Predict Client's Churn

Sparkify is a digital music service similar to **Netease Cloud Music** or **QQ Music**. Many of the users stream their favorite songs in Sparkify service everyday, either using free tier that places advertisements in between the songs, or using the premium subscription model where they stream music as free, but pay a monthly flat rate. User can upgrade, downgrade or cancel their service at anytime.

This is a Customer Churn Prediction Problem , there are so many similar projects, such as WSDM - KKBox's Churn Prediction Challenge competition from Kaggle, and a few helpful links are follows:

- Customer Churn Prediction using Machine Learning (How To)
- Prediction of Customer Churn with Machine Learning
- Customer Churn Prediction and Prevention
- Hands-on: Predict Customer Churn

So, our job is data mining the customers' data and implement appropriate model to predict customer churn as follow steps:

- Clean data: fill the nan values , correct the data types, drop the outliers.
- EDA: exploratory data to look features' distributions and correlation with key label (churn).
- Feature engineering: extract and found customer-features and customer-behavior-features; Implement standscaler on numerical features.
- Train and measure models: I choose logistic regression, linear svm classifier, decision tree and random forest classifier to train a baseline model and tuning a better model from best of them. It is worth mentioning that this data is unbalanced because of less churn customers, so we choose `f1 score` as a metrics to measure models' performance.

Quick Facts

A mini subset of size 125 MB of the original 12 GB customer log json data file will be used for creating the prediction model. The small dataset has 286'500 log entries with 18 unique columns.

The schema and info of dataset is given below:

```
root
 |-- artist: string (nullable = true)
 |-- auth: string (nullable = true)
 |-- firstName: string (nullable = true)
 |-- gender: string (nullable = true)
 |-- itemInSession: long (nullable = true)
 |-- lastName: string (nullable = true)
 |-- length: double (nullable = true)
 |-- level: string (nullable = true)
 |-- location: string (nullable = true)
 |-- method: string (nullable = true)
```

```
|-- page: string (nullable = true)
|-- registration: long (nullable = true)
|-- sessionId: long (nullable = true)
|-- song: string (nullable = true)
|-- status: long (nullable = true)
|-- ts: long (nullable = true)
|-- userAgent: string (nullable = true)
|-- userId: string (nullable = true)
```

Column's Name	Description
artist	The artist being listened to
auth	Whether or not the user is logged in
firstName/lastName	Name of the user
gender	Gender of the user
itemInSession	Item number in session
length	Length of time for current row of specific log
level	Free or Paid user
location	Physical location of user, including City and State
method	Get or Put requests
page	Which page are user on in current row
registration	Users registration number
sessionId	Session ID
song	Song currently being played
status	Web status
ts	Timestamp of current row
userAgent	Useragent of post or get in browser of users
userId	User ID

Exploratory Data Analysis

We use the `Cancellation Confirmation` events of `page` column to define the customer churn, and perform some exploratory data analysis to observe the behavior for users who stayed vs users who churned.

- churn

```
df_cleaned_cancel.dropDuplicates(['userId']).select('Churn').groupby('Churn').count().collect()
```

```
[Row(Churn=1, count=52), Row(Churn=0, count=173)]
```

So, there are 52 users have churned events in the dataset, it's about 23.1% churned rate. The rate of churn and not churn is roughly 1:3, so this is an unbalanced dataset.

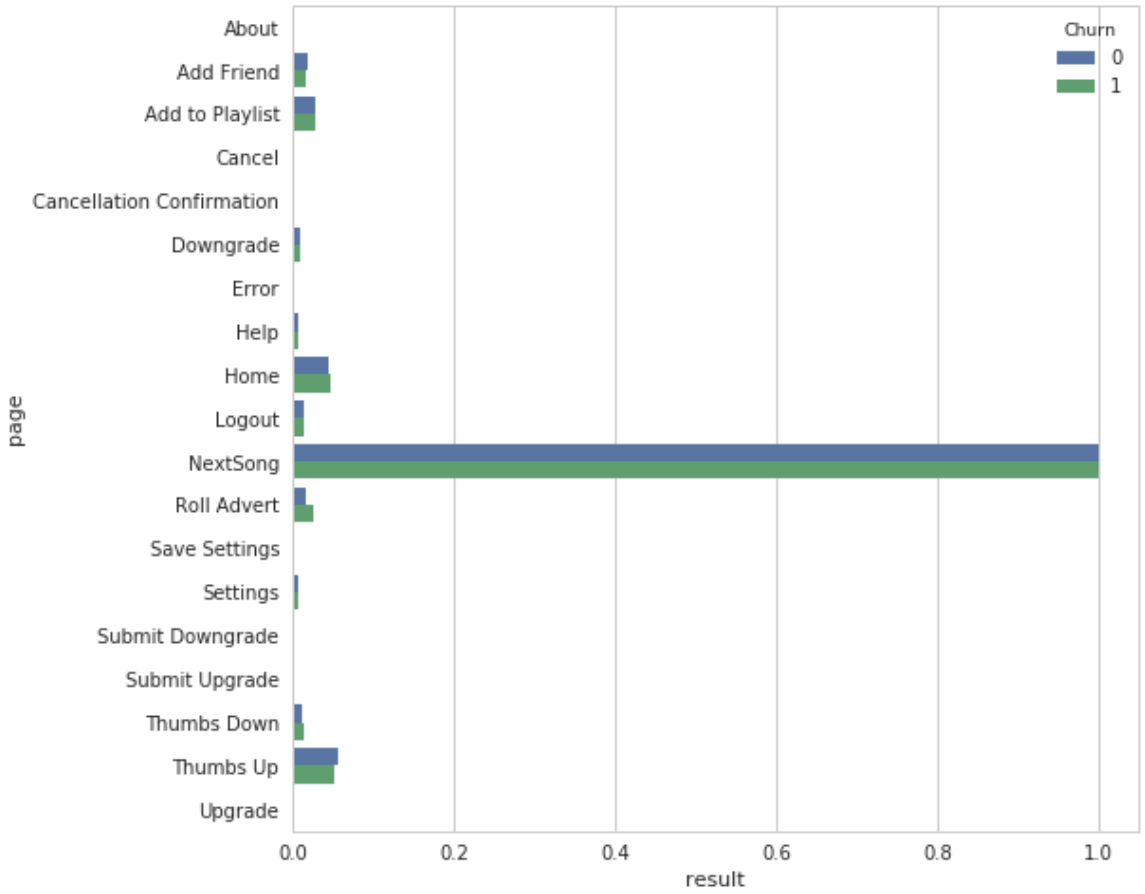
- gender

```
+-----+------+-----+
|Churn|gender|count|
+-----+------+-----+
|    0|     M|   89|
|    0|     F|   84|
|    1|     F|   20|
|    1|     M|   32|
+-----+------+-----+
```

Can we say the gender has effect on Churn or not ? We calculate the p-value and result is 0.20 over 0.05, so, we can't say like that.

- page

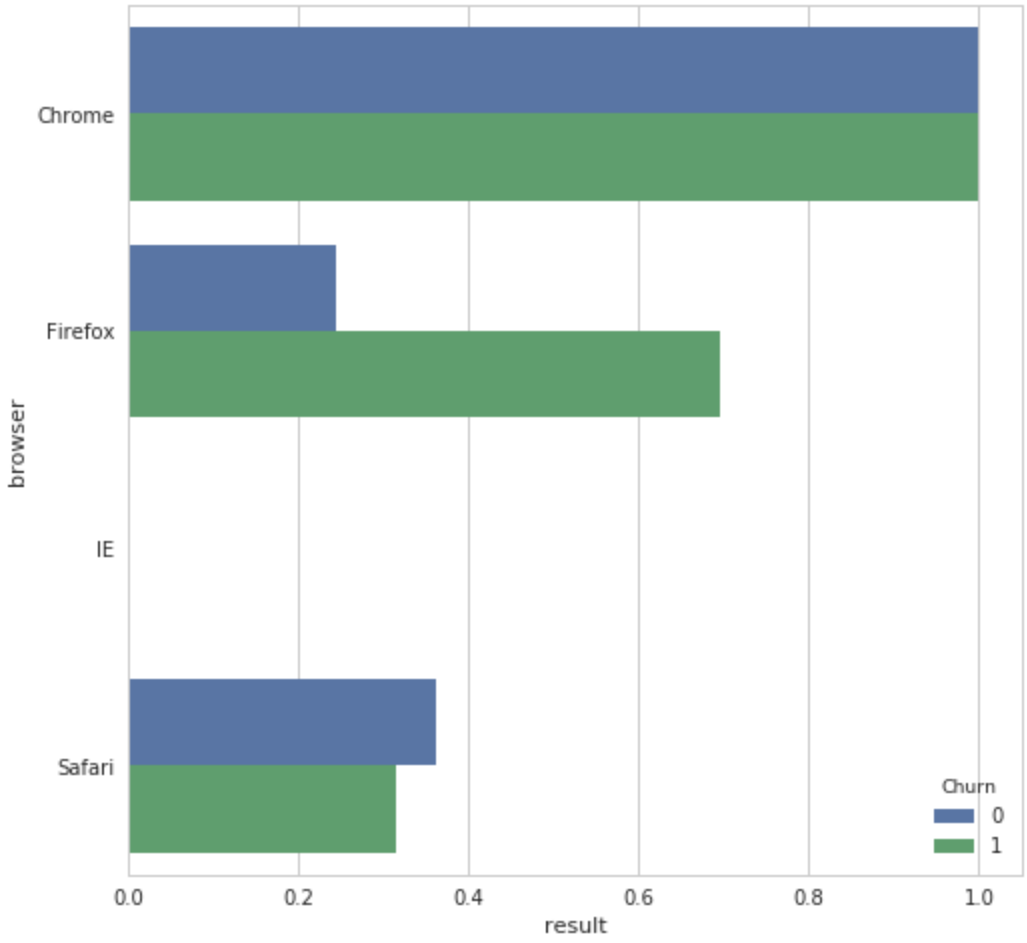
We count each item in `page` column of different group and normalized data.

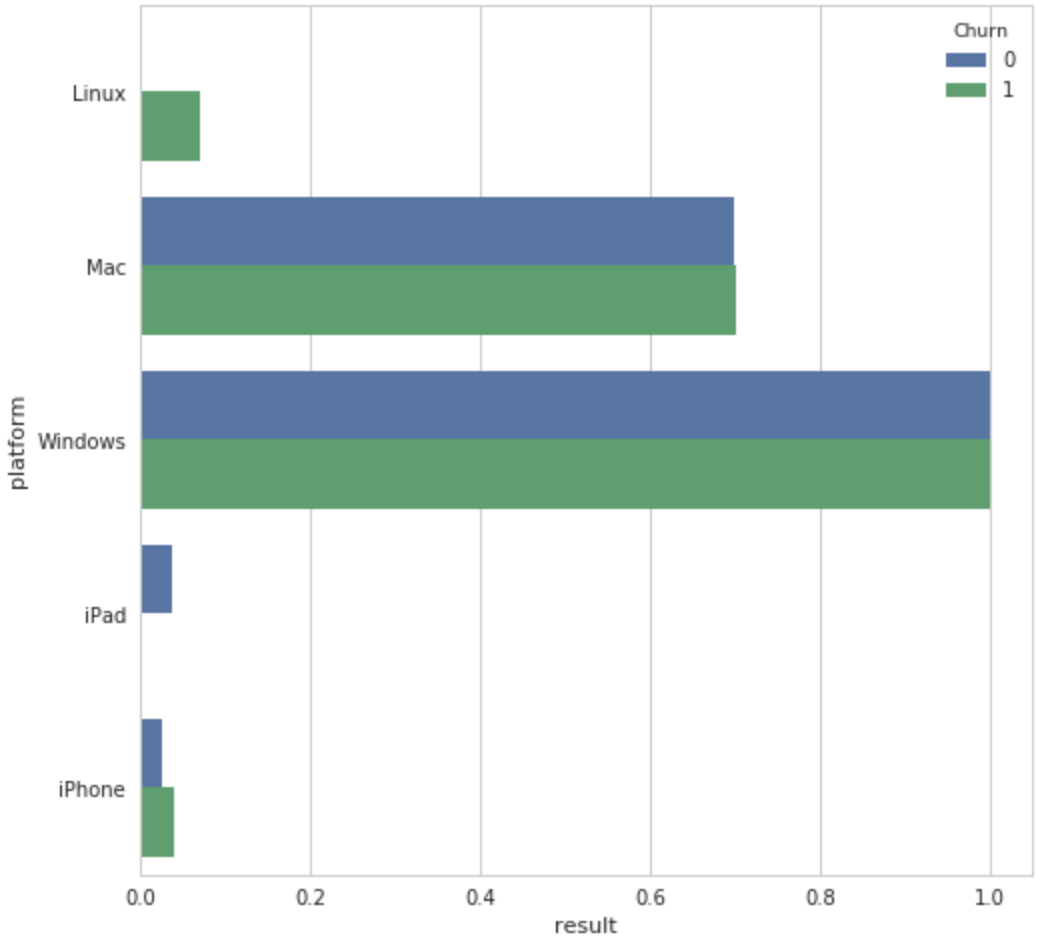


Obviously, NextSong has accounted for most of customers' events. Thumbs Up ,Thumbs Down , Home and Add to Playlist have effect on churn too.

- userAgent

We extract the browser and platform of customers from userAgent column.

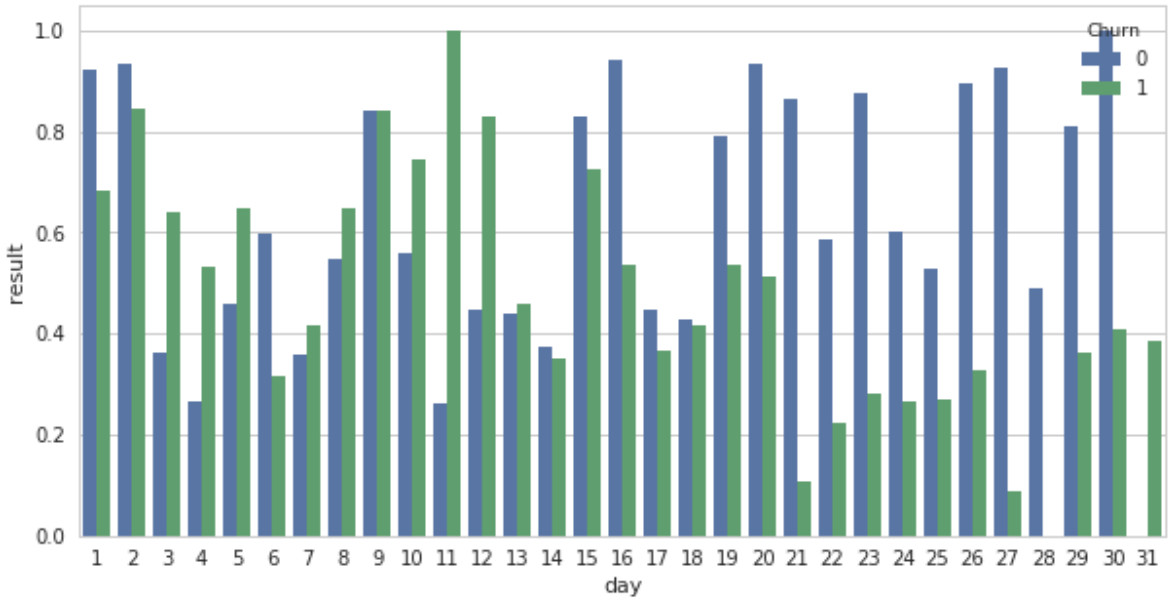

Chrome
Firefox
IE
Safari
browser
0.0
0.2
0.4
0.6
0.8
1.0
result
Churn
0
1

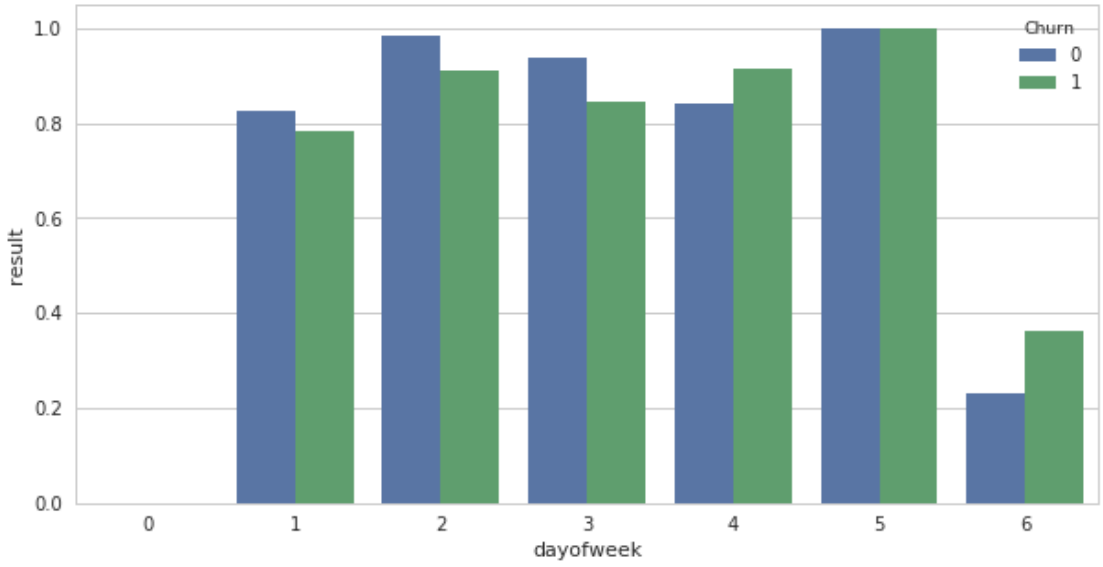


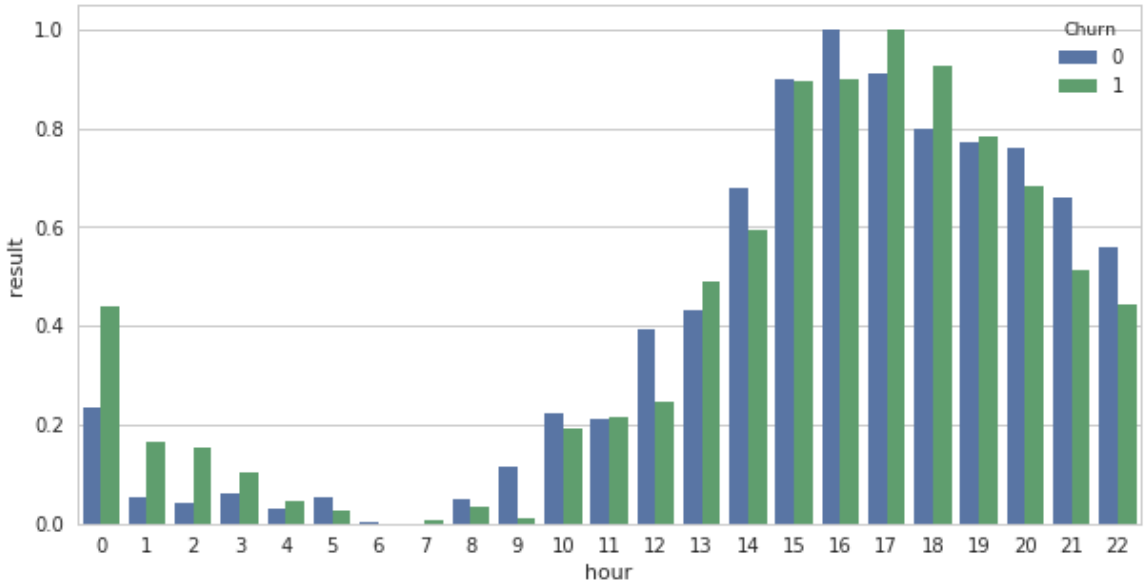
Customers using safari and iPad have more proportion in churn.

- time

We extract day-of-month, day-of-week and hour from `ts` column.


Churn
0
1
result
day
0.0
0.2
0.4
0.6
0.8
1.0
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31


Churn
0
1
result
dayofweek
0.0
0.2
0.4
0.6
0.8
1.0
0 1 2 3 4 5 6



Customers from churn group have more events after 15th in one month, and have less events in weekend.

Feature Engineering

On the basis of the above EDA, we can create features as follows:

- Categorical Features
  - gender
  - level
  - browser
  - platform
- Numerical Features
  - mean,max,min,std of length of users
  - numbers of each item in page (ThumbsUp …
  - number of unique songs and total songs of users
  - number of unique artists of users
  - percentage of operations after 15th in a month
  - percentage of operations in workday

We implement label encoding on categorical features and standard scaler on numerical features.

Modeling

We split the full dataset into train and test sets. Test out the baseline of four machine learning methods: Logistic Regression, Linear SVC, Decision Tree Classifier and Random Forest Classifier.

	LogisticRegression	DecisionTreeClassifier	RandomForestClassifier	LinearSVC
f1_test	0.608586	0.519847	0.468519	0.702076
pred_time	0.144996	0.131740	0.159543	0.092269
train_time	14.586288	5.076892	4.352907	60.177756

Though the LinearSVC spent more training time, but it can get the highest f1 score 0.702. And the LogisticRegression has a medium training time and f1 score, maybe I can tuning it to get a higherscore. So I'll choose LinearSVC and LogisticRegression to tuning in next section, the result is as follows:

- Linear SVC

Training time:

```
start = time.time()
cvModel_SVM = crossval.fit(train)
end = time.time()
print(f'Model tuning is done, spent {end-start}s.')
```

```
Model tuning is done, spent 936.2487051486969s.
```

F-1 Score:

```
pred = cvModel_SVM.transform(test)

print('Accuracy: {}'.format(evaluator.evaluate(pred.select('label','prediction'), {evaluator.metricName: "accuracy"})))
print('F-1 Score:{}'.format(evaluator.evaluate(pred.select('label','prediction'), {evaluator.metricName: "f1"})))
```

```
Accuracy: 0.7037037037037037
F-1 Score:0.7045177045177045
```

- LogisticRegression

Training time:

```
start = time.time()
cvModel_LR = crossval.fit(train)
end = time.time()
print(f'Model tuning is done, spent {end-start}s.')
```

```
Model tuning is done, spent 156.82159566879272s.
```

F-1 Score:

```
pred = cvModel_LR.transform(test)

print('Accuracy: {}'.format(evaluator.evaluate(pred.select('label','prediction'), {evaluator.metricName: "accuracy"})))
print('F-1 Score:{}'.format(evaluator.evaluate(pred.select('label','prediction'), {evaluator.metricName: "f1"})))
```

```
Accuracy: 0.7037037037037037
F-1 Score:0.702075702075702
```

As we can see in above, the logistic regression (0.7021) can get a nearly f1 score as the linear svm classifier(0.7045). But the logistic regression saves 83.3% time spending than the latter, considering this is only a quit mini dataset and our purpose is scaling this up to the total 12G dataset, so, the logistic regression is the best model from now on in this project.

Conclusion

Reflection

In this project I set out to predict customers' churn problem with the dataset of a music streaming service named Sparkify. This is a binary classification problem , so I choose four supervised learning algorithm to found a model. After evaluated and tuning, I find out the logistic regression is the suitable model for this project because of its balanced and high f1-score (0.7021) and time spending.

By the way, I once fell into the trap of data leakage ,so that all of the models can achieve a performance that seems too good (1 for f1-score) to be true. I had to go back to check my feature engineering, and found I put the cancellation confirmation which is the flag of churn in the features, what a awkward thing! And that teach us you must be careful and patient when you are working.

Improvement

There are only about 76 samples in the mini dataset above, so the model could be improved by being trained on a bigger dataset and tuning hyper parameters based on it.

Another improvement could be to try out more features or deep learning models.