The University of Texas at Arlington

WEB AND SOCIAL ANALYTICS

INSY 5377-001

SUMMER 2022

Project Report

Spotify Users Churn Prediction

Date Of Submission: 06/17/2022

Submitted by:

Team 13

Mantena, Surendra Varma (1001956606)

Instructor:

Dr. Riyaz Sikora

Table of Contents

1. Introduction

1.1. Project Overview

Spotify is a music streaming service that is available on smartphones and computers. In the first quarter of 2022, Spotify had 182 million premium subscribers worldwide, up from 158 million in the same quarter of 2021. Spotify users can listen to songs using either the free or premium subscription plans, which include advanced features and are ad-free. Users can upgrade, downgrade, or cancel their services at any time. The data in this project is about user interactions with the services, such as playing songs, adding them to playlists, rating them with a thumbs up or down, adding a friend, logging in or out, upgrading or downgrading the service etc.

1.2. Business Problem

Maintaining customer satisfaction and identifying users who may cancel service are primary concerns for the service provider because it is cheaper to gain new customers than to keep existing ones. Businesses must keep their customers to thrive. Churn (the process of losing customers) is thus a significant business issue.

1.3. Goal

The goal of this project is to analyze user activity logs and create a classifier to identify users who are likely to churn — cancel their subscription to the Spotify music streaming service.

1.4. Challenges

- Imbalanced dataset
- The dataset consists of multiple target variables and is a multi-class classification dataset. We want to predict if a user will be churned or not by converting multi-class classification to binary classification.
- Managing Null and Empty values/observations.

2. Data Description

Our initial dataset was downloaded from Kaggle [1]:

https://www.kaggle.com/code/yukinagae/sparkify-project-churn-prediction/notebook

The data set contains user activity logs from October 1, 2018, to December 1, 2018. Data logs are generated whenever a user interacts with a music streaming app, whether it is playing songs, adding them to playlists, rating them with a thumbs up or down, adding a friend, logging in or out, changing settings, etc. The original dataset is 12GB, but for this project, we used a subset dataset with 18 columns and 286500 rows. The full data set contains logs from 22277 different users, whereas the subset only includes 225 user's activities. Twelve of the eighteen columns are strings, and it appears that mostly of them are categorical variables.

Data Schema: The dataset is described by three types of columns.

1. <u>User identification data:</u>
   - userId (string)
   - firstName (string)
   - lastName (string)
   - Location (string)
   - Gender (string)
   - userAgent (string)

2. <u>Session/Account information</u>
   - sessionId (int)
   - level (string)
   - auth (string)
   - itemInSession (int)
   - length (double)

3. <u>Activity information</u>
   - song (string)
   - artist (string)
   - page (string)
   - activity timestamp (int)
   - registration timestamp (int)

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
	ts	userId	sessionId	page	auth	method	status	level	itemInSess	location	userAgent	lastName	firstName	registratio	gender	artist	song	length
0	1.54E+12	30	29	NextSong	Logged In	PUT	200	paid	50	Bakersfiel	Mozilla/5.(	Freeman	Colin	1.54E+12	M	Martha Til	Rockpools	277.8902
1	1.54E+12	9	8	NextSong	Logged In	PUT	200	free	79	Boston-Ca	"Mozilla/5	Long	Micah	1.54E+12	M	Five Iron F	Canada	236.0942
2	1.54E+12	30	29	NextSong	Logged In	PUT	200	paid	51	Bakersfiel	Mozilla/5.(	Freeman	Colin	1.54E+12	M	Adam Lam	Time For N	282.8273
3	1.54E+12	9	8	NextSong	Logged In	PUT	200	free	80	Boston-Ca	"Mozilla/5	Long	Micah	1.54E+12	M	Enigma	Knocking C	262.713
4	1.54E+12	30	29	NextSong	Logged In	PUT	200	paid	52	Bakersfiel	Mozilla/5.(	Freeman	Colin	1.54E+12	M	Daft Punk	Harder Bet	223.6077
5	1.54E+12	9	8	NextSong	Logged In	PUT	200	free	81	Boston-Ca	"Mozilla/5	Long	Micah	1.54E+12	M	The All-Am	Don't Leav	208.3
6	1.54E+12	9	8	NextSong	Logged In	PUT	200	free	82	Boston-Ca	"Mozilla/5	Long	Micah	1.54E+12	M	The Velvet	Run Run R	260.4665
7	1.54E+12	30	29	NextSong	Logged In	PUT	200	paid	53	Bakersfiel	Mozilla/5.(	Freeman	Colin	1.54E+12	M	Starflyer 5	Passenger	185.4428
8	1.54E+12	30	29	Add to Pla	Logged In	PUT	200	paid	54	Bakersfiel	Mozilla/5.(	Freeman	Colin	1.54E+12	M			
9	1.54E+12	30	29	NextSong	Logged In	PUT	200	paid	55	Bakersfiel	Mozilla/5.(	Freeman	Colin	1.54E+12	M	Frumpies	Fuck Kitty	134.4779
10	1.54E+12	9	8	NextSong	Logged In	PUT	200	free	83	Boston-Ca	"Mozilla/5	Long	Micah	1.54E+12	M	Britt Nicol	Walk On T	229.8775
11	1.54E+12	9	8	Roll Adver	Logged In	GET	200	free	84	Boston-Ca	"Mozilla/5	Long	Micah	1.54E+12	M			
12	1.54E+12	30	29	NextSong	Logged In	PUT	200	paid	56	Bakersfiel	Mozilla/5.(	Freeman	Colin	1.54E+12	M	Edward Sh	Jade	223.5818
13	1.54E+12	9	8	NextSong	Logged In	PUT	200	free	85	Boston-Ca	"Mozilla/5	Long	Micah	1.54E+12	M	Tesla	Gettin' Bet	203.064
14	1.54E+12	9	8	Thumbs Up	Logged In	PUT	307	free	86	Boston-Ca	"Mozilla/5	Long	Micah	1.54E+12	M			
15	1.54E+12	30	29	NextSong	Logged In	PUT	200	paid	57	Bakersfiel	Mozilla/5.(	Freeman	Colin	1.54E+12	M	Stan Mosk	So-Called I	246.7
16	1.54E+12	9	8	NextSong	Logged In	PUT	200	free	87	Boston-Ca	"Mozilla/5	Long	Micah	1.54E+12	M	Florence +	You've Go	168.6461
17	1.54E+12	74	217	NextSong	Logged In	PUT	200	free	0	Tallahasse	"Mozilla/5	Williams	Ashlynn	1.54E+12	F	Tokyo Poli	Citizens Of	166.1122
18	1.54E+12	30	29	NextSong	Logged In	PUT	200	paid	58	Bakersfiel	Mozilla/5.(	Freeman	Colin	1.54E+12	M	Orishas	Represent	222.2232
19	1.54E+12	9	8	NextSong	Logged In	PUT	200	free	88	Boston-Ca	"Mozilla/5	Long	Micah	1.54E+12	M	Ratatat	Swisha	229.7726
20	1.54E+12	74	217	NextSong	Logged In	PUT	200	free	1	Tallahasse	"Mozilla/5	Williams	Ashlynn	1.54E+12	F	Manolo Ga	Carbon Y F	283.7416
21	1.54E+12	30	29	NextSong	Logged In	PUT	200	paid	59	Bakersfiel	Mozilla/5.(	Freeman	Colin	1.54E+12	M	Downhere	Here I Am	223.9212
22	1.54E+12	54	53	NextSong	Logged In	PUT	200	paid	0	Spokane-S	Mozilla/5.(	Warren	Alexi	1.53E+12	F	Modjo	What I Me	250.9318
23	1.54E+12	9	8	NextSong	Logged In	PUT	200	free	89	Boston-Ca	"Mozilla/5	Long	Micah	1.54E+12	M	MÃƒÂƒÃ‚Â	Sticky Swe	231.2616
24	1.54E+12	74	217	NextSong	Logged In	PUT	200	free	2	Tallahasse	"Mozilla/5	Williams	Ashlynn	1.54E+12	F	David Bow	Sorrow (1!	174.4191
25	1.54E+12	30	29	NextSong	Logged In	PUT	200	paid	60	Bakersfiel	Mozilla/5.(	Freeman	Colin	1.54E+12	M	Skillet	Rebirthing	233.3253
26	1.54E+12	54	53	NextSong	Logged In	PUT	200	paid	1	Spokane-S	Mozilla/5.(	Warren	Alexi	1.53E+12	F	Edwyn Col	You'll Nev	216.842

Fig 1. Sample dataset

The Page column holds the valuable information of user interactions with a music streaming app. Overall, the page column in our dataset has 22 different variables as shown in Fig 2.

```
['NextSong',
 'Add to Playlist',
 'Roll Advert',
 'Thumbs Up',
 'Downgrade',
 'Thumbs Down',
 'Home',
 'Logout',
 'Help',
 'Login',
 'Upgrade',
 'Add Friend',
 'About',
 'Settings',
 'Submit Upgrade',
 'Submit Downgrade',
 'Error',
 'Save Settings',
 'Cancel',
 'Cancellation Confirmation',
 'Register',
 'Submit Registration']
```

Fig 2. Page column Variables

3. Research Questions

1. Find out if the usage activity of paid and free users has changed over time?
2. Top ten songs played by the most unique users?
3. Find out all the activities of subscription downgraded users before clicking the downgrade submit button?
4. Find out all the activities of subscription upgraded users before clicking the upgrade submit button?
5. Find out the length (in sec) of the longest session?
6. Top 15 Artists whose songs were trending?
7. Which region has the most users of the application?
8. What is the average app usage by cancelled (no longer using app) and not cancelled (still using app) users?

4. Methodology

4.1. Loading Dataset

First Load the Dataset into a data frame (df). We use Python Jupyter Notebook in this exercise for data preprocessing, cleaning, modeling, and making predictions. Fig 3 shows the sample data frame table.

```
# Load the dataset
import pandas as pd
df = pd.read_csv('dataset.csv', sep=',',index_col=[0])
df.head()
```

	ts	userId	sessionId	page	auth	method	status	level	itemInSession	location	userAgent	lastName	firstName	registration	ge
0	1538352117000	30.0	29	NextSong	Logged In	PUT	200	paid	50	Bakersfield, CA	Mozilla/5.0 (Windows NT 6.1; WOW64; rv:31.0) G...	Freeman	Colin	1.538173e+12	
1	1538352180000	9.0	8	NextSong	Logged In	PUT	200	free	79	Boston-Cambridge-Newton, MA-NH	"Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebK...	Long	Micah	1.538332e+12	
2	1538352394000	30.0	29	NextSong	Logged In	PUT	200	paid	51	Bakersfield, CA	Mozilla/5.0 (Windows NT 6.1; WOW64; rv:31.0) G...	Freeman	Colin	1.538173e+12	
3	1538352416000	9.0	8	NextSong	Logged In	PUT	200	free	80	Boston-Cambridge-Newton, MA-NH	"Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebK...	Long	Micah	1.538332e+12	
4	1538352676000	30.0	29	NextSong	Logged In	PUT	200	paid	52	Bakersfield, CA	Mozilla/5.0 (Windows NT 6.1; WOW64; rv:31.0) G...	Freeman	Colin	1.538173e+12	

Fig 3. Sample data frame table.

4.2. Data preprocessing and Cleaning:

In data science, preprocessing is an important step. The techniques used to collect data are frequently not effectively managed, which leads to missing numbers, out-of-range values (such as Income: 100), and incorrect data combinations (such as Sex: Male, Pregnant: Yes). Data analysis that has not been thoroughly checked for these issues may yield false results. Therefore, before performing an analysis, it is crucial to consider the representation and quality of the data.

Handling Missing Data:

You also need to check the percentage of missing values in each column.

- percentage of data in a column is missing, then you need to drop the column.
- percentage of data in a column is missing, then you can:
  - Fill the missing values will representative data Or
  - you can remove rows containing missing values

The percentage of data missing in each column is shown in Fig. 4. 2.9 percent of the entries are missing from the userId, location, userAgent, lastName, firstname, registration, and gender columns. 20.38 percent of the values for columns like gender, song, and length are missing.

```
df.isnull().sum()/len(df) * 100
```

```
ts                0.000000
userId            2.913089
sessionId         0.000000
page              0.000000
auth              0.000000
method            0.000000
status            0.000000
level             0.000000
itemInSession     0.000000
location          2.913089
userAgent         2.913089
lastName          2.913089
firstName         2.913089
registration      2.913089
gender            2.913089
artist           20.381152
song             20.381152
length           20.381152
dtype: float64
```

Fig 4. Missing/Null values

A userId with a Null value represents a user(s) who are currently signing in (do not have an account yet) or registering. The same users also have Null values in the Location, Name, Gender, and other columns. Rows with null userId are thus dropped.

The values of artist, song and length columns are null only when the Page columns values is not 'Next page'. Even though the percentage of missing values are large we cannot drop those values from the data frame because they help in making predictions. So, we fill the missing values with previous row values by filtering them based on user activity timestamp.

Handling Duplicated Data:

Presence of duplicated data cause bias in our data analysis. So, we need to check them and remove them from the dataset.

4.3. Data Visualization on Research Questions:

4.3.1. **Research Question 1:** What is the usage activity of paid and free users over time?

Fig. 5 shows clearly that paid users are growing while free users are steadily declining. This might be because of existing customers discontinuing the service, free users downgrading the service, paid users upgrading the service, and new users joining the services.

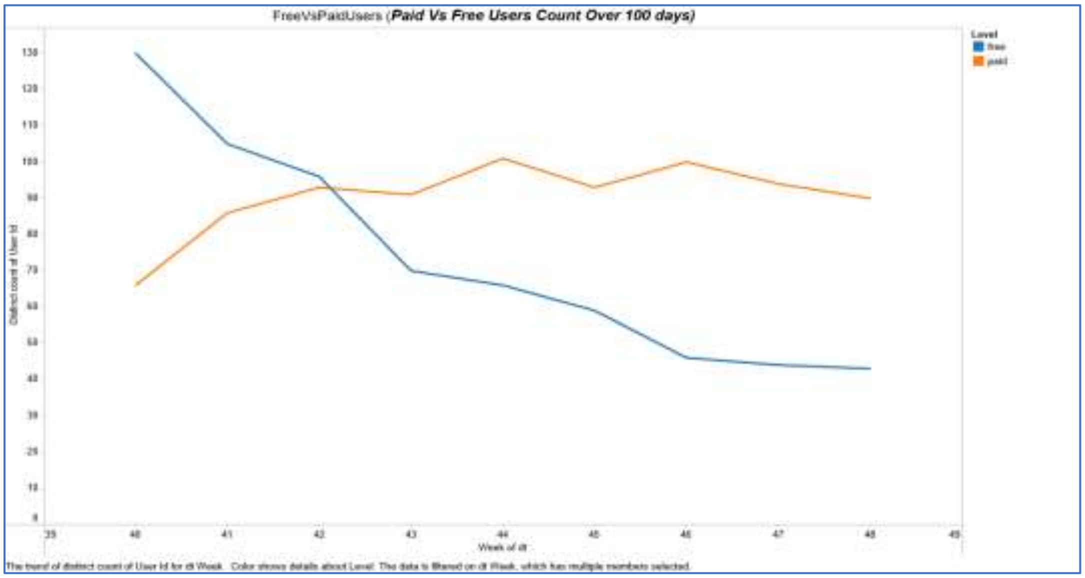

Fig 5. Paid Users VS Free Users count over 100 days

4.3.2. **Research Question 2:** Top ten songs played by the most unique users?

You are the one is most played song (189 times) followed by Reverly (177 times)

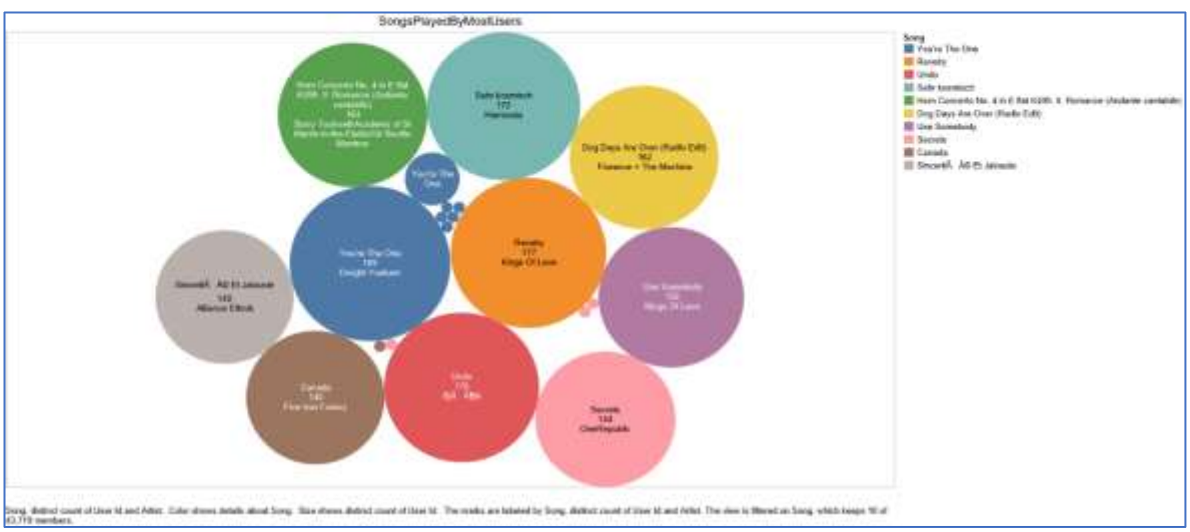

Fig 6: Top ten songs played by the most

4.3.3. **Research Question 3:** Find out all the activities of subscription downgraded users before clicking the downgrade submit button?

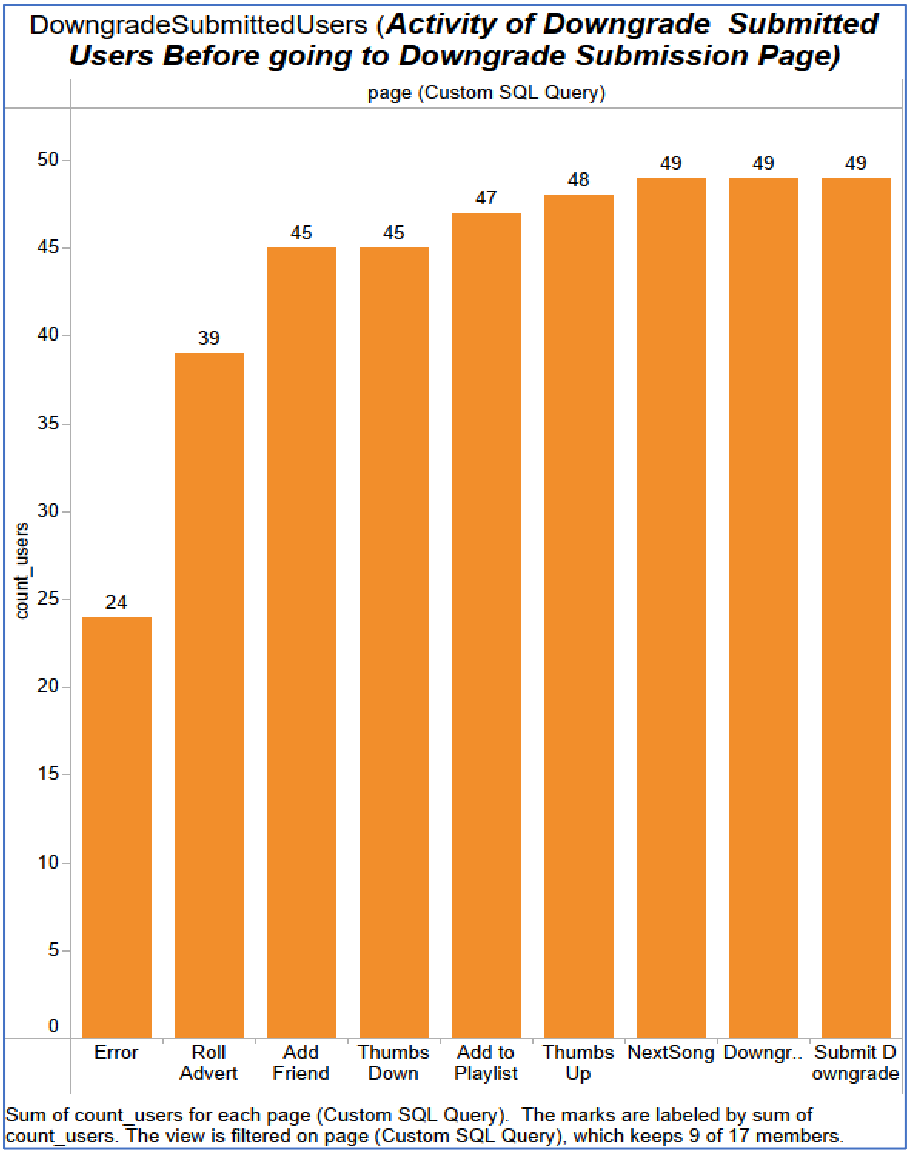

Fig 7: All activities of subscription downgraded users before clicking the downgrade submit button

The submit downgrade button has been clicked by 49 paid users, as can be shown in Fig. 7. Out of 49 users, 45 users gave the songs a thumbs down, 39 users saw advertisements despite having a paid membership, and 24 users experienced Errors while using the application. These instances could lead to a user's subscription being downgraded.

4.3.4. **Research Question 4:** Find out all the activities of subscription upgraded users before clicking the upgrade submit button?

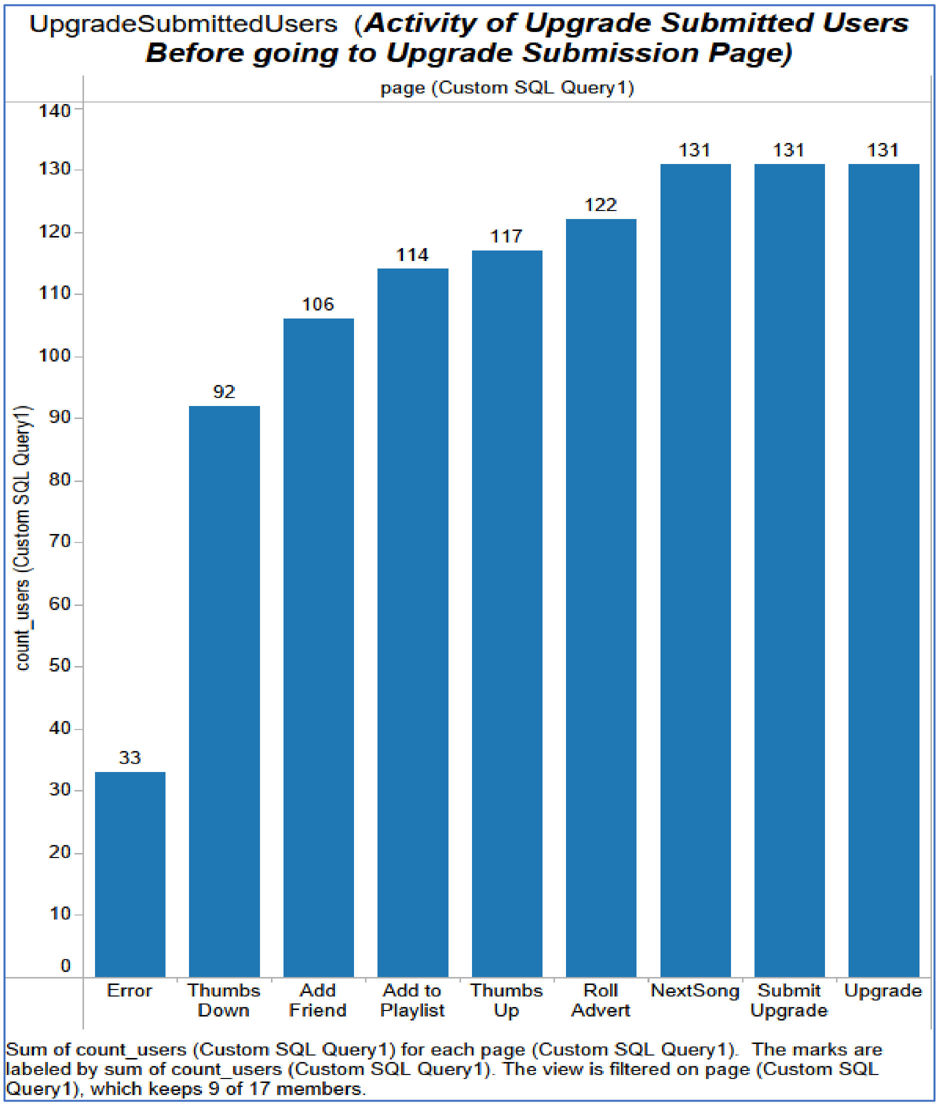

Fig 8: All activities of subscription upgraded users before clicking the upgrade submit button

There have been 131 clicks on the submit downgrade button. Only 33 out of 139 free users had application errors, while 122 free users saw advertisements. These can be reasons for customers to upgrade their subscriptions.

Research Question 4 and 5 Comparistion Graph:

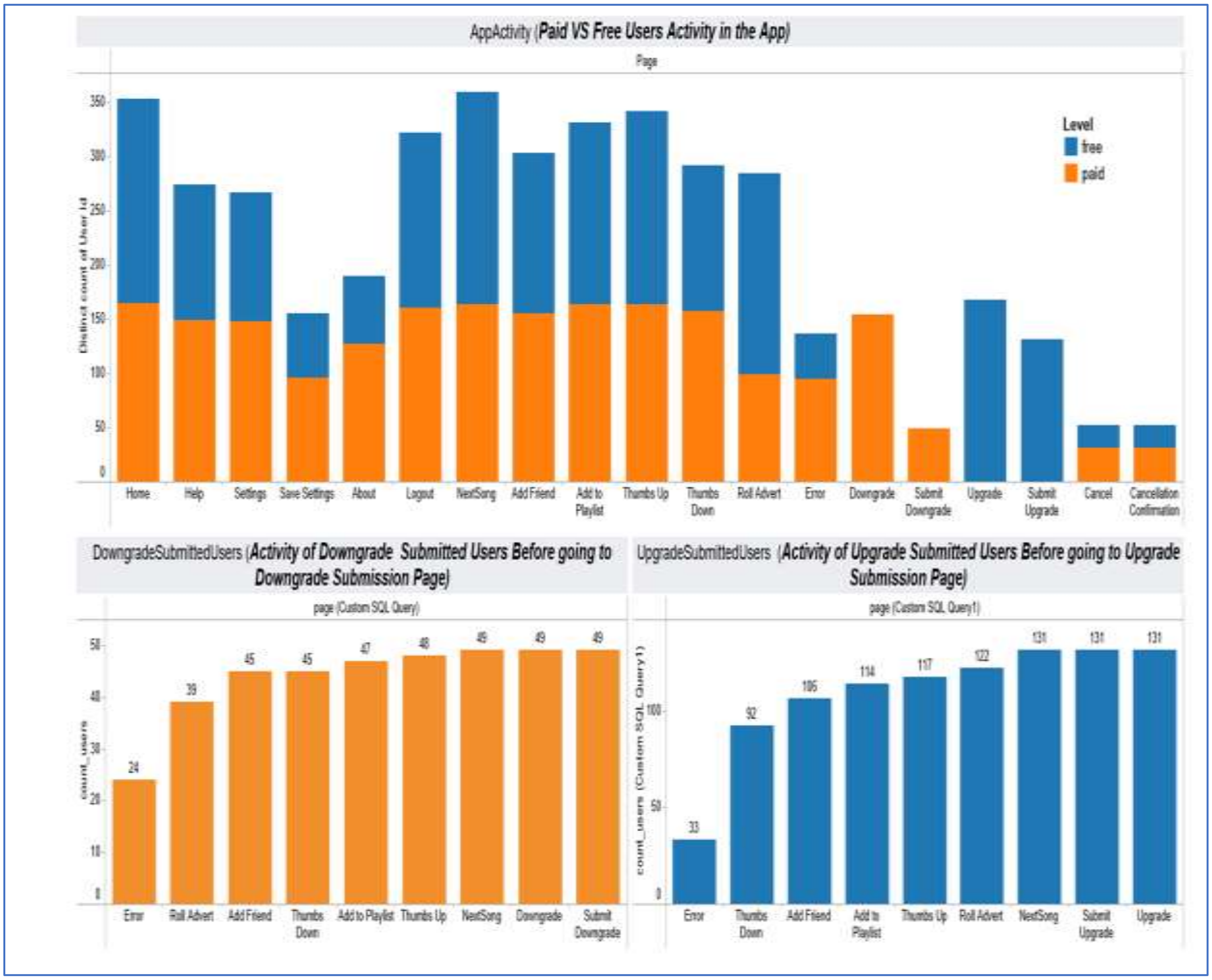

Fig 9: Comparison Graph

Figure 9's top graph displays the number of paid and free users' activities. The graph makes it obvious which users can click "Submit Downgrade": only paid users and only free users can click on submit upgrade.

Additionally, there are more than 225 and almost 350 users clicked on next song, which explains that some people have upgraded and downgraded their subscriptions multiple times.

The comparison graph also reveals that paying users face twice as many errors as free users.

4.3.5. **Research Question 5:** Find out the length (in sec) of the longest session?

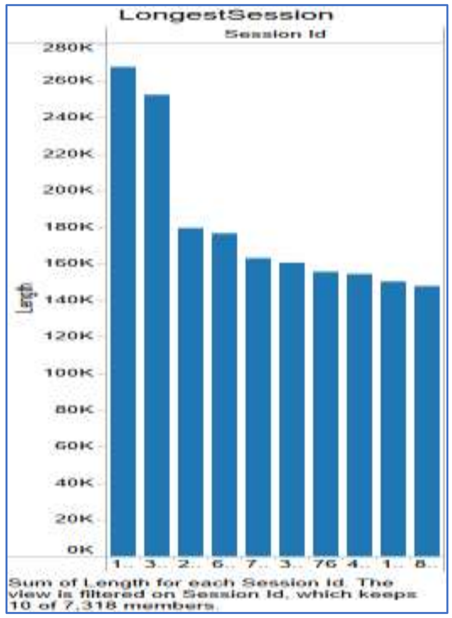

Fig 10. Longest session by length (Time in secs).

The session with id 100 has the longest session of 265,000 seconds.

4.3.6. **Research Question 6:** Top 15 Artists whose songs were trending?

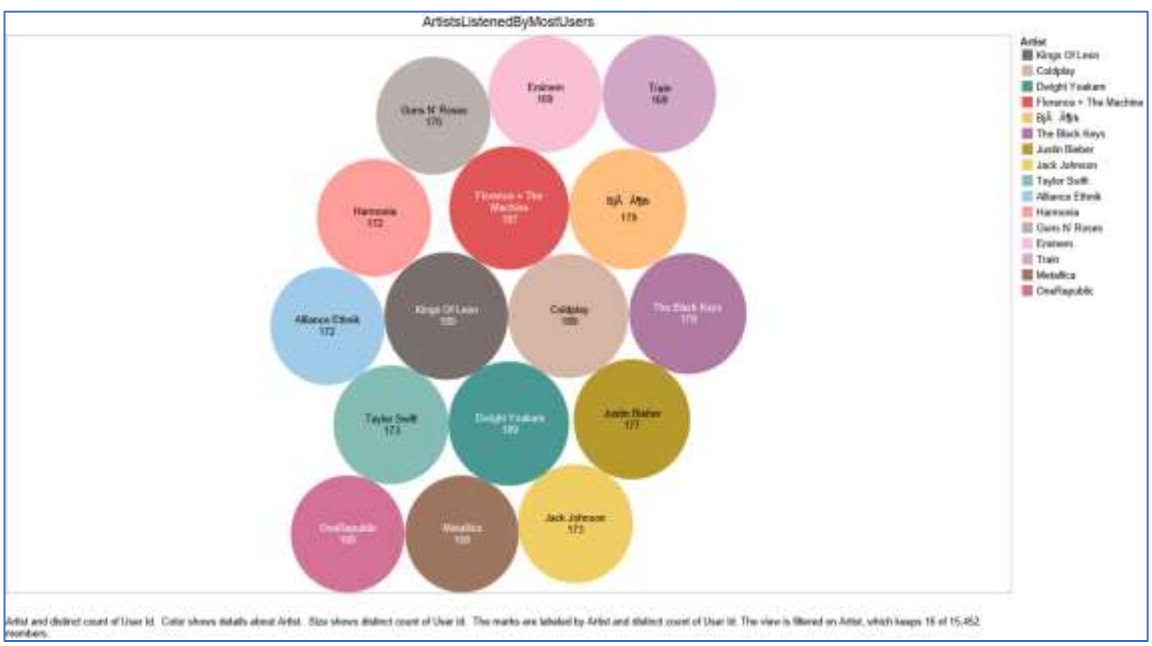

Fig 11. Top 15 Artists

Most played artist are Kings of Leon (199 unique users) and Coldplay (189 unique users)

4.3.7. **Research Question 7:** Which region has the most users of the application?

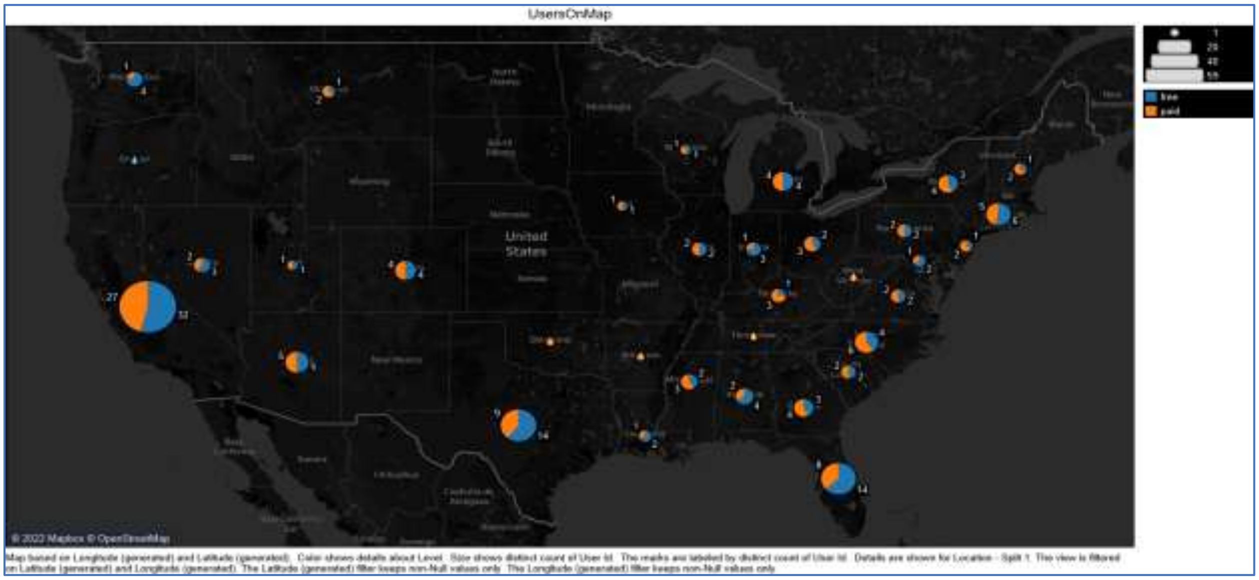

Fig 12. Most users by Location

The most users of the application, both free and paid, are found in California, as is evident from the map.

4.3.8. **Research Question 8:** What is the average app usage by cancelled (no longer using app) and not cancelled (still using app) users?

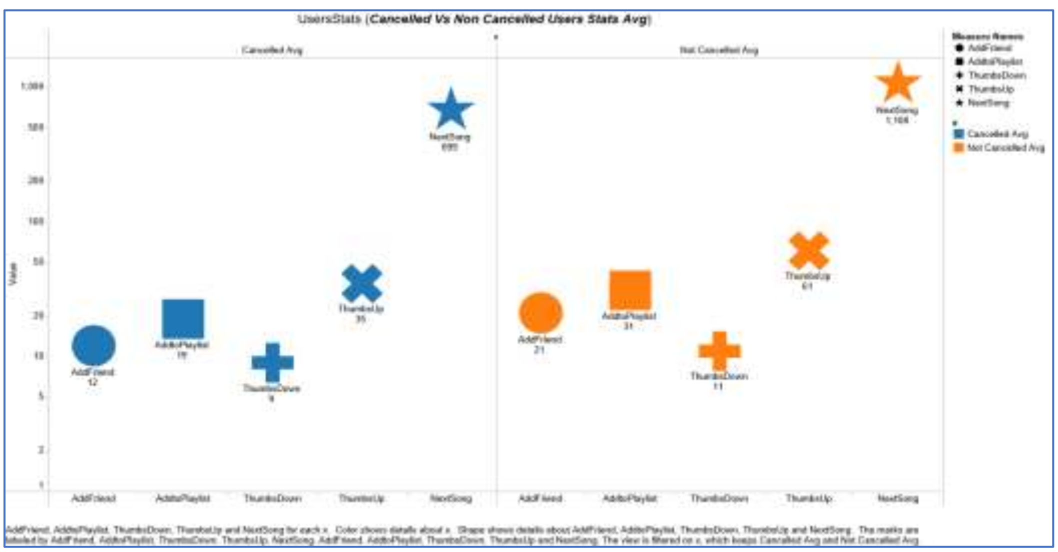

Fig 13. Cancelled Vs Non-Cancelled User Stats Average

According to the graph, non-cancelled consumers use the service on average twice as much as cancelled ones. The users who are more likely to stop using the service are therefore not regular or active consumers. Users who have not been cancelled are regular users who utilize the application frequently.

4.4. Feature Extraction

The next step was to create the churn column, which can be calculated by looking at the page column and seeing if we have “Cancellation Confirmation” event. Users with this type of event-data row will be marked as churning.

1 — users who cancelled their subscription within the observation period (Cancellation confirmation events)

0 — users who kept the service throughout

After defining churn, we find that 23.1 % of users are churned in the dataset. The below figure shows Top 5 rows of churn and non-churn users.

```
225 rows.
23.1% users churned.
+------+-----+
|userId|churn|
+------+-----+
|100010|    0|
|200002|    0|
|   125|    1|
|    51|    1|
|   124|    0|
+------+-----+
only showing top 5 rows
```

Fig 14. Sample Churn and Non-Churn users

New features are created/extracted from the current features to make predictions as shown in Fig 15.

```
Index(['is_female', 'is_male', 'registration', 'distinct_artists',
       'total_listen_time', 'distinct_sessions', 'distinct_songs',
       'total_is_logged_in', 'total_is_cancelled', 'total_is_paid',
       'total_is_get', 'total_is_put', 'total_is_help', 'total_is_thumbs_up',
       'total_is_submit_downgrade', 'total_is_upgrade', 'total_is_thumbs_down',
       'total_is_roll_advert', 'total_is_downgrade',
       'total_is_cancellation_confirmation', 'total_is_error',
       'total_is_submit_upgrade', 'total_is_settings', 'total_is_cancel',
       'total_is_next_song', 'total_is_about', 'total_is_add_to_playlist',
       'total_is_home', 'total_is_add_friend', 'total_is_404', 'total_is_307',
       'total_is_200', 'total_is_windows', 'total_is_macintosh',
       'total_is_ipad', 'total_is_iphone', 'total_is_compatible',
       'total_is_linux', 'distinc_locs'],
      dtype='object')
```

Fig 15. Newly created features.

The data frame with the newly created features now only has 225 rows. Each row represents a distinct user. The values of each feature are nothing but the sum of user actions of that feature. If user 1 clicks on the next page 795 times, the value of the "total is nextpage" column for that user is 795. When a user clicks the downgrade button 3 times, the "total is downgrade" value for that user is three.

The SQL query that is developed to create a data frame with new features containing sum values is shown in Fig. 16.

Fig 16. SQL query to create a data frame with new features

A Following execution of the above SQL query, the table is exported as a csv file and reloaded as a data frame to python Jupyter, as shown in Fig 17 below.

userid	is_female	is_male	registration	distinct_artists	total_listen_time	distinct_sessions	distinct_songs	total_is_logged_in	total_is_cancelled	total_is_pa
10	0	1	1538159495000	595	196283.64286	6	664	795	0	
100	0	1	1537982255000	1773	800610.93134	35	2407	3214	0	
100001	1	0	1534627466000	134	51394.57974	4	138	186	1	
100002	1	0	1529934689000	185	55241.07011	4	195	218	0	
100003	1	0	1537309344000	50	19009.66260	2	51	77	1	
...	...	...	...	...	...	...	...	...	...	
95	1	0	1538211832000	1211	524036.51135	33	1599	2091	0	
96	1	0	1537149749000	1298	547663.00754	19	1714	2176	0	
97	1	0	1536019842000	1360	599764.33035	30	1792	2404	0	
98	0	1	1538069638000	1644	734321.21079	28	2264	2889	0	
99	1	0	1531811983000	443	154774.94432	11	509	614	0	

225 rows × 39 columns

Fig 17. Final dataset for predictions.

4.5. Modelling

The goal of our predictive model is to determine which clients are likely to leave and which are not. As a result, the problem is fundamentally one of binary classification. The classes are Cancelling vs. Non-Cancelling.

If non-cancelling users are labeled as canceling, the company may take actions that mislead the customer and cause them to cancel the service. It is also critical to correctly categorize Churning customers. Then our classifier should be precise in classifying both types of customers.

We choose to run following models on our data:

- Logistic Regression
- Random Forest Classifier
- MLP Classifier – Neural Networks
- Decision Tree Classifier
- AdaBoost Classifier
- Gradient Boost Classifier

4.6. Evaluation Metrics

On imbalance datasets, accuracy would be not a correct metric to evaluate. The F1 score is a balance of precision and recall. When predicting churn, precision aims to ensure that it is really a churn, whereas recall aims to avoid missing any true churns, which is why F1 score is used to evaluate model performance.

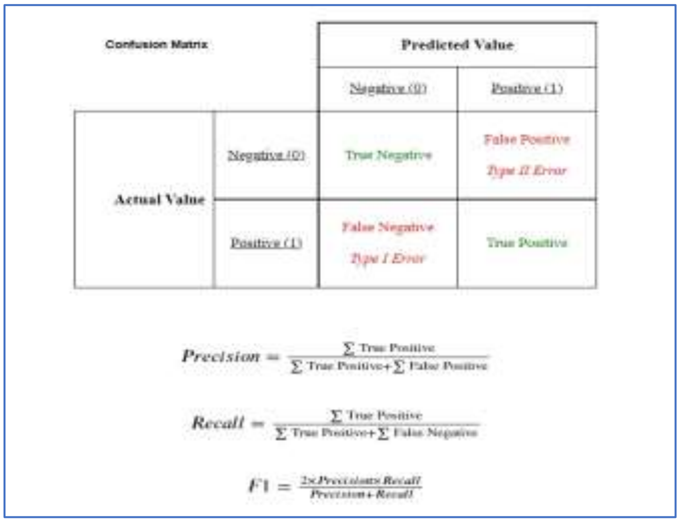

Fig 18. Evaluation Metrics

5. Results and Discussion

The initial results of classification Tree models such as Random Forest Classifier, Decision Tree Classifier, AdaBoost Classifier, Gradient Boost Classifier provided 95 to 100 percent of f1 scores, whereas logistic regression, MLP classifier, SVM, and KNN provided 60 to 70%. We can clearly see from the results that the classification tree models are overfitting.

We tried to use PCA techniques to the dataset in order to obtain better results based on the most significant values. However, when we use PCA, all the models' f1 scores are reduced by 5 to 10%, but the difference between classification model scores and regression model scores is large (approximately 30 to 40%), which explains why PCA is not useful in model selection.

In addition, we tried OLS Regression (as shown in Fig 19), and the results show that the R-square and Adj. R-squared values are equal to '1,', significant variables (p>|t| <0.001) and indicating that there is strong multicollinearity among features.

OLS Regression Results

Dep. Variable:	total_is_cancellation_confirmation	**R-squared:**	1.000
Model:	OLS	**Adj. R-squared:**	1.000
Method:	Least Squares	**F-statistic:**	2.632e+08
Date:	Thu, 04 Aug 2022	**Prob (F-statistic):**	0.00
Time:	19:18:38	**Log-Likelihood:**	1837.4
No. Observations:	225	**AIC:**	-3617
Df Residuals:	196	**BIC:**	-3518
Df Model:	28		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
is_female	-0.0010	0.001	-1.128	0.261	-0.003	0.001
is_male	-0.0010	0.001	-1.128	0.261	-0.003	0.001
registration	1.928e-15	1.76e-15	1.097	0.274	-1.54e-15	5.4e-15
distinct_artists	9.922e-09	2.08e-07	0.048	0.962	-4e-07	4.19e-07
total_listen_time	-7.342e-11	1.52e-09	-0.048	0.961	-3.06e-09	2.92e-09
distinct_sessions	-9.21e-08	1.43e-06	-0.065	0.949	-2.91e-06	2.72e-06
distinct_songs	-2.646e-08	3.16e-07	-0.084	0.933	-6.5e-07	5.97e-07
total_is_logged_in	-0.2639	3.44e-06	-7.67e+04	0.000	-0.264	-0.264
total_is_cancelled	0.6806	8.51e-06	8e+04	0.000	0.681	0.681
total_is_paid	4.957e-07	1.9e-05	0.026	0.979	-3.7e-05	3.79e-05
total_is_get	0.2361	3.15e-06	7.49e+04	0.000	0.236	0.236
total_is_put	0.1806	2.51e-06	7.19e+04	0.000	0.181	0.181
total_is_help	-0.0556	2.27e-06	-2.45e+04	0.000	-0.056	-0.056
total_is_thumbs_up	-1.461e-07	2e-06	-0.073	0.942	-4.1e-06	3.81e-06
total_is_submit_downgrade	3.059e-07	1.33e-05	0.023	0.982	-2.59e-05	2.65e-05
total_is_upgrade	-0.0556	3.5e-06	-1.59e+04	0.000	-0.056	-0.056
total_is_thumbs_down	-1.803e-07	2.42e-06	-0.074	0.941	-4.95e-06	4.59e-06
total_is_roll_advert	-0.0556	1.29e-06	-4.29e+04	0.000	-0.056	-0.056
total_is_downgrade	-0.0556	1.78e-06	-3.12e+04	0.000	-0.056	-0.056
total_is_error	-0.0556	4.77e-06	-1.17e+04	0.000	-0.056	-0.056
total_is_submit_upgrade	6.381e-07	1.37e-05	0.047	0.963	-2.63e-05	2.76e-05
total_is_settings	-0.0556	2.33e-06	-2.38e+04	0.000	-0.056	-0.056
total_is_next_song	-1.521e-07	2.05e-06	-0.074	0.941	-4.2e-06	3.9e-06
total_is_about	-0.0556	3.04e-06	-1.83e+04	0.000	-0.056	-0.056
total_is_add_to_playlist	-1.8e-07	2.2e-06	-0.082	0.935	-4.51e-06	4.15e-06
total_is_home	-0.0556	1.63e-06	-3.41e+04	0.000	-0.056	-0.056
total_is_add_friend	-2.435e-07	2.32e-06	-0.105	0.916	-4.81e-06	4.32e-06
total_is_404	-7.673e-17	2.2e-17	-3.489	0.001	-1.2e-16	-3.34e-17
total_is_307	-5.593e-17	1.7e-17	-3.290	0.001	-8.95e-17	-2.24e-17
total_is_200	-8.968e-17	1.42e-17	-6.322	0.000	-1.18e-16	-6.17e-17
total_is_windows	0.0833	1.04e-06	8.03e+04	0.000	0.083	0.083
total_is_macintosh	0.0833	1.04e-06	8.03e+04	0.000	0.083	0.083
total_is_ipad	0.0833	1.04e-06	8.03e+04	0.000	0.083	0.083
total_is_iphone	0.0833	1.04e-06	8.03e+04	0.000	0.083	0.083
total_is_compatible	7.149e-10	1.65e-08	0.043	0.966	-3.19e-08	3.33e-08
total_is_linux	0.0833	1.04e-06	8.03e+04	0.000	0.083	0.083
distinc_locs	-0.0021	0.002	-1.119	0.264	-0.006	0.002

Omnibus:	33.120	**Durbin-Watson:**	0.011
Prob(Omnibus):	0.000	**Jarque-Bera (JB):**	50.770
Skew:	0.849	**Prob(JB):**	9.45e-12
Kurtosis:	4.592	**Cond. No.**	1.36e+16

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.36e+16. This might indicate that there are strong multicollinearity or other numerical problems.

Fig 19. OLS Regression Results

5.1. Multicollinearity:

Regressors are orthogonal when there is no linear relationship between them. Unfortunately, linear dependencies frequently exist in real life data, which is referred to as multicollinearity. Multicollinearity could result in significant problems during model fitting. For example, multicollinearity between regressors may result in large variances and covariances for the OLS estimators, which could lead to unstable/poor parameter estimates. In practice, multicollinearity often pushes the parameter estimates higher in absolute value than they really should be. Further, coefficients have been observed to switch signs in multicollinear data. In sum, the multicollinearity should prompt us to question the validity and reliability of the specified model.

Multicollinearity be detected by looking at eigenvalues as well. When multicollinearity exists, at least one of the eigenvalues is close to zero (it suggests minimal variation in the data that is orthogonal with other eigen vectors).

We used VIF (Variable inflation factor) to find out the features that has strong relationship with the target. If a strong relationship exists between the target and at least one other regressor, the VIF will be high. What is high? Textbooks usually suggest 5 or 10 as a cutoff value above which the VIF score suggests the presence of multicollinearity. So, which one, 5 or 10? If the dataset is very large with a lot of features, a VIF cutoff of 10 is acceptable. Smaller datasets require a more conservative approach where the VIF cutoff may needed to be dropped to 5. Fig 20 below shows the results VIF results.

We did not remove every regressor with a VIF value greater than 5, but we did remove one regressor each time because removing every regressor with a VIF value greater than 5 resulted in an error. Instead, after dropping a feature, we check the VIF values every time and run models to see if there is any improvement in model scores.

Finally, we removed the "total is add friend," "total is put," "total is cancelled," "distinct artists," and "total is add playlist" features from the data frame to achieve consistent predictive model scores.

```
from statsmodels.stats.outliers_influence import variance_inflation_factor
x_temp = sm.add_constant(X)

vif = pd.DataFrame()
vif["VIF Factor"] = [variance_inflation_factor(x_temp.values, i) for i in range(x_temp.values.shape[1])]
vif["features"] = x_temp.columns
print(vif.round(1))
```

```
      VIF Factor                  features
0   1.382533e+12                 is_female
1   1.383807e+12                   is_male
2   1.100000e+00              registration
3   7.080000e+02          distinct_artists
4   1.045600e+04         total_listen_time
5   1.800000e+01         distinct_sessions
6   3.711100e+03            distinct_songs
7   9.007199e+15        total_is_logged_in
8   1.093009e+09        total_is_cancelled
9   2.900000e+00             total_is_paid
10  4.069395e+11              total_is_get
11           inf              total_is_put
12  9.450810e+08             total_is_help
13  7.113000e+02        total_is_thumbs_up
14  2.500000e+00  total_is_submit_downgrade
15  2.333077e+09          total_is_upgrade
16  4.140000e+01      total_is_thumbs_down
17  8.133646e+10      total_is_roll_advert
18  1.047142e+11        total_is_downgrade
19  3.550840e+08            total_is_error
20  4.100000e+00    total_is_submit_upgrade
21  2.719579e+09         total_is_settings
22  2.124264e+05        total_is_next_song
23  3.118158e+08            total_is_about
24  2.130000e+02   total_is_add_to_playlist
25  2.738833e+11             total_is_home
26  9.390000e+01       total_is_add_friend
27           NaN              total_is_404
28           NaN              total_is_307
29           NaN              total_is_200
30  2.251800e+15          total_is_windows
31  1.125900e+15        total_is_macintosh
32  9.007199e+15             total_is_ipad
33  7.569075e+13           total_is_iphone
34  2.000000e+00       total_is_compatible
35  7.901052e+13            total_is_linux
36  0.000000e+00             distinc_locs
```

Fig 20: Multicollinearity: VIF results

5.2. Training Models Results:

The below are images are model scores for each model.

- Logistic Regression

```
Best cross-validation score: 76.89
Best parameters:  {'C': 0.0001}


Cross validation, Mean Score metrics for Logistic regression are as follows:

fit_time : 0.5204836527506511
score_time : 0.5205631256103516
test_accuracy : 76.88888888888889
test_precision : 59.12296296296296
test_recall : 76.88888888888889
test_f1 : 66.84453558137768
test_AUC : 48.42426854454896
```

Fig 21: Results of Logistic Regression with f1 score 66.8%

- Random Forest Classifier

```
#Intialize the model - Using GridSearchCV to find the n_neighnors hyperparameters
RF_model = RandomForestClassifier()

#Data Partition
param_grid = {'n_estimators': [2,3,4,5,6,7]}
cv = StratifiedKFold(n_splits=3, random_state=0, shuffle=True)
grid = GridSearchCV(RF_model, param_grid, cv = cv, return_train_score=False)
grid.fit(X, y)

print("Best Parameter: {}".format(grid.best_params_))
print("Best Cross Vlidation Score: {}".format(grid.best_score_))


#Run Best Estimator model
bestModel_RF = grid.best_estimator_

scores = cross_validate(bestModel_RF,  X, y, scoring=scoring, cv=3,
                        return_train_score=False)

print("Mean Score metrics for Random Forest Classifier are as follows:\n")

for key, value in scores.items():                    # Obtain mean or median values of each performoce metrics
    print("{} : {} ".format(key,np.mean(value)*100))
```

```
Best Parameter: {'n_estimators': 2}
Best Cross Vlidation Score: 0.7511111111111112
Mean Score metrics for Random Forest Classifier are as follows:

fit_time : 0.5207935969034831
score_time : 0.5207141240437825
test_prec : 70.23118580765639
test_rec : 75.55555555555556
test_f1 : 70.50185516976167
test_AUC : 46.58309246935986
```

Fig 22: Results of Random Forest Classifier with f1 score 70.5%

- MLP Classifier – Neural Networks

```
Mean Score metrics for MLP Classifer model are as follows:

fit_time : 1.6907533009847004
score_time : 0.664830207824707
test_prec : 59.12296296296296
test_rec : 76.88888888888889
test_f1 : 66.84453558137768
test_AUC : 50.0
```

Fig 23: Results of MLP Classifier with f1 score 66.8%

- Decision Tree Classifier

```
Best Parameter: {'max_depth': 1}
Best Cross Vlidation Score: 0.7644444444444444


Mean Score metrics for Decesion Tree Classifier are as follows:

fit_time : 0.06647109985351562
score_time : 0.5902290344238281
test_prec : 59.12296296296296
test_rec : 76.88888888888889
test_f1 : 66.84453558137768
test_AUC : 57.00943817736815
```

Fig 24: Results of Decision Tree Classifier with f1 score 66.8%

- AdaBoost Classifier

```
Mean Score metrics for AdaBoost Classifier model are as follows:

fit_time : 5.634419123331705
score_time : 1.6173760096232097
test_accuracy : 68.44444444444444
test_precision : 63.89649887149888
test_recall : 68.44444444444444
test_f1 : 65.75935386994537
test_AUC : 50.442879981864365
```

Fig 25: Results of AdaBoost Classifier with f1 score 65.7%

- Gradient Boost Classifier

```
Mean Score metrics for GradientBoosting Classifier model are as follows:

fit_time : 35.743117332458496
score_time : 0.753339131673177
test_accuracy : 67.55555555555556
test_precision : 65.23701037301888
test_recall : 67.55555555555556
test_f1 : 65.56357137924934
test_AUC : 52.42353310215664
```

Fig 26: Results of Gradient Boost Classifier with f1 score 65.5%

Best Model: Random Forest Trees was found to be the winning model. 70.5 percent of the F1 Score was obtained. The data exploration observation and feature engineering may be more informative and stable with the entire dataset. The model could also be improved.

6. Conclusions:

Let's stand back and consider the entire journey.

With regard to a Spotify music streaming business, we wanted to predict user churn. Each step of the machine learning workflow uses python and SQL. For that, a binary classifier for Churner and Active Users was required. To remove log events without a user ID, we first cleaned the data and looked for any missing values in the dataset. Then, we conducted numerous data analyses to see how different indicators could help in distinguishing between Churned and Active users. Based on whether a user visited the pages for cancellation confirmation and downgrade submission or not, we determined the customer churn indicator. We then retrieved categorical and numerical variables during the features engineering process. In order to do that, we made use of the data exploration's observed indications. We also looked at the previous user activities to indicate the user's behavior prior to the churn event. Additionally, we looked for highly correlated variables and removed them from the dataset.

Finally, using cross validation and grid search to fine-tune the various models, we did model training by trying out a variety of models, ranging from simple to complicated ones. The F1 measure was used to compare their performances.

Some results from the analysis that we want to convey to Spotify Business include:

- Most users are switching from paid to free subscriptions. One of the reasons could be "Errors" and "Roll Adverts" that need to be fixed.
- Users who have been canceled generally utilize the application far less than other users do. To enhance user usage, businesses must draw customers with new ideas.

7. Potential Upgrades:

We might test other models and algorithms. However, in order to have a more accurate model for determining if a customer is likely to churn or not, we would like to perform more extensive data exploration and feature engineering first. In exchange, we would:

- More temporal features that indicate the service consumption over the last N days should be added.
- Apply more SQL/PySpark best practices to improve the data analysis and feature engineering procedures for effective data exploration, model training, and model testing.
- Due to potential statistical discrepancies with the huge dataset, do data exploration on larger batches of data subsets before using the big dataset.
- Performing better hyperparameter tuning for other model algorithms

8. Acknowledgments and References:

I. Multicollinearity: https://www.datasklr.com/ols-least-squares-regression/multicollinearity

II. https://scikit-learn.org/stable/