

Table of Contents

[Abstract](#_gjdgxs)

[Introduction](#_30j0zll)

[Data Preparation](#_1fob9te)

[Data collection](#_3znysh7)

[Preprocessing](#_2et92p0)

[Data summary](#_tyjcwt)

[Model building](#_3dy6vkm)

[Classification of coupon category – Mitali Ingole](#_1t3h5sf)

[Model building:](#_4d34og8)

[Decision tree](#_2s8eyo1)

[C5.0](#_17dp8vu)

[Random forest](#_3rdcrjn)

[Support vector machines](#_26in1rg)

[Model Validation](#_lnxbz9)

[Ensemble](#_35nkun2)

[Conclusions](#_1ksv4uv)

[Association rules and clustering of unsold coupons – Nancy Ploydanai](#_44sinio)

[Coupon sales classification – Wenbo Zhao](#_2jxsxqh)

[Introduction](#_z337ya)

[Classification](#_1y810tw)

[Findings](#_4i7ojhp)

[Prediction of coupons purchased – Tianzi Xu](#_2xcytpi)

[Introduction](#_1ci93xb)

[Model Analysis](#_3whwml4)

[Analysis](#_2bn6wsx)

[Conclusions](#_qsh70q)

[Appendix](#_3as4poj)

[Classification of coupon category](#_1pxezwc)

[Association Rules - Nunt Ploydanai](#_2p2csry)

[Coupon sales classification – Webber Zhao](#_147n2zr)

[Prediction – Tianzi Xu](#_3o7alnk)

# Abstract

The key finding in this dataset is to predict which coupons a customer will buy. As you can see from the public e-commerce website like Groupon.com, LivingSocial.com or TripAlertz.com, they give some recommendation of products and services based on consumers’ interests. To satisfy and fulfill customers’ need, we would like to find some pattern or model that efficiently helps businesses enhance their profit, and be more competitive.

The dataset is obtained from the website kaggle.com. Link is as below:

<https://www.kaggle.com/c/coupon-purchase-prediction>

We established models to classify coupons based on their category and if those coupons would be good, bad or unsold using classification techniques and ensemble learnings. We also established a model in order to predict the number of coupons purchased using regression trees and so on. This report is in order to introduce our analysis, including main problem proposal, data cleaning and transformation process, initial analysis about data distribution and relationship among variables, model building (individual section), analysis (executive summary) and conclusions (insights from analysis). We also attached our code in the Appendix.

# Introduction

The dataset consists of information about coupons from Japan’s leading website called "Ponpare." The site offers huge discounts on a variety of products such as food, concerts, jewelry, etc. The company wants to predict the type of coupon a customer will buy in a particular time frame given his past purchase and browsing behavior. The company thus wants to improve its recommender system to enhance user experience and at the same time provide a better quality of service. Given that the dataset is a real world problem and covers verticals like finance, marketing, and sales, we found this dataset interesting to better our data mining skills using the techniques learned in this class. Therefore, the supreme goal of this project is to apply the optimal recommended model to this particular coupon company to earn more money.

From the website, following datasets are available:

1. coupon\_list\_test.csv
2. coupon\_area\_test.csv
3. coupon\_list\_train.csv
4. coupon\_area\_train.csv

6. coupon\_detail\_train.csv

7. user\_list.csv

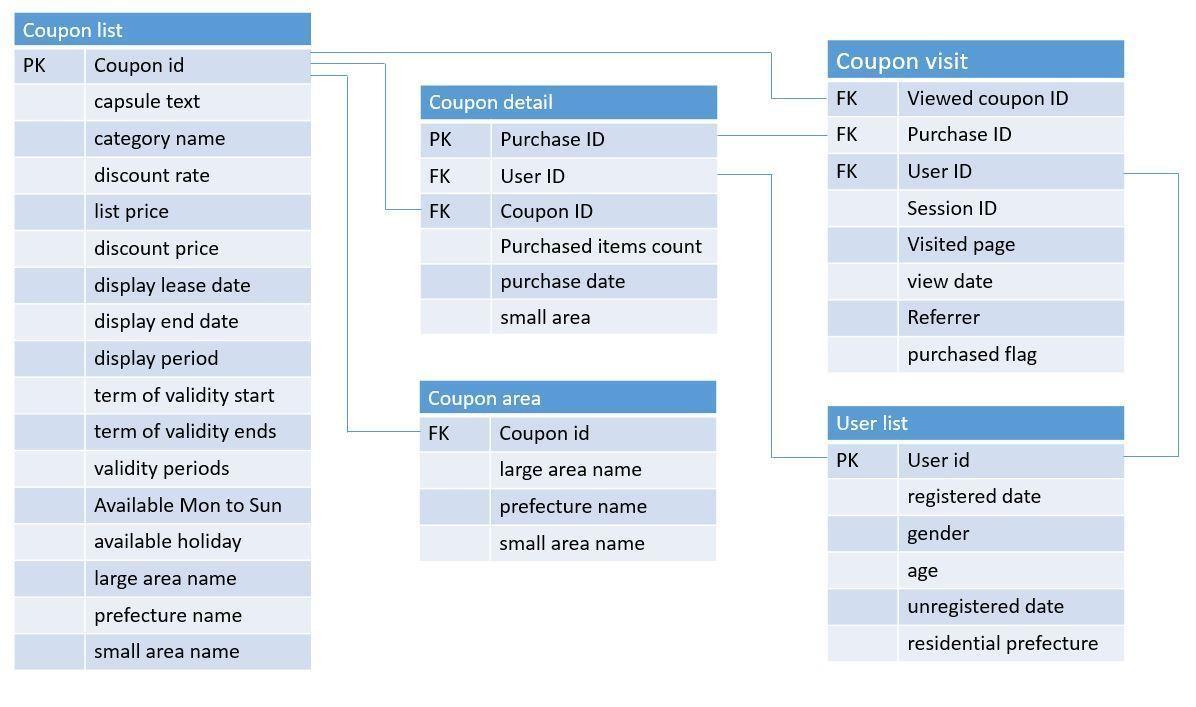
1. coupon\_visit\_train.csv
2. prefecture\_locations

We want to choose five datasets from them: coupon\_list\_train, coupon\_area\_train, coupon\_detail\_train, user\_list, coupon\_visit\_train.

There are five main datasets that can be integrated to extract information and generalize the testing set.

1. coupon\_list\_train is about coupons listing areas for training set, a dataset which is about details of purchases. There are 24 variables in this dataset. They are capsule text, category name, discount rate, list price, discount price, sales lease data, sales end date, sales period, the term of validity start, the term of validity ends, validity periods, if it is available from Monday to Sunday (there are seven variables), if it is available on holiday and before holiday, large area name of shop location, prefecture name of shop, coupon ID and small area name of shop location.
2. coupon\_area\_train: It is about areas of shops which these coupons should be applied. It includes small areas, listed prefecture name and coupon ID.
3. coupon\_detail\_train: It is about information of purchased coupons. Purchased items count, purchase date, purchase ID, user ID, coupon ID and small area of shops.
4. user\_list: It is about information of all users, including user ID, registered date, gender, age, unregistered date and residential prefecture.
5. coupon\_visit\_train: It is about the viewing log of users browsing coupons during the training set time period that includes userID, purchased flag, view data, referrer, and session ID.

To better understand the relationships between the variables in the database, ER Diagram is shown in figure 1.



**Figure 1 ER Diagram**

# Data Preparation

## Data collection

The training set of this dataset is transaction and browsing data for 22873 users from 2011-07-01 to 2012-06-23. The testing set is data for the same users from 2012-06-24 to 2012-06-30.

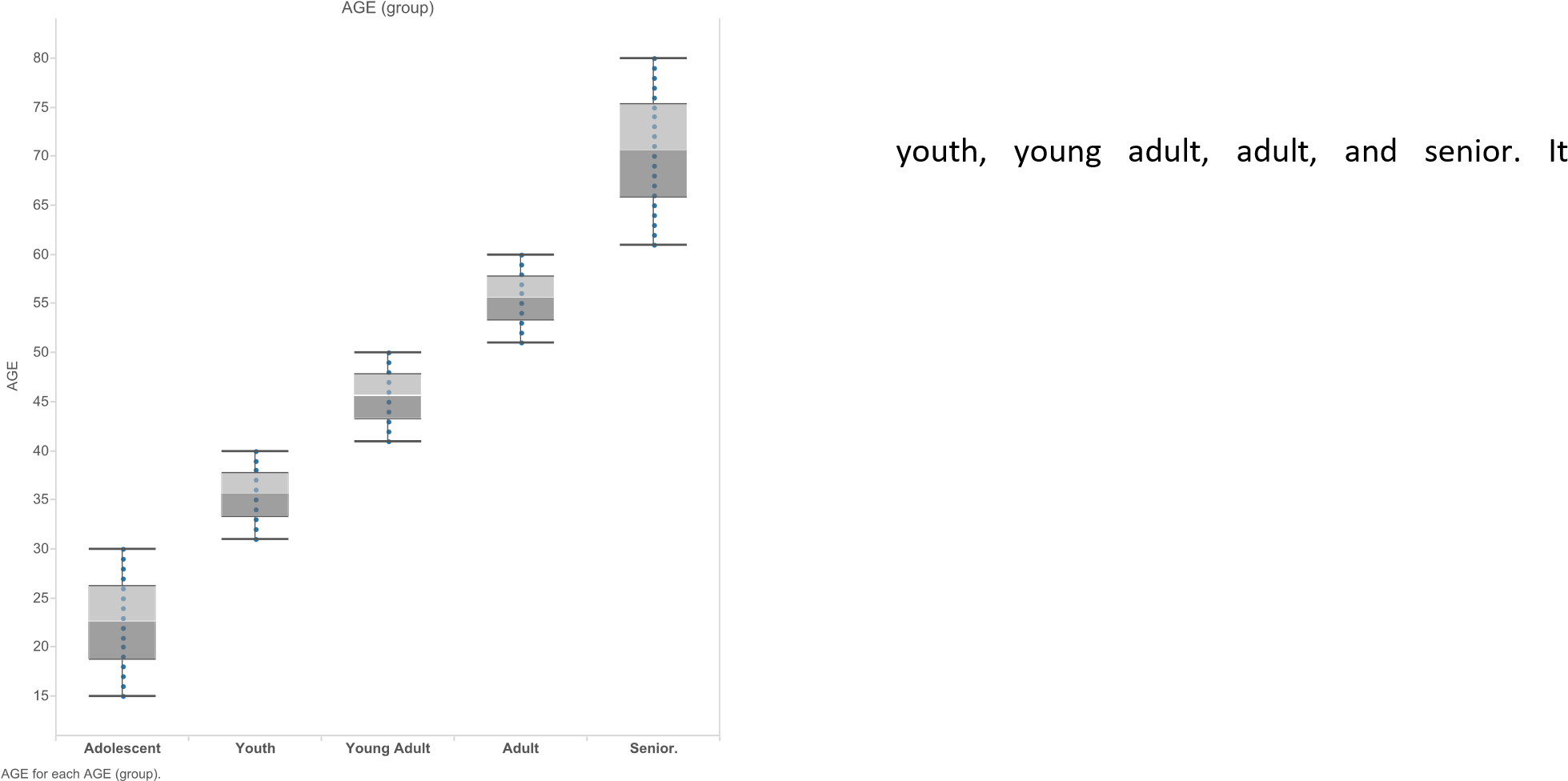
Ponpare is one of the famous coupon sites in Japan. The company intended to find preference of customers to avoid wasting resources on products and services which customers are not interested in. The company wants to predict which category of coupons these customers may buy through information of users and coupons. The company will use this model to recommend coupons to users based on their interest when they browse the website.

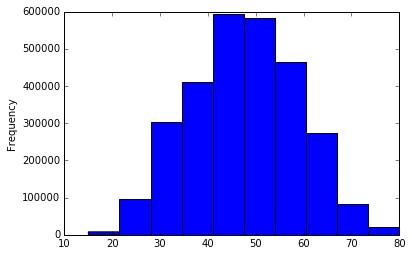
The datasets have been merged according to the requirement of the different analysis described in the following sections.

## Preprocessing

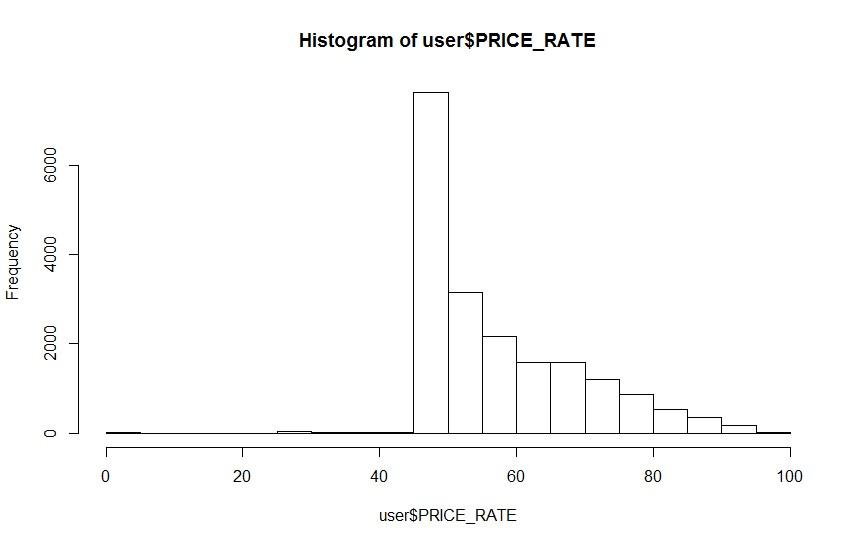
The variables in our data set contain both numeric and categorical type. Transformation on some of the important variables is described in the following sections.

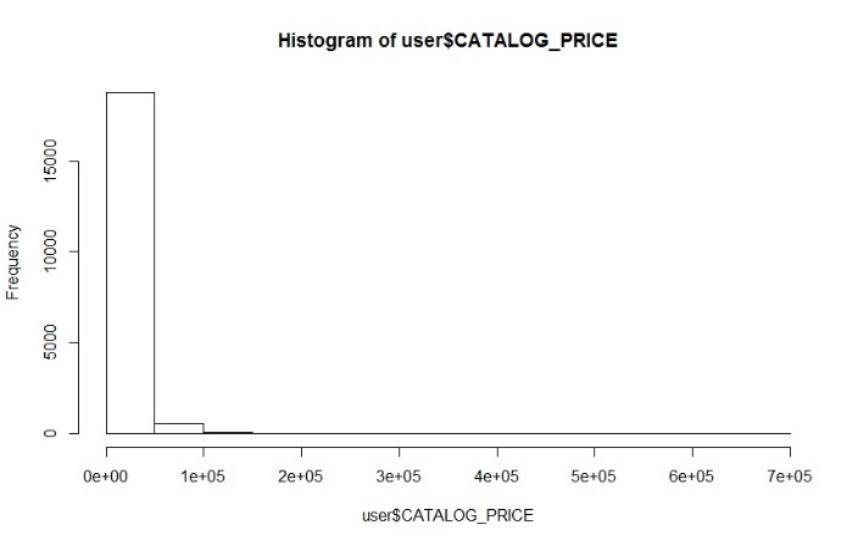
**Variable – AGE**

We convert numerical age variable into a categorical one. Since, it is normally distributed, as seen in figure 2 we bin it into 5 bins. Figure 3 shows box plots for all five binned variables. The new age bins are adolescent, youth, young adult, adult and senior. Some techniques require numerical values to calculate similarity and distance. Hence, each technique involved in this analysis uses age variable either as categorical or numerical.



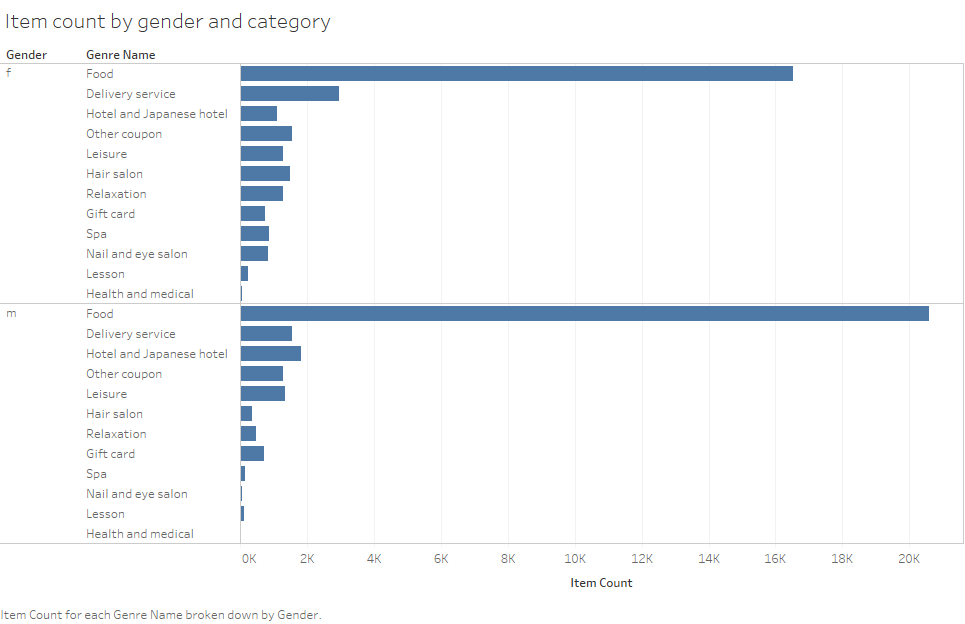
**Figure 2 Histogram of age variable**



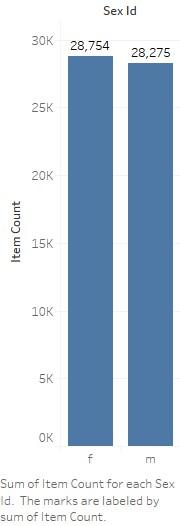
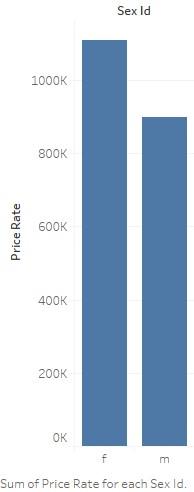
**Figure 4 Histogram of Price rate**

# Data summary



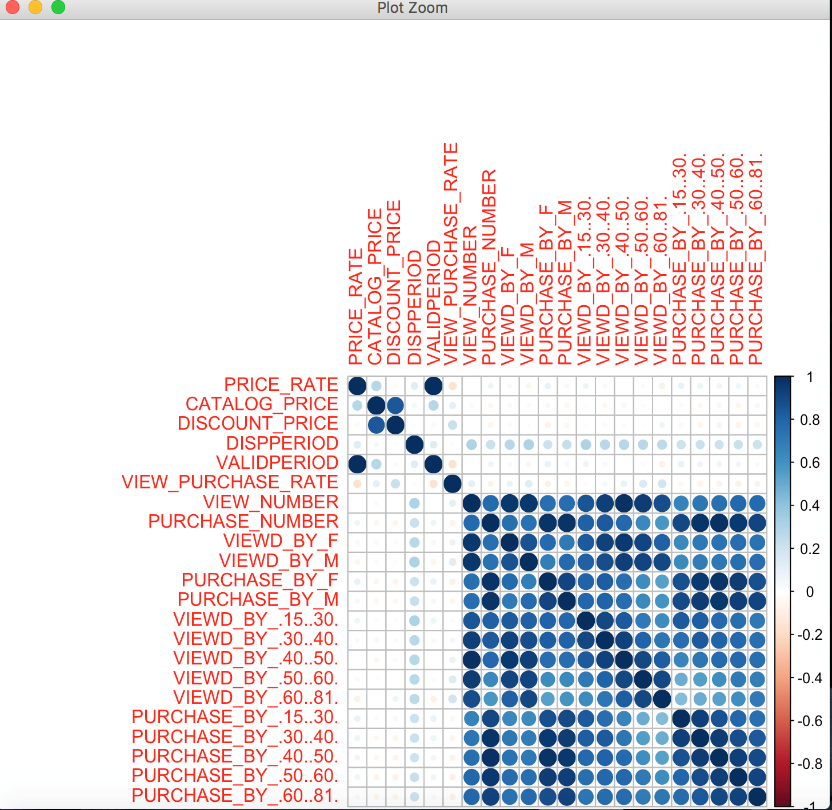
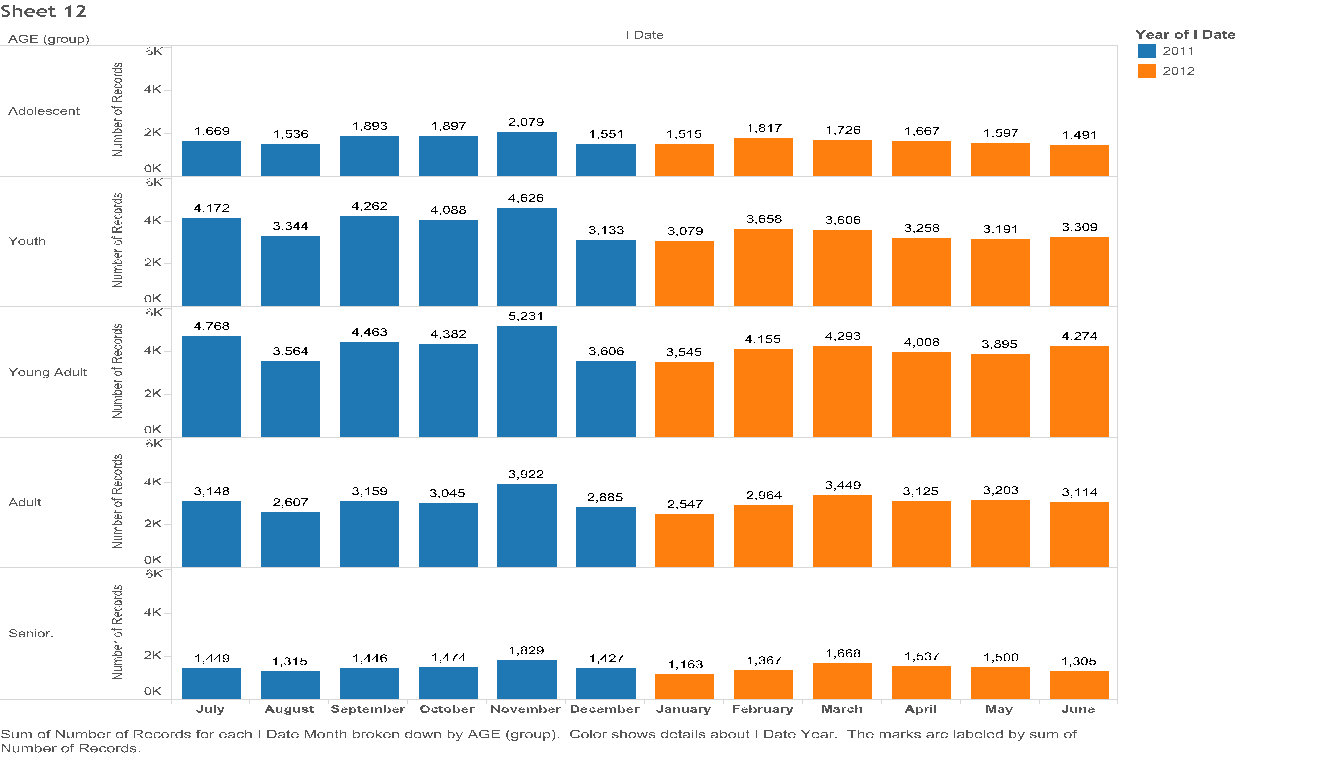


**Figure 7 Distribution of item count**

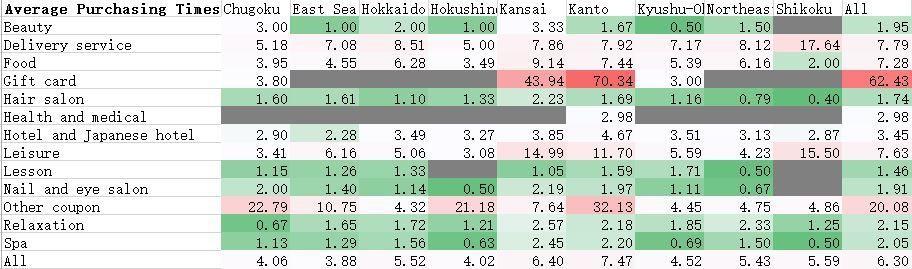




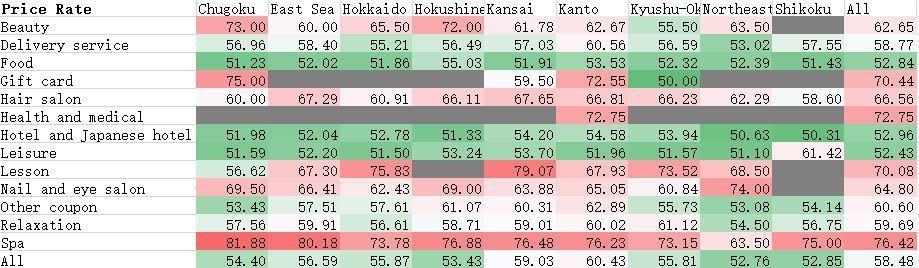
1. The average discount rates of different coupon category in different area are listed in the table below. The red color presents a higher discount rate. The green color represents a lower discount rate. The grey color represents that coupon category is not sold in this area. Hence, according to this pivot table, we can figure out that coupons of Spa, Health & Medical, Lesson, Gift card and Beauty have high discount rates. The coupon discount rates among different area are not variant.



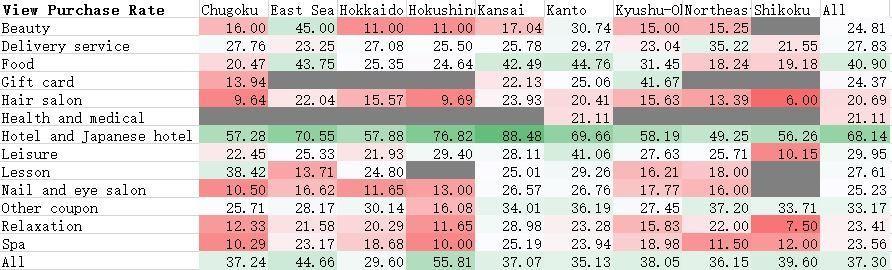
**Figure 10 bar plot**







**Table 2 pivot table of category by rate and location**



**Table 3 pivot table of purchase rate b category and location**

# Model building

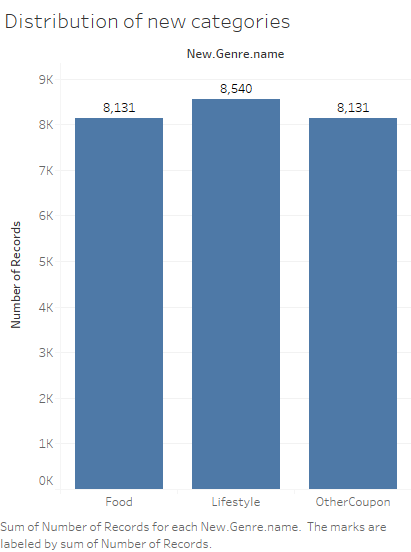
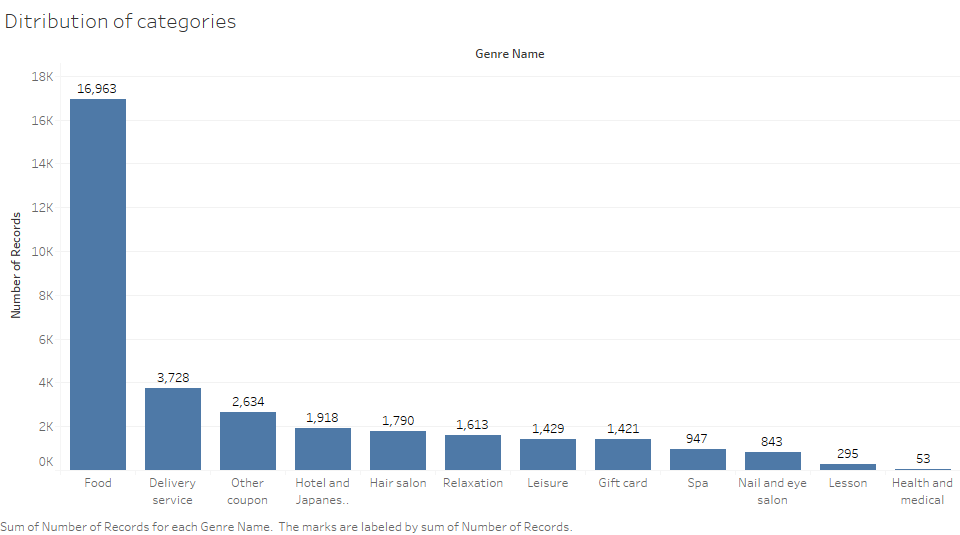
## Classification of coupon category – Mitali Ingole

**Purpose:**

This section explores classifying the coupon categories. I have combined datasets coupon\_list\_train.csv , user\_list and coupon\_detail\_train. There were test datasets only for coupon list and detail and not for user\_list. There was no corresponding information from the former test files in user\_list. Thus, we decided to combine the above three datasets and divide them into testing and training for model validation. The coupon categories are listed as follows in table 4.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Hotel and Japanese hotel | Health and Medical | Nail and eye salon | Leisure | Hair salon | Food |
| Spa | Relaxation | Lesson | Other coupons | Delivery service | Gift card |

**Table 4 categories**

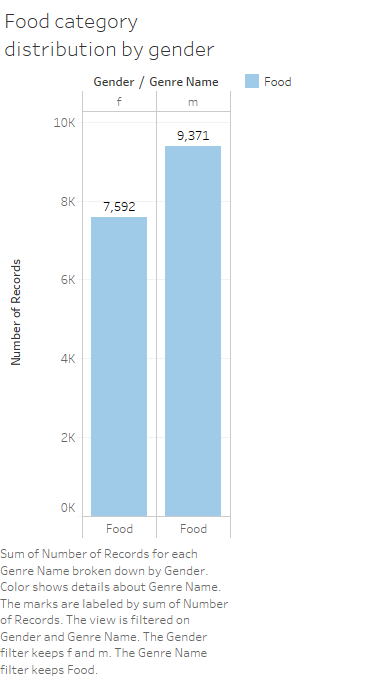
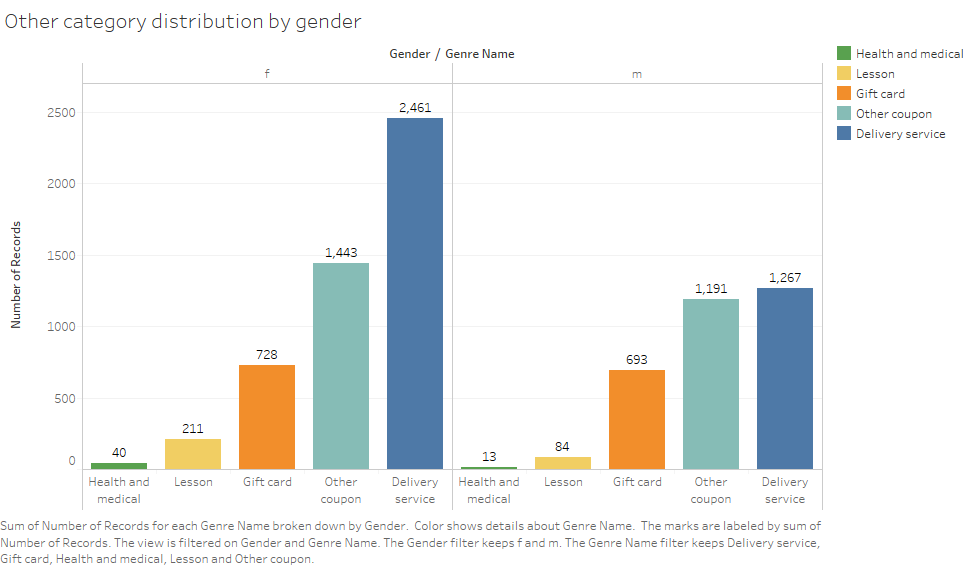


**Figure 12 category distribution** 

There are in total 12 number of categories. As seen from figure 12, there are more samples for “Food” category than for the rest of the coupons. To take care of this non uniform distribution, and also to make the multi classification problem in to a 3 class problem rather than a 12 class problem, the categories were grouped together and reduced to three which are, Food, Lifestyle and Other coupons. Each of these new categories consisted of following original coupon categories:

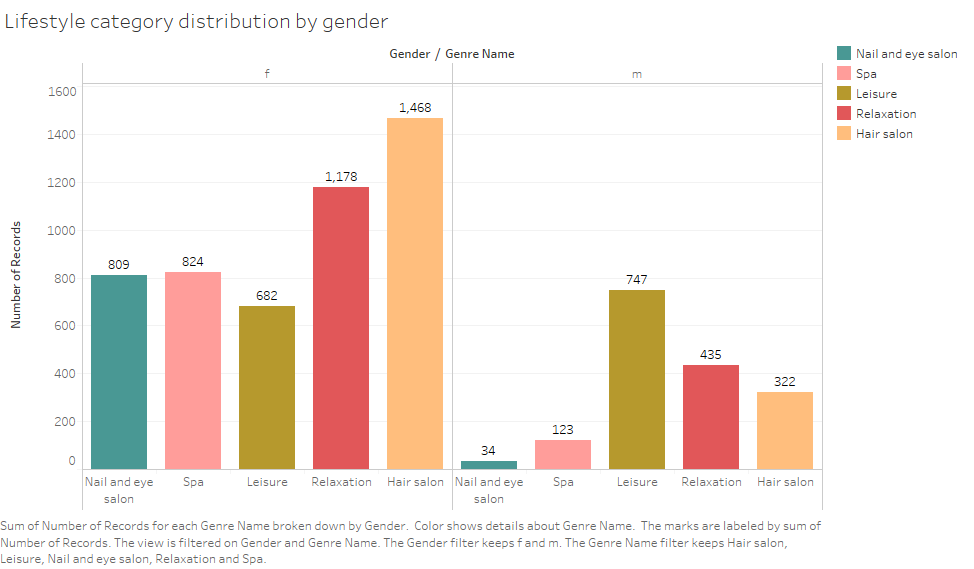
* Food coupon – all food coupons
* Lifestyle – Hotel and Japanese hotel, Hair salon, Relaxation, Leisure, Spa, Nail and Eye salon
* Other coupon – Gift card, Lesson, other coupons, health and medical and delivery services

Stratified sampling was also performed to make sure that the number of coupons in each category are nearly the same to avoid over prediction for the categories with higher counts. The distribution of these new categories is shown in figure 13.

**Figure 14 Food category**

**distribution**



**Figure 16 lifestyle distribution**

Figures 12, 13 and 14 describe the distribution of the three categories by gender. Food category is mostly preferred by males whereas the delivery services are preferred by females. All of the categories in lifestyle are highly preferred my females than males except for the exception of hotel industry. The other categories are almost negligibly used by males.

## Model building:

The techniques used for classification are decision trees, C5.0, random forests, support vector machines, treebag and gradient boosted models. These models were built using 80% of the data for training and 30% for testing.

Each of these models were compared against performance metrics listed below. Bagged and Boosted model were compared against accuracy and corresponding Kappa value.

Since, it’s a multiclass classification problem, the metrics used for testing the performance of these models are –

* Accuracy: total number of instances that have been classified correctly
* Area under the curve: how well does the classifier label a randomly chosen positive class against a randomly chosen negative class
* macroF1: mean of F1 measure across all labels
* macroPrecision: mean of precision across all labels
* macroRecall: mean of recall across all labels

### Decision tree

Decision tree for coupon classification is shown in figure 17.

**Observations:**

* The variable that differentiates between the categories is “item count”.
* Customers tend to buy “Food” coupons in more quantity than the rest. They also tend to be more expensive than the rest of the coupons. This goes to show that customers buy “Food” coupons in bulk.
* “Lifestyle” and “Other coupons” are bought either in 1 or 2 counts. This shows that these coupons are cheaper than “Food” coupons and are also provided discounts of over 50% which is not present in “Food” coupons.
* “Other coupons” consist of health and medical services, delivery services and gift cards. From the tree, we can see that these are the coupons with a discount rate of more than 62% which makes sense since these would be coupon categories the customers would like discounts on.
* “Lifestyle” coupons are more than 5245 yen in price. Hotel industry, spas and beauty form the bulk of this category and are the ones with the highest price.

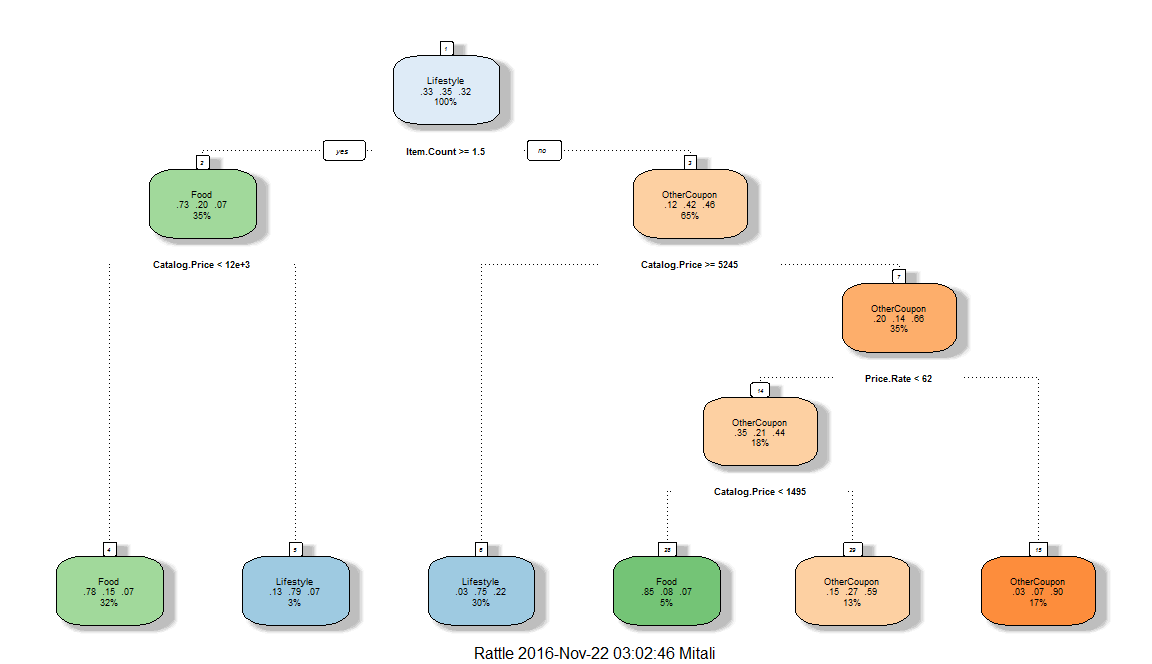
**Performance:**

|  |  |  |  |
| --- | --- | --- | --- |
| ***Category*** | ***Precision*** | ***Recall*** | ***F1*** |
| ***Food*** | 0.89 | 0.78 | 0.84 |
| ***Lifestyle*** | 0.72 | 0.74 | 0.73 |
| ***OtherCoupon*** | 0.69 | 0.77 | 0.73 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *AUC* | *Accuracy* | *macroPrecision* | *macroRecall* | *macroF1* |
| 0.8525 | 0.7841 | 0.7864 | 0.7835 | 0.7832 |

**Table 5 performance metrics for decision tree**

The performance statistics work fairly well for this model with macro precision, recall and F1 measures of about 78% and AUC 0.8525 as shown in table 5.



**Figure 17 Decision tree**

### C5.0

**Observations:**



|  |  |
| --- | --- |
| 100.00% | Catalog.Price |
| 100.00% | Item.Count |
| 91.18% | Price.Rate |
| 88.32% | Dispperiod |
| 72.57% | Discount.Price |
| 47.12% | Sun\_1 |
| 43.81% | Tue\_1 |
| 42.25% | Fri\_1 |
| 37.24% | Hol\_0 |

**Table 6 attribute usage**

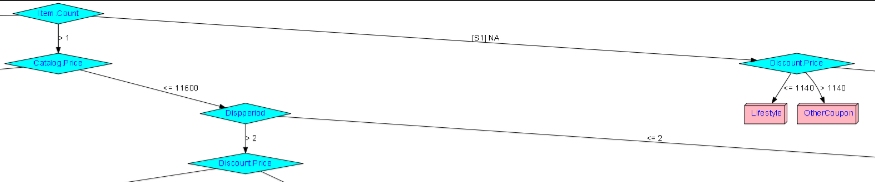
**Performance:**

|  |  |  |  |
| --- | --- | --- | --- |
| ***Category*** | ***Precision*** | ***Recall*** | ***F1*** |
| ***Food*** | 0.96 | 0.95 | 0.95 |
| ***Lifestyle*** | 0.91 | 0.90 | 0.90 |
| ***OtherCoupon*** | 0.89 | 0.91 | 0.90 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *AUC* | *Accuracy* | *macroPrecision* | *macroRecall* | *macroF1* |
| 0.95 | 0.92 | 0.92 | 0.92 | 0.92 |

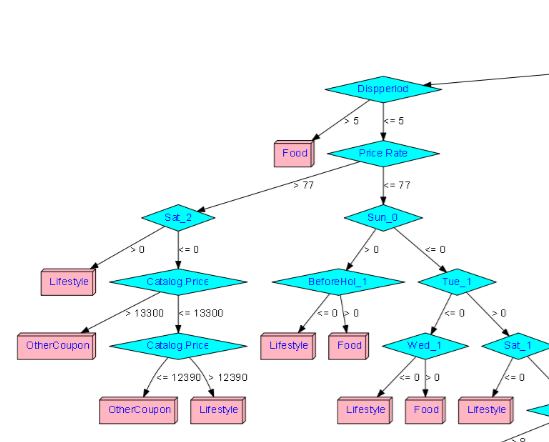
**Table 7 performance metrics for c5.0**

I could not find a better way to output C5.0 tree. I found a GraphViz library online (http://r-project-thanos.blogspot.de/2014/09/plot-c50-decision-trees-in-r.html ) to plot the tree. Since, the tree has too many branches, the plots attached below are only up to level 5 of the tree.



**Figure 18 C5.0 - parent node**

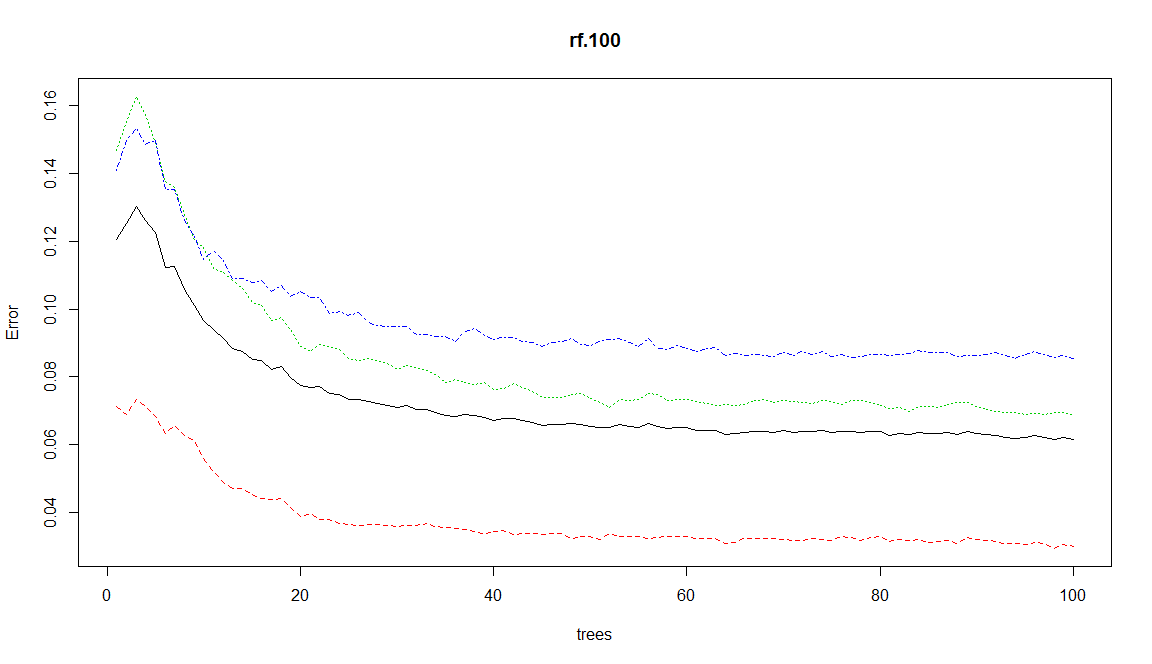




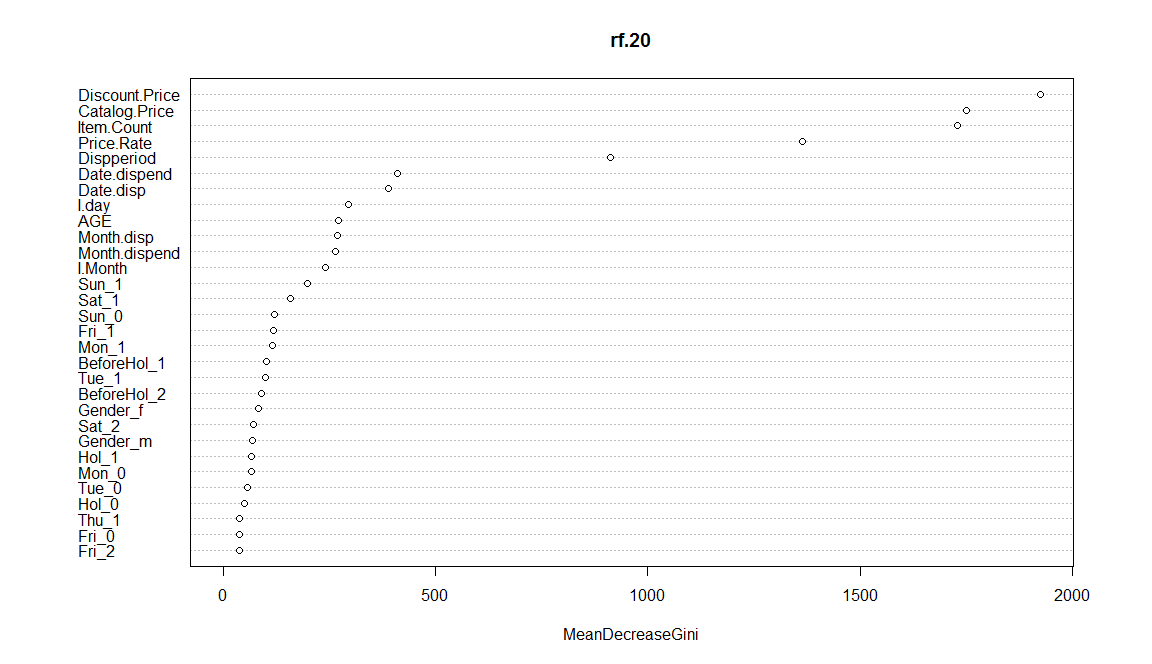
**Figure 19 Sub tree of C5.0**

### Random forest

**Observations:**



**Figure 20 Number of trees vs error**



**Figure 21 Variable importance**

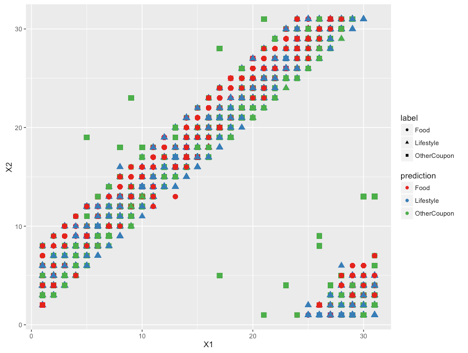
**Performance** 

|  |  |  |  |
| --- | --- | --- | --- |
| ***Category*** | ***Precision*** | ***Recall*** | ***F1*** |
| ***Food*** | 0.96 | 0.95 | 0.95 |
| ***Lifestyle*** | 0.92 | 0.92 | 0.92 |
| ***OtherCoupon*** | 0.90 | 0.92 | 0.91 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *AUC* | *Accuracy* | *macroPrecision* | *macroRecall* | *macroF1* |
| 0.9568 | 0.9391 | 0.9394 | 0.9390 | 0.9391 |

**Table 8 performance metrics for random forest**

### Support vector machines



**Figure 22 SVM OVO results**

The results are shown in table 9. Figure 22 describes how the SVM model attempts to differentiate between the different categories. It is neither helping in differentiating categories nor providing any coherent information. Hence, we decided to drop this model.

### Model Validation

The results were validated using cross validation and boosting (Stochastic Gradient boosting discussed in this section later).

#### Cross validation



|  |  |  |  |
| --- | --- | --- | --- |
| *Validation method* | *Model* | *AUC* | *Accuracy* |
| *Cross Validation* | Decision trees | 0.78 | 0.74 |
| C5.0 | 0.94 | 0.92 |
| Random forest | 0.88 | 0.84 |
| Treebag | 0.96 | 0.95 |
| *Boosting* | Gradient Boosting | 0.91 | 0.87 |

**Table 10 Cross validation results**

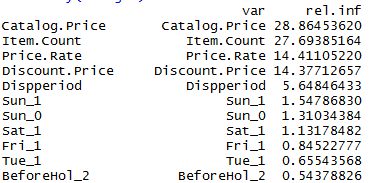
**Observations:**

* Random forest seems to overfit as compared to the standalone model discussed in table 8. Tree bag also seems to overfit with a very high accuracy of 95%.
* Decision tree seems to work the same as the stand alonemodel. C5.0 shows overfitting.
* From cross validation, we can say that decision tree works best for this dataset when compared to the stand alone model. However, if accuracy is considered as the sole metric for performance, then we can use either C5.0 or random forest to classify the categories.
* Also, random forest are more complex than decision trees. Thus, in order to compare theses two techniques, we would need more amount of data to validate these results.

Note: The difference between tree bag and random forest is that the former uses only a subset of random features of all the features as opposed to the latter that uses all features for splitting the data.

#### Stochastic Gradient boosting





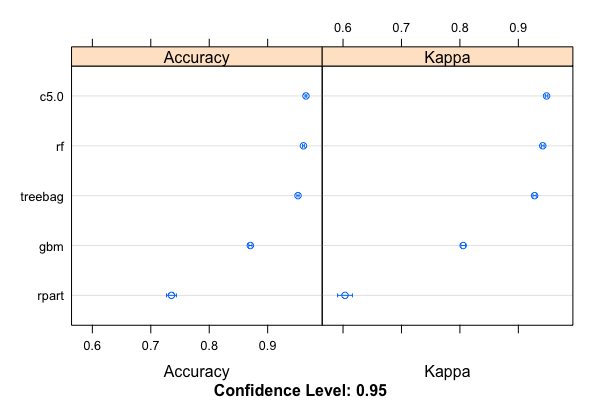
**Figure 23 Gradient boosting results**

### Ensemble

In order to compare all the models developed so far based on accuracy, an ensemble was created with decision tree(rpart), C5.0, GBM (gradient boosting), treebag and random forest.



|  |  |  |
| --- | --- | --- |
| ***Category*** | ***AUC*** | ***Accuracy*** |
| ***C5.0*** | 0.97 | 0.96 |
| ***Decision tree*** | 0.78 | 0.73 |
| ***Random forest*** | 0.96 | 0.98 |
| ***Treebag*** | 0.95 | 0.78 |
| ***GBM*** | 0.91 | 0.87 |



**Table 12 Ensemble results**

**Observations:**

* Figure 24 shows comparison of five models with respect to accuracy and corresponding Kappa value. Table 12 shows AUC and accuracy values for each model.
* C5.0, random forest and tree bag can be seen as the best models for classifying the categories. Given the increasing complexity associated with random forest and treebag as the data set increases, the company can choose C5.0 as the classification algorithm.
* Random forest seems to be a really good model for the given dataset but given that it has an AUC of nearly 1, it requires validation with the help of more samples.
* Also, from the results of C5.0 in tables 6 and 7, we can draw the final conclusions for classification that item count and catalog price are important variables in classifying the different categories.

## Conclusions

* C5.0 decision tree model can be used to classify this dataset.
* We learn from the decision tree model that customers tend to stack up on coupons from the food category (figure 12). The company can thus, focus on advertising more coupons from food industry and earn profit.
* The customers don’t tend to buy many coupons from the “other coupon” category which provide discounts of nearly 50%. From the category distribution in figure 15 we can see that the company does not advertise much of these coupons and thus, can drop this entire category.
* This category however, contains Delivery services and can be merged with the “Food” category. Delivery services can be then marketed toward the customers and gained profit from.
* The lifestyle coupons look expensive since they cost more than 5245 Yen. This category is preferred more by females as is evident from figure 14. The company can target this category towards women and generate profits in the listed areas.

## Association rules and clustering of unsold coupons – Nancy Ploydanai

The objective of Data Analysis and Model Processing is using Data Mining and Machine Learning algorithms to predict the coupon category that consumer will be more likely to buy in the particular time. We would like to explore some pattern or efficiently model that helps business improve their recommender systems, directly fulfill consumers’ need, to be more competitive, leads to enhance company profit.

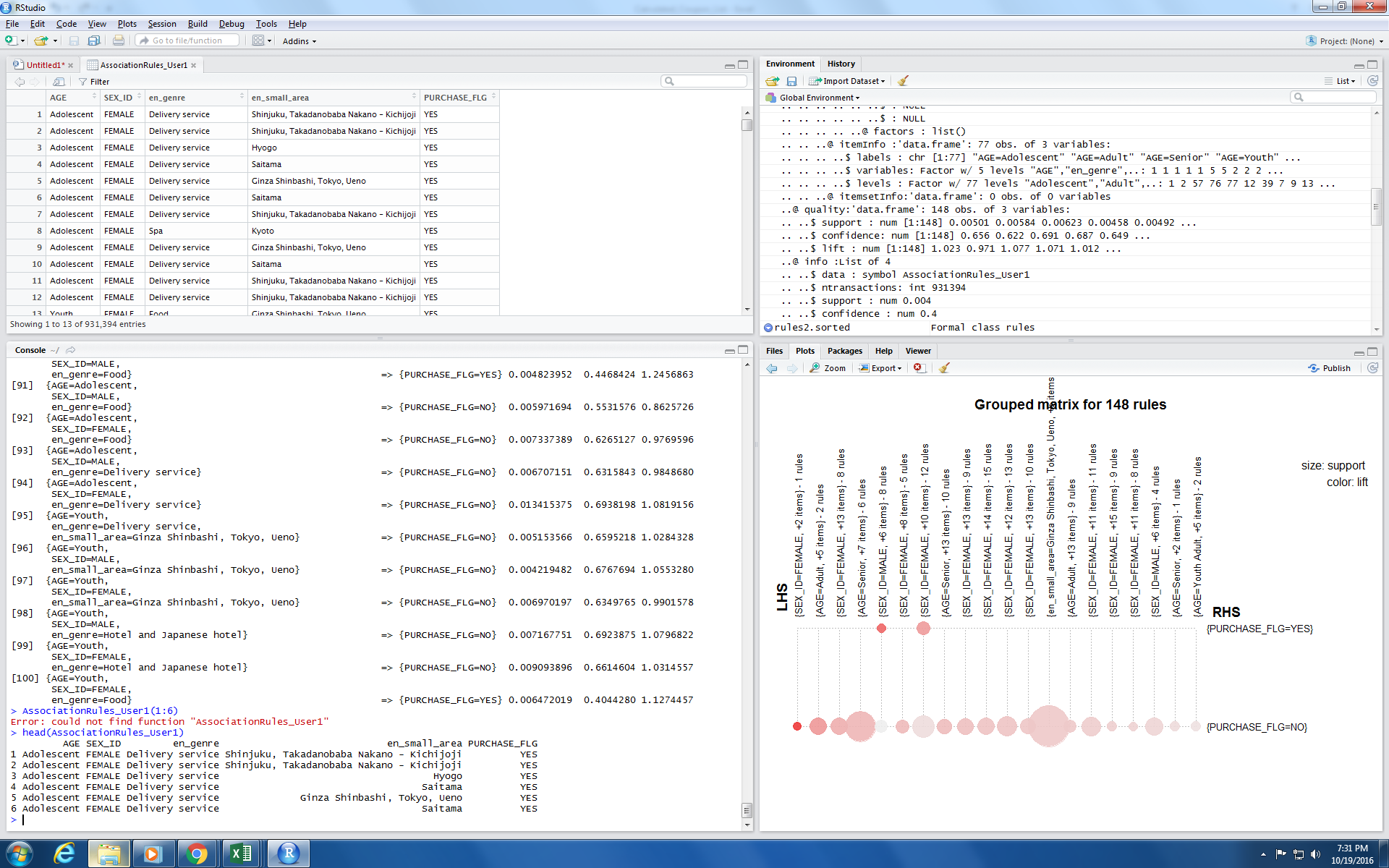
WHETHER CONSUMER WOULD/WOULD NOT BUY A COUPON?

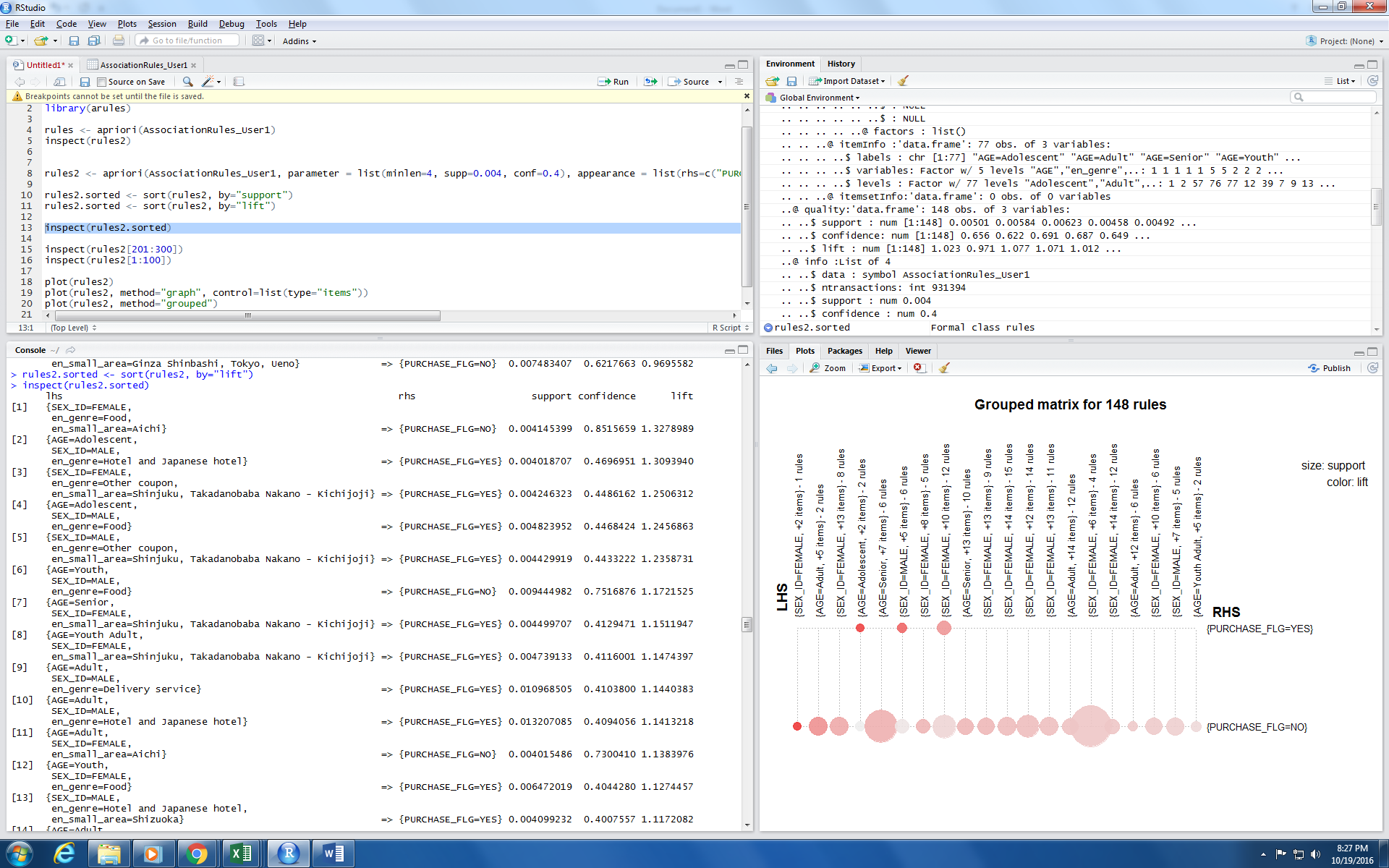
First of all, we are going to talk about the association rules mining. We would like to see how likely each consumer would or would not buy a coupon. From the data, we use table that contains information of Age, Gender, Coupon category that each user has viewed, the area of the particular coupon effective from each web user. Then, predict if consumer is more likely to purchase the coupon (YES) or do not purchase the coupon (NO) using Association analysis.

Association analysis is good for exploring the hidden relationships in the data set. The most common methods for studying the association rules are Frequent Item set Generation and Rule Generation.

In this section, we are going to talk about the Association Rules. We are now focusing on analyzing and predicting whether customers would/would not buy a coupon. The association rules represent the attributes that occur frequently in a data set. We would like to know those customers under the condition of particular age, gender, coupon genre that they have viewed, and the area of the effective coupon usable, how consumer would/would not like to buy (a) coupons.

We created a table for the Association Rules analysis.





Let’s focus on the case that consumers will buy the coupon. From the result above, basically, male consumers with have larger probability to buy a coupon than female consumers. Food coupon does not attract consumers very well. Consumers are interested to buy coupons that are usable in Shinjuku area.

[8] {SEX\_ID=MALE,

en\_genre=Hotel and Japanese hotel,

en\_small\_area=Shizuoka} => {PURCHASE\_FLG=YES} 0.004099232 0.4007557 1.1172082

[58] {SEX\_ID=MALE,

en\_genre=Other coupon, en\_small\_area=Shinjuku, Takadanobaba Nakano - Kichijoji}

=> {PURCHASE\_FLG=YES} 0.004429919 0.4433222 1.2358731

[59] {SEX\_ID=FEMALE,

en\_genre=Other coupon, en\_small\_area=Shinjuku, Takadanobaba Nakano - Kichijoji}

=> {PURCHASE\_FLG=YES} 0.004246323 0.4486162 1.2506312

[72] {AGE=Youth Adult,

SEX\_ID=FEMALE, en\_small\_area=Shinjuku, Takadanobaba Nakano - Kichijoji}

=> {PURCHASE\_FLG=YES} 0.004739133 0.4116001 1.1474397

[82] {AGE=Senior,

SEX\_ID=FEMALE, en\_small\_area=Shinjuku,Takadanobaba Nakano - Kichijoji}

=> {PURCHASE\_FLG=YES} 0.004499707 0.4129471 1.1511947

[87] {AGE=Adolescent,

SEX\_ID=MALE, en\_genre=Hotel and Japanese hotel}

=> {PURCHASE\_FLG=YES} 0.004018707 0.4696951 1.3093940

[90] {AGE=Adolescent,

SEX\_ID=MALE, en\_genre=Food}

=> {PURCHASE\_FLG=YES} 0.004823952 0.4468424 1.2456863

[100] {AGE=Youth,

SEX\_ID=FEMALE, en\_genre=Food}

=> {PURCHASE\_FLG=YES} 0.006472019 0.4044280 1.1274457

[129] {AGE=Adult,

SEX\_ID=MALE, en\_genre=Hotel and Japanese hotel}

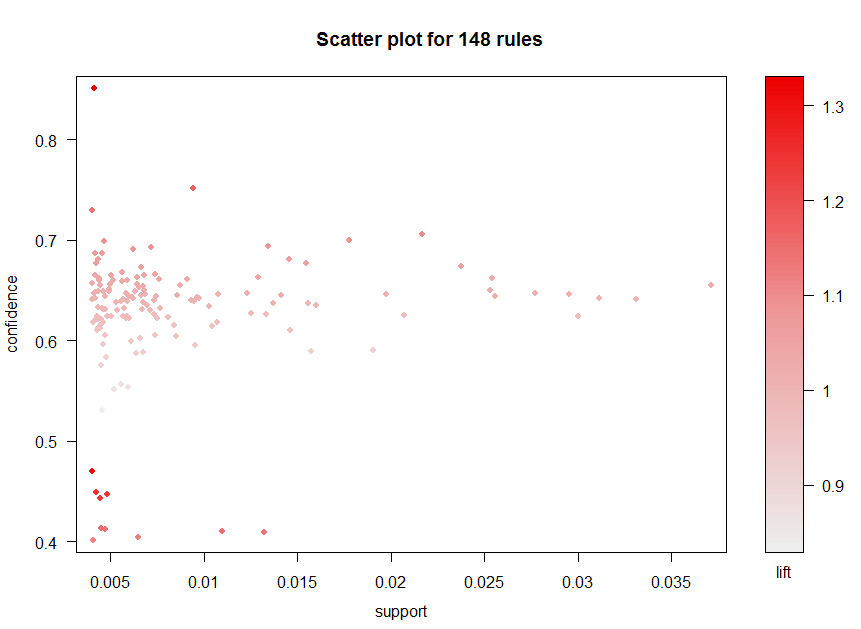
=> {PURCHASE\_FLG=YES} 0.013207085 0.4094056 1.1413218

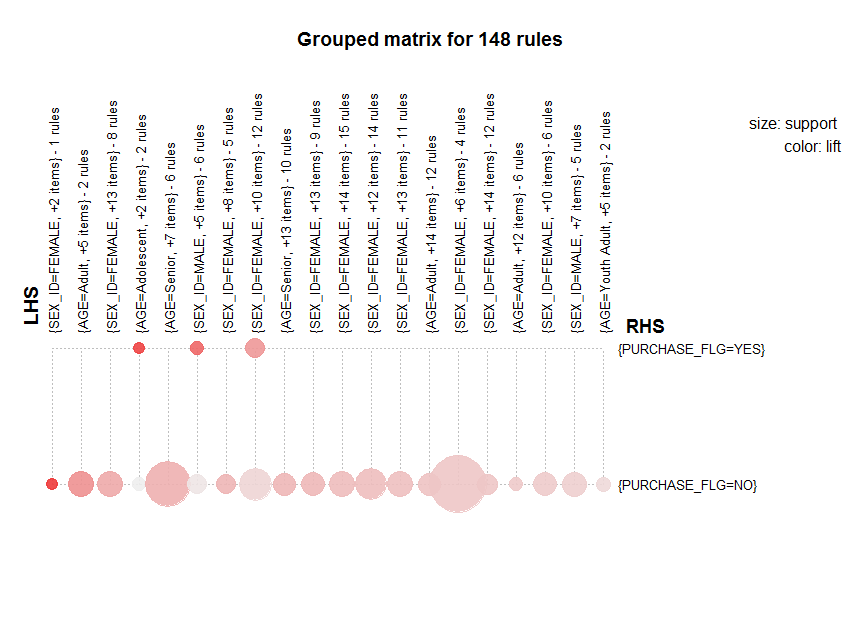
[138] {AGE=Adult,

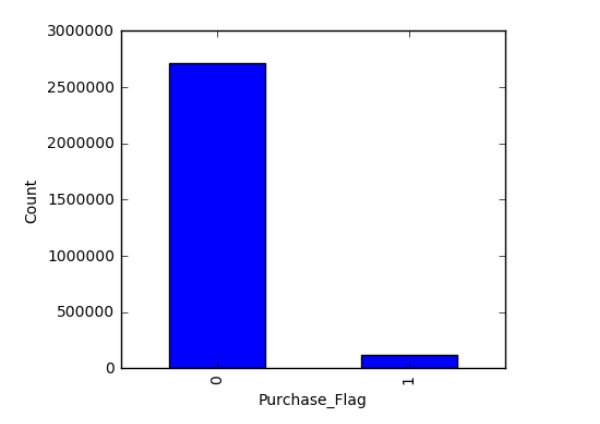
SEX\_ID=MALE, en\_genre=Delivery service}

=> {PURCHASE\_FLG=YES} 0.010968505 0.4103800 1.1440383







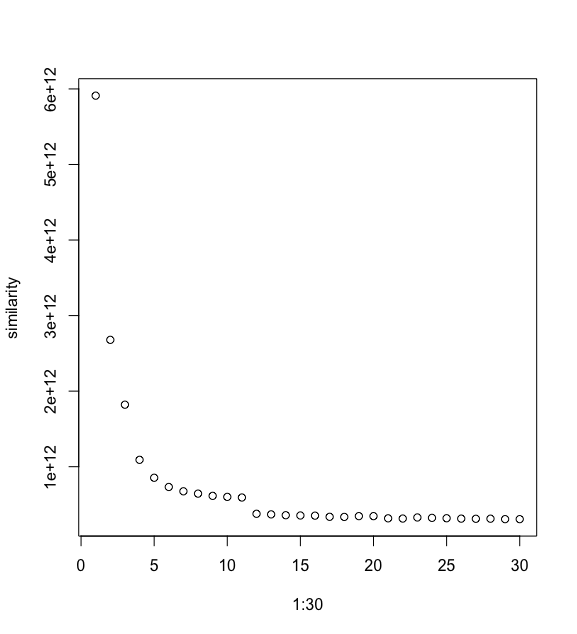


In all data mining area, we are all would like to enhance the model and improve the performance. Therefore, we consider the rules that satisfy the constraint of minimum support and confidence. We learn by trial and error to see which values are reasonable. Also, the result should be explainable, minimize the size of result, and improve the computational expensive.

**CLUSTERING ANALYSIS – UNSOLD COUPON CLUSTERING**

The next technique that we would like to perform is cluster analysis, which is a technique of unsupervised learning. The ultimate goal of clustering is to find a category or grouping of unlabeled data. We group data into each category by calculating the distance between objects.

In this case, we are going to cluster the unsold coupons since there are larger number than the sold ones. K-means clustering will be applied in this part. To archive the k-means algorithm, we need to measure the similarity between the objects. The coupon features that we use to calculate the similarity distance in this analysis are: Price rate, Catalog price, Display period, Purchase number, and Discount rate.

**How to decide the appropriate number of cluster?**

**Cluster3** shows low catalog price coupon with low purchase number

**Cluster4** is coupons with high price rate, low discount rate but pretty high discount rate

**Cluster5** consists of high-end coupon with low purchase number and low discount rate.

However, we would like to see roughly what is the main of unsold coupon category in each cluster. We use Word Clouds graphical to represent the word frequency that appear more frequently in each cluster.

**Unsold Coupons in each Cluster:**



We divide the unsold coupons into five different group based on their price and user’s profile by using clustering analysis. This makes company more understand what coupon they should be more focus on. Company can create promotion/campaign for selling differently for each group. To be a deeper analysis, in each group, they are also able to do vertical cluster coupons from the knowing which coupon categories in each cluster are most unsold by considering the Word Clouds visualization.

## Conclusions

* According to the data, there are many variables eliminated since they are unnecessary for analyzing, such as Coupon display from-end, Date of valid coupon from-end. Coupons those can be used on weekend are definitely has higher purchase number than the one that only weekday usable.
* From the association rules generated, female consumers are always interested in coupon that is valid in the “Shinjuku” are, which represents the big shopping center and business area. Senior people would have more potential to purchase the coupon in leisure, relaxation, spa, and Japanese hotel than other age groups since these kinds of coupon have high price rate.
* We sketchily divide the unsold coupons into five different group based on their price and user’s profile by using clustering analysis. This makes company more understand what coupon they should be more focus on. Company can create promotion/campaign for selling differently for each group. To be a deeper analysis, in each group, they are also able to do vertical cluster coupons from the knowing which coupon categories in each cluster are most unsold by considering the Word Clouds visualization.

## Coupon sales classification – Wenbo Zhao

### Introduction

In order to help the website to sell more coupons and generate more profit, we decided to discover insights that specify what kind of coupon would have a high sales volume.

The coupons’ sales volume varies from 0 to 345. There are totally 18540 coupons in the training set for us to develop prediction models. To determine whether a coupon is popular, I set a criteria of 10 for the coupon sales, which indicates that coupons which were sold equal to or more than 10 times are regarded as good coupons. Besides, I also assign coupons that never being sold to the category of unsold coupons. About 14% and 11% of all coupons are good coupons and unsold coupons, respectively. I categorize the rest 75% of the coupons into the group of bad coupons.

[10, +]: Good Coupon

[1, 10]: Bad Coupon

0: Unsold Coupon

To predict the coupon popularity, 18 features were utilized as independent variables. I briefly grouped them into 5 groups.

1. **Price features**, in which price rate, discount price and price after discount are included. For example, if a product’s original price is 1000 and its price rate is 40%, then the discount price and price after discount are 400 and 600 respectively.
2. **Period features** includes the period displayed on the website and coupon valid period.
3. **Usage features**. There are 9 features in this group to specify whether coupons can be applied on each day in a week and before or on holidays.
4. **Area features** tell us where coupons can be applied. 3 features, which are small, middle and large area respectively, are included this group.
5. **Coupon’s genre**: only 1 feature, genre, is in this group. The feature includes values like deliver service and food.

We used 5 different classifiers to do the prediction. They are Decision Tree, Naive Bayes, K Nearest Neighbor, Logistic Regression and Support Vector Machine. Since Logistic Regression and Support Vector Machine are binary classifiers, we applied one-against-rest method on them to predict the 3-classes dependent variables.

### Classification

After tuning model parameters, the total accuracy and Good Coupon prediction precision for each classifier are as follow:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Classifiers** | Decision Tree | SVM | KNN | Log regression | Naive Bayes |
| **Accuracy** | 40.75% | 54.50% | 70.90% | 70.62% | 70.06% |
| **Classifiers** | Decision Tree | SVM | KNN | Log regression | Naive Bayes |
| **Good Coupon Precision** | 27.62% | 29.98% | 42.69% | 39.02% | 32.79% |

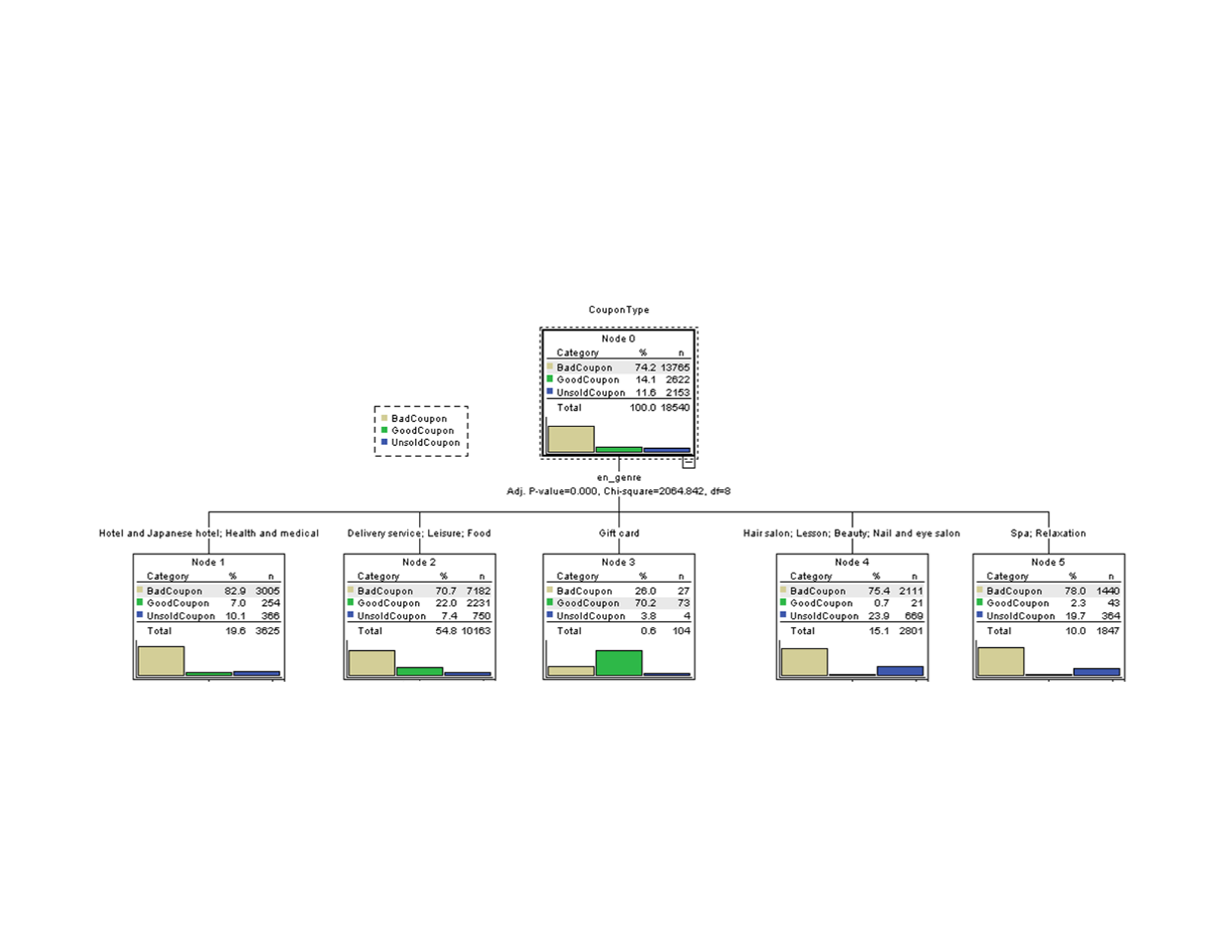


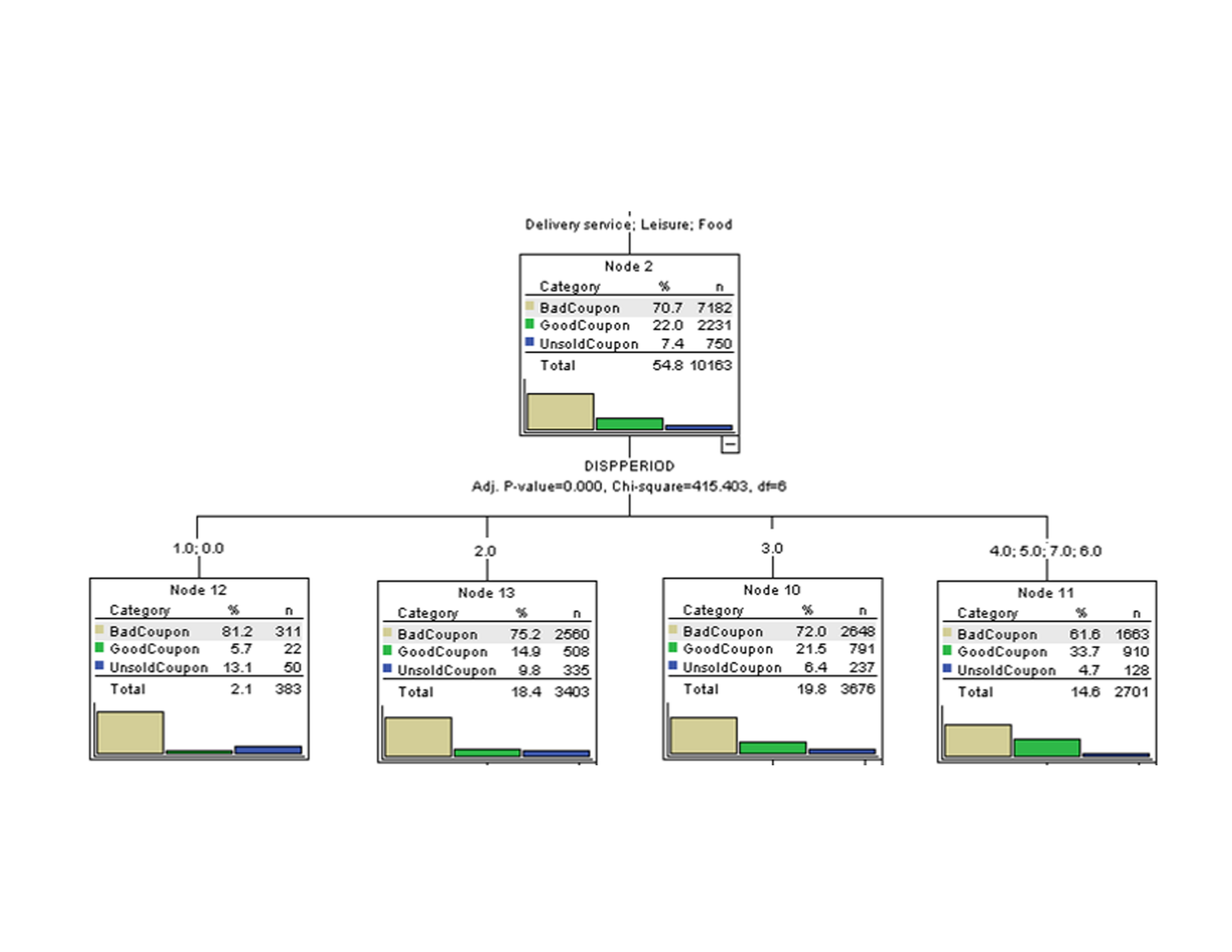
|  |  |  |  |
| --- | --- | --- | --- |
| **Ensemble Methods** | Hard | Soft | Weighted Soft |
| **Accuracy** | 70.87% | 74.01% | 74.08% |
| **Good Coupon Precision** | 42.26% | 50.66% | 50.94% |

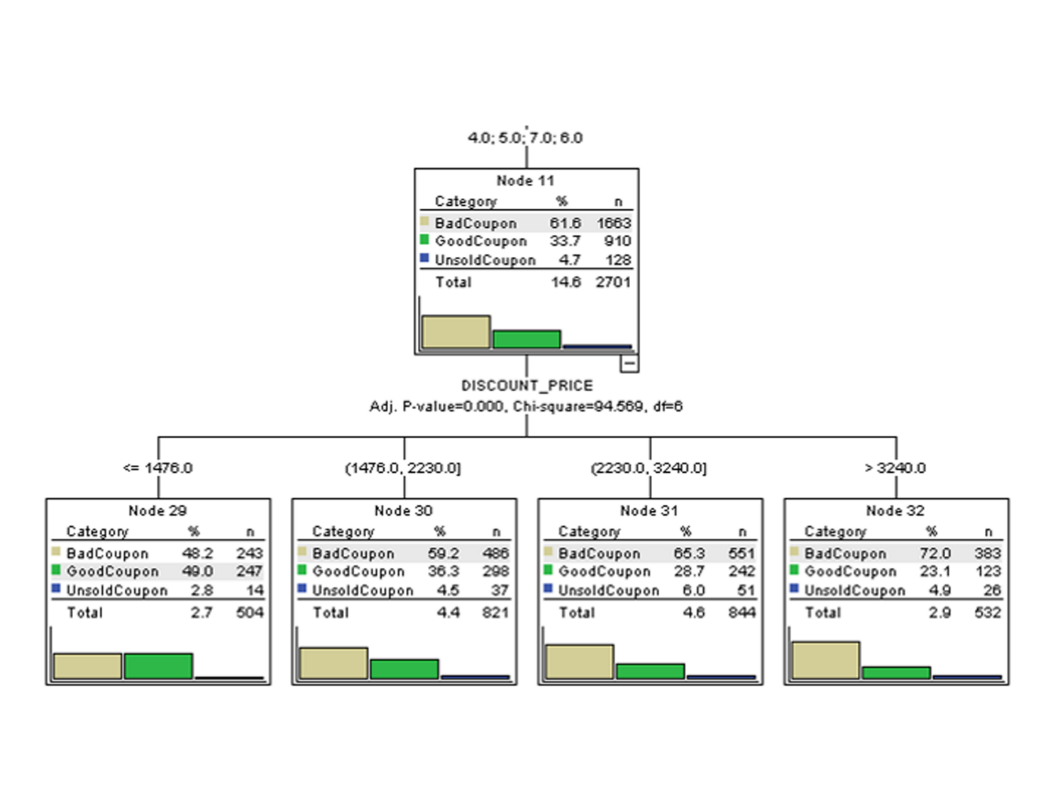
In the hard ensemble learning, the prediction accuracy and the precision of Good Coupon prediction is improved. The total accuracy of hard ensemble learning was 70.87% and the Good Coupon precision reached 42.26%. The soft ensemble provided a better performance with 74.01% of total accuracy and 50.66% of Good Coupon precision. The weighted soft ensemble learning further improved performances. The high performance of ensemble learning indicated that each classifier gave an output from its own aspect. After gathering all of them, we are able to discover enough information which we are looking for.

### Findings

From the decision tree, we discovered some helpful information to optimize the business. The most important variable to decide whether a coupon will have a high sales volume is Coupon Category. For the coupon of Food, Leisure and Delivery Service, Display Period influences sales volume significantly. The longer the display period is, the higher the probability with which a coupon is sold well. Then, Price after discount is the decision maker. No matter the display period is, a lower price gives a higher probability for a coupon to be sold well. Therefore, if restaurants, delivery service providers or leisure centers plan to use coupon to promote their business, they should reduce their price and extend the coupon display period as long as possible.







## Prediction of coupons purchased – Tianzi Xu

### Introduction

The problem I am exploring in the data is to predict coupon purchase number. I used stepwise selection, KNN regression, support vector machine regression and regression trees to predict the value of PURCHASE\_NUMBER (response variable).

My numeric variables contain:

DISCOUNT\_PRICE: price after using coupons, PRICE\_RATE: rate of discount; PRICE: original price of products and service; VALIDPERIOD: Validity period (day); DISPPERIOD: Sales period (day); USABLE\_DATE\_HOLIDAY: Is available on holiday; SEX\_ID: Gender; USABLE\_DATE\_(day): Is available (day); USABLE\_DATE\_BEFORE\_HOLIDAY: Is available on the day before holiday; GENRE\_NAME: Category name; large\_area\_name: Large area name of shop location.

I also used small\_area\_name and ken\_name since I am not sure which one is better for model building.

I transferred categorical variables to dummy variables to make it easier to do regression. In addition, I am not sure that among ken\_name, large\_area\_name and small-area\_name, which one contributes more to the mode. I keep all of them and try each one on the model.

To do KNN regression, I normalized numeric variables to compute distances among observations.

To evaluate models, I used adj-R2 to evaluate the model since adj-r2 is most direct evaluation for performance of modes.

### Model Analysis

First I create three simple linear models respectively includes small\_area\_name, large\_area\_name and ken\_name. The adjusted r-squared are around 64%, not widely different. Thus, to make model simpler, I used large\_area\_name in the following steps. Then I tried feature selection, adjusted R-squared did not improve any more.

I tried KNN regression, spliting dataset into training set and testing set. The model comes from training set can achieve 70%. However, for testing set, the model can not fit well. This technique doesn’t work. It means that, for current dataset, there are not too many similar features among the coupons.

I also tried support vector machien regression. It did not work neither.

At last, I run regression trees algorithm and prune the tress through cross-validation method. Small\_area\_name, large\_area\_name and ken\_name does not make differences within this techniques. The r-squared achieves nearly 80%. To get more insight from regression tree, I take view\_number out because most of groups are split by this attribute. Then I get another regression tree, even though I get a much lower r-square. The second regression tree gave us more details.

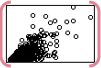
Findings:

From the initial analysis, my findings are as follows:

* The small\_area\_name, large\_area\_name and ken\_name make no difference in the model.
* The following graph is the correlation between price\_rate and validperiod. The x axis is price\_rate. This graph told us that when price\_rate increases, the period of validity becomes longer.

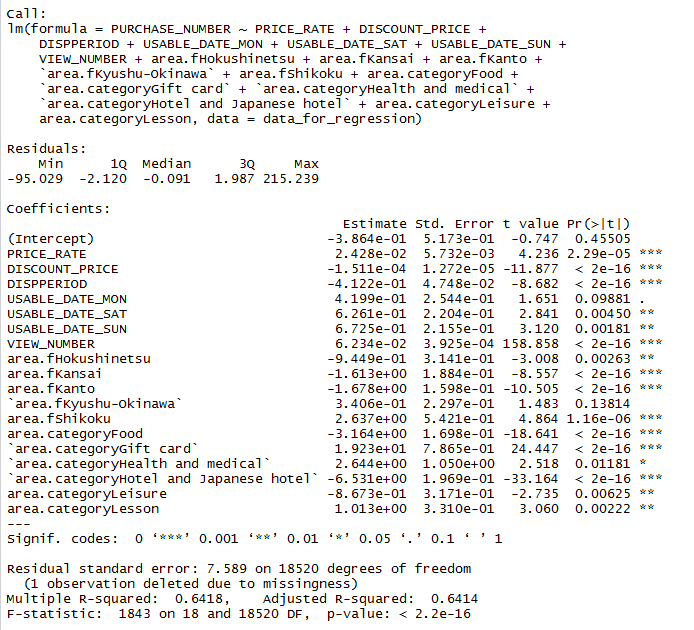


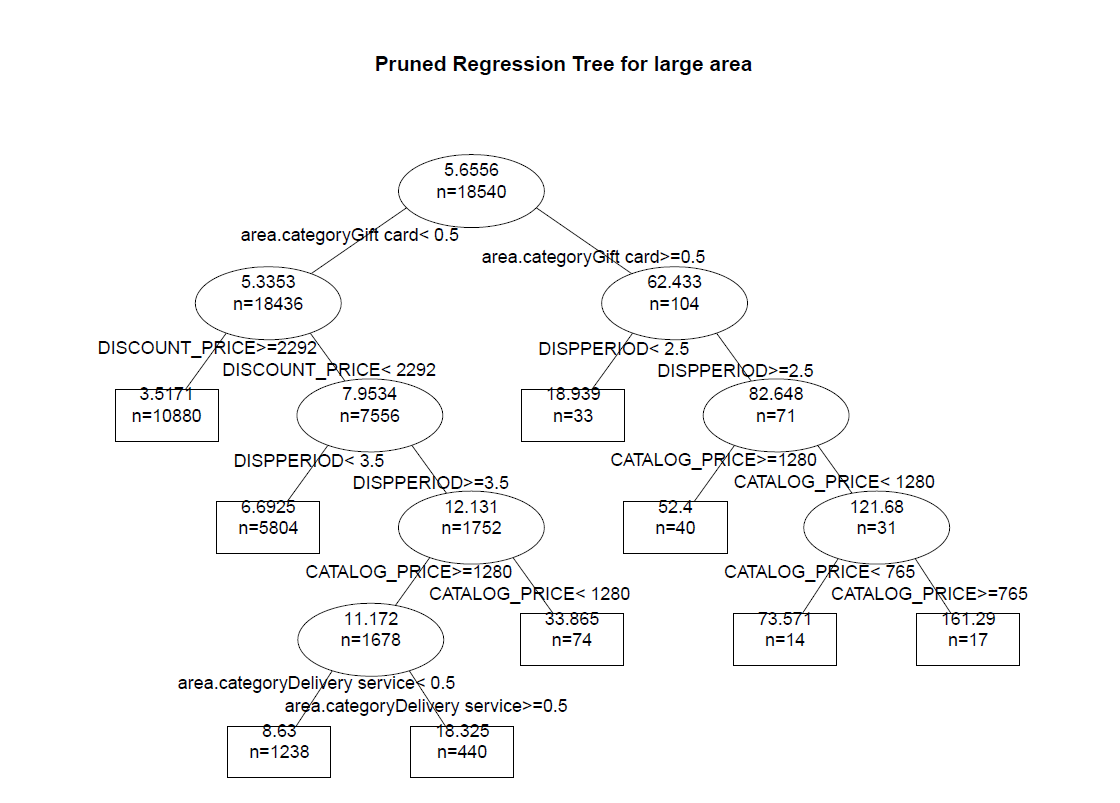
* The following graph is the correlation between view\_number and purchase\_number. It shows us that they are high correlated with each other.



From the stepwise regression, here are my findings:

* When price\_rate and view\_number increase, sales volume will increase.
* For categorical variables about usable\_data, we concluded that : If the coupon is usable on Monday, Saturday and Sunday, its purchase\_number will be higher.
* For categories of coupons, we concluded that: Gift card, health and medical as well as lesson coupons are sold more. Food, hotel and leisure coupons are sold less.
* For shop location (using Large\_area\_name), Coupons which should be used in
* Hokushinetsu, Kansai and Kanto have lower purchase number. Coupons which should be used in Shikoku and Kyushu-Okinawa have higher purchase number.





# Analysis

* C5.0 model can be used to classify the coupons categories.
* Gift cards are available more than other coupons on the website but are not sold as much. The company can cut short of these coupons and advertise coupons from food and delivery services which are bought most.
* These coupons can also be made available in bigger quantities on weekends, since their purchase behavior indicates huge sales on Saturdays and Sundays. Holidays can also be used to advertise coupons from Lifestyle category.
* Women can be targeted to sell coupons from lifestyle markets. The company can benefit from these sales by sending them emails before holidays or weekends to make most profit.
* Food is the biggest category from which most of the coupons are sold and are preferred by men and womed. The company can approach new restaurants/cafes to not only advertise the upcoming restaurants but also sell their coupons.

# Conclusions

* Divided sales volume into categories with respect to different genders, different ranges of age or different area.
* Apply both prediction and classification models on categorized sales volume to discover more patterns.
* Conduct feature engineering to gain more information.
* More customer understanding by performing segmentation based on further features: by lifestyle, demographics, etc.

# Appendix

## Classification of coupon category

library(partykit)

library(dplyr)

library(gbm)

library(sandwich)

library(party)

library(zoo)

library(gmum.r)

library(gbm)

library(nnet)

library(caretEnsemble)

library(kernlab)

library(cvAUC)

library(MASS)

library(pROC)

library(ROCR)

library(e1071)

library(rpart)

library(ggplot2)

library(lattice)

library(caret)

library(randomForest)

library(plyr)

library(rattle)

library(rpart.plot)

library(RColorBrewer)

library(C50)

library(cwhmisc)

#data preprocessing

head(balanced\_data)

balanced\_data$Capsule.Text<-NULL

balanced\_data$Ken.Name<-NULL

balanced\_data$Large.Area.Name<-NULL

balanced\_data$Small.Area.Name<-NULL

balanced\_data$New.Genre.name <- as.factor(balanced\_data$New.Genre.name)

balanced\_data$Item.Count <- as.numeric(balanced\_data$Item.Count)

for(level in unique(balanced\_data$Usable.Date.Mon)){

balanced\_data[paste("Mon", level, sep = "\_")] <- ifelse(balanced\_data$Usable.Date.Mon == level, 1, 0)

}

for(level in unique(balanced\_data$Usable.Date.Tue)){

balanced\_data[paste("Tue", level, sep = "\_")] <- ifelse(balanced\_data$Usable.Date.Tue == level, 1, 0)

}

for(level in unique(balanced\_data$Usable.Date.Wed)){

balanced\_data[paste("Wed", level, sep = "\_")] <- ifelse(balanced\_data$Usable.Date.Wed == level, 1, 0)

}

for(level in unique(balanced\_data$Usable.Date.Thu)){

balanced\_data[paste("Thu", level, sep = "\_")] <- ifelse(balanced\_data$Usable.Date.Thu == level, 1, 0)

}

for(level in unique(balanced\_data$Usable.Date.Fri)){

balanced\_data[paste("Fri", level, sep = "\_")] <- ifelse(balanced\_data$Usable.Date.Fri == level, 1, 0)

}

for(level in unique(balanced\_data$Usable.Date.Sat)){

balanced\_data[paste("Sat", level, sep = "\_")] <- ifelse(balanced\_data$Usable.Date.Sat == level, 1, 0)

}

for(level in unique(balanced\_data$Usable.Date.Sun)){

balanced\_data[paste("Sun", level, sep = "\_")] <- ifelse(balanced\_data$Usable.Date.Sun == level, 1, 0)

}

for(level in unique(balanced\_data$Usable.Date.Holiday)){

balanced\_data[paste("Hol", level, sep = "\_")] <- ifelse(balanced\_data$Usable.Date.Holiday == level, 1, 0)

}

for(level in unique(balanced\_data$Usable.Date.Before.Holiday)){

balanced\_data[paste("BeforeHol", level, sep = "\_")] <- ifelse(balanced\_data$Usable.Date.Before.Holiday == level, 1, 0)

}

for(level in unique(balanced\_data$I.Year)){

balanced\_data[paste("IYear", level, sep = "\_")] <- ifelse(balanced\_data$I.Year == level, 1, 0)

}

for(level in unique(balanced\_data$Year.disp)){

balanced\_data[paste("Yeardisp", level, sep = "\_")] <- ifelse(balanced\_data$Year.disp == level, 1, 0)

}

for(level in unique(balanced\_data$Year.dispend)){

balanced\_data[paste("YeardispEnd", level, sep = "\_")] <- ifelse(balanced\_data$Year.dispend == level, 1, 0)

}

for(level in unique(balanced\_data$Gender)){

balanced\_data[paste("Gender", level, sep = "\_")] <- ifelse(balanced\_data$Gender == level, 1, 0)

}

balanced\_data$Year.dispend<-NULL

balanced\_data$Usable.Date.Mon<-NULL

balanced\_data$Usable.Date.Tue<-NULL

balanced\_data$Usable.Date.Wed<-NULL

balanced\_data$Usable.Date.Thu<-NULL

balanced\_data$Usable.Date.Fri<-NULL

balanced\_data$Usable.Date.Sat<-NULL

balanced\_data$Usable.Date.Sun<-NULL

balanced\_data$Usable.Date.Holiday<-NULL

balanced\_data$Usable.Date.Before.Holiday<-NULL

balanced\_data$I.Year<-NULL

balanced\_data$Year.disp<-NULL

balanced\_data$Gender<-NULL

balanced\_data$Item.Count..bin.<-NULL

balanced\_data$Number.of.Records<-NULL

head(balanced\_data)

#data normalization

data<-NULL

norm <- function(x){(x-min(x))/(max(x)-min(x))}

#catalog price

CatalogPrice <- norm(balanced\_data$Catalog.Price)

data<-cbind(balanced\_data, CatalogPrice)

#Discount price

DiscountPrice <- norm(balanced\_data$Discount.Price)

data<-cbind(data, DiscountPrice)

#Dispperiod

DispPeriod <- norm(balanced\_data$Dispperiod)

data<-cbind(data, DispPeriod)

#Item.Count

ItemCount <- norm(balanced\_data$Item.Count)

data<-cbind(data, ItemCount)

#Price.Rate

PriceRate <- norm(balanced\_data$Price.Rate)

data<-cbind(data, PriceRate)

#age

Age <- norm(balanced\_data$AGE)

data<-cbind(data, Age)

data$Catalog.Price<-NULL

data$Discount.Price<-NULL

data$Dispperiod<-NULL

data$Item.Count<-NULL

data$Price.Rate<-NULL

data$AGE<-NULL

data$Capsule.Text<-NULL

data$Ken.Name<-NULL

data$Large.Area.Name<-NULL

data$Small.Area.Name<-NULL

head(data)

set.seed(1234)

#function to calculate performance metrics:

#Accuracy, Precision, Recall, F measure,macroPrecision,macroRecall,macroF1

perfMetrics <- function(t,test){

n = sum(t) # number of instances

nc = nrow(t) # number of classes

diag = diag(t) # number of correctly classified instances per class

rowsums = apply(t, 1, sum) # number of instances per class

colsums = apply(t, 2, sum) # number of predictions per class

p = rowsums / n # distribution of instances over the actual classes

q = colsums / n # distribution of instances over the predicted classes

accuracy = sum(diag)/nrow(test)

precision = diag / colsums

recall = diag / rowsums

f1 = 2 \* precision \* recall / (precision + recall)

#Macro-averaged Metrics

macroPrecision = mean(precision)

macroRecall = mean(recall)

macroF1 = mean(f1)

result <- data.frame(accuracy,precision, recall, f1,macroPrecision,macroRecall,macroF1)

return(result)

}

#trainig and testing sets

ind <- sample(2, nrow(balanced\_data), replace=TRUE, prob=c(0.8, 0.2))

train <- balanced\_data[ind==1,]

test <- balanced\_data[ind==2,]

nrow(test)

nrow(train)

#Decision trees

model.tree <- rpart(New.Genre.name ~ . , data=train)

pred.model.tree <- predict(model.tree, test, type = "class")

t<-table(pred.model.tree,test$New.Genre.name)

confusionMatrix(t)

#AUC

predictions <- as.numeric(predict(model.tree, test, type = 'class'))

multiclass.roc(test$New.Genre.name, predictions)

result<-perfMetrics(t,test)

result[,2:4]

result[1,c(1,5:7)]

fancyRpartPlot(model.tree)

#C5.0

model.c50 <- C5.0(train[-3], train$New.Genre.name)

pred.model.c50 <- predict(model.c50, test, type = "class")

t<-table(pred.model.c50,test$New.Genre.name)

confusionMatrix(t)

summary(model.c50)

plot(model.c50, subtree = 3)

C5.0.graphviz(model.c50,'C:\\Users\\Mitali\\Documents\\MS\\plot.txt',

col.question ='cyan')

#AUC

predictions <- as.numeric(predict(model.c50, test, type = 'class'))

multiclass.roc(test$New.Genre.name, predictions)

result<-perfMetrics(t,test)

result[,2:4]

result[1,c(1,5:7)]

#Support vector machines

# One versus one is solving K(K-1)/2 subproblems (one for each pair)

x<-data[,-1]

head(x)

sv.ovo <- SVM(x=x, y=data$New.Genre.name, class.type="one.versus.one", verbosity=0)

preds <- predict(sv.ovo, x)

acc.ovo <- sum(diag(table(preds, data$New.Genre.name)))/sum(table(preds, data$New.Genre.name))

pred <- predict(sv.ovo,test)

t <- table(pred,test$New.Genre.name)

confusionMatrix(t)

result<-perfMetrics(t,test)

result[,2:4]

result[1,c(1,5:8)]

#AUC

predictions <- as.numeric(predict(sv.ovo, test, type = 'prob'))

multiclass.roc(test$New.Genre.name, predictions)

#Random Forest

#n=10

rf.10 <- randomForest(New.Genre.name ~ ., data=train, ntree=10)

pred.rf.10 <- predict(rf.10, test, type = "class")

t <- table(pred.rf.10,test$New.Genre.name)

confusionMatrix(t)

varImpPlot(rf.10)

#performance metrics

result<-perfMetrics(t,test)

result[,2:4]

result[1,c(1,5:8)]

#AUC

predictions <- as.numeric(predict(rf.10, test, type = 'response'))

multiclass.roc(test$New.Genre.name, predictions)

#n=20

rf.20 <- randomForest(New.Genre.name ~ ., data=train, ntree=20)

pred.rf.20 <- predict(rf.20, test, type = "class")

t <- table(pred.rf.20,test$New.Genre.name)

confusionMatrix(t)

varImpPlot(rf.20)

plot(rf.20)

#AUC

predictions <- as.numeric(predict(rf.20, test, type = 'response'))

multiclass.roc(test$New.Genre.name, predictions)

#performance metrics

result<-perfMetrics(t,test)

result[,2:4]

result[1,c(1,5:7)]

#n=25

rf.25 <- randomForest(New.Genre.name ~ ., data=train, ntree=25)

pred.rf.25 <- predict(rf.25, test, type = "class")

t <- table(pred.rf.25,test$New.Genre.name)

confusionMatrix(t)

varImpPlot(rf.25)

plot(rf.25)

#AUC

predictions <- as.numeric(predict(rf.25, test, type = 'response'))

multiclass.roc(test$New.Genre.name, predictions)

#performance metrics

result<-perfMetrics(t,test)

result[,2:4]

result[1,c(1,5:7)]

rf.100 <- randomForest(New.Genre.name ~ ., data=train, ntree=100)

pred.rf.100 <- predict(rf.100, test, type = "class")

t <- table(pred.rf.100,test$New.Genre.name)

confusionMatrix(t)

varImpPlot(rf.100)

plot(rf.100)

#performance metrics

result<-perfMetrics(t,test)

result[,2:4]

result[1,c(1,5:7)]

#AUC

predictions <- as.numeric(predict(rf.100, test, type = 'response'))

multiclass.roc(test$New.Genre.name, predictions)

#CROSS VALIDATION

##DT

train\_control <- trainControl(method="cv", number=10)

model <- train(New.Genre.name~., data=balanced\_data, trControl=train\_control, method="rpart")

print(model)

pred <- predict(model, test, type = "raw")

t <- table(pred,test$New.Genre.name)

confusionMatrix(t)

#performance metrics - AUC

predictions <- as.numeric(predict(model, test, type = 'raw'))

multiclass.roc(test$New.Genre.name, predictions)

##RF

train\_control <- trainControl(method="cv", number=10)

mrf <- train(New.Genre.name~., data=balanced\_data, trControl=train\_control, method="cforest")

proc.time()

print(mrf)

pred <- predict(model, test, type = "raw")

t <- table(pred,test$New.Genre.name)

confusionMatrix(t)

#performance metrics - AUC

predictions <- as.numeric(predict(model, test, type = 'raw'))

multiclass.roc(test$New.Genre.name, predictions)

set.seed(seed)

control <- trainControl(method="repeatedcv", number=10, repeats=3)

metric <- "Accuracy"

# Stochastic Gradient Boosting

fit.gbm <- train(New.Genre.name~., data=balanced\_data, method="gbm", metric=metric, trControl=control, verbose=FALSE)

summary(fit.gbm)

print(fit.gbm)

# summarize results

boosting\_results <- resamples(list(c5.0=fit.c50, gbm=fit.gbm))

summary(boosting\_results)

dotplot(boosting\_results)

#performance metrics - AUC

predictions <- as.numeric(predict(fit.gbm, test, type = 'raw'))

multiclass.roc(test$New.Genre.name, predictions)

#stacking ensembles

control <- trainControl(method="repeatedcv", number=10, repeats=3, savePredictions=TRUE, classProbs=TRUE)

algorithmList <- c( 'rpart', 'rf', 'gbm', 'treebag')

set.seed(seed)

ensemble.model <- caretList(New.Genre.name~., data=balanced\_data, trControl=control, methodList=algorithmList)

results <- resamples(ensemble.model)

summary(results)

dotplot(results)

#--------------------------------------------------------------------------------------------------------------------------------------------------------------

## Association Rules - Nunt Ploydanai

install.packages("arules")

library(arules)

rules <- apriori(AssociationRules\_User1)

inspect(rules2)

rules2 <- apriori(AssociationRules\_User1, parameter = list(minlen=4, supp=0.004, conf=0.4), appearance = list(rhs=c("PURCHASE\_FLG=NO", "PURCHASE\_FLG=YES"), default="lhs"), control = list(verbose=F))

rules2.sorted <- sort(rules2, by="support")

rules2.sorted <- sort(rules2, by="lift")

inspect(rules2.sorted)

inspect(rules2[201:300])

inspect(rules2[1:100])

plot(rules2)

plot(rules2, method="graph", control=list(type="items"))

plot(rules2, method="grouped")

# For clustering

# Prepare data

coupon\_train <- read.csv("coupon\_list\_train.csv", as.is = T)

coupon\_test <- read.csv("coupon\_list\_test.csv", as.is = T)

couponClus <- read.csv("Calculated\_Coupon\_List.csv", as.is = T)

couponClus$X <- NULL

couponClus$DISPFROM <- NULL

couponClus$DISPEND <- NULL

couponClus$COUPON\_ID\_hash <- NULL

couponClus$VIEW\_PURCHASE\_RATE <- NULL

couponClus[5:19] <- list(NULL)

couponClus[5:7] <- list(NULL)

couponClus$VIEW\_PURCHASE\_RATE <- NULL

couponClus[6:19] <- list(NULL)

couponClus$Discount\_rate<- (couponClus$DISCOUNT\_PRICE/couponClus$CATALOG\_PRICE)

couponClus$VALIDPERIOD <- NULL

couponClus$DISCOUNT\_PRICE <- NULL

couponClus$VALIDPERIOD <- NULL

couponClus$VIEW\_NUMBER <- NULL

couponClus$Disc\_Rate <- couponClus$DISCOUNT\_PRICE/couponClus$CATALOG\_PRICE

#coupon\_train$is\_train <- 1

#coupon\_test$is\_train <- 0

#Combining both test and train data

#all\_coupon <- rbind(coupon\_train, coupon\_test)

# Prepare for cluster

#str(all\_coupon)

# Add attributes to be clustered here. Avoid categorical data.

#attributes <- c("PRICE\_RATE", "CATALOG\_PRICE", "DISCOUNT\_PRICE", "DISPPERIOD","PURCHASE\_NUMBER")

#mathcing attributes from testand train data to new data frame

#coupon\_cluster <- all\_coupon[names(all\_coupon) %in% attributes]

#coupon\_cluster <- all\_coupo[,names(all\_coupon) %in% c("COUPON\_ID\_hash", "cluster")]

#wss <- vector(mode = "numeric", length = 10)

#for (n in 1:10) {

# wss[n] <- sum(kmeans(coupon\_cluster, n)$withinss)

#}

#plot(1:10, wss)

similarity <- vector(mode = "numeric", length = 30)

for (n in 1:30) {

similarity[n] <- sum(kmeans(couponClus, n)$withinss)

}

plot(1:30, similarity)

#coupon\_kmeans <- kmeans(coupon\_cluster, 4)

#coupon\_kmeans$centers # 1: low price low rate, 4: high rate high price.

#all\_coupon$cluster <- coupon\_kmeans$cluster

coupon\_kmeans <- kmeans(couponClus, 5)

coupon\_kmeans$centers # 1: low price low rate, 4: high rate high price.

couponClus$cluster <- coupon\_kmeans$cluster

Calculated\_Coupon\_List$cluster <- coupon\_kmeans$cluster

couponClus$cluster <-NULL

coupon\_data <- couponClus

coupon\_target <- coupon\_kmeans$cluster

coupon\_data <- read.csv("Calculated\_Coupon\_List.csv", as.is = T)

coupon\_data$X <- NULL

coupon\_data$DISPFROM <- NULL

coupon\_data$DISPEND <- NULL

coupon\_data$COUPON\_ID\_hash <- NULL

coupon\_data$VIEW\_PURCHASE\_RATE <- NULL

coupon\_data[2:3] <- list(NULL)

coupon\_data[3:4] <- list(NULL)

coupon\_data[4:18] <- list(NULL)

coupon\_data[6:19] <- list(NULL)

coupon\_data$VIEW\_PURCHASE\_RATE <- NULL

write.csv(coupon\_data,"coupon\_data.csv")

write.csv(coupon\_target,"coupon\_target.csv")

coupon\_cat <- read.csv("Calculated\_Coupon\_List.csv", as.is = T)

coupon\_cat <- coupon\_cat$en\_capsule

write.csv(coupon\_cat,"coupon\_cat.csv")

# Descriptive statistics

Calculated\_Coupon\_List$en\_capsule <- as.factor(Calculated\_Coupon\_List$en\_capsule)

Calculated\_Coupon\_List$en\_genre <- as.factor(Calculated\_Coupon\_List$en\_genre)

Calculated\_Coupon\_List$is\_train <- as.factor(Calculated\_Coupon\_List$is\_train)

show\_attr <- c("en\_capsule", "en\_genre", "VISIT", "PURCH", "is\_train", "PRICE\_RATE", "CATALOG\_PRICE", "DISCOUNT\_PRICE")

c <- 1

summary(Calculated\_Coupon\_List[Calculated\_Coupon\_List$cluster==c, names(Calculated\_Coupon\_List) %in% show\_attr])

c <- 2

summary(all\_coupon[all\_coupon$cluster==c, names(all\_coupon) %in% show\_attr])

c <- 3

summary(all\_coupon[all\_coupon$cluster==c, names(all\_coupon) %in% show\_attr])

c <- 4

summary(all\_coupon[all\_coupon$cluster==c, names(all\_coupon) %in% show\_attr])

coupon\_cluster <- Calculated\_Coupon\_List[,names(Calculated\_Coupon\_List) %in% c("COUPON\_ID\_hash", "en\_capsule", "cluster")]

hist(Visit\_Purchase$PURCHASE\_FLG)

attrs <- c("PURCHASE\_FLG", "COUPON\_ID\_hash", "USER\_ID\_hash")

visit <- Visit\_Purchase[,names(Visit\_Purchase) %in% attrs]

visit\_table <- data.table(visit)

group\_table <- visit\_table[,list(C\_VISIT = .N, C\_PURCH = sum(PURCHASE\_FLG==1)),by=COUPON\_ID\_hash]

coupon\_train <- merge(Calculated\_Coupon\_List, group\_table, by.x = "COUPON\_ID\_hash", by.y = "COUPON\_ID\_hash", all.x = T)

coupon\_test <- merge(coupon\_test, group\_table, by.x = "COUPON\_ID\_hash", by.y = "VIEW\_COUPON\_ID\_hash", all.x = T)

coupon\_train[is.na(coupon\_train)] <- 0

plot(coupon\_train$C\_VISIT, coupon\_train$C\_PURCH) #Looks like there is an outlier

cor(coupon\_train$C\_VISIT, coupon\_train$C\_PURCH) #0.7789

#unsold

unsold <- Visit\_Purchase[which(Visit\_Purchase$PURCHASE\_FLG < 1),]

counts <- table(unsold$en\_genre)

barplot(counts, main="unsold")

unsold[1:2] <- list(NULL)

unsold[4:5] <- list(NULL)

unsold[5:21] <- list(NULL)

unsold$DISCOUNT\_PRICE <- NULL

unsold$Disc\_Rate <- unsold$DISCOUNT\_PRICE/unsold$CATALOG\_PRICE

unsold$cluster <- NULL

unsold[is.na(unsold)] <- 0

wss <- vector(mode = "numeric", length = 10)

for (n in 1:10) {

wss[n] <- sum(kmeans(unsold, n)$withinss)

}

plot(1:10, wss)

coupon\_kmeans <- kmeans(unsold, 4)

coupon\_kmeans$centers # 1: low price low rate, 4: high rate high price.

unsold$cluster <- coupon\_kmeans$cluster

Calculated\_Coupon\_List$cluster <- coupon\_kmeans$cluster

C1 <-subset(coupon\_cluster, cluster== "1")

C2 <-subset(coupon\_cluster, cluster== "2")

C3 <-subset(coupon\_cluster, cluster== "3")

C4 <-subset(coupon\_cluster, cluster== "4")

C5 <-subset(coupon\_cluster, cluster== "5")

# Number of visit and purchase of each user and coupon

group\_table <- visit\_table[,list(VISIT = .N, PURCH = sum(PURCHASE\_FLG==1)), by=list(COUPON\_ID\_hash, USER\_ID\_hash)]

visit\_user\_count <- group\_table

write.csv(visit\_user\_count,"visit\_user\_count.csv")

# Number of visit and purchanse of each genre

genre\_table <- data.table(coupon\_train)[,list(COUPON = .N, VISIT = sum(C\_VISIT), PURCH = sum(C\_PURCH)), by=en\_genre]

write.csv(genre\_table, "genre\_table.csv")

Good <- Good\_Coupon\_List[which(Good\_Coupon\_List$CouponType == "Yes"),]

write.csv(Good,"Good.csv")

Bad <- Good\_Coupon\_List[which(Good\_Coupon\_List$CouponType == "No"),]

write.csv(Bad,"Bad.csv")

# Number of visit and purchanse of each user

group\_table <- visit\_table[,list(VISIT = .N, PURCH = sum(PURCHASE\_FLG==1)), by=USER\_ID\_hash]

user\_list <- merge(user\_list, group\_table, by="USER\_ID\_hash", all.x = T, all.y = F)

hypothesis\_1 <- table(user\_list$AGE, user\_list$SEX\_ID, user\_list$PURCH)

write.csv(user\_list,"user\_groups.csv")

hist(user\_list$PURCH)

hist(user\_list$VISIT)

hist(user\_list$AGE)

# How many time user visit each genre

coupon.genre <- coupon\_train[,names(coupon\_train) %in% c("COUPON\_ID\_hash", "en\_genre")]

coupon.genre <- rbind(coupon.genre, coupon\_test[,names(coupon\_test) %in% c("COUPON\_ID\_hash", "en\_genre")])

visit.genre <- merge(visit, coupon.genre, by.x = "COUPON\_ID\_hash", by.y = "COUPON\_ID\_hash", all.x = T, all.y = F)

visit.genre <- visit.genre[!is.na(visit.genre$en\_genre),]

visit.genre <- data.table(visit.genre)

visit.genre.count <- visit.genre[,list(G\_VISIT = .N, G\_PURCH = sum(PURCHASE\_FLG==1)), by=list(COUPON\_ID\_hash, en\_genre)]

visit.genre.count <- as.data.frame(visit.genre.count)

rm(visit.genre)

# How many time user visit each cluster

visit.cluster <- merge(visit, coupon\_cluster, by.x = "COUPON\_ID\_hash", by.y = "COUPON\_ID\_hash", all.x = T, all.y = F)

visit.cluster <- visit.cluster[!is.na(visit.cluster$cluster)]

visit.cluster <- data.table(visit.cluster)

visit.cluster.count <- visit.cluster[,list(Cl\_VISIT = .N, Cl\_PURCH = sum(PURCHASE\_FLG==1)), by=list(USER\_ID\_hash, cluster)]

visit.cluster.count <- as.data.frame(visit.cluster.count)

coupon\_train[,'C\_PURCH'] <- as.numeric(as.character(coupon\_train[,'C\_PURCH']))

coupon\_train$G\_PURCH[coupon\_train$C\_PURCH == 0] <- 0

coupon\_reg <- coupon\_train

coupon\_reg$CATALOG\_PRICE <- NULL

Visit\_Purchase[4:12] <- list(NULL)

Visit\_Purchase[7:8] <- list(NULL)

Visit\_Purchase[8:24] <- list(NULL)

Visit\_Purchase$PREF\_NAME <- NULL

Visit\_Purchase[is.na(Visit\_Purchase)] <- 0

Visit\_Purchase <- Visit\_Purchase[which(Visit\_Purchase$PRICE\_RATE > 1),]

write.csv(Visit\_Purchase,"visit\_purchase.csv")

visit222 <- merge(Visit\_Purchase, visit\_user\_count, by.x = "COUPON\_ID\_hash", by.y = "COUPON\_ID\_hash", all.x = T, all.y = F)

visit222 <- rbind(visit\_user\_count, Visit\_Purchase[,names(Visit\_Purchase) %in% c("VISIT", "PURCH")])

Visit\_Purchase <- merge(Visit\_Purchase, visit\_user\_count, by="COUPON\_ID\_hash", all.x = T, all.y = F)

user\_list <- merge(user\_list, group\_table, by="USER\_ID\_hash", all.x = T, all.y = F)

Visit\_reg <- Visit\_Purchase[,list(VISIT = .N, PURCH = sum(PURCHASE\_FLG==1)), by=list(COUPON\_ID\_hash, USER\_ID\_hash)]

group\_table <- visit\_table[,list(VISIT = .N, PURCH = sum(PURCHASE\_FLG==1)), by=list(COUPON\_ID\_hash, USER\_ID\_hash)]

set.seed(1234)

ind <- sample(2, nrow(Regress\_coupon), replace=TRUE, prob=c(0.8, 0.2))

train <- Regress\_coupon[ind==1,]

test <- Regress\_coupon[ind==2,]

lm\_model <- glm(PURCHASE\_FLG ~ PRICE\_RATE + CATALOG\_PRICE + DISCOUNT\_PRICE + DISPPERIOD + VISIT + PURCH,

data = train,

family = binomial(link = "logit"))

summary(lm\_model)

library(pscl)

pR2(lm\_model)

#predict <- predict(lm\_model, type = 'response')

#table(train$PURCHASE\_FLG, predict > 0.5)

pred = predict(lm\_model, newdata=test)

accuracy <- table(pred, test[,"PURCHASE\_FLG"])

round(accuracy, digits = 0)

sum(diag(accuracy))/sum(accuracy)

prob <- predict(lm\_model, newdata=test, type="response")

pred <- prediction(prob, test$PURCHASE\_FLG)

perf <- performance(pred, measure = "tpr", x.measure = "fpr")

plot(perf)

auc <- performance(pred, measure = "auc")

auc <- auc@y.values[[1]]

auc

library(ROCR)

ROCRperf <- performance(pred, 'tpr','fpr')

plot(ROCRperf, colorize = TRUE, text.adj = c(-0.2,1.7))

library(pROC)

# Compute AUC for predicting Class with the variable CreditHistory.Critical

f1 = roc(Class ~ CreditHistory.Critical, data=train)

plot(f1, col="red")

set.seed(5678)

ind <- sample(2, nrow(Regress\_coupon), replace=TRUE, prob=c(0.8, 0.2))

train1 <- Regress\_coupon[ind==1,]

test1 <- Regress\_coupon[ind==2,]

lm\_model <- glm(PURCHASE\_FLG ~ PRICE\_RATE + CATALOG\_PRICE + DISCOUNT\_PRICE + DISPPERIOD + VISIT + PURCH,

data = train1,

family = binomial(link = "logit"))

summary(lm\_model)

library(pscl)

pR2(lm\_model)

predict <- predict(lm\_model, type = 'response')

table(train$PURCHASE\_FLG, predict > 0.5)

library(ROCR)

ROCRpred <- prediction(predict, train1$PURCHASE\_FLG)

ROCRperf <- performance(ROCRpred, 'tpr','fpr')

plot(ROCRperf, colorize = TRUE, text.adj = c(-0.2,1.7))

set.seed(84569)

ind <- sample(2, nrow(Regress\_coupon), replace=TRUE, prob=c(0.8, 0.2))

train2 <- Regress\_coupon[ind==1,]

test2 <- Regress\_coupon[ind==2,]

lm\_model <- glm(PURCHASE\_FLG ~ PRICE\_RATE + CATALOG\_PRICE + DISCOUNT\_PRICE + DISPPERIOD + VISIT + PURCH,

data = train2,

family = binomial(link = "logit"))

summary(lm\_model)

library(pscl)

pR2(lm\_model)

predict <- predict(lm\_model, type = 'response')

table(train2$PURCHASE\_FLG, predict > 0.5)

library(ROCR)

ROCRpred <- prediction(predict, train2$PURCHASE\_FLG)

ROCRperf <- performance(ROCRpred, 'tpr','fpr')

plot(ROCRperf, colorize = TRUE, text.adj = c(-0.2,1.7))

full\_model <- glm(PURCHASE\_FLG ~ PRICE\_RATE + CATALOG\_PRICE + DISCOUNT\_PRICE + DISPPERIOD + VISIT + PURCH,

data = train,

family = binomial(link = "logit"))

backwards = step(full\_model)

summary(backwards)

set.seed(7155168)

ind <- sample(2, nrow(Regress\_coupon), replace=TRUE, prob=c(0.7, 0.3))

train1 <- Regress\_coupon[ind==1,]

test1 <- Regress\_coupon[ind==2,]

lm\_model1 <- glm(PURCHASE\_FLG ~ PRICE\_RATE + CATALOG\_PRICE + DISCOUNT\_PRICE + DISPPERIOD + VISIT + PURCH,

data = Regress\_coupon,

family = binomial(link = "logit"))

summary(lm\_model1)

library(pscl)

pR2(lm\_model1)

#predict <- predict(lm\_model, type = 'response')

#table(train$PURCHASE\_FLG, predict > 0.5)

pred = predict(lm\_model1, newdata=test1)

accuracy <- table(pred, test1[,"PURCHASE\_FLG"])

round(accuracy, digits = 0)

sum(diag(accuracy))/sum(accuracy)

prob <- predict(lm\_model1, newdata=Regress\_coupon, type="response")

pred <- prediction(prob, Regress\_coupon$PURCHASE\_FLG)

perf <- performance(pred, measure = "tpr", x.measure = "fpr")

plot(perf)

auc <- performance(pred, measure = "auc")

auc <- auc@y.values[[1]]

auc

library(ROCR)

ROCRperf <- performance(pred, 'tpr','fpr')

plot(ROCRperf, colorize = TRUE, text.adj = c(-0.2,1.7))

write.csv(Calculated\_Coupon\_List,"Model\_coupon.csv")

## Coupon sales classification – Webber Zhao

table = pd.read\_csv('Coupon\_for\_Model.csv')

x = pd.get\_dummies(table.drop(['COUPON\_ID\_hash','PURCHASE\_NUMBER','CouponType'],axis=1))

nx = normalize(x)

nx\_area =pd.get\_dummies(normalize(table).drop(['COUPON\_ID\_hash','PURCHASE\_NUMBER','CouponType','en\_large\_area'],axis=1))

nx\_rate = pd.get\_dummies(normalize(table).drop(['COUPON\_ID\_hash','PURCHASE\_NUMBER','CouponType','PRICE\_RATE'],axis=1))

y = table['CouponType'].replace(t2d)nbrsclf = nbrs.KNeighborsClassifier(n\_neighbors=30,p=2,weights='distance')

treeclf = tree.DecisionTreeClassifier(max\_leaf\_nodes=47,class\_weight='balanced')

svmclf = svm.SVC(class\_weight='balanced',probability=True)

logclf = lin.LogisticRegression(class\_weight='balanced')

nbclf = nb.MultinomialNB()

estimators = [nbrsclf,treeclf,svmclf,logclf,nbclf]

Xs = [nx,x,x,nx\_area,nx\_rate]

estimator\_number = len(estimators)

skf = StratifiedKFold(n\_splits=5, shuffle=True, random\_state=123)

y\_actual = np.array([])

y\_predicts = [np.array([])]\*estimator\_number

proba\_predict = [np.zeros((0,len(set(y))))]\*estimator\_number

k\_fold\_list = [tt for tt in skf.split(x,y)]

for fold\_i in range(len(k\_fold\_list)):

y\_actual = np.append(y\_actual,y[k\_fold\_list[fold\_i][1]])

for i in range(estimator\_number):

for train\_index, test\_index in k\_fold\_list:

estimators[i].fit(Xs[i].iloc[train\_index],y[train\_index])

y\_predicts[i] = np.append(y\_predicts[i],estimators[i].predict(Xs[i].iloc[test\_index]))

proba\_predict[i] = np.vstack((proba\_predict[i],estimators[i].predict\_proba(Xs[i].iloc[test\_index])))

hard\_predict = pd.concat([pd.Series(x).replace(d2t) for x in y\_predicts],axis=1).apply(lambda s: s.value\_counts().index[0],axis=1)

proba\_sum = np.zeros((y\_actual.size,3))

for proba in proba\_predict:

proba\_sum += proba

soft\_predict = pd.DataFrame(proba\_sum).apply(lambda s:s.argmax(),axis=1).replace(d2t)

weighted\_proba\_sum = np.zeros((y\_actual.size,3))

precision\_weight\_sum = np.zeros((1,3))

for index in range(estimator\_number):

precision = precision\_score(y\_actual,y\_predicts[index],average=None)

weighted\_proba\_sum += proba\_predict[index] \* precision

precision\_weight\_sum += precision

weighted\_soft\_predict = pd.DataFrame(weighted\_proba\_sum/precision\_weight\_sum).apply(lambda s:s.argmax(),axis=1).replace(d2t)

## Prediction – Tianzi Xu

head(Coupon\_view,3)

#transform categorical variables to dummy variables

area.f = factor(Coupon\_view$en\_large\_area)

dummies = model.matrix(~area.f)

head(dummies,5)

area.category = factor(Coupon\_view$en\_genre)

dummies\_category = model.matrix(~area.category)

head(dummies\_category,5)

olddata=Coupon\_view[,-c(1,17:20)]

Coupon=cbind(olddata,dummies,dummies\_category)

head(Coupon)

#scatterplot for numeric variables

pairs(~.,data = Coupon[,c(2:6,16,17)])

data\_for\_regression=Coupon[,2:38]

regular\_fit=lm(PURCHASE\_NUMBER~.,data=Coupon[,2:38])

summary(regular\_fit)

head(data\_for\_regression,3)

#normalize the data for knn and SVM regression

normalize <- function(x) {

return ((x - min(x)) / (max(x) - min(x)))

}

mm\_Coupon\_large\_area <- as.data.frame(lapply(Coupon[,2:38] , normalize))

head(mm\_Coupon\_large\_area,5)

#split training and testing set

set.seed(1234)

ind <- sample(2, nrow(mm\_Coupon\_large\_area), replace=TRUE, prob=c(0.8, 0.2))

train.mm\_Coupon\_large <- mm\_Coupon\_large\_area[ind==1,]

test.mm\_Coupon\_large <- mm\_Coupon\_large\_area[ind==2,]

#KNN regression

library(lattice)

library(ggplot2)

library(caret)

knnfit=knnreg(test.mm\_Coupon\_large[,c(1:15,17:37)],test.mm\_Coupon\_large[,16],k=5)

summary(knnfit)

predictions=predict(knnfit,mm\_Coupon\_large\_area[,c(1:15,17:37)])

rmse=mean((mm\_Coupon\_large\_area$PURCHASE\_NUMBER-predictions)^2)

SSE=sum((mm\_Coupon\_large\_area$PURCHASE\_NUMBER-predictions)^2)

TSS=sum((mm\_Coupon\_large\_area$PURCHASE\_NUMBER-mean(mm\_Coupon\_large\_area$PURCHASE\_NUMBER))^2)

r.squared=1-SSE/TSS

r.squared

#test overfitting of KNN model

knnfit=knnreg(train.mm\_Coupon\_large[,c(1:15,17:37)],train.mm\_Coupon\_large[,16],k=5)

predictions=predict(knnfit,test.mm\_Coupon\_large[,c(1:5,7:54)])

summary(predictions)

SSE=sum((test.mm\_Coupon\_large$PURCHASE\_NUMBER - predictions)^2)

TSS=sum((test.mm\_Coupon\_large$PURCHASE\_NUMBER - mean(test.mm\_Coupon\_large$PURCHASE\_NUMBER))^2)

r.squared=1-SSE/TSS

r.squared

#SVM regression

SVMfit=ksvm(mm\_Coupon\_large\_area$PURCHASE\_NUMBER~.,mm\_Coupon\_large\_area)

summary(SVMfit)

predictions\_svm=predict(SVMfit,mm\_Coupon\_large\_area)

SSE\_svm=sum((mm\_Coupon\_large\_area$PURCHASE\_NUMBER-predictions\_svm)^2)

TSS=sum((mm\_Coupon\_large\_area$PURCHASE\_NUMBER-mean(mm\_Coupon\_large\_area$PURCHASE\_NUMBER))^2)

r.squared\_svm=1-SSE\_svm/TSS

r.squared\_svm

#stepwise regression

library(MASS)

min.model = lm(PURCHASE\_NUMBER ~ 1,data=data\_for\_regression)

biggest = formula(lm(PURCHASE\_NUMBER~.,data=data\_for\_regression))

fwd.model = step(min.model, direction='forward', scope=biggest)

fullmodel=lm(PURCHASE\_NUMBER~., data = data\_for\_regression)

backward.model=step(fullmodel, direction = "backward" )

step\_model= step(fullmodel, direction="both")

fit\_backward=lm(PURCHASE\_NUMBER ~ PRICE\_RATE + DISCOUNT\_PRICE + DISPPERIOD + USABLE\_DATE\_MON + USABLE\_DATE\_SAT + USABLE\_DATE\_SUN + VIEW\_NUMBER + area.fHokushinetsu + area.fKansai + area.fKanto + `area.fKyushu-Okinawa` + area.fShikoku + area.categoryFood + `area.categoryGift card` + `area.categoryHealth and medical` + `area.categoryHotel and Japanese hotel` + area.categoryLeisure + area.categoryLesson, data=data\_for\_regression)

summary(fit\_backward)

#regression tree model

library(rpart)

regression\_tree=rpart(PURCHASE\_NUMBER~.,method = "anova",data=data\_for\_regression)

printcp(regression\_tree)#display the results

plotcp(regression\_tree)#visualize cross-validation results

summary(regression\_tree)

par(mfrow=c(1,2))#two plots on one page

rsq.rpart(regression\_tree)

plot(regression\_tree, uniform=TRUE,main="regression tree for large area")

text(regression\_tree,use.n = TRUE, all=TRUE, cex=.6)

post(regression\_tree, file = "c:/Users/txu15/downloads/tree2.ps", title = "Regression Tree for large area ")

prune\_regression\_treefit=prune(regression\_tree,cp=0.001)

plot(prune\_regression\_treefit,unifom=TRUE,main="pruned regression tree for large area")

text(prune\_regression\_treefit,use.n = TRUE, all=TRUE,cex=.6)

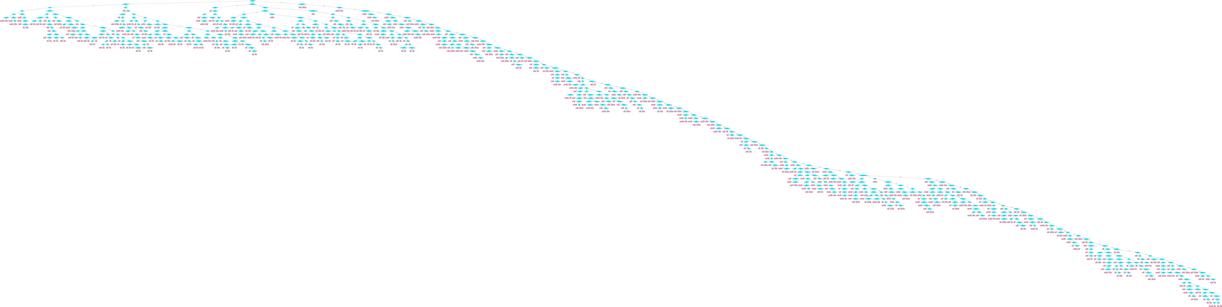
printcp(prune\_regression\_treefit)

plotcp(prune\_regression\_treefit)

summary(prune\_regression\_treefit)

par(mfrow=c(1,2))

rsq.rpart(prune\_regression\_treefit)



**Figure 105 C5.o plot**