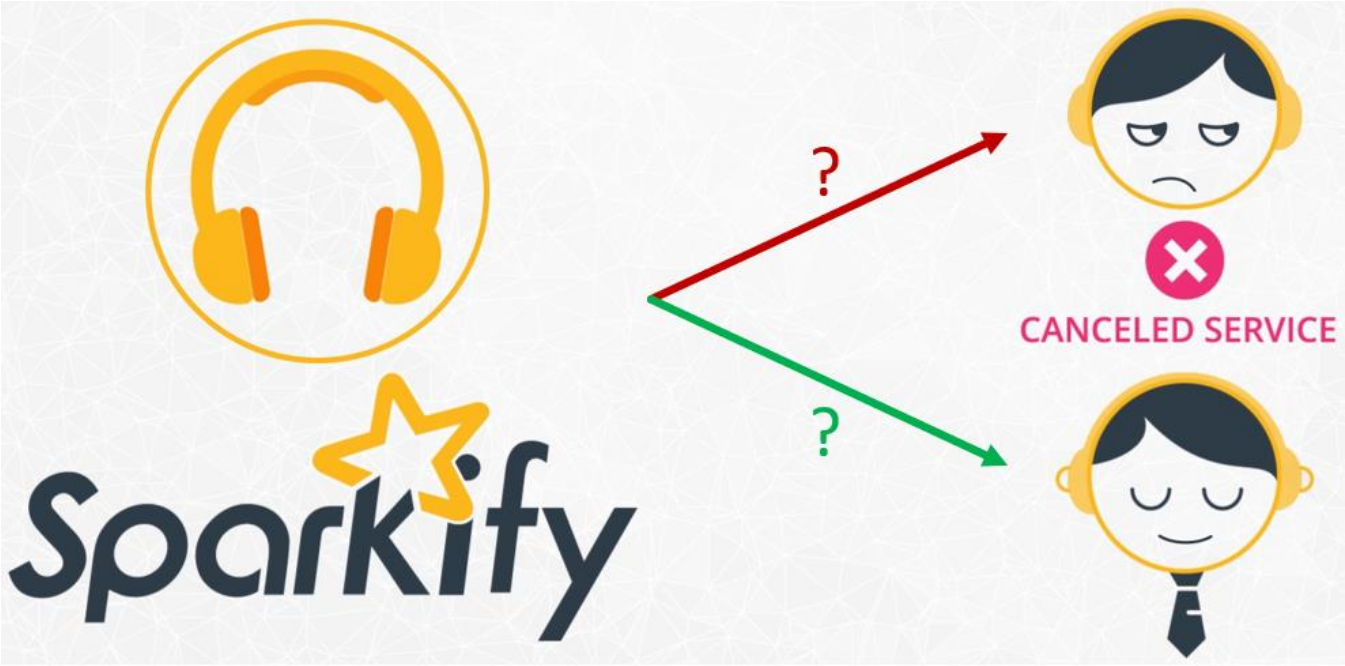
MDSC-304(P)



Mini-Project

Assisting Sparkify music service for churn prediction

Krishnakanth G

20233

Contents

Assisting Sparkify music service for churn prediction

Introduction

Predicting customers' behavior is critical for organizations, but it is also a difficult process. Fortunately, given the correct input, data science may assist in anticipating customers' demands. In this project I used users event data from Sparkify, which is an imaginary digital music service similar to Spotify or Pandora, to build a model to predict users' churn.

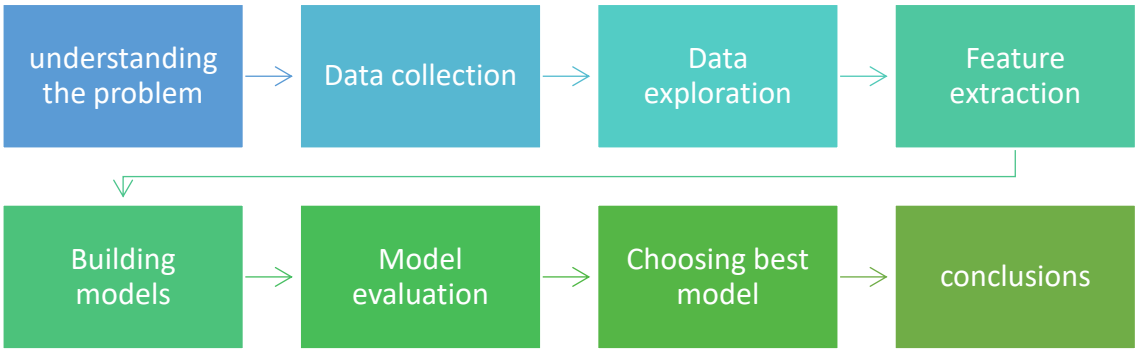
Problem Statement

Problem statement: Assisting Sparkify music service for churn prediction.

In this project, I am going to analyze Sparkify data and create a machine learning model to predict churn. The following tasks should be accomplished to do this project:

- Data cleaning
- Data exploration
- Feature extraction
- Modeling

Scope of the solution



- First, we need to understand the given problem.
- After understanding the problem, we have to collect correct data related to the problem.
- Once we have data in our we need to explore the data.
- After getting enough understanding of the data then extract the required feature from it.

- Then build different models on the data using extracted features.
- Evaluate the models using test data and pick the best out of all.
- Conclude the results based on the model selected.

Challenges

I was primarily confronted with two challenges: memory and processing power. The original data I collected is approximately 12GB; storing and processing it may require more time and resources; however, due to these constraints, I chose one percent of the total data, which is 128MB, which is efficient for creating a model.

Data Description

The following table describes the attributes of the data.

#	Column	Type	Description
1	userId	string	Unique identifier of the user, the event is related to
2	artist	string	Name of the artist related to the song related to the event
3	auth	string	“Logged in” or “Cancelled”
4	firstName	string	First name of the user
5	gender	string	Gender of the user, “F” or “M”
6	itemInSession	Big int	Item in session
7	lastName	string	Last name of the user
8	length	double	Length of the song related to the event
9	level	string	Level of the user’s subscription, “free” or “paid”. User can change the level, so events for the same user can have different levels
10	location	string	Location of the user at the time of the event
11	method	string	“GET” or “PUT”
12	page	string	Type of action: “Next Song”, “Login”, “Thumbs Up” etc.
13	registration	Big int	Registration number
14	sessionId	Big int	Session id

15	song	string	Name of the song related to the event
16	status	Big int	Response status: 200, 404, 307
17	ts	Big int	Timestamp of the event
18	userAgent	string	Agent, which user used for the event, for example, “Mozilla/5.0”

Data Cleaning

The following is the count of null values in each attribute in the data:

Attributes	**Null values**
artist	58392
auth	0
firstName	8346
gender	8346
itemInSession	0
lastName	8346
length	58392
level	0
location	8346
method	0
page	0
registration	8346
sessionId	0
song	58392
status	0
ts 0	0
userAgent	8346
userId	8346

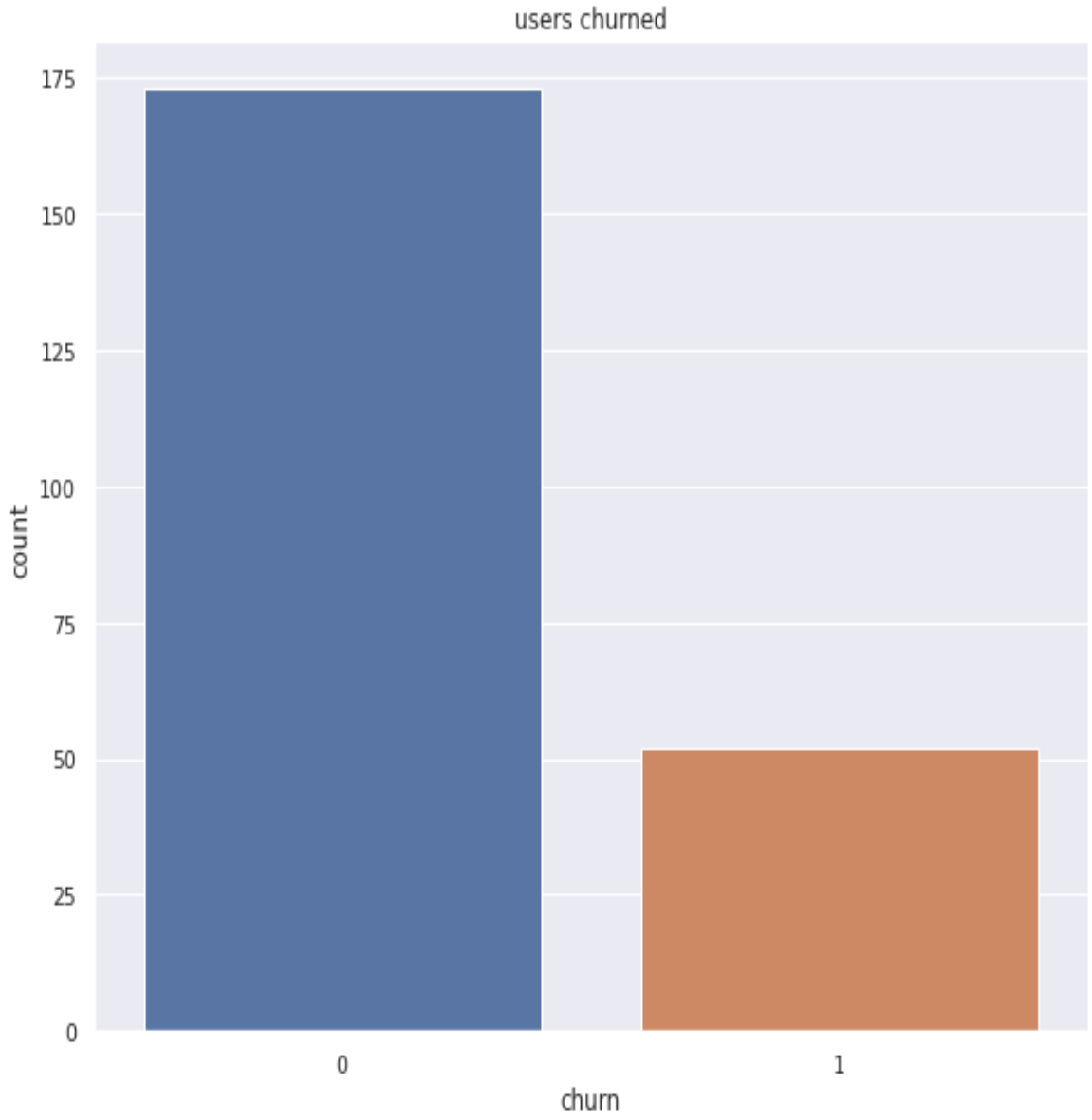
First, I, dropped null values using subset as userId and stored the data frame in df_clean. The cleaned data has 278154 records and 18 attributes. After cleaning I, started Exploring the data

and created few attributes from existing attributes that are event time, registration time, state location, downgrade event, and churn.

Data Exploration

In this section, I am exploring most of the variables to get more clarity on data.

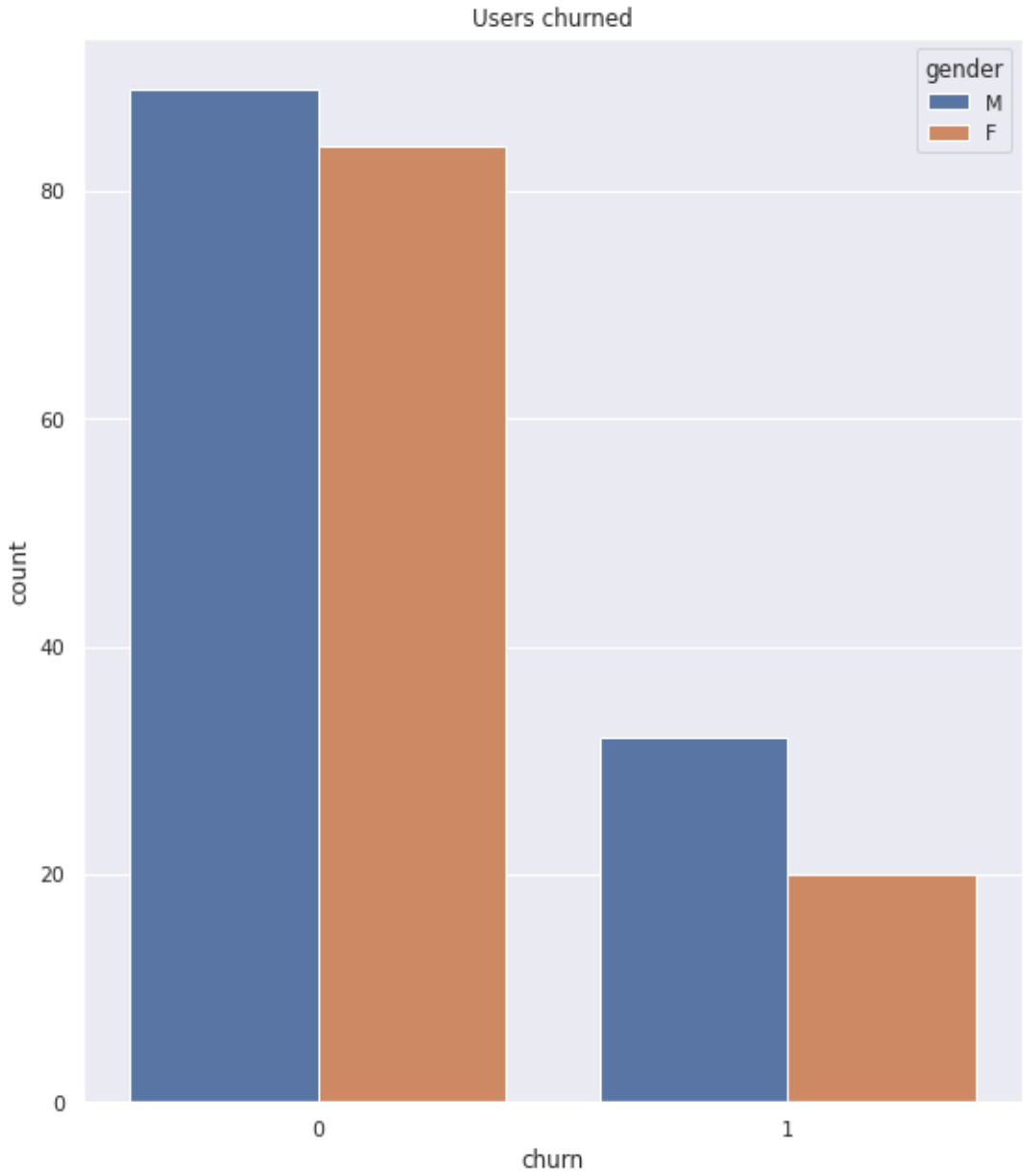
Churn: It determines whether a user stay back or cancel an event. 1 if user is cancelling the event, else 0.



From the above plot, we can observe that the following:

- Around 170 users are retaining the events.
- Around 50 users are cancelling events.

Following is the plot that represents churn counts grouped by gender,

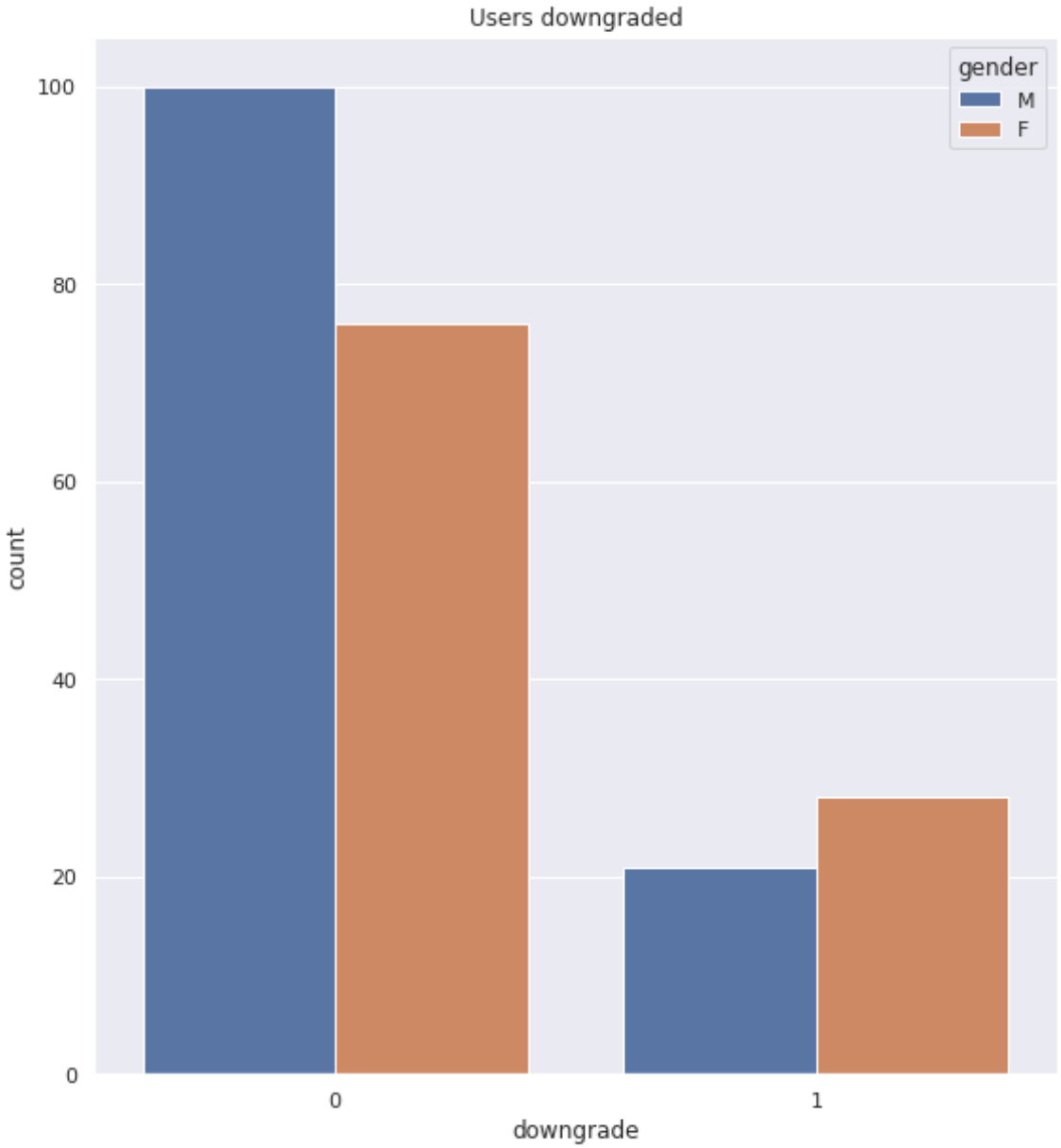


From the above plot, we can observe the following:

- Around 90 users from male and 85 users from female are retaining the events.
- Around 32 users from male and 20 users from female are cancelling events.

Downgrade: It determines whether a user downgrades an event or not. 1 if user is downgrading the event, else 0.

Following is the plot that represents Downgrade counts grouped by gender,

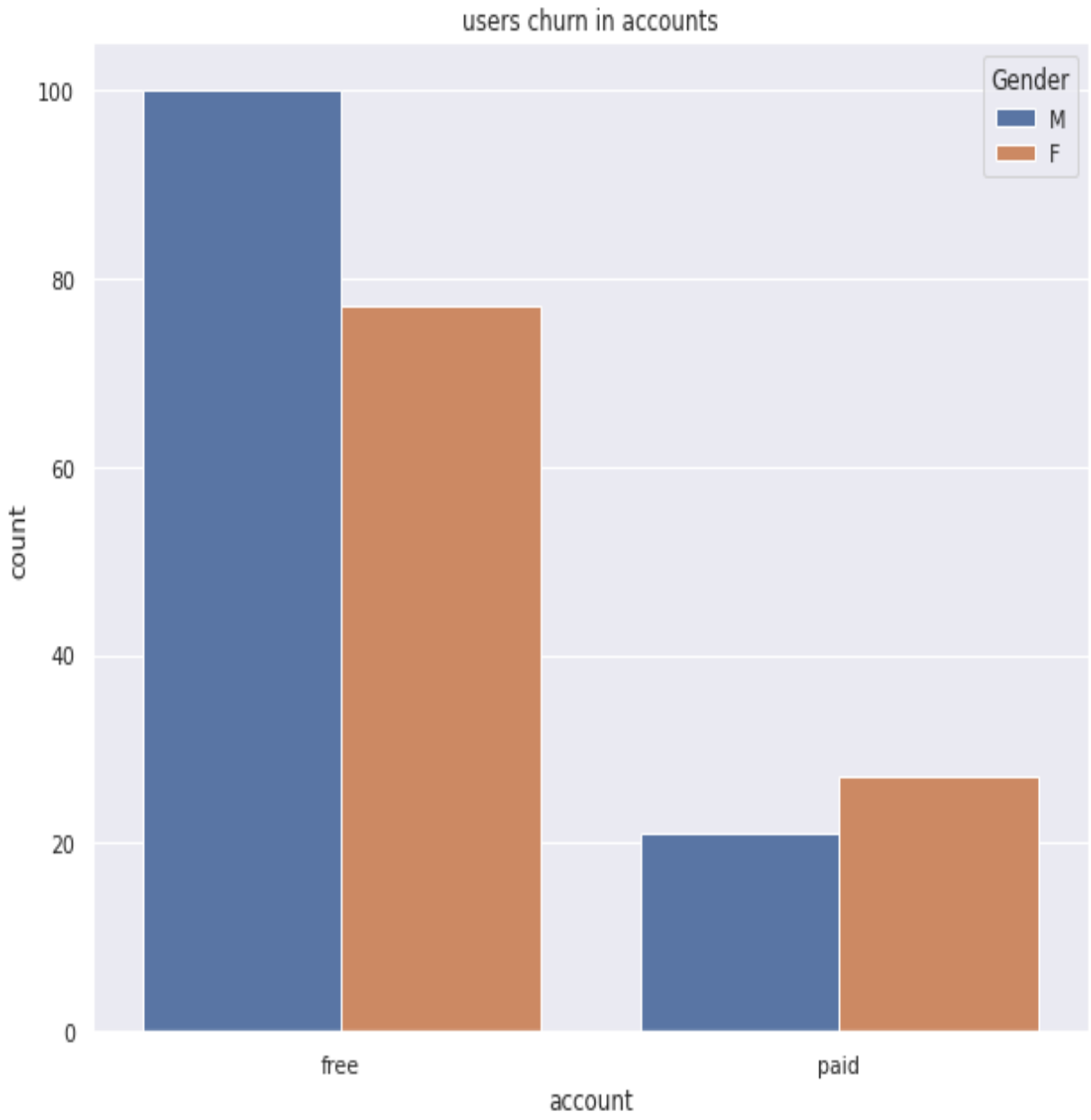


From the above plot, we can observe the following:

- Around 100 users from male and 75 users from female are downgrading event.
- Around 20 users from male and 28 users from female are not downgrading the event.

Level: Level of the user's subscription, "free" or "paid". User can change the level, so events for the same user can have different levels.

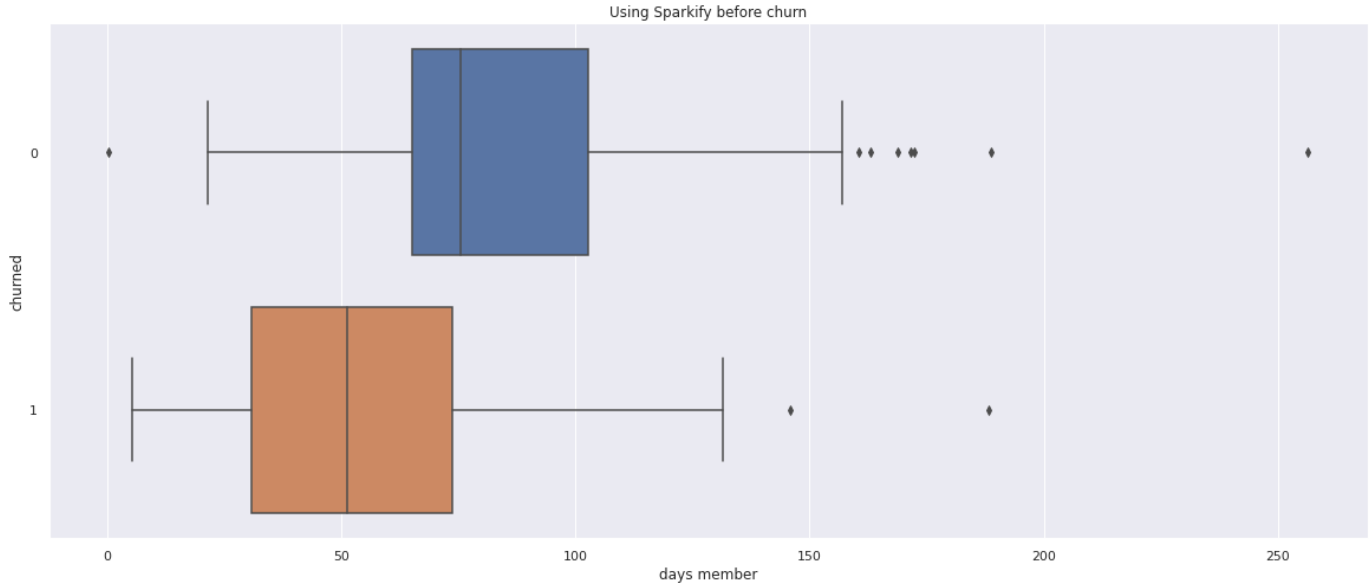
Following is the plot which has users' level (i.e., free & paid) and users churn grouped by gender,



From the above plot, we can observe the following:

- Around 100 users from male and 88 users from female are retaining the events in free level.
- Around 20 users from male and 25 users from female are cancelling events in paid level.

Life time is an attribute created by using registration date and event time stamp. Following plot shows the users churn after some life time that is number of days,

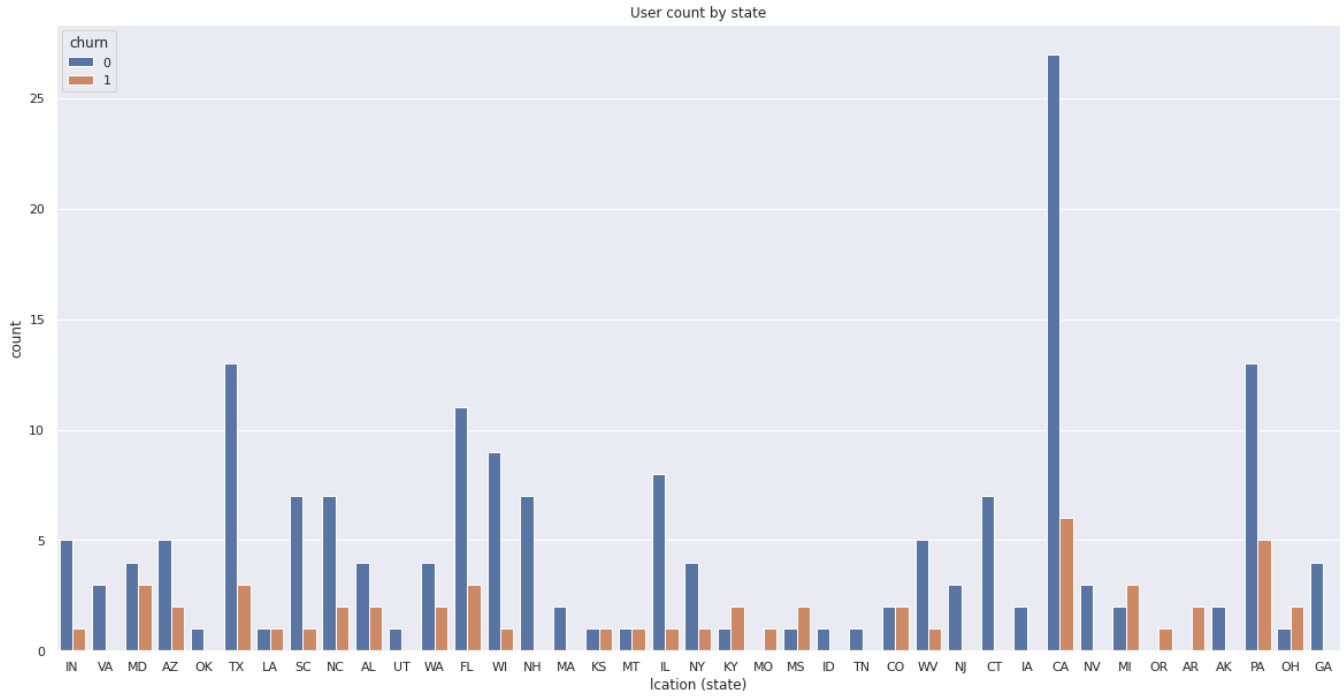


From the above plot, we can observe the following:

- Median of users retaining the event is 75 days. It is having many outliers and positively skewed.
- Median of users cancelling the event is 50 days. It is having few outliers and looks like normal.

Location: Location of the user at the time of the event.

Following is the plot showing users count by each location or state in USA,

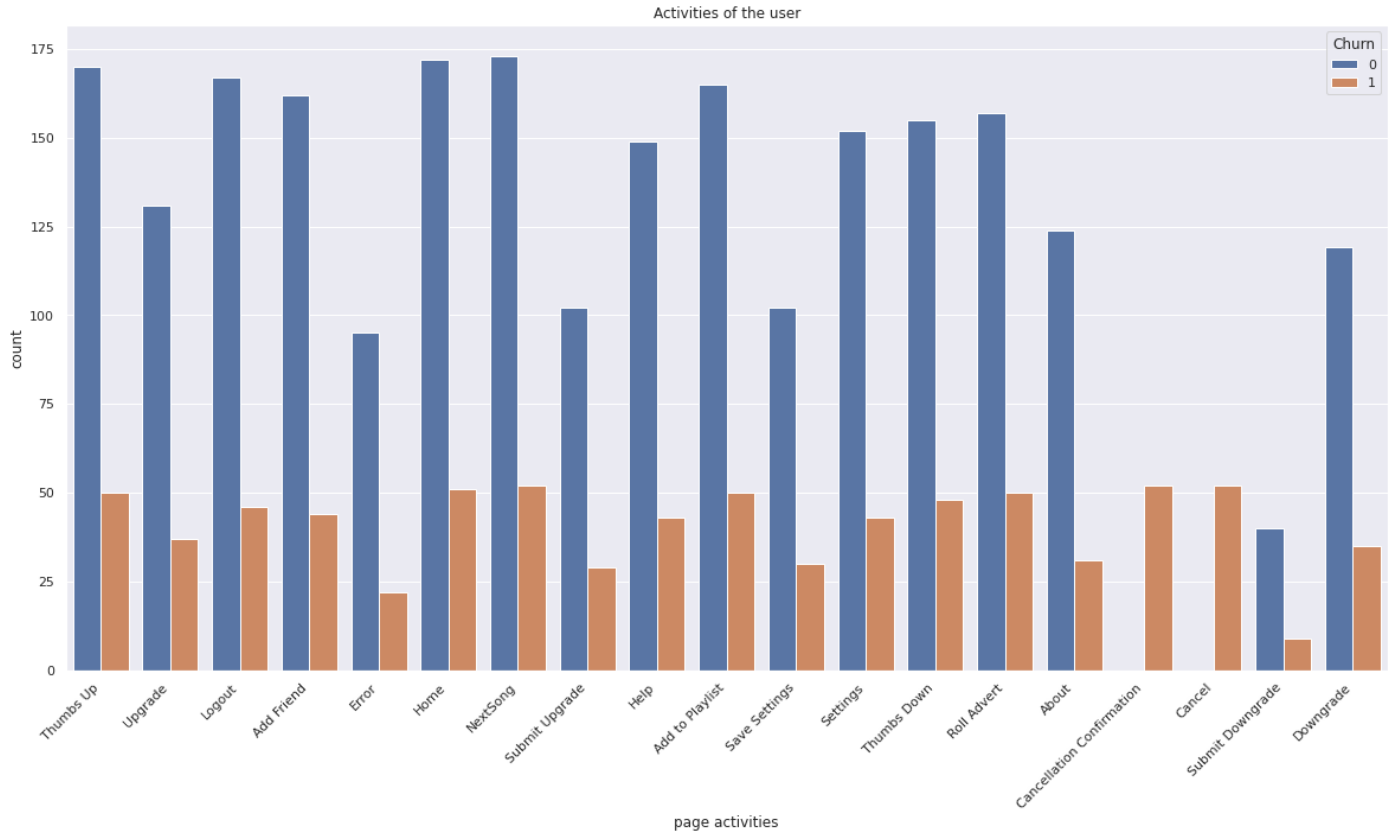


From the above plot, we can observe the following:

- The retaining rate is high compared to canceling rate in many of the states.
- In some sates retaining and canceling rates are almost same
- In very few sates canceling rate is high.

Page: It contains the different activities done by user in an event. Type of activity: “Next Song”, “Login”, “Thumbs Up”, etc.

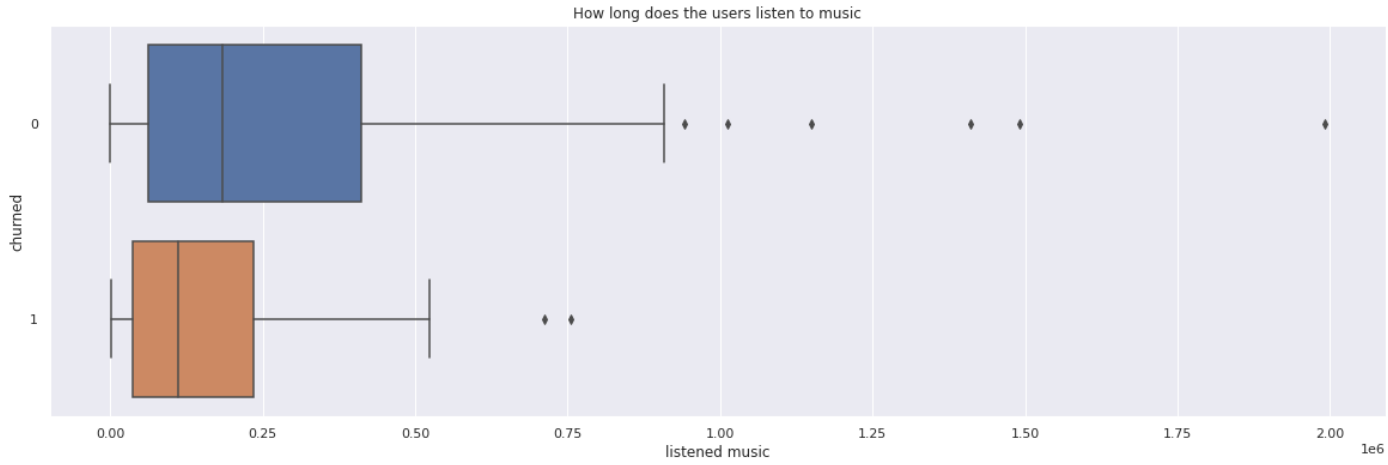
Following is the plot that shows users churn according to activities done by users in the events,



From the above plot, we can observe the following:

- The retaining rate is high compared to canceling rate in activities like upgrade, thumbs up, etc.,
- Most expected thing is users who cancels events are not going to retain so there are no users (Zero) who are retaining events in cancel conformation and cancel activities.
- The odd thing is most of the users who downgraded the event still continued to retain the events.
- All other activities we have many users retaining because they are satisfied with events.

Following is the plot that shows users churn according to time length of music in the events,



From the above plot, we can observe the following:

- Median of users cancelling the event is 33 minutes. It is having few outliers and positively skewed.
- Median of users retaining the event is 56 minutes. It is having many outliers and positively skewed.

Feature Engineering

During my exploration of the data, I discovered some important features that can be derived from the existing attributes, which are as follows:

- gender: whether a user is male (1) or female (0)
- level: whether a user is paid (0) or free (1)
- downgrade: 1 if user don't like the event else 0
- churn (label): 1 if user cancelled the event else 0
- num_songs (count): Number of songs by a user
- num_thumbs_up: Number of Thumbs up by a user
- num_thumbs_down: Number of Thumbs down by a user
- num_playlist: Number of songs a user added to his playlist
- num_friends: Number of friends a user added
- sum_listened: Length of listened songs by a user
- avg_song_session: Average number of songs in the all sessions for a user

- num_artists: Number of artists a user following
- days_member: Member of days a user stayed in event
- num_session: Number of sessions by a user
- dur_session: Duration of a session

Design of solution

I formed a data frame using the features extracted above. There are 3176 records and 15 attributes in this data frame. we have one dependent variable, churn, and fourteen independent variables among the 15 attributes. These fourteen independent variables are being used to construct machine learning models.

I divided the dataset into training (80 percent) and testing (20 percent) sets, and built a pipeline that assembles the above-mentioned features, scales them, and then runs a machine learning model. The following are the classification models I used:

- **Logistic regression**
- **Decision tree**
- **Random forest**
- **Gradient boost tree**

Metrics: Because we are interested in both precision and recall, I believe F1-score is an appropriate metric to use to assess the performance of our machine learning classifier.

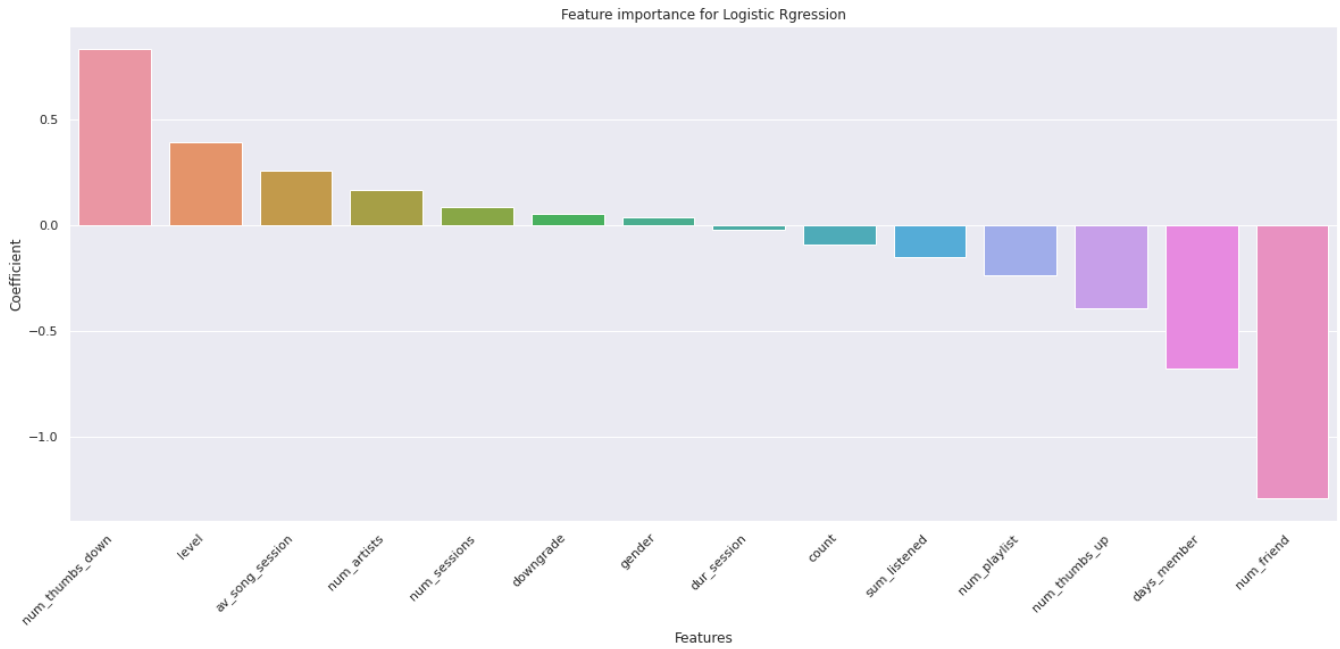
Models implementation

Logistic regression

Logistic regression is a supervised learning algorithm used in machine learning to predict the probability of a binary outcome. A binary result can only have one of two possible outcomes. Yes/no, 0/1, and true/false are some examples. In this case, we have fourteen independent variables to predict the outcome that is whether a user is going to churn or not. The model is producing the following results,

- Accuracy: 0.85
- Precision: 0.88
- Recall: 0.14
- F1-Score: 0.25

The model gave the feature importance graph as below,



From the above plot, we can infer the following

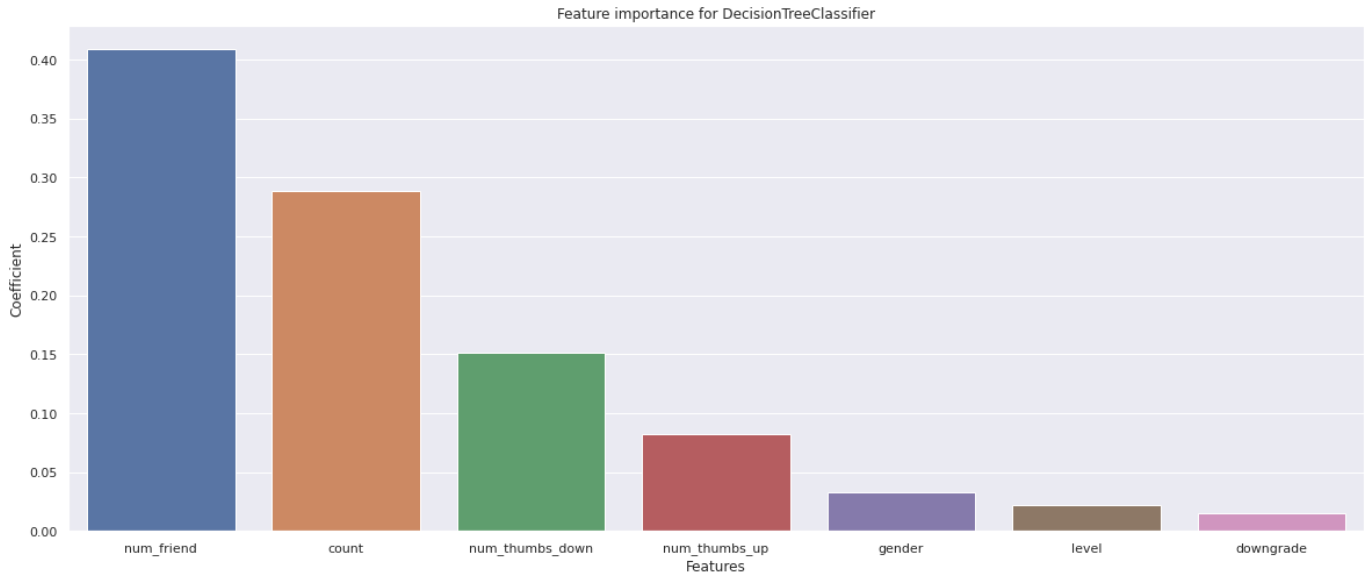
- Features that are seemingly most relevant of non-churned users are:
  - num_friend – the more friends you added, the less likely you are to churn
  - days_member – the more days you stay, the less likely you are to churn
- Features that are seemingly most relevant of churned users are:
  - num_thumbs_down – a good indicator representing how much users don't enjoy the service
  - Level – paying users seem more likely to churn than non-paying ones

Decision tree

Decision Trees are a type of Supervised Machine Learning in which data is continuously split based on a parameter. Two entities, decision nodes and leaves, can be used to explain the tree. The decisions or final outcomes are represented by the leaves. And the data is split at the decision nodes. The model is producing the following results,

- Accuracy: 0.95
- Precision: 0.95
- Recall: 0.74
- F1-Score: 0.83

The model gave the feature importance graph as below,



From the above plot, we can infer the following

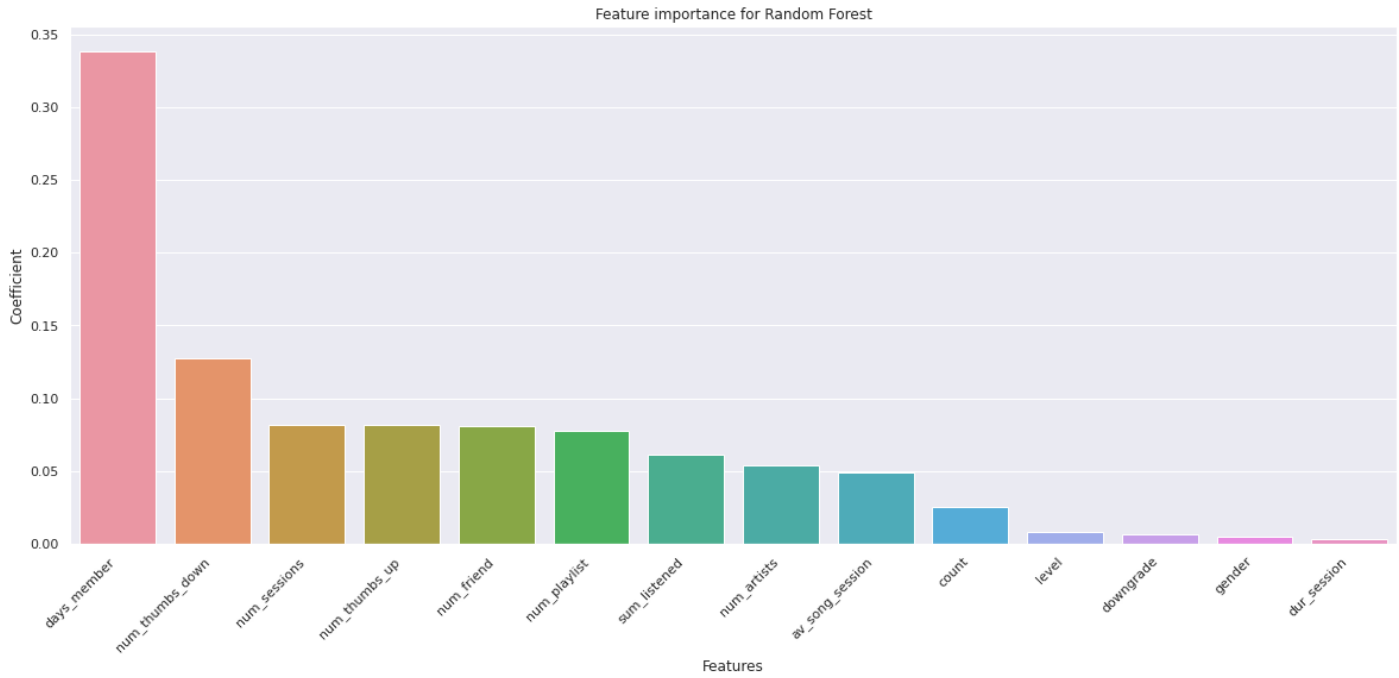
- We have two most important features that are number of friends and number of songs in this model
- We have some features like number of thumbs up and down with medium importance
- All others are having very less importance

Random forest

Random forest is a consensus algorithm used in supervised machine learning to solve regression and classification problems. Each random forest is comprised of multiple decision trees that work together as an ensemble to produce one prediction. The model is producing the following results,

- Accuracy: 0.94
- Precision: 1.0
- Recall: 0.65
- F1-Score: 0.79

The model gave the feature importance graph as below,



From the above plot, we can infer the following

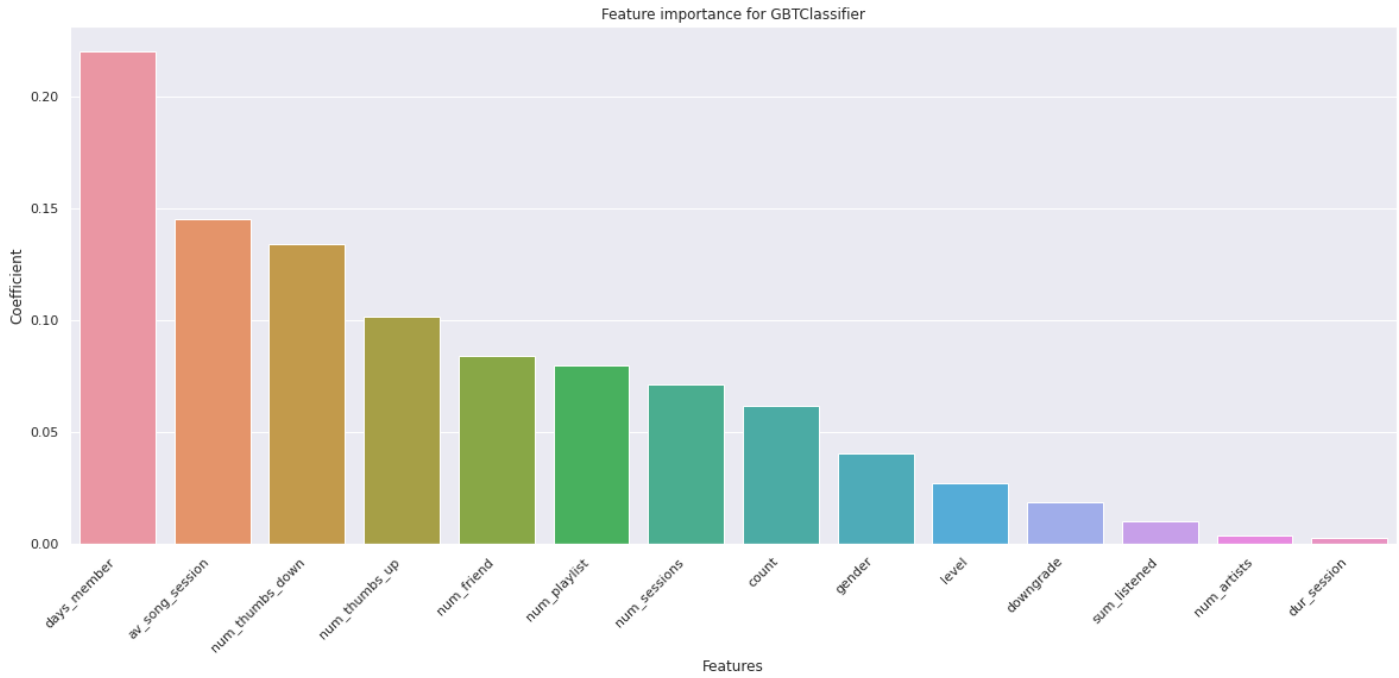
- We have days member as important feature in this model
- We have some features like number of thumbs up and down, number of sessions, number of friends, number of playlists, etc. with medium importance
- All others features are having very less importance

Gradient boost tree

Gradient boosting is a machine learning technique used in regression and classification tasks, among others. A gradient-boosted trees model is built in a stage-wise fashion as in other boosting methods, but it generalizes the other methods by allowing optimization of an arbitrary differentiable loss function. The model is producing the following results,

- Accuracy: 0.98
- Precision: 1.0
- Recall: 0.9
- F1-Score: 0.95

The model gave the feature importance graph as below,



From the above plot, we can infer the following

- We have days member as most important feature in this GBT
- We have some features like number of thumbs up and down, average number of songs in the sessions, with little bit high importance compared to others
- Number of friends, number of sessions, number of songs and number of playlists are having average importance
- All others features are having very less importance

Results and Analysis

Below is the table formed from the four models and their corresponding metrices

CLASSIFIER	ACCURACY	PRECISION	RECALL	F1-SCORE
LOGISTIC REGRESSION	0.85	0.88	0.14	0.25
DECISION TREE	0.95	0.95	0.74	0.83
RANDOM FOREST	0.94	1.0	0.65	0.79
GRADIENT BOOST TREE	0.98	1.0	0.90	0.95

From all of the models, we can see that the Gradient boost tree is the best because its accuracy and F-1 score are high and close to each other, indicating that it is doing well. Other models provide the most accuracy, but when it comes to the F-1 score, they fall short. As a result, the optimal model is the gradient boost tree.

Conclusion

In this project I learnt that, if we have limited resources, add more data rather than fine-tuning the weights on our machine-learning algorithm. Balanced sampling can produce the approximate results as whole data. choosing the right metric for evaluation of the models. Apart from this, I have learned how to infer from feature importance graphs.

I selected the F-1 score as a metric because we require both precision and recall. We can see that all models provide good accuracy, but when it comes to F-1 scores, they fall short. Only the Gradient boost tree has a high accuracy and F1-score, indicating that this model is optimal. We can now apply the model to our entire 12GB data set.

References

https://medium.com/@olivier.klein/sparkify-udacity-data-scientist-nanodegree-capstone-project-65e3181ea2b0

https://media.githubusercontent.com/media/angang-li/sparkify/master/mini_sparkify_event_data.json