THE DATA SCIENCE NANODEGREE CAPSTONE PROJECT

Project Name: Sparkify

Date: September 24, 2019

Author: Itoro Michael Ikon

PROJECT DEFINITION

Project Overview

Sparkify is a fictitious music streaming service similar to actual music streaming services such as Spotify and Pandoras. The goal of this project is to build a machine learning model from the event log data generated by Sparkify, to predict whether a user is likely to churn. Churn is defined in this case as the outright cancellation of the Sparkify service by a subscriber.

The Sparkify service profits by serving ads to subscribers in the free tier of the service, as well as by taking a monthly fee from users the paid tier. Hence, churn prediction is important since users at risk of churning can be identified in advance and offered incentives to remain in the streaming service, thereby increasing the revenue accruing to the Sparkify.

Problem Statement

The problem solved by this project is the detection of users who will cancel their membership in the Sparkify service based on the event log data generated by users of the platform. The detection is performed by a supervised machine learning classifier model trained on the event log data generated by Sparkify. For users at risk of churning, the classifier is expected to output the value 1, while it outputs 0 for those unlikely to churn. Specifically the project will involve the following steps:

1. Use the Spark framework to preprocess the event log data to extract features for each user in the event log.
2. Split the dataset created from the extracted features into train, test and validation sets
3. Train multiple supervised learning models on the train set
4. Select the best model based on the F1 scores on the validation set and report on its performance based on the test set.

Metrics

The evaluation metric for model selection is the F1 score. The F1 score is preferable since the project dataset is imbalanced. The percentage of users who churned in the provided event logs is about 23%, therefore the F1 score is a more appropriate metric than say accuracy, which is more suited to the balanced case. The F1 score is defined as the harmonic mean of precision and recall and the formula is shown below:

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$

while recall is defined as:

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives}$$

Recall directly focuses on the actual positives in the dataset and their correct detection. The actual positives is comprised of the true positives and the false negatives. The churned customers are the actual positives and have a lesser size in the Sparkify dataset. A high recall value will mean that the classifier correctly classifies most of the actual positives, which are the smaller class in the dataset.

Therefore by using F1 as metric and indirectly the recall, the performance of the classifier on the churned users is significantly being taken into account.

ANALYSIS

Data Exploration

The data used for model training is the mini_sparkify_event_data.json file which stores the event logs generated by the Sparkify service. The file size is 125,467kb on file system and contains 286,500 transaction records. A snapshot of the data schema after loading the file on the Pyspark dataframe is shown below.

```
In [5]: df.printSchema()

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)
```

Illustration 1: Input data schema

The dataset schema above shows the field names in the data as well as the their type. There are 18 fields in the dataset A section of the input dataset in tabular form is shown below in illustration 2.

	artist	auth	firstName	gender	itemInSession	lastName	length	level	location	method	page	registration
0	Martha Tilston	Logged In	Colin	M	50	Freeman	277.89016	paid	Bakersfield, CA	PUT	NextSong	1538173362000
1	Five Iron Frenzy	Logged In	Micah	M	79	Long	236.09424	free	Boston-Cambridge-Newton, MA-NH	PUT	NextSong	1538331630000
2	Adam Lambert	Logged In	Colin	M	51	Freeman	282.82730	paid	Bakersfield, CA	PUT	NextSong	1538173362000

Illustration 2: A section of the input dataset in tabular form

There were no null values in the userId and sessionId fields which determine the integrity of a saved event.

Data Visualization
In addition to the illustrations above, more diagrams will be shown which further explore the information contained in the dataset.

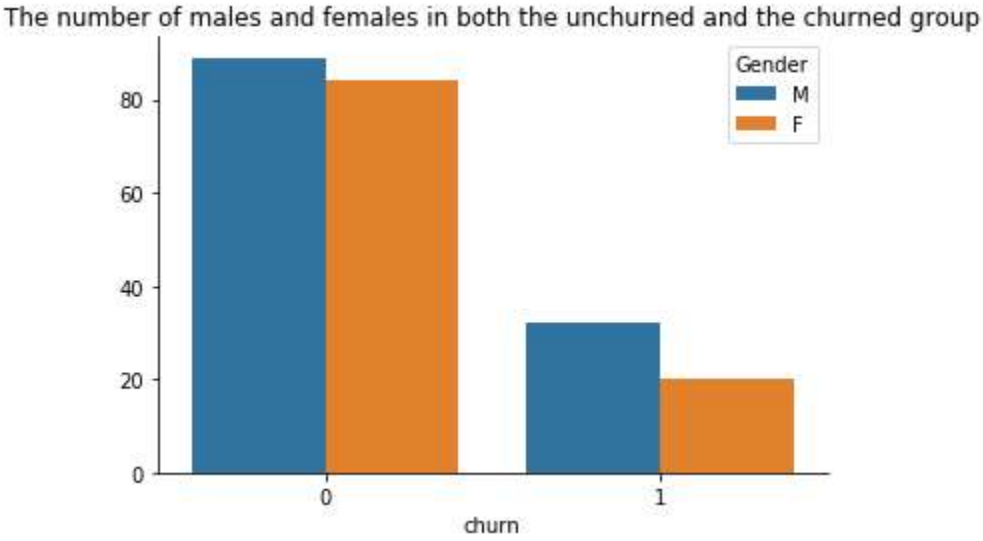

Illustration 3: Gender representation in dataset

The figure above shows the number of males and females in both, the set of users who churned and those who did not. The churned set is denoted by 1, while the retained clients are depicted with 0, in the churn axis. The number of males is consistently above the number of females in both groups.

The average time spent by users on the Sparkify platform is shown below in illustration 4.

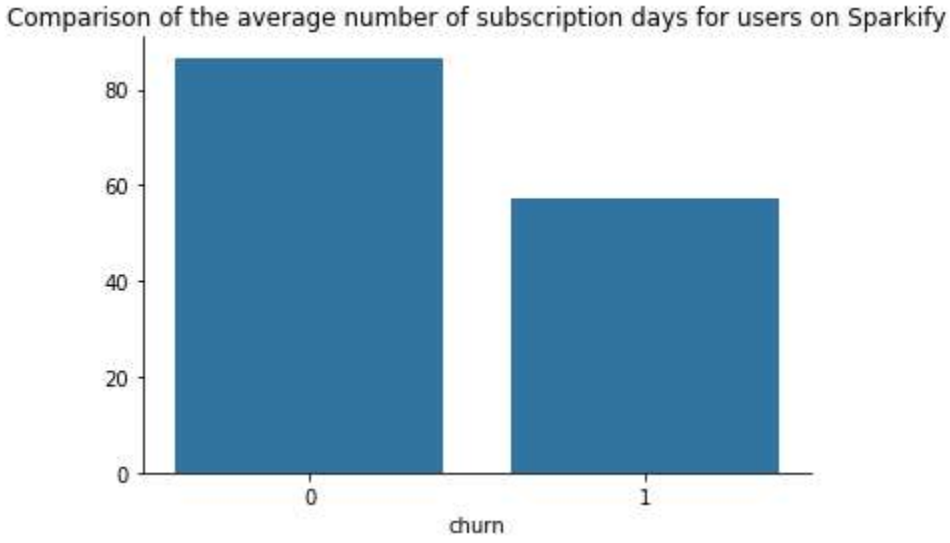

Illustration 4: The average lifetime for users

It can be seen that on average that customer's who churn have a lesser time span on the service.

METHODOLOGY

Data Preprocessing

The input data as shown above is in an event log form, however for it to be fed to a supervised learning model it has to be transformed into a set of features for each user in the dataset. This is appropriate since churn prediction is to be made per user on the platform. In the following, the steps involved in converting the dataset to a set of features will be detailed.

To create features for each user in the data, the Pyspark groupby() function is used to group the data based on each userId in the data. A listing of the preprocessing steps is given below.

1. Create a churn column. Users that have the Cancellation Confirmation event in their page field are considered to have churned. The value 1 is appended to the churn column where the page event equals Cancellation Confirmation and 0 otherwise.
2. Create the following features: gender, paid_subscriber, num_songs, thumbs_up, thumbs_down, duration.
3. Transform the created features into Pyspark vectors in the features field.
4. Apply min-max scaling to the features field to create the scaled_features field
5. Rename the churn column to label and the scaled_features field to features so that they can be recognized by Pyspark models.

userId	gender	paid_subscriber	num_songs	thumbs_up	thumbs_down	duration	churn
100010	0	0	275	17	5	55.6437	0
200002	1	1	387	21	6	70.0746	0
125	1	0	8	0	0	71.3169	1
124	0	1	4079	171	41	131.5559	0
51	1	1	2111	100	21	19.4558	1
7	1	0	150	7	1	72.7782	0
15	1	1	1914	81	14	56.5136	0
54	0	1	2841	163	29	110.7517	1
155	0	1	820	58	3	23.556	0
100014	1	1	257	17	3	85.0834	1

Illustration 5: User features generated from event log data

In illustration 5 above, the gender column is binary and represents females with the value 0 and males with 1. The paid_subscriber feature is also binary and checks if a user has ever been a paid level customer. This is meant to track how, having been a paying subscriber affects an individuals decision to churn. The num_songs column captures the the number of songs played by a user while using Sparkify. The thumbs_up column shows the number of likes given to songs in the platform by any user, while the thumbs_down shows the number of songs disliked. The duration feature displays the lifetime of each user in the service in days. The churn column is binary and the ground truth label.

Implementation

The implementation stage is the model training step. Three supervised learning models were trained on the input features. They are the LogisticRegression, DecisionTreeClassifier, and the Gradient-Boosted Trees models. The rationale is to choose the model that performs best on the validation set, and parameter tune it so as to further improve the chosen model.

The F1 score of the three models are 0.405, 0.723, and 0.635 respectively. Their corresponding accuracies are 0.56, 0.75, and 0.69. Apparently, the DecisionTreeClassifier was chosen as the best classifier for parameter tuning.

Refinement
The refinement of the chosen model was accomplished by parameter tuning the DecisionTreeClassifier based on maxDepth and the maxBins sizes. Curiously, every attempt to tune the model led to overfitting on the training set and poorer F1 scores on the validation set, therefore the default setting of the classifier is retained, since it gives the highest validation scores.

RESULTS

Model Evaluation and Validation
The performance of the best model, which is the DecisionTreeClassifier, is reported on the test set. The F1 score of the model on test set is 0.737 with an accuracy of 0.72, which verifies the robustness of the model. The final value for the maxDepth and the maxBins parameters are 5 and 32 respectively, every other parameter remains at default value as stated earlier.

Justification
The DecisionTreeClassifier is known to be more suited for datasets with categorical data. Two of the features created above are binary, which are the gender and the paid_subscriber features. This may explain the superiority of the DecisionTreeClassifier in the dataset.

The LogisticRegression model gave the poorest performance, however it was only trained for 20 iterations. Its performance could improve with more iterations and less skewed dataset. The Gradient-Boosted Trees model was promising but the drawback was the longer training time it required. It had the longest training time compared to the other models.

CONCLUSION

Reflection
This project was created to detect customer churn from event log data. The steps involved are:

1. Downloading the event log data
2. Analyzing the log data to generate features for model training using the Spark framework
3. Splitting the feature set to train, test and validation sets
4. Training a number of models on the train set and picking the best model based on the F1 score on the validation set
5. Further improvement of the chosen model based on parameter tuning.
6. Reporting the score of the final model based on the test set.

The interesting part of the project was getting decent scores on both the validation and test sets, while the challenging part was having to cope with slower execution speeds, while running Spark in local mode.

Improvement
More can be done to improve the project. For example, more features can be engineered to further extract insight from the dataset. Such new features can be, noting if a user has ever downgraded their account, checking if a user has added friends to the service, tracking how many errors a user encountered while using Sparkify, and checking how a user responds to served advert. These will enable a model to have better variance, and score.

Furthermore, other models can be trained on the dataset and their performances compared. Such models include the the support vector machines classifier (SVM), the artificial neural network and the random forest models.

In addition to the above suggestions, the data preprocessing time and the training duration for each model can also be improved by running Spark in standalone mode. This will take advantage of Spark's ability to parallelise operations to reduce processing time. As consequence, a larger input dataset can be fed to supervised learners, leading to less over-fitting and improved classifier performance.

References

Cs230.stanford.edu. (2019). Section 7. [online] Available at: http://cs230.stanford.edu/section/7/ [Accessed 25 Sep. 2019].

GitHub. (2019). linpingyu/Sparkify. [online] Available at: https://github.com/linpingyu/Sparkify/blob/master/Sparkify.ipynb [Accessed 25 Sep. 2019].

GitHub. (2019). udacity/machine-learning. [online] Available at: https://github.com/udacity/machine-learning/blob/master/projects/capstone/report-example-1.pdf [Accessed 25 Sep. 2019].