國立交通大學

數據科學概論期末報告

機器學習與深度學習之應用

—以電子商務資料為例

第 25 組

組員：張誌安、郭玟慧、

王肇揚、林志遠

指導教授：盧鴻興 教授

中華民國一零九年一月

1. Introduction

In this semester, professor has taught how to use regression, classification and text mining to predict the variable. So, in this report, we use two data sets to predict customer decision of recommendation and customers' yearly Amount Spent by these three methods, both data set are related to E-commerce, and our report will be divided into two parts, chapter 1-3 are according the first dataset, and chapter 4 is according the second dataset.

1-1. Feature Variables

In the first section, we use the first dataset to do category prediction and make the prediction by words, there has 10 feature variables in this data set, including：

- **Clothing ID**: Integer Categorical variable that refers to the specific piece being reviewed.
- **Age**: Positive Integer variable of the reviewers age.
- **Title**: String variable for the title of the review.
- **Review Text**: String variable for the review body.
- **Rating**: Positive Ordinal Integer variable for the product score granted by the customer from 1 Worst, to 5 Best.
- **Recommended IND**: Binary variable stating where the customer recommends the product where 1 is recommended, 0 is not recommended, **this variable is also what we want to predict.**
- **Positive Feedback Count**: Positive Integer documenting the number of other customers who found this review positive.
- **Division Name**: Categorical name of the product high level division.
- **Department Name**: Categorical name of the product department name.
- **Class Name**: Categorical name of the product class name.

1-2. Data Visualization

Next, we visualize some data to understand its distribution faster and easier.

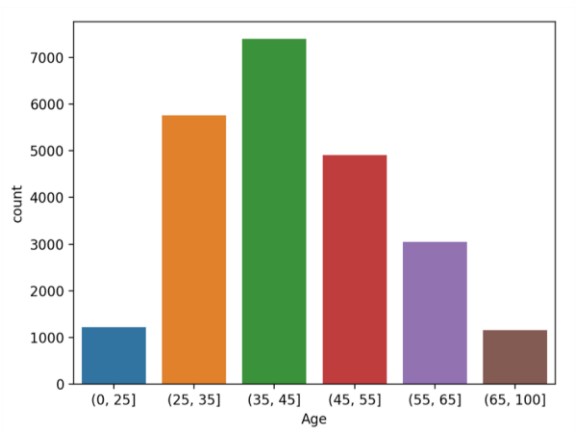


Figure 1.

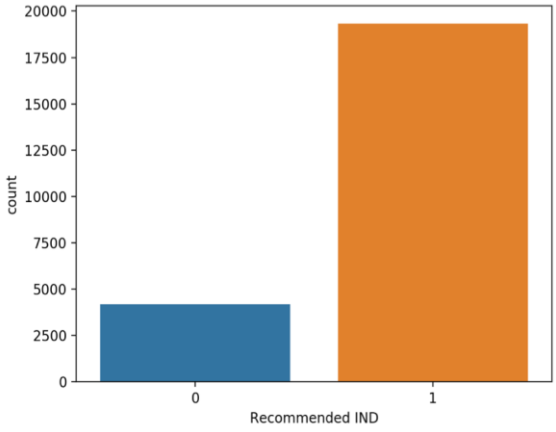


Figure 2.

The age distribution of the reviewers is shown in figure 1, most reviewers' age are fall in the range of 35 to 44 years old, suggesting that the core market segment for this clothing brand is women between 34 and 45. And overall, we also calculated the percentage of recommended comments, which is about to 0.82, the number of recommended comments (shown as figure 2) is much higher than non-recommended comments.

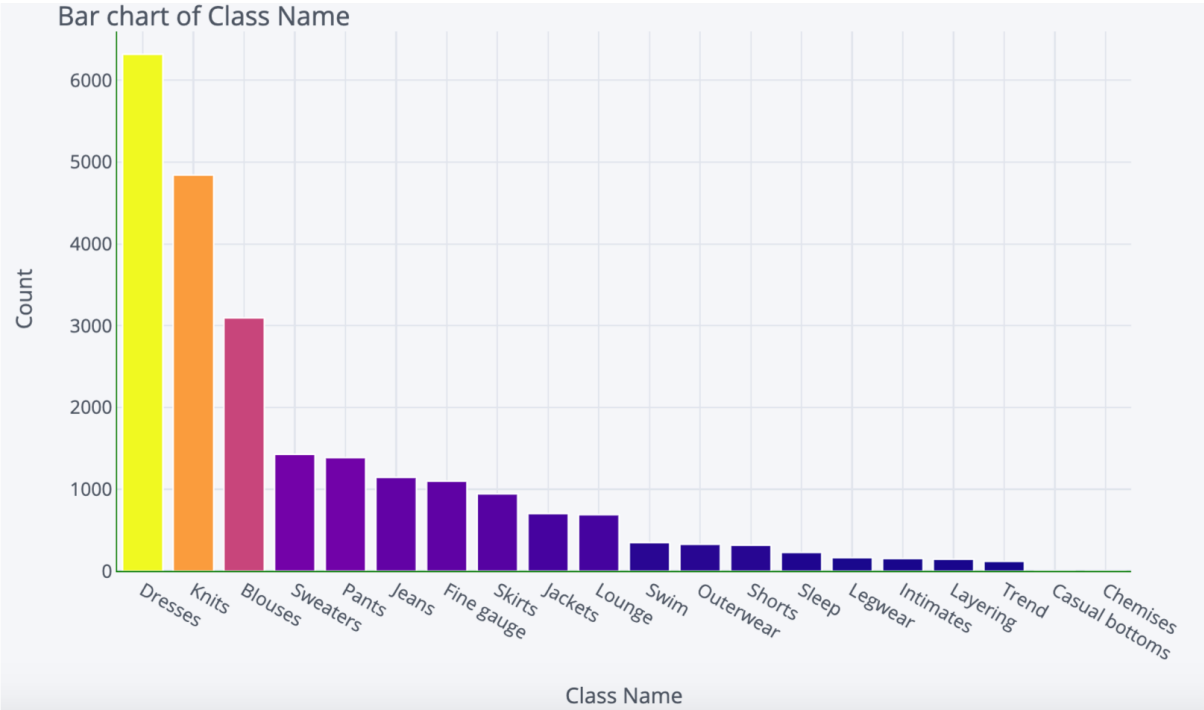


Figure 3.

In figure 3, we can find that dress have the highest number of reviews across all product classes, knits, blouses and sweater also the popular classes in customers' reviews. However, chemises is the last one in all classes.

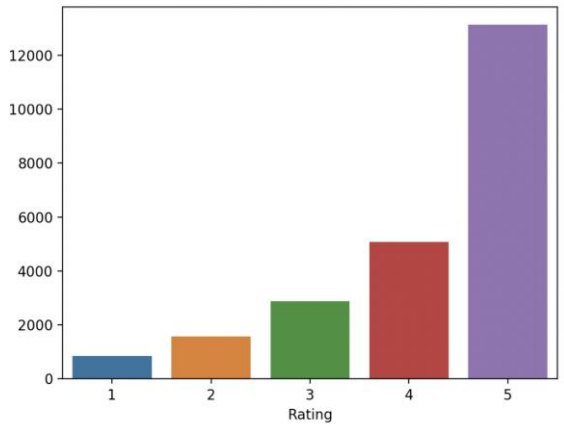


Figure 4.

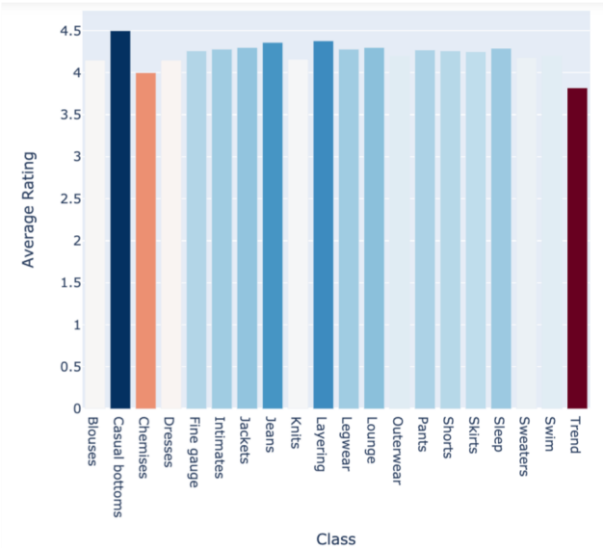


Figure 5.

In figure 4, it seems that most of the ratings are positive, and the average rating between the classes looks closely. Now, let's check the rate of recommendations of every class of the products (figure 5), we can find that the ratio of recommendation is relatively large in all classes.

2. Classification Models-1

2-1. Data Preprocessing

As for the data preprocessing, we remove the text columns (‘Title’ & ‘Review Text’) in this section, because and we will do text mining in chapter 3.

	Clothing ID	Age	Rating	Recommended IND	Positive Feedback Count	Division Name	Department Name	Class Name
9444	72	25	5	1	0	NaN	NaN	NaN
13767	492	23	5	1	1	NaN	NaN	NaN
13768	492	49	5	1	0	NaN	NaN	NaN
13787	492	48	5	1	0	NaN	NaN	NaN
16216	152	36	5	1	0	NaN	NaN	NaN
16221	152	37	5	1	0	NaN	NaN	NaN
16223	152	39	5	1	0	NaN	NaN	NaN
18626	184	34	5	1	5	NaN	NaN	NaN
18671	184	54	5	1	0	NaN	NaN	NaN
20088	772	50	5	1	0	NaN	NaN	NaN
21532	665	43	5	1	0	NaN	NaN	NaN
22997	136	47	5	1	1	NaN	NaN	NaN
23006	136	33	5	1	0	NaN	NaN	NaN
23011	136	36	5	1	0	NaN	NaN	NaN

Figure 6.

According to figure 6, we find that in the last 3 columns, there each has 14 missing values, and all are in the same rows, so we remove these 14 rows of the data.

Next, we also convert the columns ‘Clothing ID’ , 'Rating’ , 'Division Name’ , 'Department Name’ , 'Class Name’ into dummy variables.

However, we split 80% of the data as training set, and 20% of the data as testing set.

2-2. Building Classification

In this section, we build six models to predict customer decision of recommendation, the results are presented as below:

Naïve Bayes:

```
Accuracy: 0.9116080937167199
              precision    recall  f1-score   support

           0       0.77      0.73      0.75       845
           1       0.94      0.95      0.95      3850

    accuracy                           0.91      4695
   macro avg       0.85      0.84      0.85      4695
weighted avg       0.91      0.91      0.91      4695
```

Decision Tree:

```
Accuracy: 0.9173588924387647
              precision    recall  f1-score   support

           0       0.73      0.85      0.79       845
           1       0.97      0.93      0.95      3850

    accuracy                           0.92      4695
   macro avg       0.85      0.89      0.87      4695
weighted avg       0.92      0.92      0.92      4695
```

SVM:

```
Accuracy: 0.9271565495207668
              precision    recall  f1-score   support

           0       0.74      0.91      0.82       845
           1       0.98      0.93      0.95      3850

    accuracy                           0.93      4695
   macro avg       0.86      0.92      0.89      4695
weighted avg       0.94      0.93      0.93      4695
```

Logistic Regression:

```
Accuracy: 0.9116080937167199
              precision    recall  f1-score   support

           0       0.77      0.73      0.75       845
           1       0.94      0.95      0.95      3850

    accuracy                           0.91      4695
   macro avg       0.85      0.84      0.85      4695
weighted avg       0.91      0.91      0.91      4695
```

Random Forest:

```
Accuracy: 0.9233226837060703
              precision    recall  f1-score   support

           0       0.77      0.82      0.79       845
           1       0.96      0.95      0.95      3850

    accuracy                           0.92      4695
   macro avg       0.86      0.88      0.87      4695
weighted avg       0.93      0.92      0.92      4695
```

Neural Network:

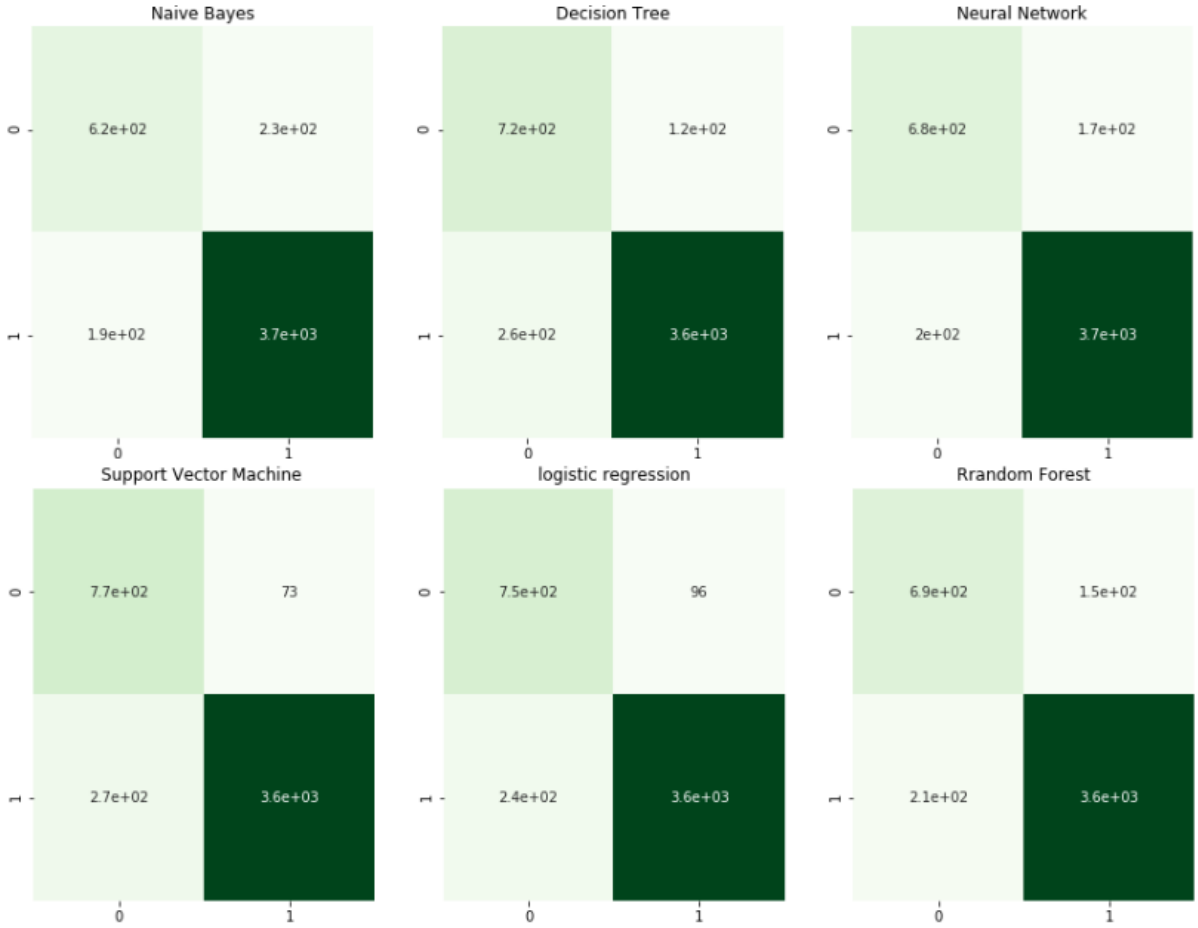
```
Accuracy: 0.9218317358892438
              precision    recall  f1-score   support

           0       0.77      0.80      0.79       845
           1       0.96      0.95      0.95      3850

    accuracy                           0.92      4695
   macro avg       0.86      0.87      0.87      4695
weighted avg       0.92      0.92      0.92      4695
```

2-3. Model Evaluation

The models we had are naïve bayes, decision tree, SVM, logistic regression, random forest and neural network. In this section, we will use several methods to evaluate our models and find the best one.

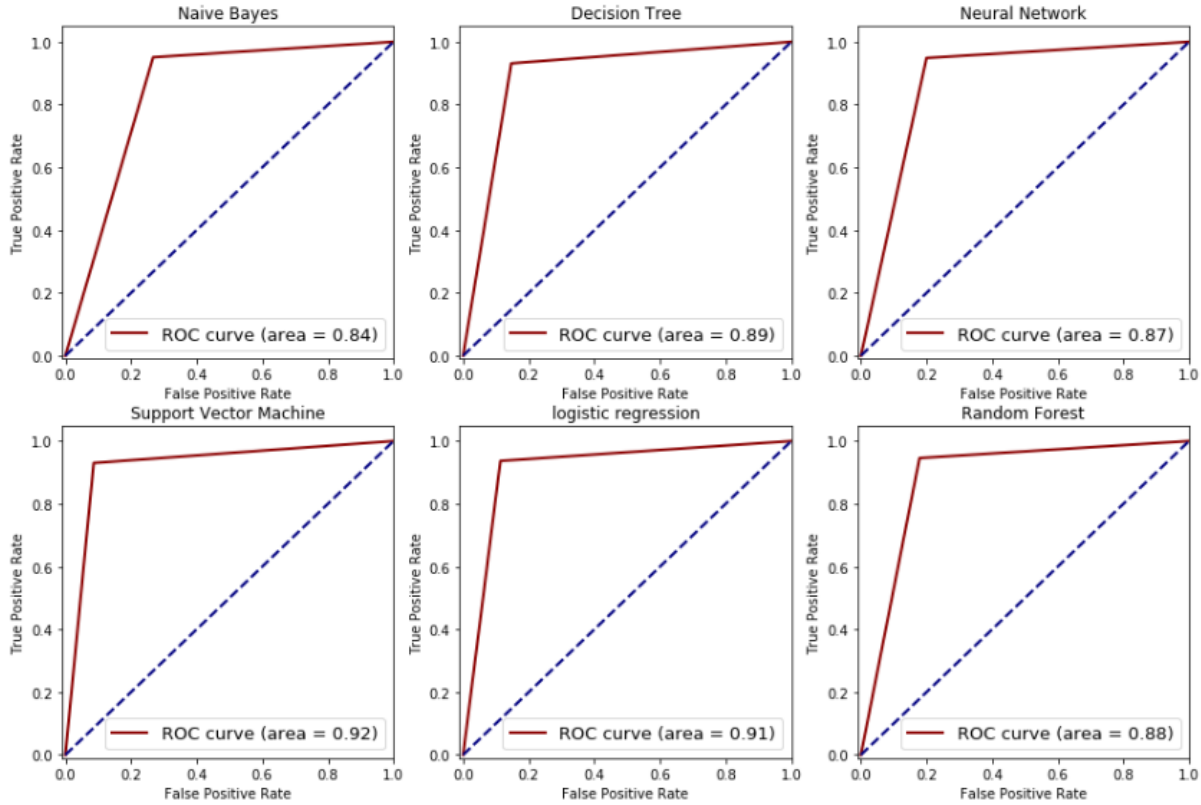
Model	accuracy
Logistic Regression	0.927583
Support Vector Machines	0.927157
Neural Network	0.921832
Random Forest	0.921832
Decision Tree	0.917359
Naive Bayes	0.911608

We evaluate models by accuracy first. All of them are up to 90%. Logistic regression and SVM model are the highest two and they are almost the same.



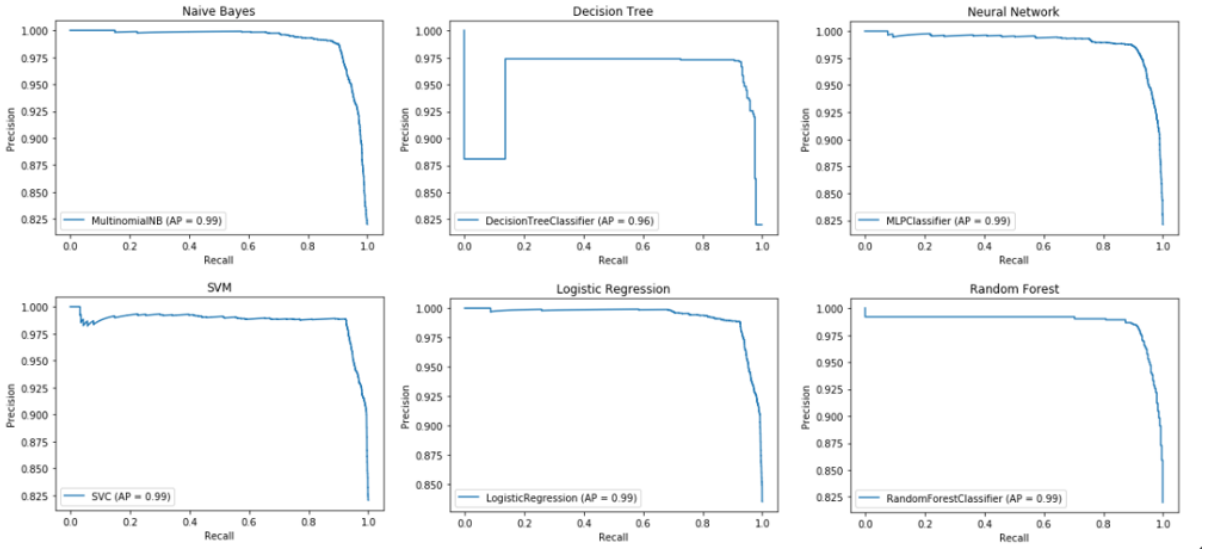
Then we evaluate models by confusion matrix. The vertical axis is predicted label and the horizontal axis is true label. The diagonal cells are the number that we predicted correctly and the other cells are we predicted incorrectly. As an E-commerce clothing company, we hope not to predict unrecommended as recommended since we just can not find out the problem which

led to unrecommended by the customer. Therefore, we hope the number in upper right cell is as small as possible. In this six models, SVM is the smallest one.



The next method is ROC, which is a graph composed of true positive rate and false positive rate. When the red line is closer to the upper left corner, which means the model is better. By the way, we can also calculate RUC, which is the area under the red line. The bigger of RUC, the better of the model. In this method, SVM is still the best model.

However, our data is not balanced, we have almost 80% recommended but only have 20% unrecommended. This may lead to “accuracy paradox” that ROC can not detect. For example, we can simply predict all of them are recommended and we will have approximately 80% accuracy without using any machine learning algorithm. We can not do it in business world. So we use PRC to evaluate whether our data has this problem or not.



The vertical axis is presion and the horizontal axis is recall. If the blue line is smooth enough, which means the model is not suffered from the imbalance problem. The result shows that we do not need to worry about the accuracy paradox.

Form the perspective of accuracy, confusion matrix, ROC and PRC, SVM model is obviously our best classifier to help us predict the customer will recommend or unrecommend.

3. Classification Models-2

3-1. Data Preprocessing

Next, we try using the words data of previous selected data set, which is customer reviews, to predict the customer decision of recommendation. In this section, we'll use four machine learning or deep learning models to train these words data, and see whether the fitted models could do the label classification more precisely or not.

We import the required packages for future uses at the beginning.

```
import pandas as pd
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import AdaBoostClassifier
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import precision_recall_fscore_support
from xgboost import XGBClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import classification_report
from keras.models import Sequential
from keras.preprocessing import sequence
from keras.preprocessing.text import Tokenizer
from keras.layers import Dense, Dropout, Activation, Flatten, TimeDistributed, RepeatVector
from keras.layers.normalization import BatchNormalization
from keras.layers.recurrent import SimpleRNN
from keras.layers.embeddings import Embedding
from keras.optimizers import Adam
from keras.callbacks import EarlyStopping, ModelCheckpoint
from keras.layers.recurrent import LSTM
import matplotlib.pyplot as plt
%matplotlib inline
```

Choose the first 10000 words to be used on training and testing.

```
#取前10000筆資料的文字作為字庫
bag_of_words = data['Review Text'][:10000]
rec = data['Recommended IND'][:10000]
```

Split the data into training set (75%) and testing set (25%).

```
X_train, X_test, y_train, y_test = train_test_split(bag_of_words, rec , test_size = 0.25)
X_train
```

Before training the model. We use the tf-idf algorithm to do the data preprocessing. The first thing is to remove the stop words. Stop words are words like “and”, “the”, “him”, which are presumed to be uninformative in representing the content of a text, and which may be removed to avoid them being construed as signal for prediction. Sometimes, however, similar words are useful for prediction, such as in classifying writing style or personality. In this case, the customer decision of recommendation is not based on these conjunction words, article words, and so on. So we would remove the words by a list of “general English stop words” to decrease the bias of influence by them. After calculating the weight of each words and transforming them into vectors. We could now implement the model training.

3-2. Building Classification

The first two models are both categories of ensemble learning techniques, which are adaboost (adaptive boosting) and xgboost (extreme gradient boosting). For adaboost, in every iteration, it increases the weight of sample whose predicted value was not correct, then put these weighted data to the next classifier in order to make the model more suitable. And Xgboost use gradient boosting trees to fit the model, it’s now the one of the most popular model for data scientists. After training without tuning any parameters, we found that the accuracy of these

two models are both about 86%. However, in these two models, we could see that the recall(*.48* & *.25*, respectively) and precision(*.65* & *.72*, respectively) value for "Not recommended" label are not as good as the accuracy value. In our opinion, the inbalanced data may cause the consequence. We could remove some positive sample to balance the ration between label "0" and "1" in the future to deal with the problem.

```
ada_predict = ada_model.predict(X_test)
#print(ada_predict)
print(classification_report(y_test, ada_predict, labels =[0,1]))
```

	precision	recall	f1-score	support
0	0.65	0.48	0.55	446
1	0.89	0.94	0.92	2054
accuracy			0.86	2500
macro avg	0.77	0.71	0.74	2500
weighted avg	0.85	0.86	0.85	2500

```
xgbc_predict = xgbc.predict(X_test)
print(classification_report(y_test, xgbc_predict, labels =[0,1]))
```

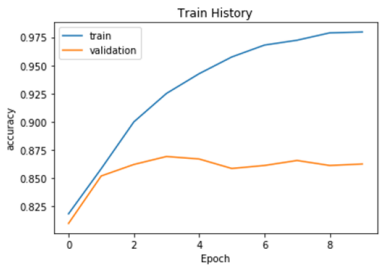
	precision	recall	f1-score	support
0	0.72	0.25	0.37	446
1	0.86	0.98	0.91	2054
accuracy			0.85	2500
macro avg	0.79	0.61	0.64	2500
weighted avg	0.83	0.85	0.82	2500

In the last two models, we use the "keras" module used in Python. The one is called RNN (Recurrent neural network), and the other is LSTM (Long short-term memory). Basically RNN has a similar structure of general neural networks, but the difference is that its nodes not only connect to the next layer of neurons, but also make in conjunction with themselves. Also, LSTM is a more sophisticated model of RNN. Instead of using tf-idf, we use nltk package to remove stop words first, and use keras tokenizer function to transform the words into vectors. After training without tuning any parameters, we found that the accuracy of these two models are about 86% and 88%. In this series, the LSTM has better accuracy than other three models.

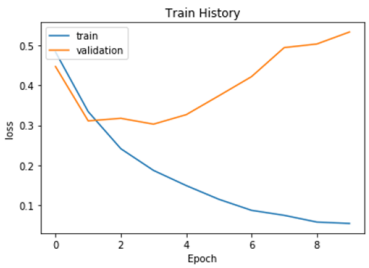
RNN

Test model accuracy: 86%

Accuracy



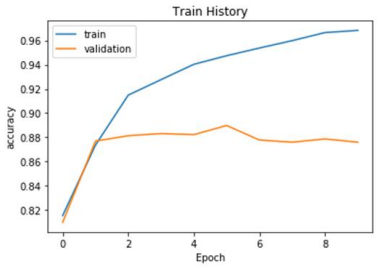
Loss



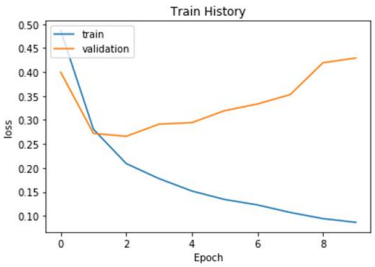
LSTM

Test model accuracy: 88%

Accuracy



Loss



4. Multiple Regression

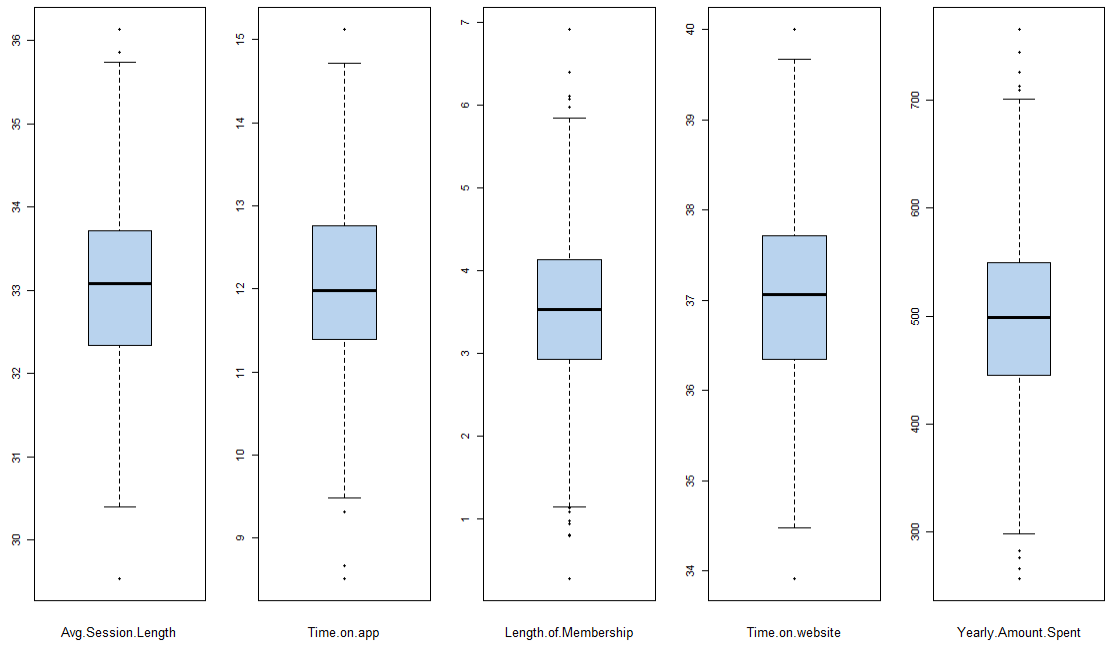
4-1. Feature Variables

Next, we used the second data set for the multiple regression. The data set is from E-commerce. There are eight variables in the data set and no missing values.

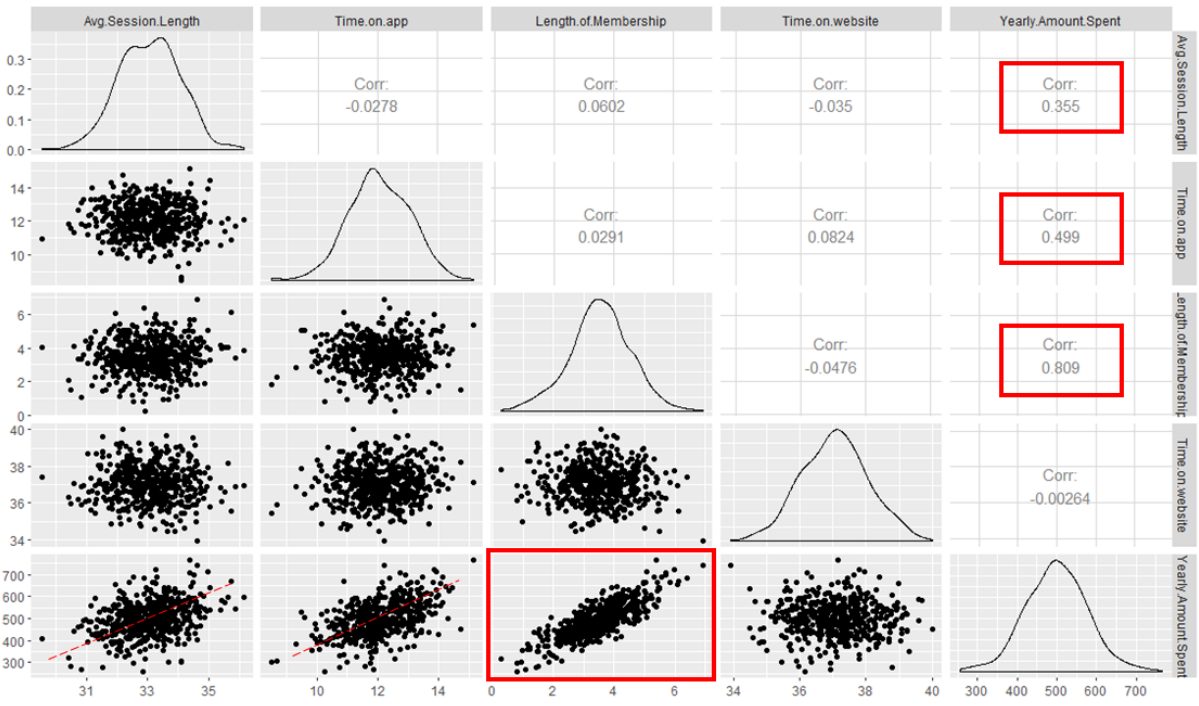
- Email: Customers' register email.
- Address: Customers' register address.
- Avatar: Average time spent on Website in minutes.
- Avg. Session Length: Average session of in-store style advice sessions.
- Time on App: Average time spent on App in minutes.
- Time on Website: Average time spent on Website in minutes.
- Length of Membership: How many years the customer has been a member.
- Yearly Amount Spent: Average consumer annual spending per session.

4-2. Data Preprocessing

We take only numeric variables for multiple regression which are "Avg. Session length", "Time on app", "Length of Membership", "Time on website", and "Yearly amount spent" and "Yearly amount "spent" is the dependent variable. Before running multiple regression, we do some simple data preprocessing. First, we run the "boxplot" to check out whether there are any outliers among each variable. According to boxplot, we can find that all five variables have outliers, so we directly delete them.



Next, we run the correlation coefficient and scatter plot to check the relation between variables. There is a positive correlation between "Yearly amount spent" and the three variables, "Avg. Session length", "Time on app", and "Length of Membership". "Length of "Membership" has the greatest positive correlation strength, while "Time on website" has a low negative correlation. In addition, you can see from the scatter plot that "Length of Membership" and "Yearly amount spent" have a potential linear relationship.

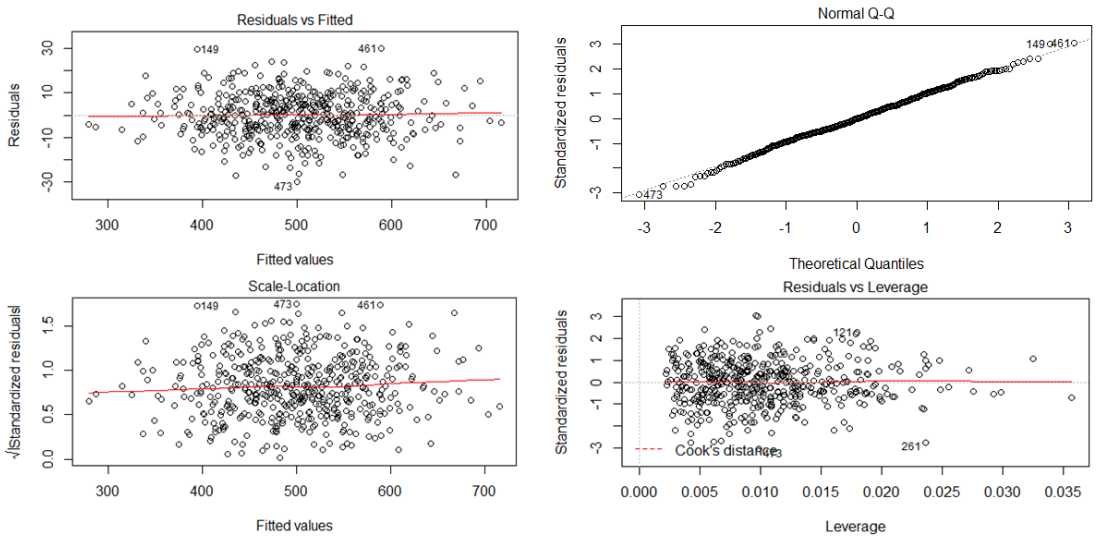


4-3. Multiple Regression

We perform multiple regression and the result. We can see that except for “Time on website”, other three variables are all significant, and the adjusted R-square of this model is 0.98, which has a very high variation explanation.

```
Call:
lm(formula = Yearly.Amount.Spent ~ Length.of.Membership + Time.on.app +
    Avg.Session.Length + Time.on.website, data = customer)

Residuals:
     Min       1Q   Median       3Q      Max
-30.3412  -6.3347  -0.0531   6.5942  30.0455

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)
(Intercept)         -1048.6313    23.4822 -44.656   <2e-16 ***
Length.of.Membership    61.5911     0.4997 123.254   <2e-16 ***
Time.on.app             38.8188     0.4673  83.072   <2e-16 ***
Avg.Session.Length      25.7087     0.4615  55.703   <2e-16 ***
Time.on.website          0.3453     0.4551   0.759    0.448
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 10.01 on 481 degrees of freedom
Multiple R-squared:  0.9813,    Adjusted R-squared:  0.9811
F-statistic:  6300 on 4 and 481 DF,  p-value: < 2.2e-16
```

Next, we analyze the residuals of regression. The upper left is the residual variation plot. We can see that the variation distribution is average spread and are not on the horizontal line. The lower left is the standardized residual plot, which is similar to the upper distribution. For the "normal plot", it can be inferred that it conforms to the normal distribution. The graph in the lower right corner shows that there are no influence data in the model.



Final, we check out whether the model meets the hypotheses. We run "normality test", use "shapiro-wilk normality test", p-value is greater than 0.05, so this model meets the normality assumption; "Independent test", use the "Durbin-Watson", p-value is greater than 0.05, this

model meets the independence assumption. Finally, in the "homogenous test", the p-value is greater than 0.05, so the model also meets the homogeneity hypothesis. All these three results confirmed that the multiple regression follow the assumption.

```
        Shapiro-Wilk normality test

data:  model$residuals
W = 0.99773, p-value = 0.7599
```

```
 lag Autocorrelation D-W Statistic p-value
   1      0.04198641      1.909291   0.346
 Alternative hypothesis: rho != 0
```

```
Non-constant Variance Score Test
Variance formula: ~ fitted.values
Chisquare = 1.027904, Df = 1, p = 0.31065
```