UCL

UCL
SCHOOL OF
MANAGEMENT

**MSc Business Analytics (with specialisation in Management Science)**
**Predictive Analytics MSIN0097 2021-22**

# Predicting the success of artists on Spotify


## Author's candidate numbers:
QSGV8
QXSB1
RKXS1
TDYS5


A report is submitted in partial fulfilment of the requirements for the module of predictive analytics MSIN0097.

The authors confirm that the work presented in this report is our own work. Where information has been derived from other sources, we can confirm that this has been indicated in the work.

Google Colab link for the Notebook file:
https://colab.research.google.com/drive/1Fe94BV082O_hazYx3CVcBVuFrJ3gCbdd?usp=sharing

**Group Number: 15**

**Date: 24/03/2021**

**Word Count: 2000**

# Contents

# Figures



# Tables

# 1. Framing the problem

Spotify is one of the biggest music streaming platforms in the world which connects established and up-and-coming artists with listeners worldwide. Spotify playlists are one of the key channels connecting the listeners with music they want to hear. Inclusion in popular playlists will immediately expose an artist to a wider audience, in turn leading to higher streaming levels. Subsequently, playlists are an important strategic asset for artists. The inclusion of an artist in a playlist can be the differentiating factor between success and failure.

The purpose of this project is to identify the most impactful metrics that maximise an artists' chances of success ceteris paribus.

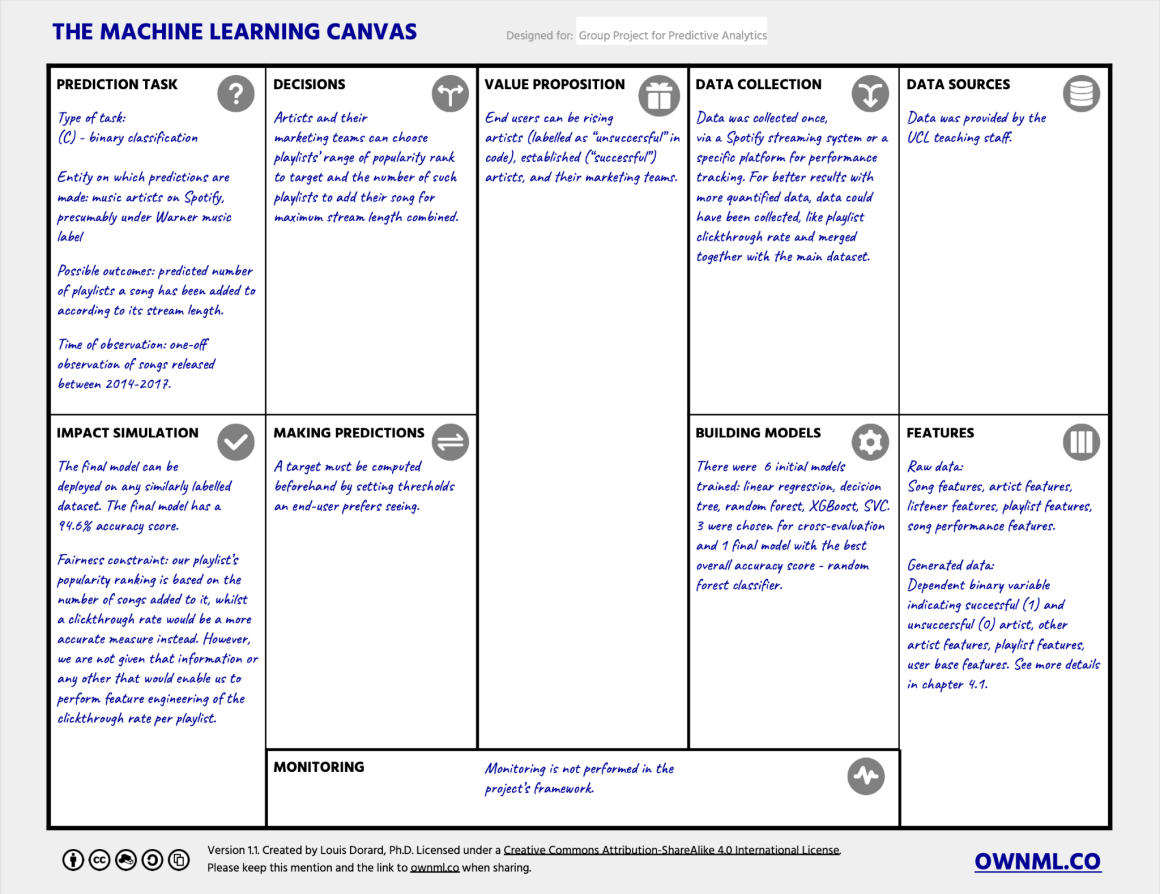


***Figure 1*** *Machine learning canvas*

# 2. Data cleaning

The main dataset was composed of 3.8M rows and 44 columns, where each row represents an instance of a track being listened to by a user. For example, this contains information on the track name, the artist name, how the track was accessed, and what user accessed it. The *region_code* column was dealt with as follows: the try-except method was used to convert strings of the column to an integer type, and for the rare countries where the conversion did not succeed (due to code containing letters), they were converted manually to UN codes, so that country codes were consistent throughout. Columns that were deemed irrelevant for future analysis were dropped.

The second data set used is *playlist_df*. It consists of 1.9M observations and two variables: playlist id and playlist name. The number of rows in the two dataframes is different, hence, they could not be concatenated horizontally, so it was explored how they could be merged instead. Since there were multiple "id"s for the same name of a playlist, it was decided to ignore the playlist dataframe and only use the main one, which is addressed as "df" in the Jupyter notebook.

Lastly, duplicates and null values were dropped to prepare the data for machine learning methods.

# 3. Exploratory data analysis

Platform access:

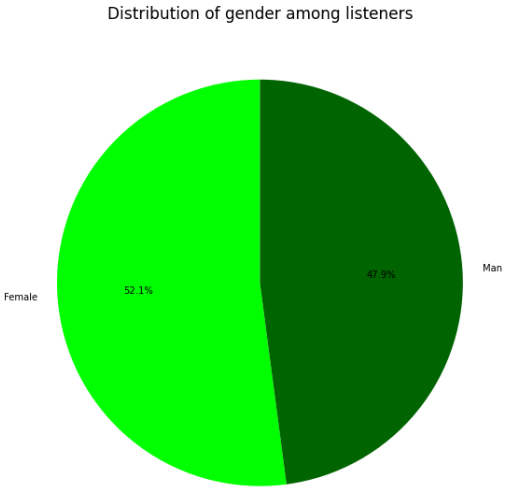


***Figure 3.1*** *Distribution of gender*

The gender split is quite even. Female users slightly outnumber male users by 4.2% (fig. 3.1).

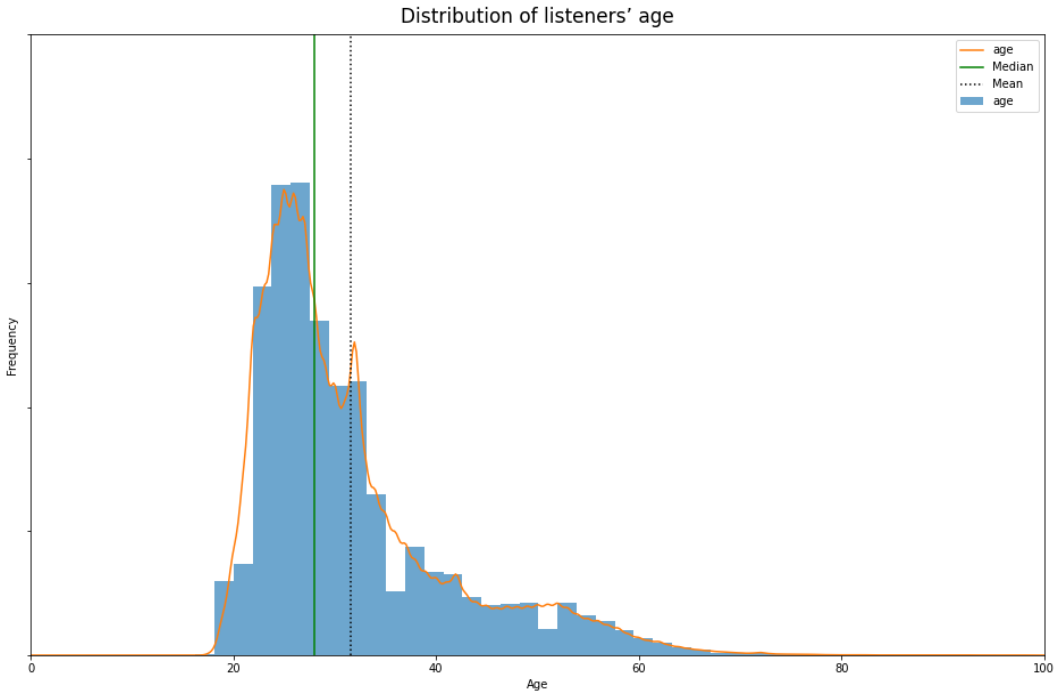


***Figure 3.2*** *Distribution of listeners' age*

Spotify listeners' age follows a Poisson distribution with the majority aged 20-35. The mean is slightly higher than the median age value, thus suggesting that the data is slightly right-skewed, i.e. listeners' age is skewed to higher values (fig. 3.2).

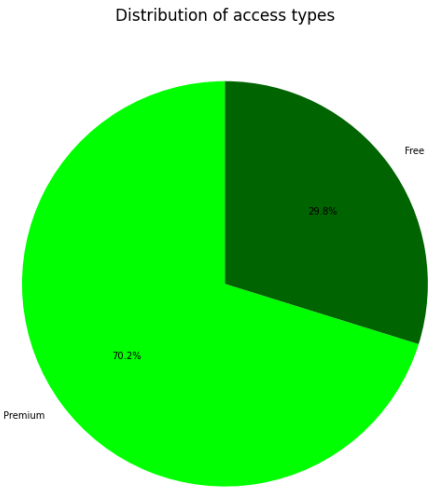


***Figure 3.3*** *Distribution of access types*

70.2% of the streams in the dataset come from the paid “Premium” accounts, whilst only 29.8% are from accounts that are free (fig. 3.3).

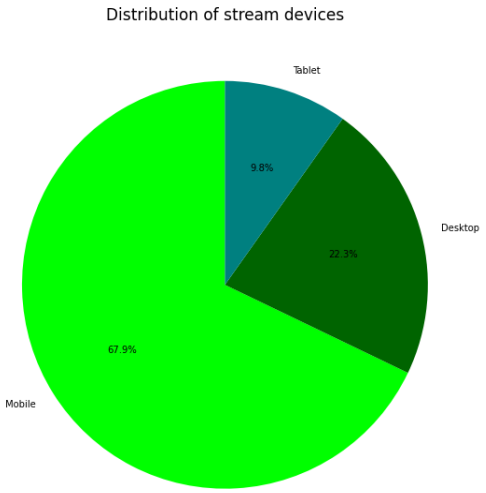


***Figure 3.4*** *Distribution of stream devices*

The most popular way to stream on Spotify is via mobile. This accounted for 67.9% of the observations. Mobile streaming is three times more popular than using a desktop – 22.3%. Tablets are the least used devices for streaming at only 9.8% (fig. 3.4)

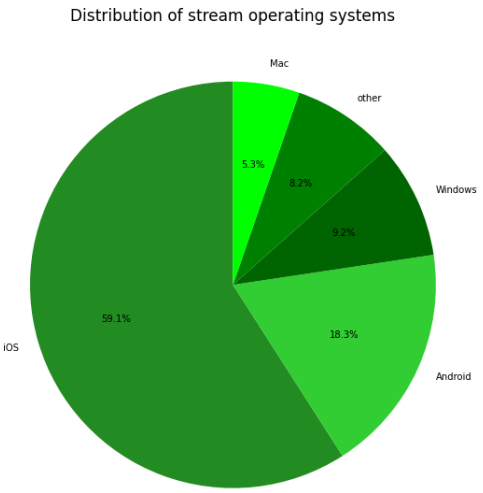


***Figure 3.5*** *Distribution of stream operating systems*

iOS is used by 59.1% of the people and is the most common operating system in the dataset (fig. 3.5). All other operating systems add up to 40.1% of the total: one third less than iOS usage. Overall, the second most common operating system is Android (18.3%), followed by Windows (9.2%), Mac (5.3%) and other smaller platforms (8.2%).

Creatives:

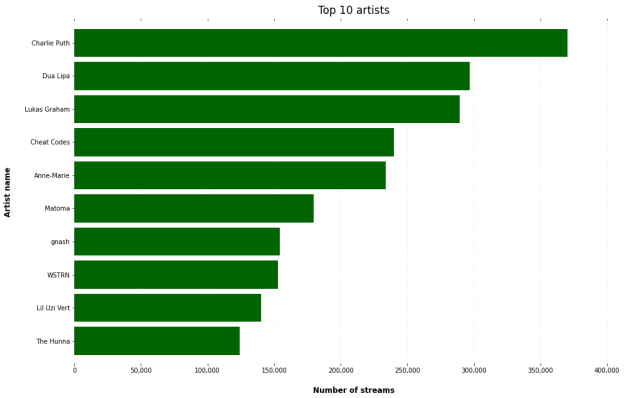


***Figure 3.6*** *Top 10 artists*

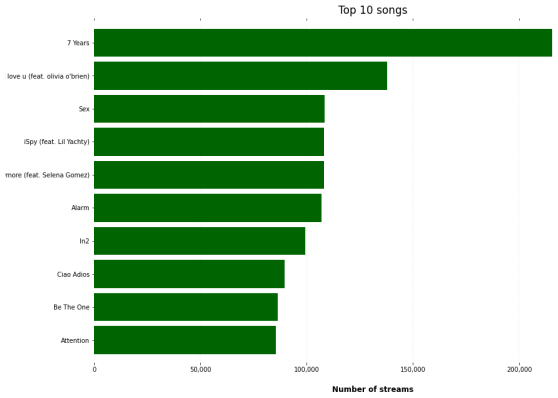


***Figure 3.7*** *Top 10 songs*

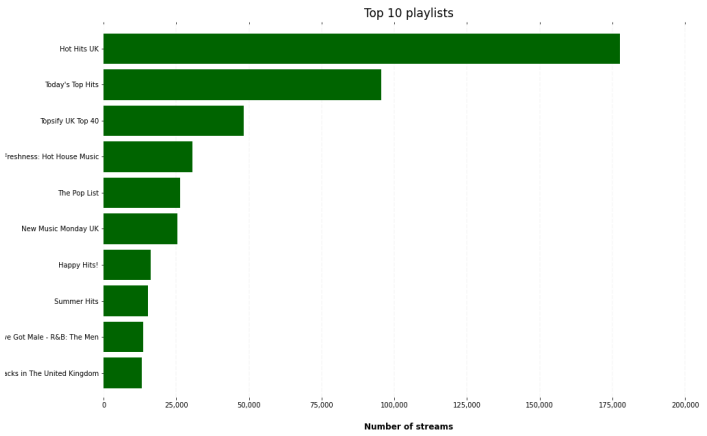


***Figure 3.8*** *Top 10 playlists*

The top 10 artists, songs and playlist names are presented in descending order by the total number of streams (figs. 3.6-8).

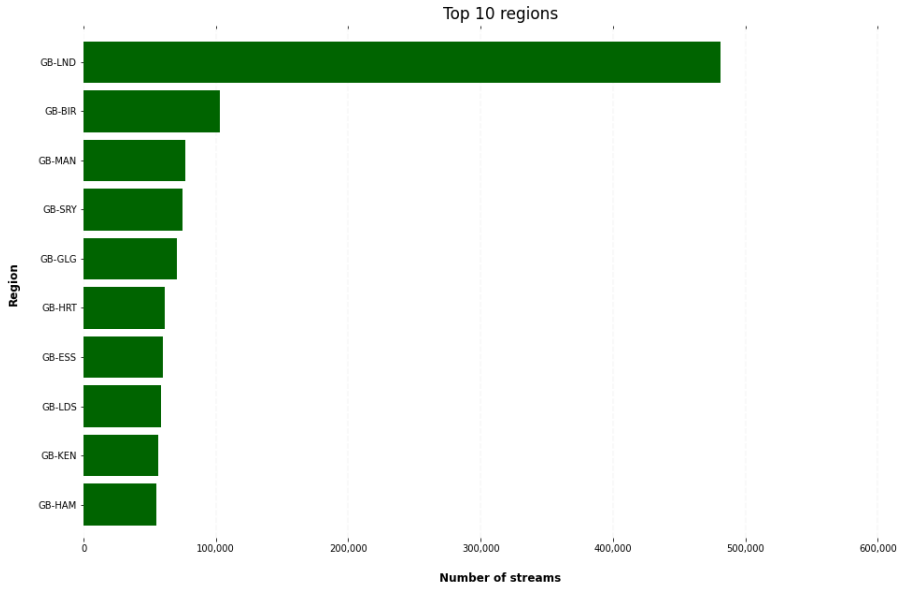


***Figure 3.9*** *Top 10 regions*

London has the most streamed tracks by a very large margin (fig. 3.9). The second biggest region for streaming is Birmingham and the third is Manchester. The difference in streams is biggest between London and Birmingham, then onwards the number of tracks streamed is gradually decreasing. This is likely because London has the biggest population of all cities in the UK– currently 9,540,000 live in the capital (World Population Review, 2022) which amounts to 14% of the total UK population.

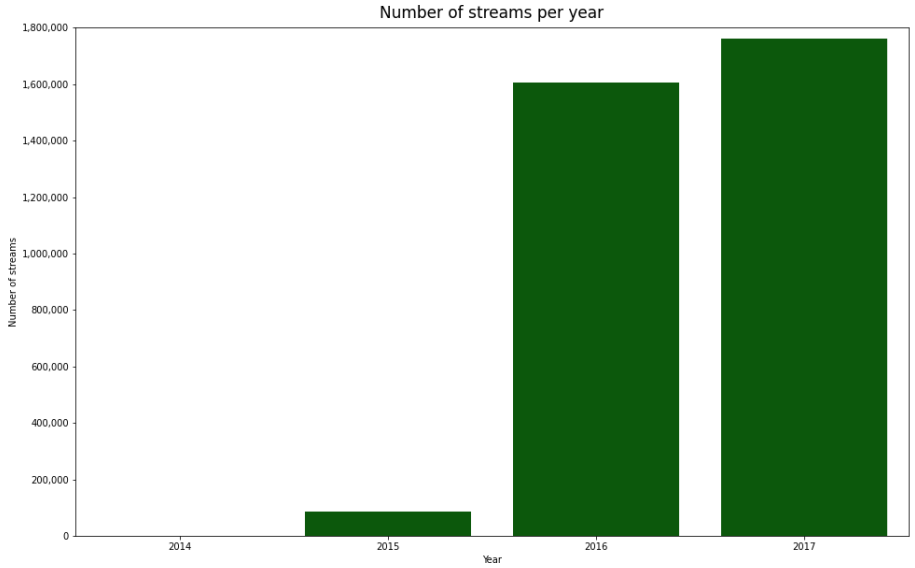


***Figure 3.10*** *Number of streams per year*

Number of streams per year (fig. 3.10) can be split into two parts – 2014/2015 have very few observations whilst 2016/2017 have much higher numbers and account for the vast majority of the total. This pattern is unlikely to be due to a sudden rise in the popularity of Spotify in 2016, and more probably arises from an increase in data collection for 2016/2017 compared to 2014/2015.

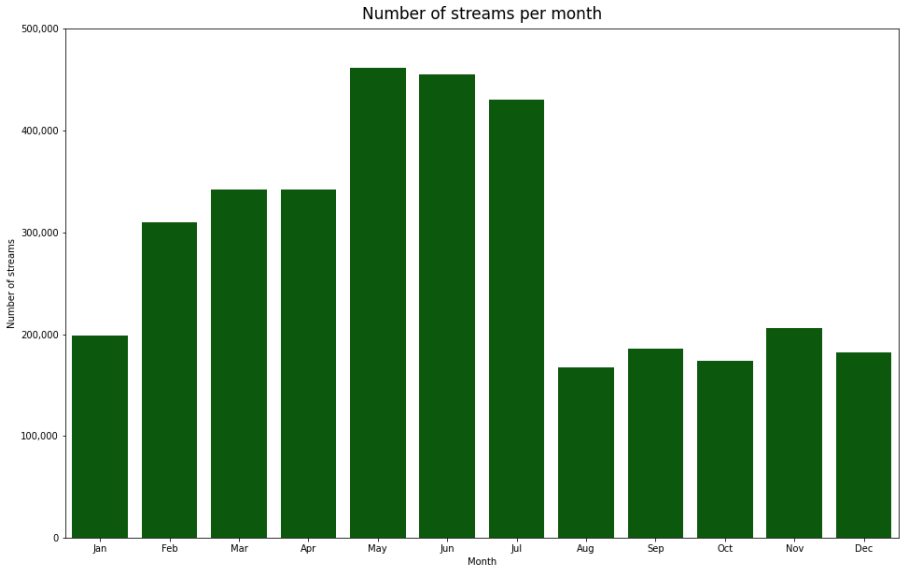


***Figure 3.11*** *Number of streams per month*

Number of streams per month (fig. 3.11) allows seasonal trends to be examined. Streams start off with 200,000 in January,  May-July has the highest number of total track streams at 400,000-450,000, and then there is a drop off where August-December numbers plateau at about 200,000 per month again. While this suggests summer is the most popular time for music streaming on Spotify, this might also be due to variances in data collection methods or collection efficacy over time.

# 4. Data pre-processing

Data pre-processing started by creating a dependent variable – the future target for the classification task. The new feature is a boolean type that identifies an artist as successful (1) or not (0) based on whether the artist_name was featured in one of the four most popular playlists by stream length: 'Hot Hits UK', 'Massive Dance Hits', 'The Indie List', 'New Music Friday UK'.

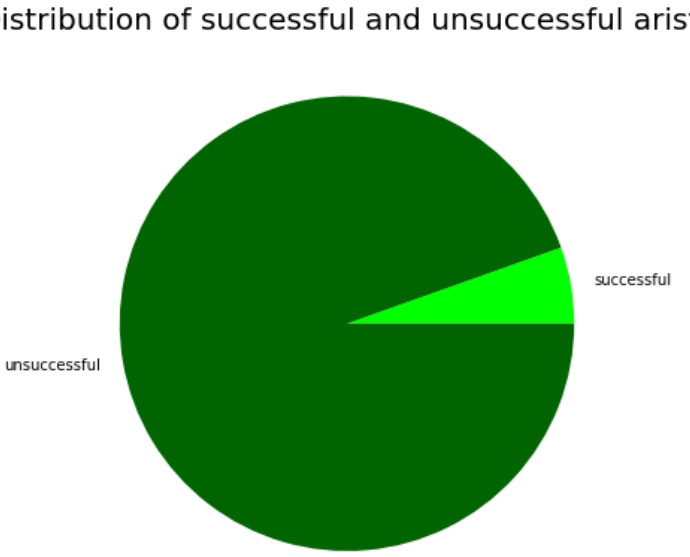


***Figure 4*** *Distribution of successful and unsuccessful artists*

Then, three artist features were created: stream count, the total number of users, and passion score for an artist measuring how dedicated the artist's audience is. This is followed by playlist features, including the same three features as for artists, as well as artist frequency (measuring how many times a specific artist appears in a playlist) and playlist variation (measuring how many unique tracks are included in a particular playlist). The main assumption was that if the "playlist_id" column is null, the song is not included in any playlist. Lastly, gender percentage breakdown and age vector quantization were performed as part of user-based feature engineering. Overall, 10 new features were created.

Next, it was checked which columns remain non-numerical. Using the Info() function revealed that only the artist_name column was an object data type, so this was encoded into integers using LabelEncoder().

Finally, the response variable 'successful_artist' was specified and assigned to the variable Y. The rest of the features were assigned to X, scaled using RobustScaler() to make them more robust to outliers, and the dataset was split into train and test sets. The dataset was stratified on the Y variable to account for the imbalance in classes.

# 5. Model training and comparison

The worst performing model was the decision tree with a harmonic mean equaling 0.88. The second worst was the SVM model with a 0.91 f1-score. Logistic regression, KNeighborsClassifier and XGBoost performed about the same with a 0.93 accuracy score. Random forest performed the best with 0.95 accuracies and 0.94 weighted average precision score.

After training the models, they were compared by plotting confusion matrices alongside each other (fig. 5). Random forest is truly performing as the best model with 86.82% true values for unsuccessful artists and 7.75% true values for successful values. These metrics are normal in this particular case, as the data is severely imbalanced, with much fewer instances in class 1 (successful artists). Therefore the models are better at identifying unsuccessful artists because most artists, in general, are considered unsuccessful.

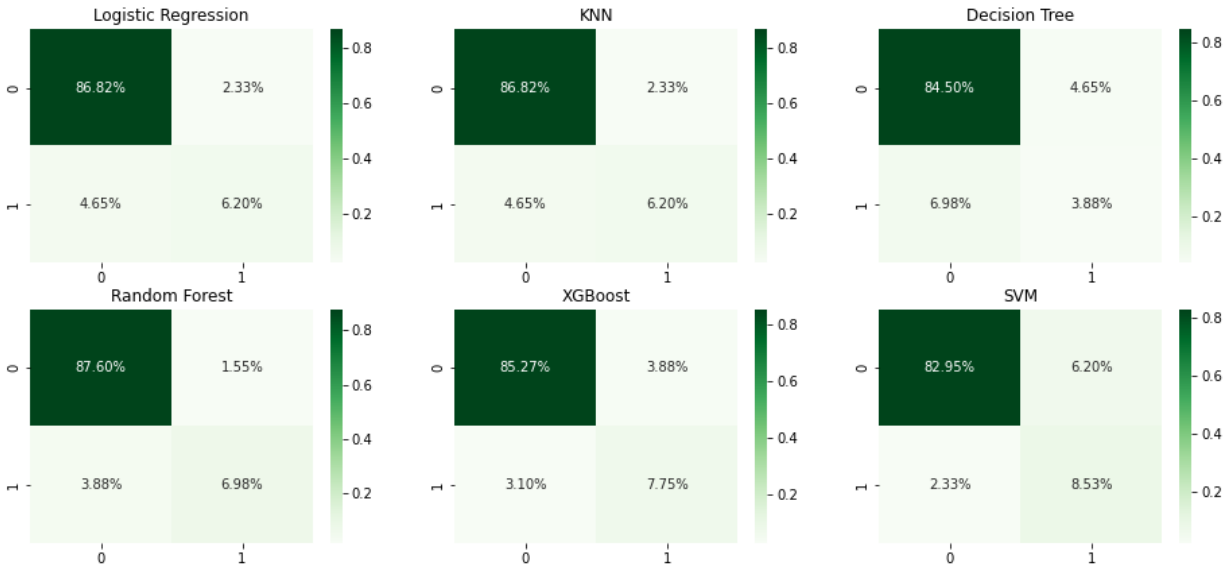


***Figure 5*** *Confusion matrix heatmaps of trained models*

# 6. Cross-validation

Cross-validation is used on the six machine learning models to shortlist the best-performing ones to be fine-tuned.

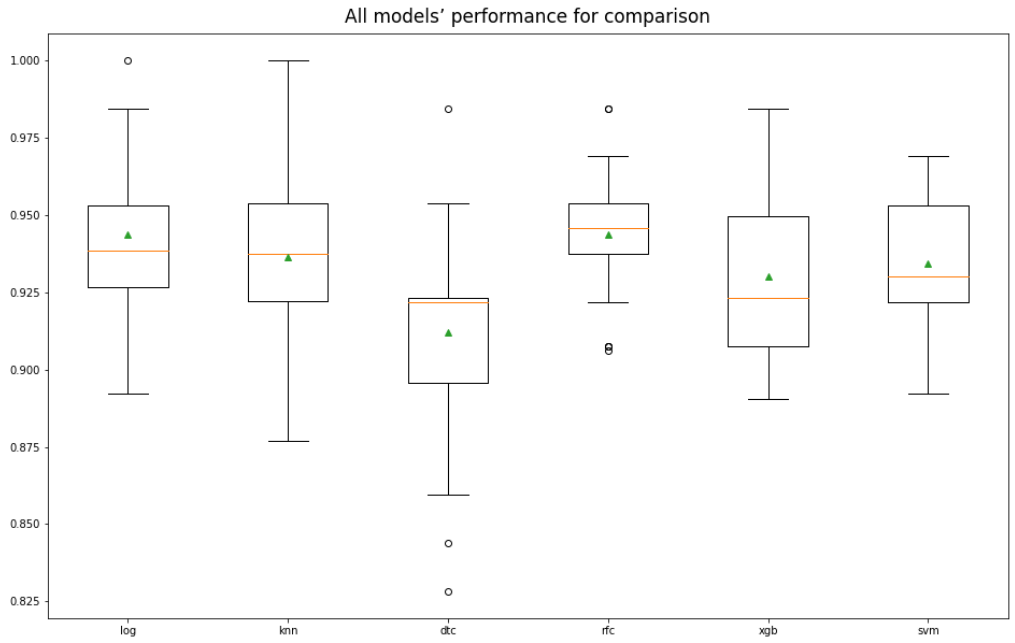


***Figure 6*** *Models' performance for comparison*

Half of the models has outliers as indicated by values lying beyond the 'whiskers' in figure 6. The decision tree suffers from extreme values the most, which might explain its lowest scores in every metric as seen below. Orange lines indicate accuracy mean, green triangles - mean accuracy per model.

| Rank | Model name | Cross-validation | Mean | Std |
|---|---|---|---|---|
| **1** | **SVM** | **95.3%** | **93.4%** | **2.2%** |
| **2** | **XGBoost** | **95.3%** | **93.0%** | **2.5%** |
| **3** | **Random forest classifier** | **94.8%** | **94.4%** | **2.1%** |
| 4 | Logistic regression | 94.2% | 94.4% | 2.3% |
| 5 | KNN | 94.2% | 93.6% | 2.4% |
| 6 | Decision tree classifier | 93.0% | 91.2% | 3.5% |

***Table 1*** *Models' accuracy scores*

Table 1 ranks the final accuracy scores of the models in descending order by cross-validation score and mean accuracy. Top three models were selected for the next stage.

# 7. Fine-tuning

As the first step in fine-tuning, grid searches were performed on SVM, XGBoost, and random forest classifier, none of which led to improvement in the accuracy score. A voting classifier was then run in an attempt to collectively boost the model's accuracy via ensemble learning techniques. However, this also did not have an impact on accuracy. For a shorter runtime and ease of further evaluation, the fine-tuned random forest model was used as a final model to evaluate feature importance and tweak the number of independent variables if needed.

```
artist_freq              0.225921
total_user               0.189881
stream_count             0.167078
playlist_passion_score   0.137240
dependent_perc           0.039149
youngadult_perc          0.035400
artist_passion_score     0.034679
female_perc              0.033392
senior_perc              0.028622
artist_name              0.027653
playlist_var             0.027257
adult_perc               0.027026
male_perc                0.026701
```

***Figure 7.1*** *Numeric feature importance*

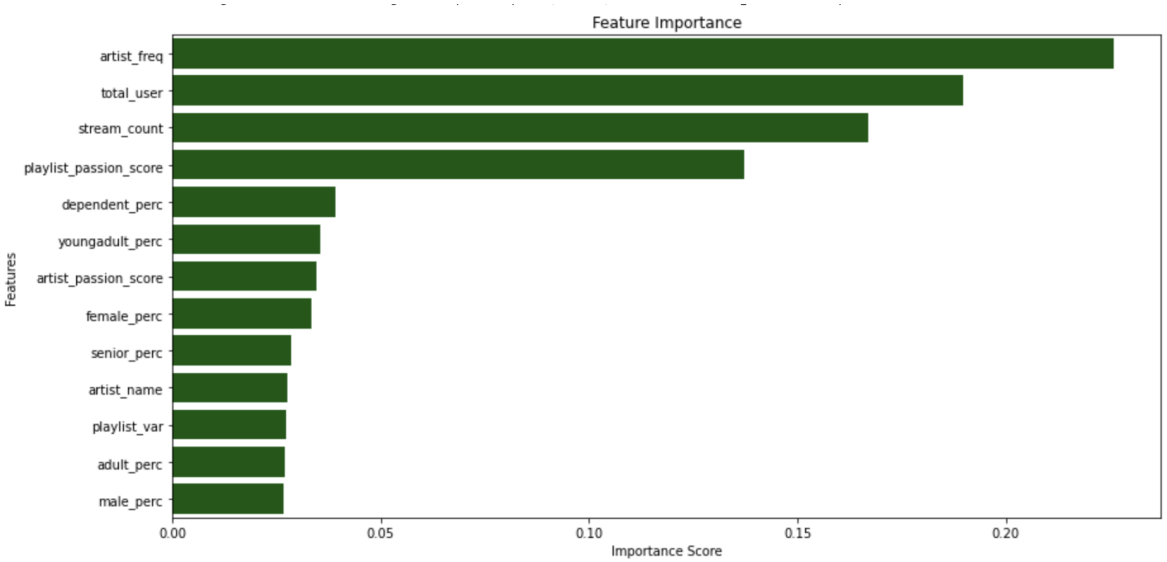


***Figure 7.2*** *Visual feature importance*

Reporting feature importance was done in two ways: by listing the values numerically (fig. 7.1), and visually (fig. 7.2). Diminishing returns in performance occur after dropping just 1 feature, therefore all of the original features were kept to maintain maximum accuracy.

# 8. Performance evaluation

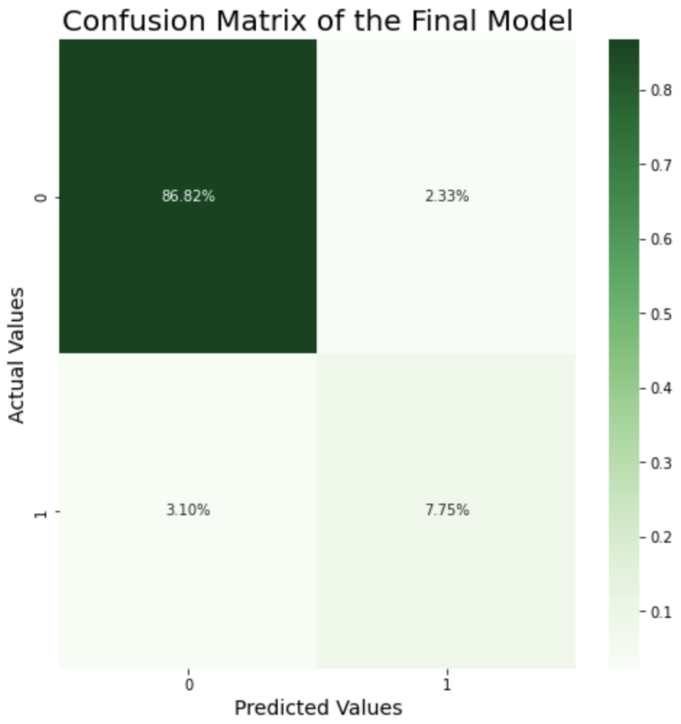


***Figure 8.1*** *Final confusion matrix*

The rate of correctly classified unsuccessful artists and successful artists is 86.82% and 7.75% respectively. 3.10% of observations were classified as successful artists whilst actually being unsuccessful (this is the false positive rate). 2.33% of observations were false negatives, meaning they were predicted to be unsuccessful but were in fact, successful artists. Overall 94.57% of the predictions are accurately classified – this is a good score considering the data at hand, thus, the model performed well on the test set.

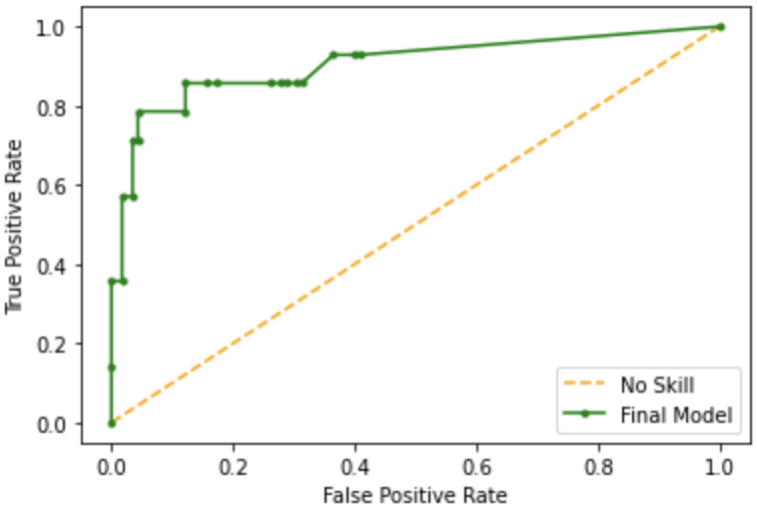


***Figure 8.2*** *ROC curve*

An ROC curve is used to deepen the analysis of predictive power and goodness of fit of the fine-tuned random forest. For an ROC curve, it is desirable to be as close as possible to the top-left corner of the graph. The area under the curve is 0.876, the maximum possible score is 1. Hence, the classification accuracy of the fine-tuned random forest is adequate. The dotted yellow line represents an untrained "no skill" classifier which gives a recall equal to 0.5, akin to a coin toss. The green line shows the final model's predictions; the bigger the area under the curve, the better the predictive power is.

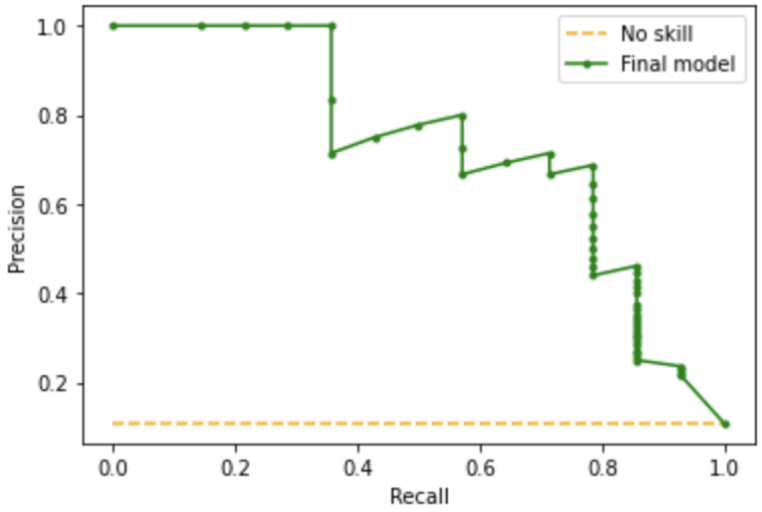


***Figure 8.3*** *Precision-recall curve*

Given the imbalance between successful and unsuccessful artists, the precision-recall relationship is analysed to further evaluate the success of the model's classifications. Precision quantifies the number of predicted successful artists by the model which actually belongs to that class. Recall quantifies the number of 'successful' predictions out of all true successful instances in the data. AUC score is desired to be as high as possible, with the maximum being 1. The higher the score the higher both precision and recall will be. Thus, the model will return more accurate results (precision) whilst the majority of results returned is positive (recall). The models' AUC is 0.688 which is satisfactory for the analysis.

# 9. Concluding remarks

This project used a secondary dataset to investigate features that impact artists' success in terms of being added to the most popular playlists. Initial data analysis and exploration revealed interesting trends, but very little in terms of useful inputs for machine learning models. Considerable feature engineering and data pre-processing and transformation had to be undertaken before any classification models could be trained on data that served as functional representations of artist success and what factors may affect it.

Six machine learning models were tested and evaluated, with a final random forest classifier being chosen as the best possible predictor of artist success within this project scope. The final model can be applied in a practical setting both by an artists' marketing team and the artists themselves to give insight into what may predict or impact their success. It is suggested that artists' release frequency, song's total listeners, and the number of streams (not length) are the most influential features that add tracks to top playlists and thus have a higher chance of success overall. For example, it is better to create short catchy songs that would be on repeat, rather than ones with long duration.

Since there were not many explanatory variables, the model has suffered from the reduced precision of class predictions. More quantitative data would give more informative insights which could have been improved by merging additional datasets with the same tracks. A transformation pipeline would also aid in pre-processing and data splitting steps by increasing speed and efficiency. While other fine-tuning techniques would be worth applying, including more iterations within a grid search likely have led nowhere as it seems that the model has reached its maximum potential accuracy score under its current state.

# 10. Reference list

GeeksforGeeks. (2019). *SVM Hyperparameter Tuning using GridSearchCV | ML*. [online] Available at: https://www.geeksforgeeks.org/svm-hyperparameter-tuning-using-gridsearchcv-ml/ [Accessed 19 Mar. 2022].

Kästle, K. (n.d.). *Country Codes List - Nations Online Project*. [online] www.nationsonline.org. Available at: https://www.nationsonline.org/oneworld/country_code_list.htm [Accessed 15 Mar. 2022].

Koehrsen, W. (2018). *Hyperparameter Tuning the Random Forest in Python*. [online] Medium. Available at: https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74 [Accessed 19 Mar. 2022].

Scikit-learn.org. (2019). *Precision-Recall — scikit-learn 0.21.3 documentation*. [online] Available at: https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html [Accessed 23 Mar. 2022].

Stack Overflow. (n.d.). *Python - ROC for Multiclass Classification*. [online] Available at: https://stackoverflow.com/questions/45332410/roc-for-multiclass-classification [Accessed 19 Mar. 2022].

Wikipedia. (2022). *ISO 3166-2:GB*. [online] Available at: https://en.wikipedia.org/wiki/ISO_3166-2:GB [Accessed 15 Mar. 2022].

# MSIN0097 Group Coursework

ORIGINALITY REPORT

| 9% | 1% | 0% | 9% |
|---|---|---|---|
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

PRIMARY SOURCES

| | | |
|---|---|---|
| 1 | Submitted to University College London<br>Student Paper | 8% |
| 2 | Submitted to Cranfield University<br>Student Paper | 1% |
| 3 | www.scsug.org<br>Internet Source | <1% |

| | | | |
|---|---|---|---|
| Exclude quotes | On | Exclude matches | Off |
| Exclude bibliography | On | | |

# MSIN0097 Group Coursework

GRADEMARK REPORT

FINAL GRADE

67 /100

GENERAL COMMENTS

## Instructor

Good to have added machine learning canvas to the report. Some interesting findings from the visualisations but no feature engineerings were performed based on those findings. You need to reconsider the key metric for comparing the models in terms of thinking which one would provide more insights, especially for the imbalanced dataset. You shud have also considered cost matrix. Nice to have plotted the precision-recall curve, but the narratives were a little imprecise. It would be beneficial if you work on the best classification threshold that optimise the business outcome since you have started looking into precision-recall tradeoff.

PAGE 1

PAGE 2

PAGE 3

PAGE 4

PAGE 5

PAGE 6

PAGE 7

PAGE 8

PAGE 9

PAGE 10

PAGE 11

PAGE 12

PAGE 13

PAGE 14