Predicting Purchase Behavior from User Browsing History in Online Shopping Website

Chengzhang Quan, Lei Yi

1 Introduction

Online shopping has been a rapid-growing business for years. According to the survey made by UPS and analytics firm comScore Inc., the American shoppers made 51% of their purchases on the web in the year 2016. In the era of e-commerce, it is of core importance to understand user behavior in online shopping websites. Predicting user preference based on user's profile and purchase history can enable companies to effectively recommend items to target buyers, and therefore increase their revenue significantly.

In this project, we try to predict user's actual purchase behavior based on their browsing history. The data comes from 2015 recsys challenge, and is provided by YOOCHOOSE. It contains a collection of sequences of click events from many click sessions, together with whether a session results in a purchase event and detail of purchase (price and quantity). The data was collected during several months in the year of 2014, reflecting the clicks and purchases performed by the users of an on-line retailer in Europe. To protect end users privacy, as well as the retailer, all numbers have been modified. The goal of the challenge was to predict whether a user is going to buy something or not, and if he is buying, what would be the items he is going to buy.

2 Dataset

2.1 Data Description

The dataset can be downloaded at

`http://2015.recsyschallenge.com/challenge.html`

The data set contains two files, yoochoose-clicks.dat and yoochoose-buys.dat.

`yoochoose-clicks.dat` contains 9249729 sessions with a total of 33003944 click histories, each with the following categories:

- **session_id**: Each log-in has a unique session id. In one session there are one or many clicks. Session ID is represented as an integer.
- **timestamp**: the time when the click occurred. Format of YYYY-MM-DDThh:mm:ss.SSSZ

- **item_id**: the unique identifier of the item that has been clicked. Item ID is represented as an integer.
- **category**: The category of an item. The value "S" indicates a special offer, "0" indicates a missing value, a number between 1 to 12 indicates a real category identifier, any other number indicates a brand.

`yoochoose-buys.dat` contains 509696 sessions with a total of 1150753 buy histories, each with the following categories:

- **session_id**: It is the same as the session ID in click data if the purchase event is in the same session as the the click event.
- **timestamp**: the time when the buy occurred. Format of YYYY-MM-DDThh:mm:ss.SSSZ
- **item_id**: the unique identifier of item that has been bought.
- **price**: the price of the item. Each price is an integer (rescaled to conceal the actual price).
- **quantity**: the quantity in this buying. Quantity is represented as an integer.

2.2 Data Preprocessing

2.2.1 Deleting Irrelevant Information

Due to large size of the data set (1.5 Gb just for click data), we want to reduce the data size by deleting irrelevant information and summarize high dimensional data.

First of all, the time stamp contains many irrelevant information. All the browsing and purchase events were made in the year 2014, so year information is of no practical value in this dataset. The precise timestamp includes second and milisecond, which is also of little important in our analysis. Therefore, we discard year, second and milisecond in timestamp, and keep month, day, hour, minute information. This is done by the function `reformat_buy` and `reformat_click` in `reformat_data.py`. The resulted files are `yoochoose-clicks-simplified.dat` and `yoochoose-buys-simplified.dat`.

Next, since click and buy data are stored in separate files, we would like to combine them into a single file. We do this by matching the session id, and item id from the click file and buy file. In the mean time, we also discard time information for the buy data, since they are not of practical importance to our task: to predict whether users will make a purchase. This is achieved by the function `combine_click_buy` in `reformat_data.py`. The resulted data file is `yoochoose-combined-simplified.dat`. This file is approximately 1Gb in size (which is about 2/3 of the original files combined).

2.2.2 Creating an Item Catalogue

There are no item catalog in the data. However, there is information about items contained in click and buy event. For example, we can tell item price, browsing frequency, ratio of buy/click events, etc. Therefore, an item catalog is built to include these information for future use. `create_item_catalogue` in `create_item_catalogue.py` achieve this task. The item catalogue is stored in the file `item_catalogue.dat`. Note however that we have no information about an item's price if it has not been purchased at all.

2.2.3 Extracting Key Features

To predict session-level purchase event, we note that the session-level data is not homogeneous in size. Some contain more click histories, some contain fewer. In order to fit a model with fixed number of inputs, we decide to extract some key features from sessions, which we believe are useful in predicting whether the session results in a purchase. The feature we extract are listed in the following table (standard features already given are not listed):

feature	**description**
day_of_week	the day of week of the session, displayed as integers from 0 to 6, where 0 means Sunday and 6 means Saturday
period_of_month	whether the session is in the first third of the month, middle third, or the last third, displayed as integers 1,2 and 3
no_click	total number of click events in this session
duration	approximate duration of the session, measured by last click time minus first click time, measured in minutes
ave_time	mean time spent on an item, measured by duration divided by no_clicks
longest_time	the longest time spent on a single page, measured in minutes
no_item	total number of items viewed in the session
return_check	whether the user viewed an item more than once in the same session
ave_buy_rate	average buy/purchase ratio of all items viewed
promotion	whether an item is in promotion, we can tell from item category
buy	whether the session ends with a purchase

Similarly, to predict whether a user will purchase a specific item, we create a table containing the features we extract relevant to the item viewed in a session. Features which are the same to session-level features will not be listed again:

feature	description
no_click_item	number of clicks of the particular item in a session
prop_click	proportions of total number of clicks
no_click_prior	number of clicks prior to viewing the item
no_click_after	number of clicks after viewing the item
time_item	time spent on the item
prop_time	proportion of time spent on the item
time_prior	time spent prior to viewing the item
time_after	time spent after viewing the item
buy_rate	buy/view rate of the item
promotion_item	whether the item is in promotion
buy_item	whether the item is purchsed in the session

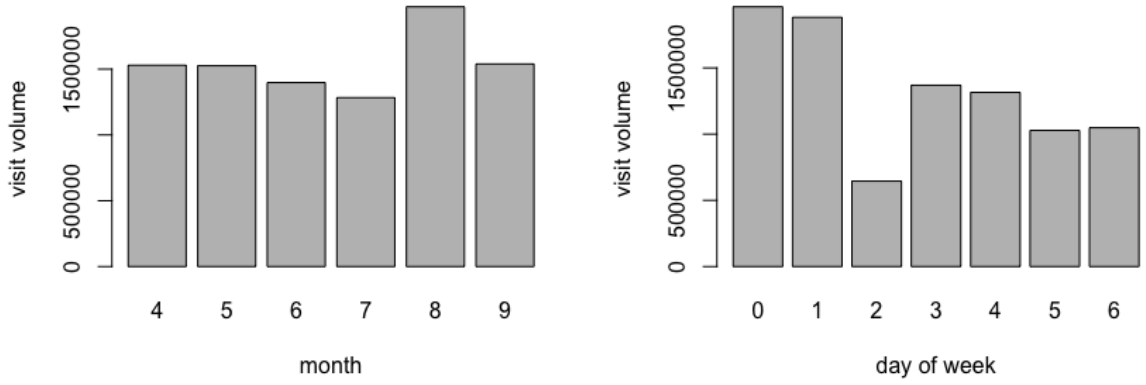
Using `extract_feature_session` and `extract_feature_item` in `feature.py`, we create data `session.dat` and `item.dat` as input for `R`. Note that `session.dat` is 350 Mb and `item.dat` is 30Mb in size.

2.3 Data Visualization

There are many interesting results we can obtain without doing detailed analysis.

2.3.1 click-month

We can visualize the visit volume in each month. There is gentle decrease from May to July, possible due to warmer weather and thus more offline shopping. There is a sudden increase in August. We propose that this is due to pre-school shopping wave.

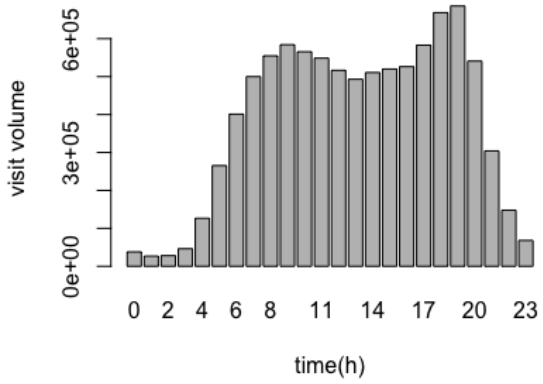


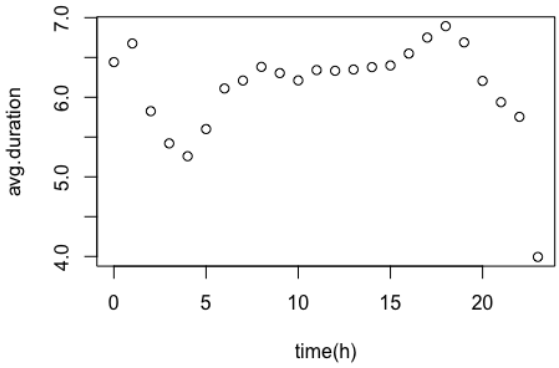
2.3.2 click-day of month

There is huge difference in visiting volume on each day of a week. The highest volume is on Sunday and Monday, while the lowest volume is on Tuesday.

2.3.3 click - time of day

The visit volume vs time of day agrees with our intuition. There are more volume during day time and less during nighttime. The peak occurs at around 6pm.



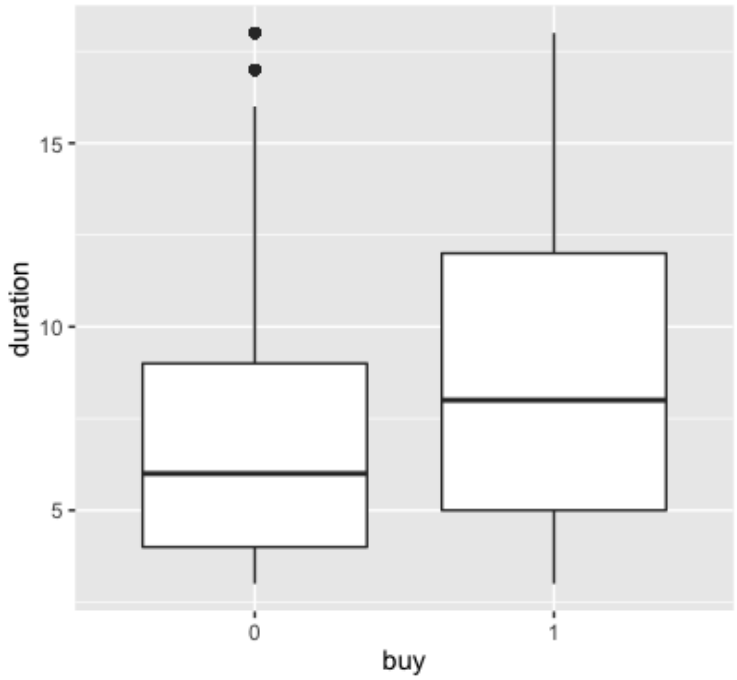


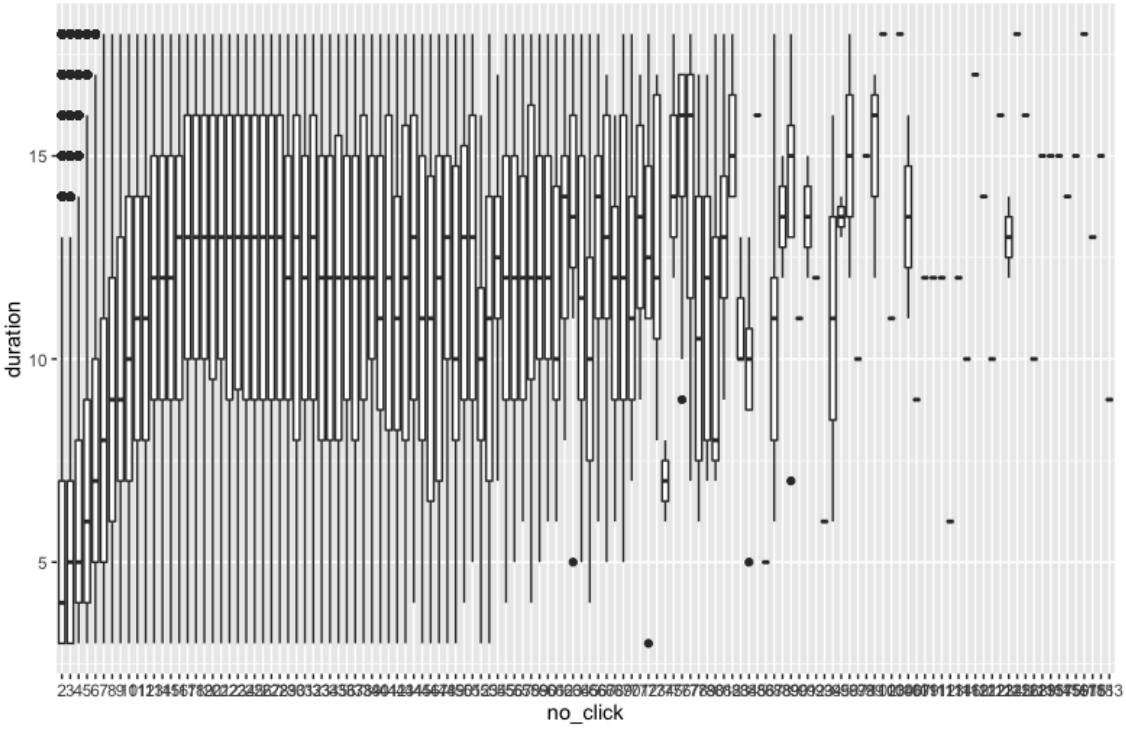
2.3.4 duration - time of day

We calculate the average duration based on different time of a day. From the figure, we can see that during evening time (17-19), people spend more time on viewing items. Compared with this, the average duration is lower at midnight.

2.3.5 duration - buy

We present a box plot of buy-vs-duration where we exclude most outliers. The shape of these two box satisfy our expectation that on average people spend more time on viewing items if they finally make a purchase.


15
10
5
duration
0
1
buy


15
10
5
duration
no_click

2.3.6 duration - no of clicks

Finally, we want to see whether duration of a session is related to number of clicks. The figure shows the box plot of duration of session corresponding to different clicks. Here we also delete most of outliers. It clearly shows that duration increases when number of clicks goes higher when click when the number of clicks is less than 12 clicks. When a session has more clicks than 15 times, duration will hold in the range of 10 to 20 minutes. However, it still shows that duration is correlated with clicking number.

3 Model

The two tasks (determine whether there is a purchase in a session, and determine whether a specific item is bought) have similar features, so we use the same method to tackle them. From now on we will focus on the first task, and explain any additional detail for the second task later.

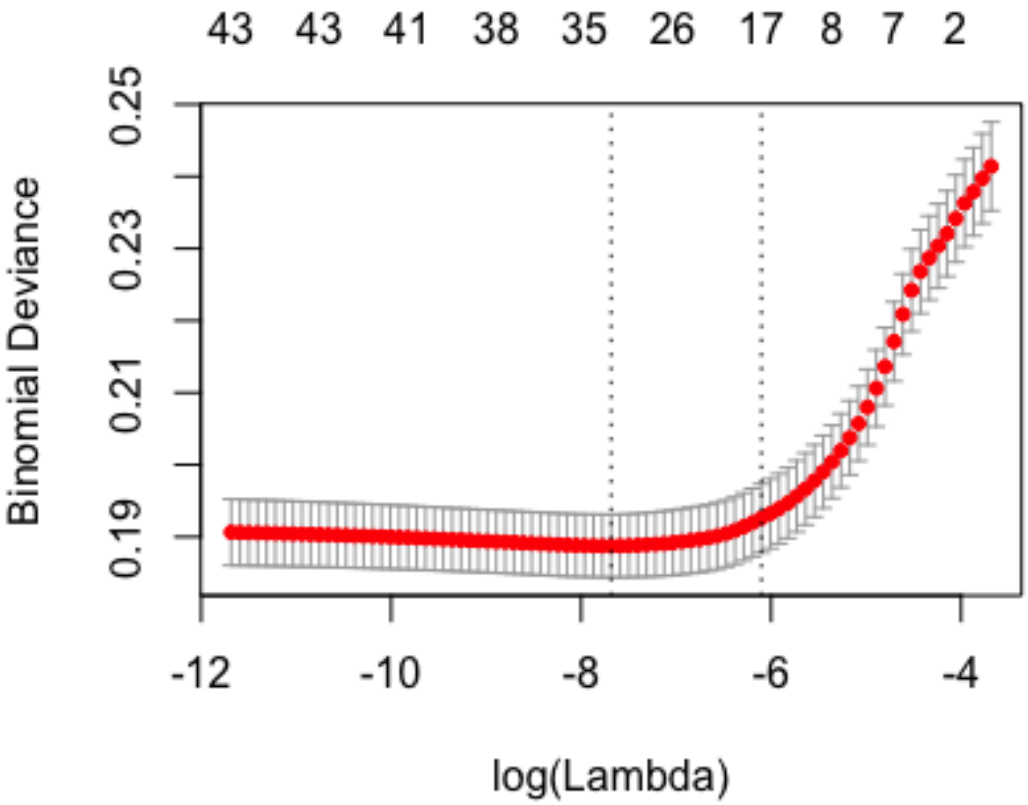
We have a total of 9249728 sessions in our dataset `session.dat`, out of which 234295 end with a purchase. Even though we are able to load it into R, it is impossible and unhelpful to use the full dataset. Therefore, we randomly subsampled 10000 sessions to do our analysis.

3.1 Logistic Regression

The extracted features, together with the original features are too much to build a concise model for prediction. Therefore, the first step is to select a subset of variables using LASSO regression. We use 10-fold cross validation to determine the optimal value of λ in order to minimize classification error. The following plot shows the relationship between misclassification error and value of λ. Observe that the minimum misclassification error is achieved when $\log \lambda$ is approximately -7, with number of variables around 35. In order to minimize number of variables used, we choose λ to be lambda1se, which corresponds to the right dashed line. Using this λ we get a series of variables useful in prediction. The variables are listed below:

```
                    [,1]
(Intercept)     -4.174777765
month5          -2.310844833
month6          -2.111814694
month7          -1.991739744
month9           0.279310174
day_of_week1    -0.200992188
day_of_week6     0.023332730
time4            0.363110375
time7            0.054938959
time9            0.367149577
time12          -0.117495448
time15           0.519123786
time19          -0.046817418
duration         0.007569061
period_of_month3 0.034808083
```

```
no_click            0.047235503
return_check1       1.274180737
promotion1          0.038803881
```



Once we obtain the variables, we then feed them into a logistic regression to find the final model. However, from the results of logistic regression, many variables are insignificant at 0.05 level. A backward selection method is then used to reduce the variable set further. In fact, only time and period of month is deleted from the model. Below is the logistic regression output using the reduced variable set:

```
Call:
glm(formula = buy ~ month + day_of_week + duration + no_click +
    return_check + promotion, family = binomial, data = data_session_subset)

Deviance Residuals:
```

```
     Min          1Q      Median          3Q         Max
-3.01369   -0.21682   -0.14289   -0.00002    3.01170

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)    -3.909097   0.200453 -19.501  < 2e-16 ***
month5        -19.340309 669.923205  -0.029  0.97697
month6        -18.629209 699.903133  -0.027  0.97877
month7        -18.810736 753.125244  -0.025  0.98007
month8         -0.489681   0.232383  -2.107  0.03510 *
month9         -0.105196   0.224148  -0.469  0.63884
day_of_week1   -0.617111   0.217391  -2.839  0.00453 **
day_of_week2   -0.652155   0.319621  -2.040  0.04131 *
day_of_week3   -0.228970   0.221130  -1.035  0.30046
day_of_week4   -0.080915   0.213270  -0.379  0.70439
day_of_week5   -0.237341   0.233278  -1.017  0.30895
day_of_week6    0.097082   0.218074   0.445  0.65619
duration        0.012182   0.004432   2.749  0.00599 **
no_click        0.051676   0.012355   4.183 2.88e-05 ***
return_check1   1.449734   0.152805   9.487  < 2e-16 ***
promotion1      0.385143   0.200290   1.923  0.05449 .
---
Signif. codes:  0 ?***? 0.001 ?**? 0.01 ?*? 0.05 ?.? 0.1 ? ? 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 2418.2  on 9999  degrees of freedom
Residual deviance: 1839.8  on 9984  degrees of freedom
AIC: 1871.8

Number of Fisher Scoring iterations: 20
```

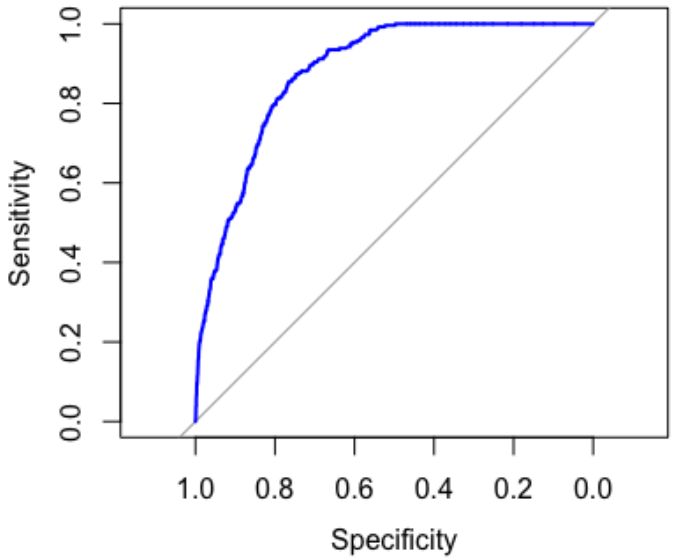
Even though some indicators are not significant alone, the variable as a whole is significant. This can be shown from Anova test:

```
Analysis of Deviance Table (Type II tests)

Response: buy
             LR Chisq Df Pr(>Chisq)
month          322.99  5  < 2.2e-16 ***
day_of_week     14.62  6    0.023399 *
duration         6.86  1    0.008825 **
no_click        18.58  1   1.626e-05 ***
```

```
return_check      98.66   1  < 2.2e-16 ***
promotion          3.92   1    0.047677 *
---
Signif. codes:  0 ?***? 0.001 ?**? 0.01 ?*? 0.05 ?.? 0.1 ? ? 1
```

From the output above, we can see that month, day of week, session duration, number of clicks, whether there is a return check, and whether there is item in promotion are all important predictors to whether a session will end up with a purchase. Compared to Sunday, all weekdays lead to a smaller probability of making a purchase. This is reasonable as people tend to be busy during weekdays and have fewer time making a decision of purchase. It is interesting to see that Saturday actually leads to an increasing probability, but it is not significant. The duration of a session is a good indicator of a potential purchase. The more time people spend, the more likely that they will end up buying something. Similarly, the more clicks a person makes, the more likely he will purchase something, probably because he wants to compare different items to finalize one. Return check is an extremely good indicator of purchase behavior. If someone views a page the second time, he might have compared it with other items and prefer the item he sees twice. In this case, the person has a high chance of actually buying the item. Last but not least, promotion items can increase the probability of purchase.



The ROC curve is shown above. The area under the curve is 0.8806, which is reasonably good for a binomial classifier.

Finally, we use the model to test another randomly selected 10000 sessions. We use threshold of 0.1, in order to have more session predicted as 1. The confusion matrix is given below:

	no buy	buy
predict no buy	9181	168
predict buy	543	108

The misclassification rate is 0.07. Note that we predict more than a third of the sessions with purchase correctly, without creating too many false positives.
In the second task we are constrained to the subset of session where there is actually a purchase event. There are 1071422 item-session pair in all sessions with a purchase. Out of them, 494398 items are bought. We want to predict which item(s) are likely to be bought by the user. Again a logistic regression model is set up, with detail omitted, below is the summary of logistic regression.

```
Call:
glm(formula = buy_item ~ no_click + longest_time + return_check +
    no_click_item + time_item + prop_time + time_prior + promotion_item,
    family = binomial, data = data_item_subset)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-6.2371  -0.8482  -0.4262   0.9739   3.5027

Coefficients:
                 Estimate Std. Error z value Pr(>|z|)
(Intercept)     -1.253973   0.075499 -16.609  < 2e-16 ***
no_click        -0.032544   0.002732 -11.913  < 2e-16 ***
longest_time    -0.021575   0.003041  -7.095 1.29e-12 ***
return_check1   -0.573957   0.060276  -9.522  < 2e-16 ***
no_click_item    1.163842   0.040874  28.474  < 2e-16 ***
time_item        0.032644   0.005230   6.242 4.33e-10 ***
prop_time        1.501394   0.113829  13.190  < 2e-16 ***
time_prior      -0.007198   0.002242  -3.211  0.00132 **
promotion_item1  0.777359   0.048881  15.903  < 2e-16 ***
---
Signif. codes:  0  ***  0.001  **  0.01  *  0.05  .  0.1  1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 13798  on 9999  degrees of freedom
Residual deviance: 10774  on 9991  degrees of freedom
AIC: 10792

Number of Fisher Scoring iterations: 5
```

Apart from similar variables to the first task, we observe that when predicting whether an item will be bought, the number of clicks on the specific item is important. Similarly, the time spent on

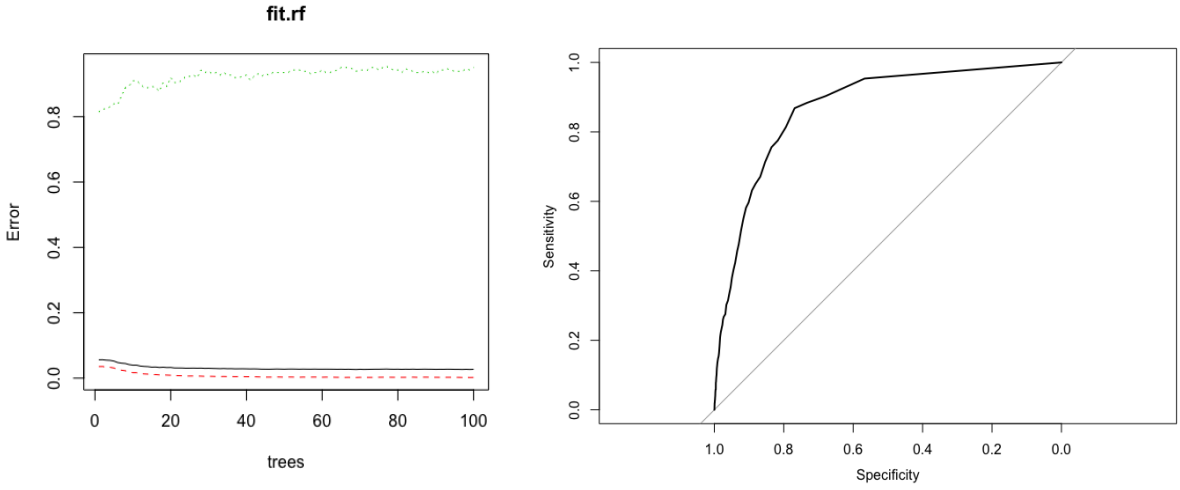
the item and proportion of total time are also significant. It is interesting to note that the more time spent before viewing an item, the less likely the item will be bought. Whether an item is in promotion is again a very good indicator of whether an item will be bought. We also observe that day of week is no longer significant in predicting whether a particular item will be bought in a session. This is probably because day of week is related to the desire for shopping, but probably does not relate to what kind of item is bought. Another interesting observation is that we create item catalogue and include buy/view ratio in our data. This is a bit of cheating since the event of purchase an item is incorporated in buy/view ratio. However, the resulted prediction variable does not include buy/view ratio.

Below is the confusion table on test data. We use a threshold of 0.5 this time. The overall accuracy is 0.74.

	no buy	buy
predict no buy	4392	1654
predict buy	1007	2947

3.2 Random Forest Model

We also use Random Forest as an alternative. We use the same training data set and testing data set as above. The parameters for our random forest model are: *mtree=5, ntree = 100*. The fitting plot is shown below. The AUC is 0.8715, slightly worse than logistic regression. The OOB error of RF model is 2.69% and the testing error of RF model is 2.74%.

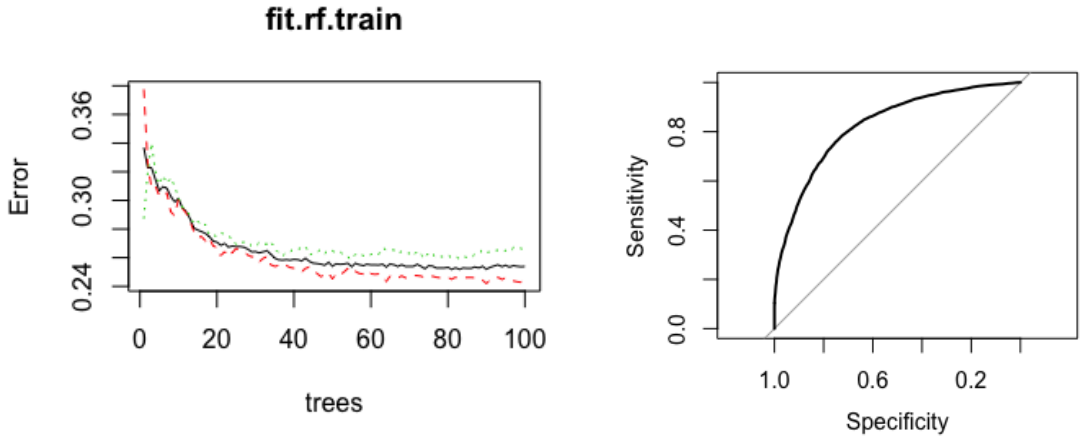


For the sake of comparison, we choose a threshold 0.15 such that predict buy/ buy rate are about the same for the two models. Under this threshold, the confusion matrix for testing is

	no buy	buy
predict no buy	9361	169
predict buy	363	107

Compared with logistic regression, we see that we actually have fewer false positives when the true positives are held equal.

For the second task, a similar procedure is followed, and the error plot and confusion matrix is given below. For this model, we still set parameters *mtree=5, ntree = 100*. The OOB error is 25.5% and the testing error is 24.5%. The AUC is 0.8286. The result is slightly better then Logistics model.



	no buy	buy
predict no buy	4134	1191
predict buy	1265	3410

3.3 Comparison of Method

As mentioned earlier, we compare the two methods by comparing their proportion of false positives, when true positives are kept at the same level (by tuning thresholds). It is clear that in both tasks random forest performs better than logistic regression. We propose the following reasons for the difference:

- RF is not sensitive to outliers, while logistic regression is.
- RF model captures the interaction between variables, while a simple logistic regression does not.

However, RF also has its drawback. There is no clear interpretation of RF coefficients, while we can give a clear and rigorous interpretation of logistic regression coefficients. The validity of setting a threshold other than 1/2 is also of question, even though we use it for the sake of comparison in this study.

Therefore, we still prefer logistic regression as our final model. It gives a clear relationship between our target variable, and all relevant variables, as explained earlier.

4 Conclusion

In this study, we have presented two models to study the factors affecting shopping behavior of online-shopping-website customers. In both models we can predict whether a series of click events will end with a purchase with high probability. We can also predict which items a customer will purchase with high probability. These results can be adapted to online recommendation systems. Based on click data, we can recommend suitable items to potential customers with a high matching rate, so that online business companies can increase their revenue significantly without creating too many spam or inefficient advertisements. There are limitations in our model. For example, we have not used fully the fact that customers with similar click histories may have similar preferences. This fact, if fully analyzed, will enable us to promote items that users never click before to potential customers. With collaborative filtering, this is possible and has been proved very effective. Another limitation is that due to computational constraints, we have not used the full data, which might give better prediction results compared to using only a portion of data.