out transactions from the transactions data set which did not give value to predictions and contained critically missing/erroneous data. Given the humongous size(22GB) of the transactions dataset such errors were likely to be present and were detected in the exploratory analysis. Based on inferences from summary statistics and observations from the data, the following cleaning tasks were identified. We used PySpark and SparkSQL for data cleaning. We filtered transactions:

- Where the Purchase Quantity was a negative value or zero
- Where the Purchase Amount was zero or missing
- Where any of these fields had NULL or negative values:
  Customer ID, Company ID, Category ID, Brand ID, Chain ID

REDUCTION

The objective of the reduction task was to eliminate records which did not give value to the model training or depreciated the quality of the data. For example, transactions which did not deal with a category or even a company associated with any of the offers pool records were pruned. This was carried out in the following steps using Python scripts.

1. We filtered transactions where the Category_ID mapped to at least one of the Category_IDs mentioned in the offers file as one data set.
2. Similarly, we filtered transactions where the Company_ID mapped to at least one of the Company_IDs mentioned in the offers file as another dataset.
3. We took a union of both the data sets to remove duplicates.
4. The reduced file of size 2GB was then used for further analysis in the project.

FEATURE ENGINEERING/ FEATURE PROPERTIES VISUALIZATION

One of the major challenges we faced was the unavailability of relevant features that could be used towards prediction of repeat purchase patterns in customers. This called for creatively engineering features while making sure that we did not detail it to a degree to cause the models to overfit and increase the generalisation error. We considered factors that could influence repeat purchase behaviours - including a few inspired from the RFM (Recency, Frequency, Monetary value) market model. The features were infact iteratively engineered, with every iteration increasing details in purchase behaviors. The features were generated by performing operations relevant joins and calculation over the four tables - Train History, Test History, Offers and Transactions files. The results were merged into two comprehensive Spark dataframes containing the engineered features - one for the Test History and other for the Train History transactions. Around 15 features were constructed keeping the following factors in mind:

- Purchase value and quantity
- Purchase frequency
- Purchase timeline
- Satisfaction level with company
- Brand affinity
- Customer loyalty to company
- Chain visit frequency

The following features were generated iteratively:

Feature	Explanation
PURCHASE_TIMES_COMPANY	Number of times the customer purchased

	goods of that company
PURCHASE_VALUE_COMPANY	Total value of the purchases made by the customer on goods of that company
PURCHASE_QUANTITY_COMPANY	Total quantity of goods belonging to that company purchased by the customer
OFFER_COMPANY_180	Number of times the customer purchased goods from the offer company in last 180 days
OFFER_COMPANY_60	Number of times the customer purchased goods from the offer company in last 60 days
OFFER_COMPANY_30	Number of times the customer purchased goods from the offer company in last 30 days
PURCHASE_TIMES_CCB	Number of times the customer purchased goods that belong to the offer category, company and brand
NEVER_BOUGHT_CCB	Whether the customer purchased goods that belong to the offer category, company and brand or not
PURCHASE_TIMES_CAT	Number of times the customer purchased goods from the offer category
NEVER_BOUGHT_CAT	Whether the customer purchased goods that belong to the offer category or not
AMOUNT_SPENT_CAT	Total amount spent by customer on the offer category
BOUGHT_CATEGORY_30	Whether the customer purchased goods that belong to the offer category or not in last 30 days
CHAIN_VISIT_FREQ	Number of times the customer visited the chain
RATIO_RET_BGHT_CC	Number of times the customer returned goods over the number of times purchased

Feature engineering was also supported by further insights into the data deploying visualizations as such:

- Distribution of the amount spent on various categories.
- Popularity of various chains measured by frequency of visits to the chains.
- Distribution of amount spent on various companies by customers.

PREDICTION

The prediction task involved iteratively testing well performing models over newly engineered features. As the dimensionality of the feature set was large, we implemented Feature Selection prior to the actual prediction to have a simpler model, increase efficiency in training and most importantly, avoid over fitting our models. Feature selection was implemented using Univariate Feature Selection that determines the contribution of each variable to the response of the classifier. It uses Mutual Information as the correlation metric.

A lot of times simple models performance are just as good as a complex ensemble of models. We exploited this possibility by looking at the prediction accuracies of simple Logistic Regression and Support Vector Machines and then moving on to more complex ensemble models.

We iteratively constructed increasingly detailed features and measured the performance of the binary classification models under consideration. The training data was divided into training and testing sets in 7:3 ratios and tested the performance (figure 2.1) on the test data using random splitting.

In figure 2.1, we consider the performance of five models Logistic Regression, SVM, Decision Trees, Random Forest and Gradient Boosted Models with various randomized splits on the training data. We notice that the ensembles, Gradient Boost and Random Forest are among the better performing models along with Logistic Regression while SVM has a very erratic performance. We reason that this is because SVM’s do not perform very well on imbalanced data sets – where the classes are not equally represented, much like our case – where repeat purchases happen much lesser.

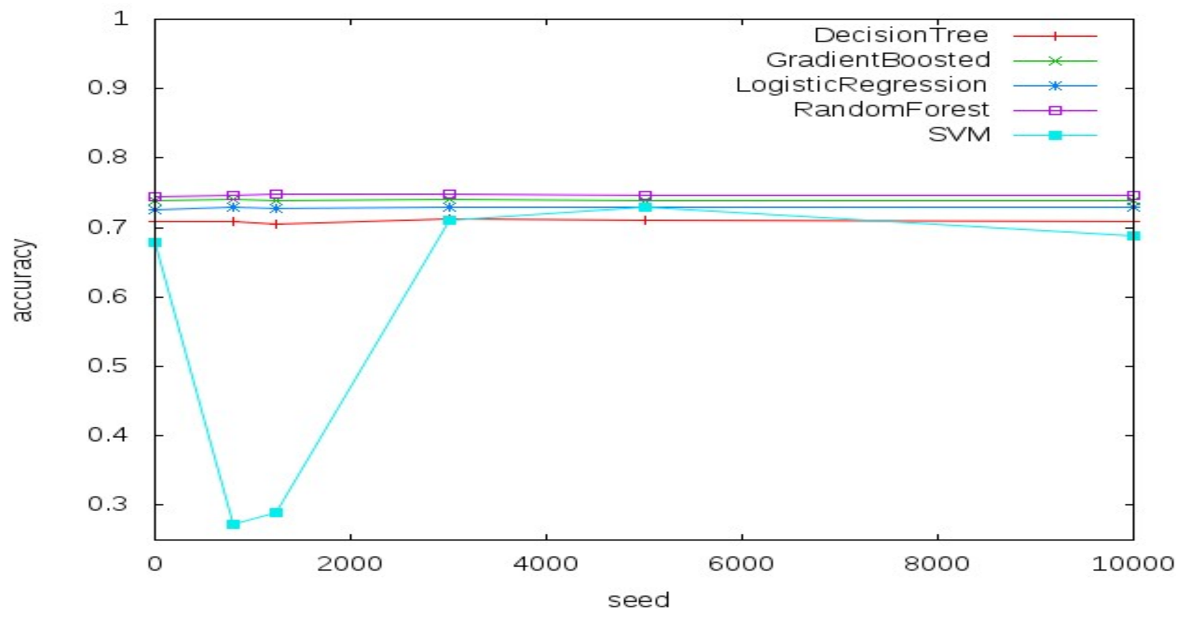
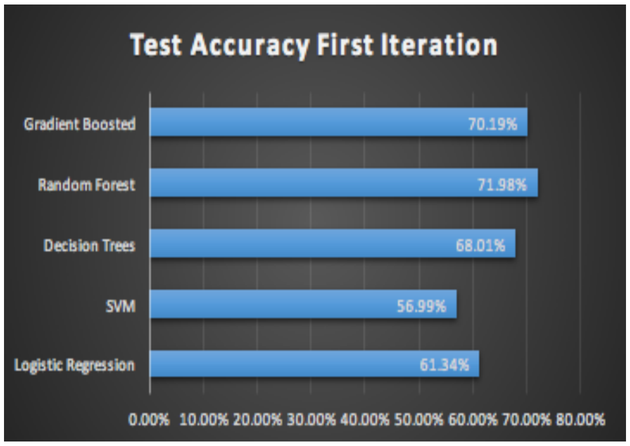


Figure 2.1: Performance of models across random splits of training data

In figure 2.2 we analyze the accuracy of predictions of the five models, comparing accuracies from the first iteration through to the final iteration after which the models started over fitting. We notice that Decision Tree and the ensembles - Random Forests and Gradient Boost are the best performing and in fact their accuracies improve by +3% through the iterations. It is important to note that iterations mentioned here are the iterations over Feature Engineering, Feature Selection and Prediction and not
Iterations of the algorithm.



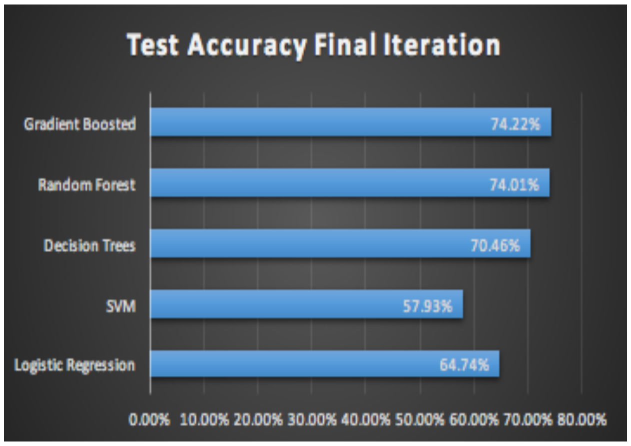


Figure 2.2 Test sample accuracies in first and final iteration

We also measured model performance using area under the ROC (Receiver Operating Characteristic) curve (figure 2.3) which was also used by Kaggle for the final evaluation. We could notice similar trends of Decision Trees and the ensemble models performing the best.

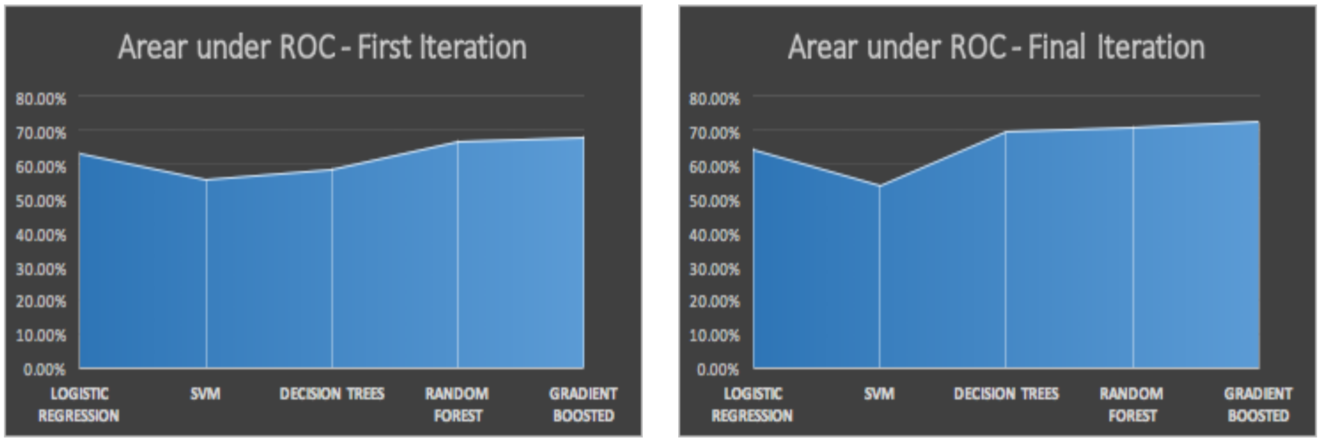


Figure 2.3 Test sample performance using area under ROC curve

In a similar manner as described above, for every iteration, we made Kaggle submissions for our better performing models, that being all except SVM. We noticed a similar increase in Kaggle score of +3 over the iterations. Random Forest and Gradient Boost gave the best predictions and we hit a final score of **55.89** with the Gradient Boost Ensemble. In the figure (figure 2.4) below, we show our progress in subsequent submissions using increasingly detailed features.

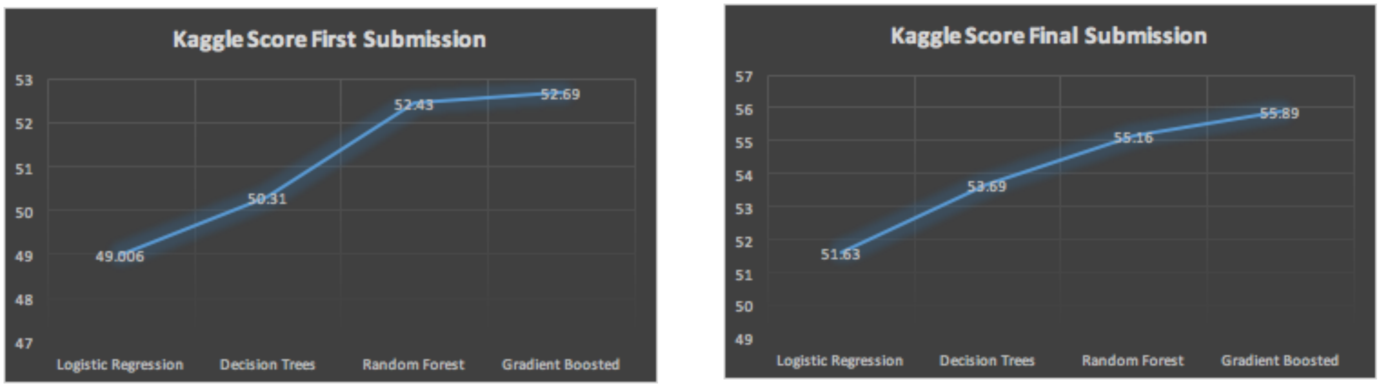


Figure 2.4 Kaggle score trends over the iterations

EVALUATION METRICS

We used Spark Evaluation Metrics such as accuracy and area under the ROC (Receiver Operating Characteristic) Curve to asses the results of the prediction task when testing on the training data (split of the training data). The ROC curve plots the True Positive Rate against the False Positive Rate against various thresholds. These evaluations have been visualized above in figures 2.1,2.2 and 2.3. Moreover, the Kaggle final score was calculated factoring in the area under the ROC curve, which too has been visualized above in figure 2.4.

DATA PRODUCT AND FUTURE SCOPE

The final data product built gives a fair prediction on the chances that a customer will become a repeat buyer of a product after an initial incentive. It models the repeat purchase patterns in customers and helps any company/brand to reach out and invest in the right subset of customers.

There are a lot of interesting dimensions one could extend this project to. We could build recommender systems which would do bundling of products using Apriori (i.e. bundle a product which is repeatedly bought with products frequently sold with it) which also builds customer brand loyalty. We could also try to see the result of ensembling our best performing models using Gradient Boost. We experimented a similar ensembling using the Majority Voting approach which did not influence the accuracy majorly so other, more creative ensembling methods could be experimented with. Moreover, we could also offer customized coupons to customers harnessing the Rule Base of Decision Trees generated.

OBSERVATIONS AND RESULTS

We further observe the critical role of selecting the right features while not over fitting the model to the data. This had to be done meticulously, doing the right degree of feature selections.

Overall ensemble models gave us the best results and we can attribute that to a number of inherent attributes of ensembles – they handle outliers, discover interactions between features and minimize the variance in the prediction error. We can also draw a comparison between Logistic Regression and SVM and see that Logistic regression fared better than SVM as it can handle imbalanced data better, where the (two)classes are not equally represented.

ALGORITHMS DEPLOYED

Spark MLLib provided support for the prediction algorithms for Logistic Regression, SVM, Decision Trees and the ensembles. Scripts were written for majority voting predictions using the best performing models.

KEY LEARNINGS

Of all the individual learnings we could enumerate, most importantly, we learned how to deal with large datasets and construct a data science pipeline for a big data problem. Building a generic data science pipeline is not an easy task. Depending on the nature and distribution of data, approaches and tool sets vary for each stage of the pipeline. Understanding the problem and properties of the dataset and consequently matching technologies to employ was a major skill the project helped us sharpen. Visualizing large range continuous data came as a challenge too and we found that MapReduce and binning techniques were effective strategies to deal with it. Feature engineering, which is covered very less in literature too was a skill indispensable to the project. Engineering the right features decided the success of the project to a major extent. On a technical ground, we understood the Apache Spark framework and employed it for data cleaning and reduction, feature generation as well prediction tasks.

CHALLENGES

One of the biggest challenges that we faced was coming up with a predictive subset of features and filtering them to avoid over fitting and making our models unnecessarily complex. Another equally challenging task was to visualize the large volume of data all of which had large range continuous values. We experimented with different prediction models such as Decision Trees, Gradient Boosted, Logistic Regression, Random Forest and SVM, with the best(ensembles) achieving 74% accuracy on sample test dataset (partition from train history), however submissions for these best performing models on Kaggle achieved the highest score of 55.89(Gradient Boosted). Moreover, an ensemble of the performing models - using Majority Voting did not show a substantial increase in accuracy which was a major blocker to bettering our score/accuracy.

Link to Github Repository (Contains the exhaustive scripts, visualizations used in the pipeline) --- https://github.com/AnnuSharma/IDS-Final-Project.git