Coupon Recommendation System

Naïve Baes

ECS171
Term Project

Yongkang Zhao	John Chen
Xiaodong Yan	Baotuan Nguyen
Yilun Yang	Nguyen Ngo
Shuxin Li	Anthony Tran
Jizhou Chen	Edwin Hao
Jiebin Sun	Cindy Huang

Naive Baes

November 30, 2015

1 Abstract

Ponpare is Japan's leading joint coupon site, offering huge discounts for a wide variety of products and services. To provide its customers with appealing offers, Ponpare utilizes a recommendation system which takes into account a user's browsing and purchasing patterns. In this paper, we will outline the development of our own recommendation system which uses Ponpare's data sets to predict a user's preferred coupons. This was achieved through employng a variety of techniques, including Random Forest, Bagging, Boosting, Matrix Completion, SVM Classification, K Nearest Neighbors, in addition to coupling an Artificial Neural Network with K-Means Clustering. Applying these techniques to a refined version of Ponpare's data resulted in a collection of methods which exhibited varying degrees of success at predicting coupon suggestions for a given user.

2 Introduction

Recommendation systems are designed to generate meaningful suggestions to a collection of users for products or offerings which might be of interest to them. Recommendation systems are primarily classified based on their method of analyzing data sources. Collaborative Filtering systems analyze historical interactions in order to develop proposals. On the other hand, Content-Based Filtering systems create inferences based on the attributes derived from profiles and products[1].

With Collaborative Filter systems, the recommendation engine relies on on past user interaction patterns in order to match users with products they may be interested in. Using this model, the recommendation system clusters new into groups based on their behavior and provides predictions based on patterns exhibited by other users within the cluster. This approach addresses the need to model a user's inclinations. By analyzing actual purchasing patterns, causal relationships among purchases are better explored, leading to a higher success rate [2]. However, clustering users based on their similarity may result overfitting or overgeneralizing, leading to esoteric suggestions which fail to consider a user's individuality. Moreover, Collaborative Filtering systems are dependent upon data for predictions, making it impossible for it to act as a "cold start" recommendation engine which lack user or product history.

With Content-Based Filtering systems, the engine matches users with products or services based on their properties. Labels and attributes are attached to each item within the system, and users are inquired for attributes they are seeking within their desired item. Using this approach, a system attempts to fully capture a user's needs and suggests the products which offer the best fit based their profile. Content-Based Filtering systems place emphasis on providing the best suggestions based on a user's immedate needs rather than predicting their future desires. Thus, the approach successfully solves problems which arise from attemping to suggest from a large amount of fungible goods, where Collaborative Filter systems are unsuitable. In addition, it is capable of acting as a "cold start" system, providing suggestion for users with little historical data. Unfortunately, this method comes with its flaws. Content-Based Filtering systems are poorly suited for suggesting items that are highly complementary but does not match the profile given.

Our recommendation system integrates aspects of both Collaborative and Content-Based systems. Like Collaborative systems, suggestions are made based on the purchase history of other users who exhibit similar behaviors. In addition, our system takes as input a summary of each user and coupon and utilizes those attributes for prediction. The result is a hybrid reccomendation engine which can employ different techniques to produce a diverse set of recommendations.

3 Data Exploration

Before constructing data into meaningful way, we want to get an idea of what each attribute look like. We think the location of the coupon may have a high impact to weather the user will purchase the coupon or not.

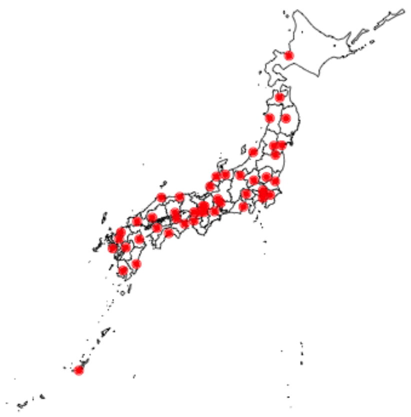
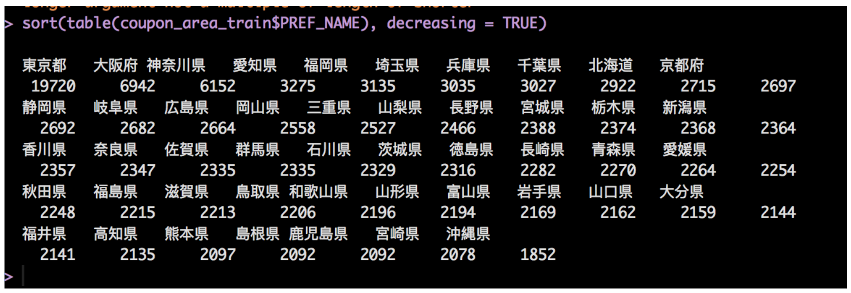


Figure 1:

There are more than a million interactions between users and coupons. Each coupon belongs to a genre, we think the genre is highly correlated with the viewing and purchasing behavior of the user and we went ahead and checked how many coupon there is in each genre. We found out that in each genre, there are plenty of coupons to justify keeping all genres in the training set.



```
> sort(table(coupon_area_train$PREF_NAME), decreasing = TRUE)

 東京都    大阪府  神奈川県    愛知県    福岡県    埼玉県    兵庫県    千葉県    北海道    京都府
  19720      6942      6152      3275      3135      3035      3027      2922      2715      2697
 静岡県    岐阜県    広島県    岡山県    三重県    山梨県    長野県    宮城県    栃木県    新潟県
   2692      2682      2664      2558      2527      2466      2388      2374      2368      2364
 香川県    奈良県    佐賀県    群馬県    石川県    茨城県    徳島県    長崎県    青森県    愛媛県
   2357      2347      2335      2335      2329      2316      2282      2270      2264      2254
 秋田県    福島県    滋賀県    鳥取県  和歌山県    山形県    富山県    岩手県    山口県    大分県
   2248      2215      2213      2206      2196      2194      2169      2162      2159      2144
 福井県    高知県    熊本県    島根県  鹿児島県    宮崎県    沖縄県
   2141      2135      2097      2092      2092      2078      1852
>
```

Figure 2:

We suspected that there might be a small group of user who purchases coupon. The plan was to treat buyers and non-buyer separately if that is indeed the case.

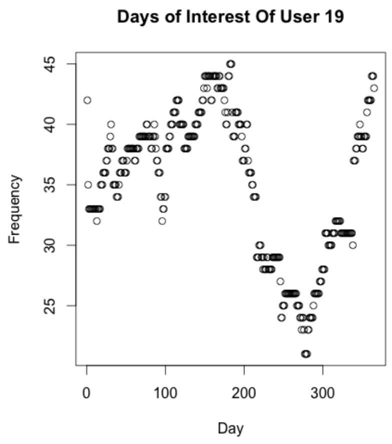


Figure 3:

Lastly, in Japan. Holiday time and Weekend time are a rare currency to people. For some company, employee are only allowed Sunday off, while some other companies allows employee to have both weekend day off.

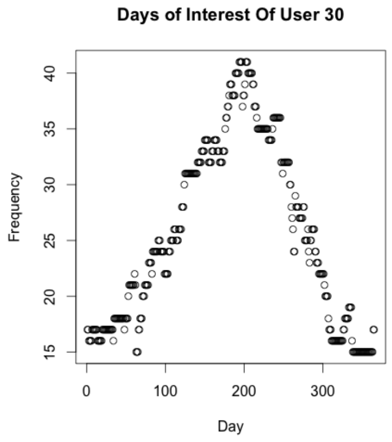


Figure 4:

So far we have found 3 aspect that might help us relate a user and a coupon. Genre, location and date of availability. In the following sections, we will talk about how we construct the data and how its used in different models.

4 Method

4.0.1 Data Processing

Since original dataset has about 16 millions observations, we use sampling method to avoid the enormous computation. We randomly select 10percent users from nearly 20000 users list. From plot, it shows that the two distribution of users' age randomly selected and all users' age are similar and the ratio of female anagist male for the small sample and original dataset is also similar. We can conclude that the sample from random sampling method can reflect meaningful information about original dataset.

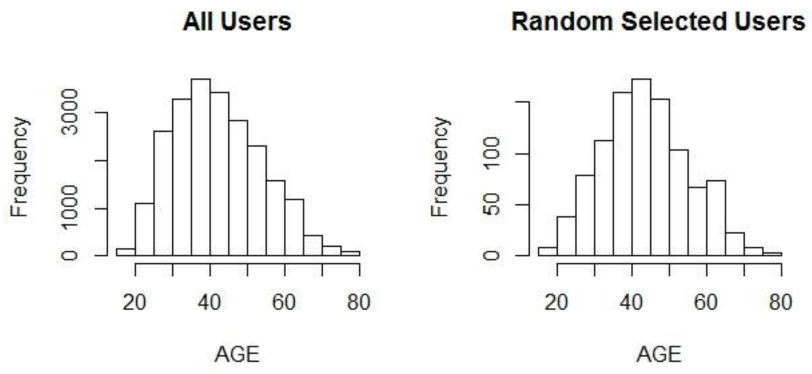


Figure 5:

4.1 ANN+Distance Calculation

Our first methodology is a two-step process. We first use an ANN to predict the characteristics of the ideal coupon for each given user. This ANN takes in information on each user as input, which includes demographic data (age, gender, location, etc.) as well as daily activity history of the user on the site. From this, the ANN outputs a vector of continuous values, which are a prediction of the characteristics of an ideal coupon for the user, which includes discount amount, coupon location, and coupon genre. We then find the distance (L1, L2, etc.) of each coupon in our dataset to this predicted value. This gives us a metric of how different every coupon is from our prediction. Finding the coupon with the minimum distances will give us our best predictions. We played around with different measurements and found the Euclidean distance, L2 to be most meaningful. The following table outlines the input as well as output of our ANN.

Artificial Neural Network – Economics Approach This approach attempts to model users' preference in order to make recommendations based on preference.

Gender	Binary	0,1	Male is 1, female is 0
Age	Integer	15,22,35,50, etc	
Location	Binary Vector	L1 L2 L3 L4 USER 1 0 0 1 0 USER 2 0 1 0 0 USER 3 1 0 0 0 USER 4 0 0 0 1	Dummy variable
Useable X1 – X365	Integer	[30 32 20 10 …] Viewed 30 coupons useable on Jan.1 Viewed 32 coupons useable on Jan.2 Viewed 20 coupons useable on Jan.3 Viewed 10 coupons useable on Jan.4	Frequency Count
Discount 10% - 100% 10 variables	Integer	normalize[2 4 5 20 93 40 50 30 5] → [0.0080 0.0161 0.0201 0.0803 0.3735 0.1606 0.2008 0.1205 0.0201] Viewed 2 coupons that has a discount from 0% - 10% Viewed 4 coupons that has a discount from 10% - 20% Viewed 5 coupons that has a discount from 20% - 30% Viewed 20 coupons that has a discount from 30% - 40% Viewed 93 coupons that has a discount from 40% - 50% Viewed 40 coupons that has a discount from 50% - 60% Viewed 50 coupons that has a discount from 70% - 80% Viewed 30 coupons that has a discount from 80% - 90% Viewed 5 coupons that has a discount from 90% - 100%	Normalized Frequency Count
Coupon Useable Location(s): 48 variables	Numeric	Same concept as Discount, but instead of counting discount, this one is counting the number of coupons that the user views is useable at different location	Normalized Frequency Count
Coupon Genres Type 12 variables	Numeric	Same concept as Coupon Useable Location. This one is counting the number of coupons that the user views is associated with what genre.	Normalized Frequency Count

Figure 6:

With this dataset, we can use Coupon Useable Location Coupon Genres Type and Discount that is associated with each user as each user's preference for coupon. Note all these are normalized per user. Which is going to be a vectors of numbers range from 0 to 1. While 1 represent the user's most preferred case, and 0 represent the least preferred case.

Stage 1: Artificial Neural Network Diagram.

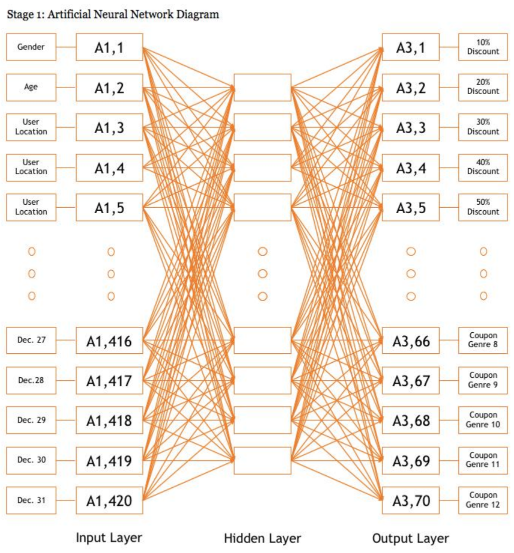


Figure 7:

Stage 2: Find Distances from our Coupon Dataset

For each coupon in our dataset, we find the L2 distance from that coupon to our prediction from the ANN. Effectively, this assigns each coupon a proximity to our prediction. We then recommend the closest coupons to the user.

4.2 Matrix Completion

The coupon problem can also be modeled as a classical recommendation system by using matrix completion. We can make a sparse matrix storing coupon purchase quantity between each user and coupon, we denote it as A. There are many blank cells in the matrix corresponding to the missing values or unreported values of purchase quantity. A can be factorized as $\mathbf{W} * \mathbf{H}^\top$, where$Wand\mathbf{H}^\top$ are the pattern matrices for users and coupons. The latent feature space (column space for user and row space for coupon) contains informative yet unknown features of user and coupon just like the principle components in PCA and factors in the factor analysis. Number of features in the latent space can be determined by cross-validation. The main idea of matrix completion is to use W and H which we can get by stochastic gradient descent to predict those missing values and make the recommendation. Below if the plot for this method.

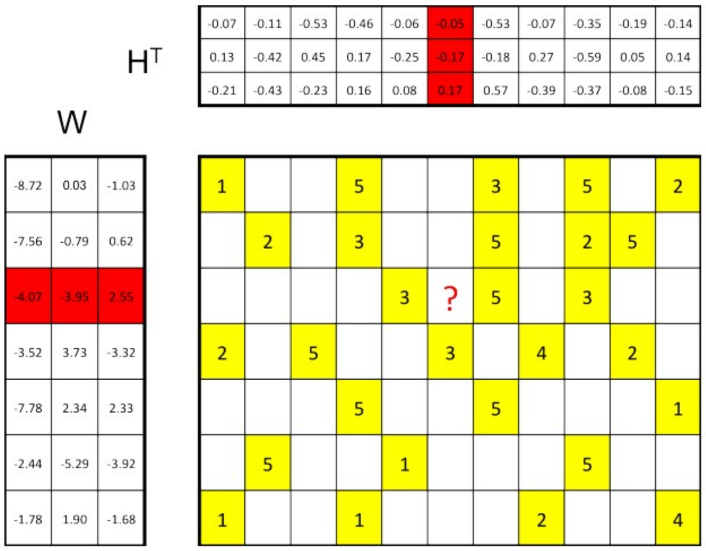


Figure 8:

Similar to other machine learning algorithms, this method also has an objective function to minimize.

$$min_{w \in R^{m*K}, H \in R^{n*k}} \sum_{(i,j) \in \Omega} (A_{ij} - \mathbf{w}_i^\top h_j)^2 + \lambda(||W||_F^2 + ||H||_F^2) \tag{1}$$

- Ω=$((i,j)|A_{ij} is obserbed)$
- Regularized terms to avoid over-fitting

This function can be optimized by Alternative Least Square($O(|\Omega|k^2 + nk^3)$)or the stochastic gradient descent($O(|\Omega|k)$),we will implement the program with stochastic gradient descent for speed concern. The W and H can be alternatively optimized as following until convergence:

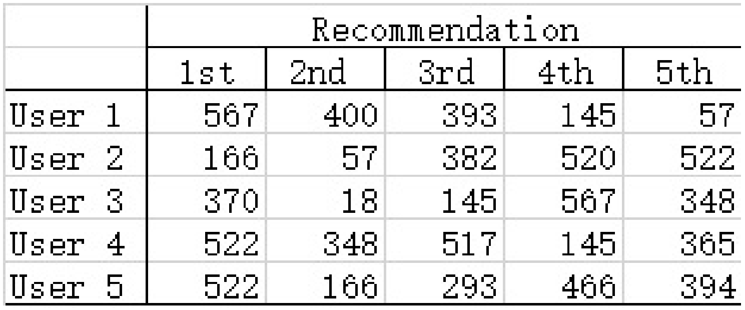
For t=1,2,... Randomly pick a pair(i,j)$\in \Omega$

$$w_i \leftarrow (1 - \frac{\eta_t \lambda n}{n_i^w}) w_i - \eta_t(\mathbf{w}_i^\top h_j - A_{ij}) h_j \tag{2}$$

$$h_i \leftarrow (1 - \frac{\eta_t \lambda n}{n_i^w})h_i - \eta_t(\mathbf{w}_i^\top h_j - A_{ij})w_j \tag{3}$$

In detail, first we need to redesign our sparse matrix for memory concern since there are so many missing values. We only stores the non-zero values and their row and column indices in our program. Next I applied cross-validation to select regularized penalty, learning rate and number of features in latent feature space. It took us nearly three hours to get the result since there are more than 1 million observations in the dataset. We found the parameters are not that sensitive as we imagined provided magnitude is correct thus we decided to pick small parameters to reduce calculation. The final parameters are 0.01 learning rate, 3 features in latent space and 8 for penalty term.

Finally, we used the matrix to calculate all missing values in the matrix and recommend each user the coupons with highest quantity prediction. Below is the recommendation coupons for first five users. Numbers represents the coupon number, the corresponding coupon hash ID and user hash ID can be retrieved in the code.



	Recommendation				
	1st	2nd	3rd	4th	5th
User 1	567	400	393	145	57
User 2	166	57	382	520	522
User 3	370	18	145	567	348
User 4	522	348	517	145	365
User 5	522	166	293	466	394

Figure 9:

4.3 SVM

Since we have learned SVM in class, I will not give too much theory in the report. Instead I will show more details on how to solve the dual problem (Sequential Minimal Optimization) to get our coefficients and prediction.

$$max_\alpha W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2}\sum_{i,j=1}^{m} y^{(i)}y^{(j)}\alpha_i\alpha_j < x^{(i)}, x^{(j)} > \tag{4}$$

$$s.t.\, 0 \leq \alpha_i \leq C, i = 1, ..., m \tag{5}$$

$$\sum_{i=1}^{m} \alpha_i y^{(i)} = 0, \tag{6}$$

$$\alpha_1 y^{(1)} + \alpha_2 y^{(2)} = -\sum_{i=3}^{m} \alpha_i y^{(i)} \tag{7}$$

Each $\alpha_i\, and\, \alpha_j$ can be rewritten in the above form. A good idea to update αs is to randomly pick out two elementsα_i, α_j to update and keep others fixed in each iteration, we keep up iterating until all alpha converge. In detail, we replaceα_i in the form of α_j and get them back into our objective function to optimize.

$$W(\alpha_1, \alpha_2, ..., \alpha_m) = W((\zeta - \alpha_j y^{(j)})y^{(i)}, \alpha_j) \tag{8}$$

Then the objective function is only a quadratic function of α_j so that optimization would be very easy and fast, after we get α_i with the equation above. Note there is still and box constraint in the dual form so that there would be a upper bound and lower bound for α_j in each iteration.

$$\alpha_2^{new} = \begin{cases} H, & \text{if} \alpha_2^{new,unclipped} > H \\ \alpha_2^{new,unclipped}, & \text{if} L \leq \alpha_2^{new,unclipped} \leq H \\ L, & \text{if} \alpha_2^{new,unclipped} < L \end{cases}$$

New α_j has to be between lower bound and upper bound to be updated, otherwise, either lower bound or upper bound would be its update. After we getα,w and $\hat{y}$ can be got by:

- At optimum : $W = \sum_{i=1}^{n} \alpha_i y_i \varphi(x_i)$
- Prediction : $\mathbf{w}^\top \varphi(\hat{x}) = \sum_{i=1}^{n} \alpha_i y_i K(x_i, \hat{x})$.

We have reformed our dataset with a binary variable "Buy or Not Buy" as our response variable and location, time, etc as our feature space. Then we can apply the binary SVM to our dataset to get the prediction. MSE of 10 folds cross-validation is 0.2230

4.4 Random Forest

Just as SVM, the response variable in random forest is "Buy or Not Buy". Random Forest is similar to bagging to build a number of decision trees on bootstrapped training samples. However, in random forest, only part of variables is selected in each split thus can decorrelate the trees and reduce variance. Since random forest is also a tree-based model, we can include all the variables we have.

We found CATALOGPRICE, DISCOUNTPRICE and age are the most important variables to explain Crime in reducing Gini index and mean accuracy. The error rate is 0.191.

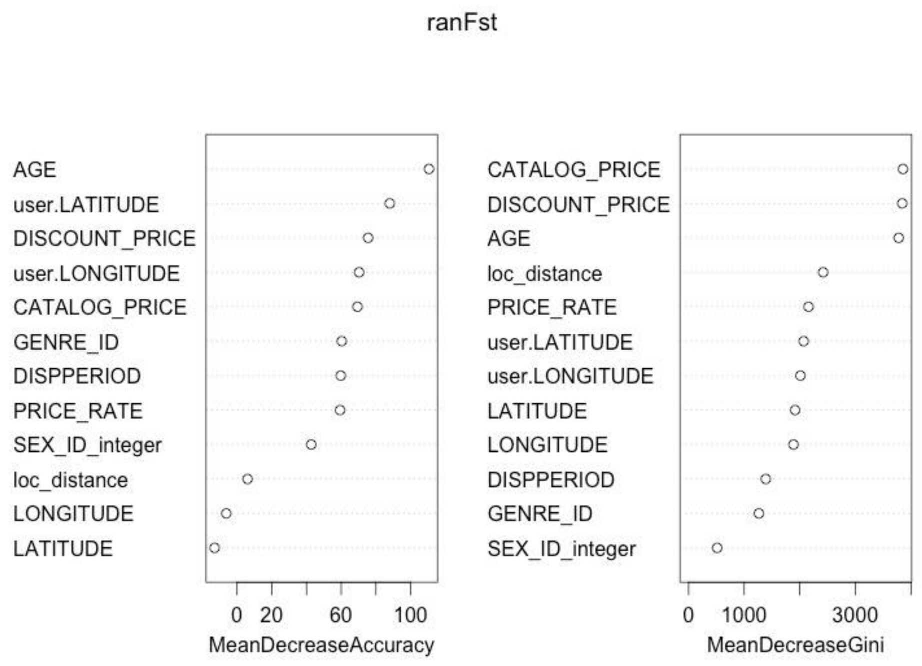


Figure 10:

4.5 Bagging

4.5.1 Normal Bagging

We fit a series of decision trees to the bootstrap samples. To predict a new observation, each tree we fitted would give us a best class prediction. We usually use the majority voting mechanism to determine the final class for that observation. In bagging, we could include all the variables we kept from the preliminary exploration because decision tree would prune thus help us to eliminate multicollinearity within the predictors.

Of course we can use the same function as random Forest to build bagging by adjusting mtry. However, our group explored another package — 'ipred'. It provides unbelievable implementation to bagging since it provides a very flexible way to apply bagging to other types of model than decision tree. The misclassification error is about 0.1964.

4.5.2 Double Bagging

We explored a method called Double Bagging to apply bagging to a not so satisfactory model—LDA and see if there is any improvement on the model. Double-bagging (Hothorn and Lausen, 2003) computes a LDA on the out-of-bag sample and uses the discriminant variables as additional predictors for the classification trees. We also use all the variables in this problem. In the double bagging, coob is not useable to calculate OOB error estimation so we have to use errorest to estimate the error rate. The error rate is about 0.204.

4.6 Boosting

Boosting is similar to bagging in that we combine the results of several classification trees. However, unlike in bagging, in boosting we fit the tree to the entire training set, but adaptively weight the observations to encourage better predictions for points that were previously misclassified. gbm package is used to fit two boosting models.

We got the best iteration number (best n.trees) by checking the performance of our model. We then plot our relative influence plot based on the best iteration number. Not surprisingly, DISCOUNTPRICE are the most influential predictors, which is consistent with the result from Random Forest. The best result is with the error rate of 0.415.

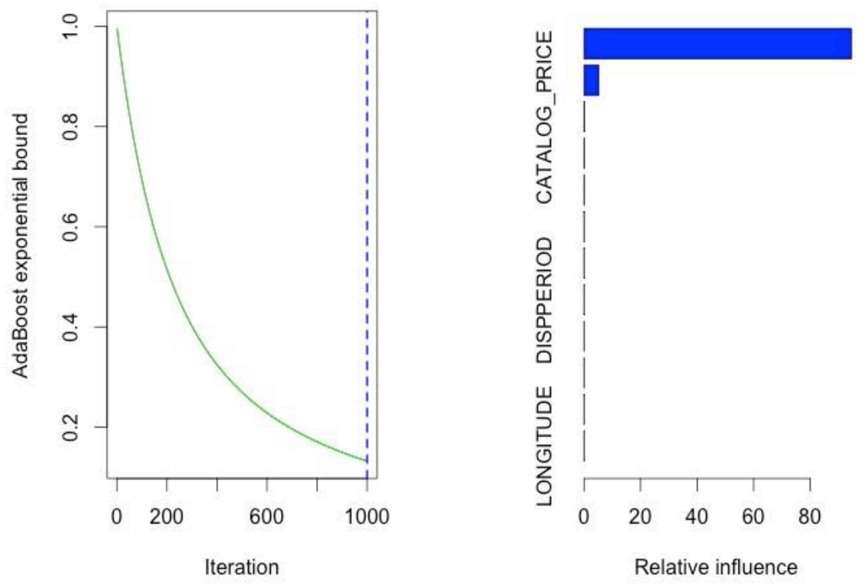


Figure 11:

4.7 KNN

Kth Nearest Neighbor Due to our observation having multiple dimensions and the nature of our problem - classifying which coupons to recommend given information on one user. Each point in our n-dimensional space represent a single interaction between a user and a coupon. This event, in addition of encapsulating information about the user and the coupon, it also contains a flag indicating whether the user has purchased this coupon. After constructing this event-relation between users and coupons. We then constructed additional event points in the n-dimensional space using 310 test coupons and all the users. For each newly constructed event points, we look at k other points (k is an odd number) that are closest to this point and count whether there are more buying events or non-buying events. We achieved this using the dist function in R. The testing result shows 0.62525percent success rate in predicting whether a user has purchased a coupon selected from the testing data. Runtime

5 Results

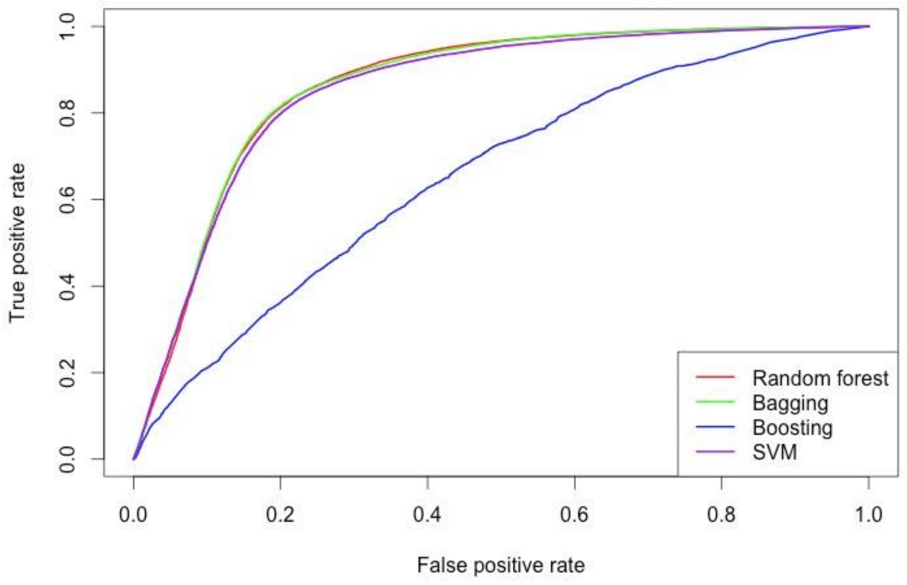


Figure 12:

Random forest is the best model
The purpose of this report is explore the effects of applying different machine learning models to solve the real life problem of predicting which tens coupons would be the most relevant to a user. We accomplished this by implementing seven different techniques found in Machine Learning. We were compared these methods to evaluate their relative performance with respect to our challenge. This project provided a great deal of insight into how recommendation systems are implemented within the industry.

One problem we realized was that the input data was relatively simple. With the growing prominence of Big Data, we expect to have a need to expand our model to capture additional features regarding customers. As we progress into the future, companies are anticipating petabytes of additional data generated daily. We must design models which optimally utilize all this data without overfitting or taking too long to converge.

A successful recommendation system, when applied to a coupon platform such as Ponpare, can level the playing field in marketing, especially for fledgling businesses. Recommendations deliver potential sales to customers which they may never otherwise reach. Automated recommendation systems have totally redefined the dynamics of the sales industry. Recommendation systems allow businesses to gain a better understanding of their customer base, allowing them to leverage economics, supply/demand, game theory, etc. Achievements in the fields could result in heightened user experience and higher revenue for the company.

References

[1] Melville, Prem and Vikas Sindhwani, *Recommender Systems, Encyclopedia of Machine Learning*, 2010.

[2] M. Tim Jones, *Recommender Systems*, 12 December 2013.

[3] Hsu, Chid-Wei, Chih-Chung Chang, and Chih-Jen Lin. *"A Practical Guide to Support Vector Classification."*, 2010. Web.
`http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf`

[4] Chapelle, Olivier, and Vladimir Vapnik. *"Model Selection for Support Vector Machines."*, Web.
`http://olivier.chapelle.cc/pub/nips99.pdf`

[5] Murphy, Kevin P. *Machine Learning: A Probabilistic Perspective.* Cambridge, MA: MIT, 2012.

[6] Zhang, Min-Ling, and Zhi-Hua Zhou. *"A K -Nearest Neighbor Based Algorithm for Multi-label Classification."* Web.
`http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.86.9501&rep=rep1&type=pdf`

[7] Hechenbichler, Klaus, and Klaus Schliep. *"Weighted K -Nearest-Neighbor Techniques and Ordinal Classification."* Web.
`http://olivier.chapelle.cc/pub/nips99.pdf`

[8] Breiman, Leo. *"Bagging Predictors."* Web.
`http://statistics.berkeley.edu/sites/default/files/tech-reports/421.pdf`

[9] Konstantin Shvachko, Hairong Kuang, Sanjay Radia, Robert Chansler. *The Hadoop Distributed File System.* Yahoo! Web.
`http://zoo.cs.yale.edu/classes/cs422/2014fa/readings/papers/shvachko10hdfs.pdf`

6 Author Contributions

Team Member	Contributions
Baotuan Nguyen	ANN, Distance Method, Conclusion
Jizhou Chen	Data Exploration, KNN
Yongkang Zhao	ANN, Data Preparation, Distance Method
Cindy Huang	Logistic Regression
Edwin Hao	Logistic Regression
Nguyen Ngo	ANN, Conclusion
Xiaodong Yan	Data Cleaning, ANN
John Chen	Abstract, Introduction, Latex Compilation, Data Cleaning
Yilun Yang	Matrix Completion, SVM, Boosting, Double Bagging
Jiebin Sun	ANN, Method
Anthony Tran	ANN
Shuxin Li	Random Forest, Bagging, Boosting