**Data Analytics and Machine Learning Project Assignment**

**Objective:**

To apply a machine learning algorithm to a dataset, performing all necessary preprocessing steps, scaling features, and conducting performance analysis of the chosen model.

### Assignment Outline:

#### ****Part 1: Dataset Selection and Understanding****

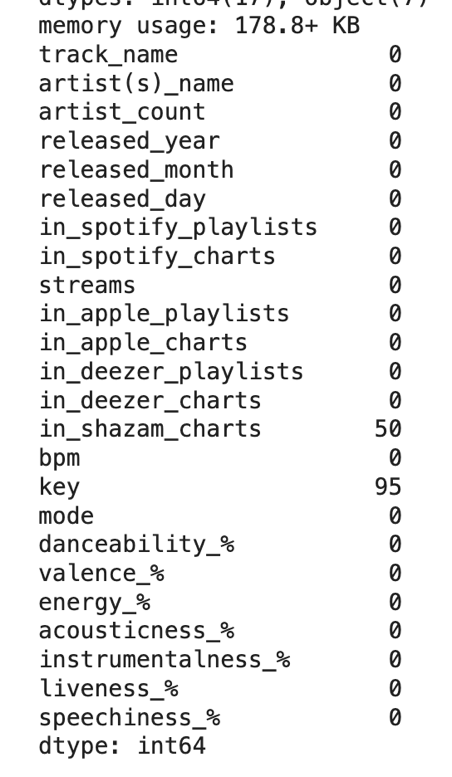
Most Streamed Spotify Songs 2023 Dataset Kaggle

* 953 rows, 23 columns
* Streams is the target column
* Release year, date, month may not all be relevant columns
* Some missing values in shazam charts and key column
* Challenges in figuring out which exact features might be most responsible for the “Next Hit”
* A lot of the columns are ordinal data which will make pre-processing easier, but some columns like Streams are string which need to be converted to integer/float

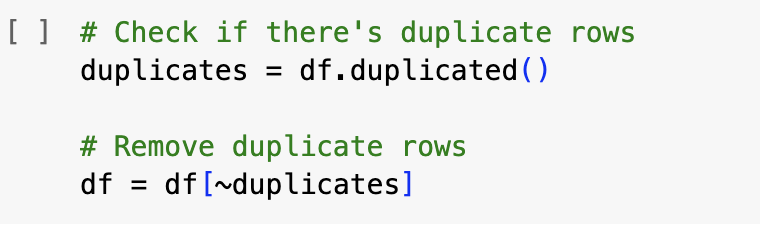
**Part 2: Data Preprocessing**

1. **Handling Missing Values**
   * Used df.info() to identify the different columns and datatypes and use it to decide how to deal with the missing values depending on the datatype.
   * Proceeded to using df.isnull() to figure out missing values in each column.
   * The results indicated that two columns (in\_shazam\_charts and key had 50 and 95 missing values respectively). Using this code helps us efficiently see each column and the sum of missing values.

A screenshot of a computer code

Description automatically generated

1. **Removing Duplicates** 
   * Check for and remove duplicate records if any. This can be done by df.duplicated()



1. **Data Type Conversion**
   * Ensure all features have the correct data types which was assessed using the df.info() code earlier. It showed that there were certain columns which had numbers but were considered as objects not as integers or floats. The code to convert these columns to integers and floats is pd.to\_numeric.
   * The conversion was necessary because we need to use numerical data for machine learning algorithms.

A screenshot of a computer

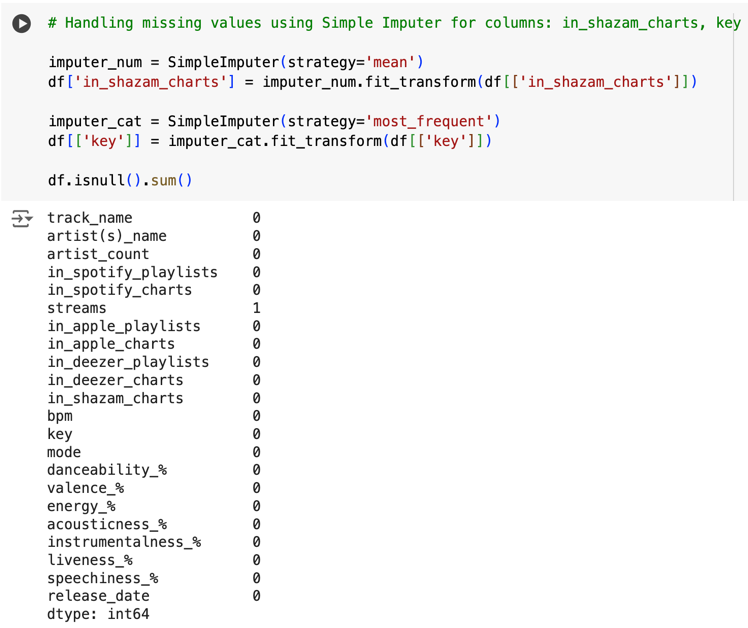
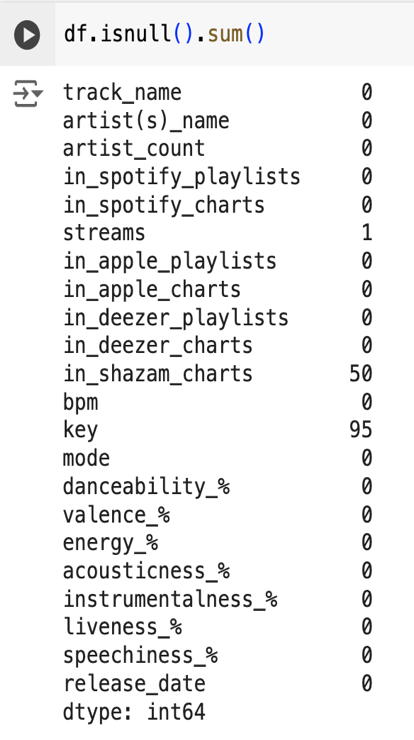
Description automatically generated

A screenshot of a computer code

Description automatically generated

* + After data conversion I checked if the columns were actually converted to integers or float and that was the result above.

1. **Handling missing values after data conversion** 
   * I checked again for missing values and found that there was one more added after converting some of the columns to numeric values.
   * In order to handle the missing values for in\_shazam\_charts and key I knew they were important columns for the regression analysis, therefore I used Simple Imputer
   * As shazam charts was a numerical column I used the mean method to fill in the missing values.
   * For the key column, the datatype was object so I knew I couldn’t use the mean or median method on the simple imputer, hence I opted for the most frequent method which finds the mode of key and uses that value to fill in the missing values in that column.



* + For the streams column, since this was the target column I didn’t want to put in a random value hence I decided to drop the value so that I can get a more accurate R-Squared score. I checked again that there were no missing values. A screenshot of a computer program

    Description automatically generated

**Part 3: Encoding and Feature Engineering and Scaling**

1. **Encoding**
   * I applied LabelEncoding and One-hot Encoding from Scikit-Learn for the categorial variables such as key, mode, track\_name, and artist\_name.
   * I used LabelEncoding for key and mode because for key column there wasn’t a huge variety of values so it was easier to assign them numbers and for mode the two values were only major and minor so that also called for numerical encoding.
   * For the track name and artist name columns there were a huge variety when examining the data and it didn’t make sense to use LabelEncoding. A screen shot of a computer code

     Description automatically generated
2. **Scaling Features**
   * I applied feature scaling techniques such as normalization or standardization.
   * This method subtracts the mean and divides by the standard deviation for each feature, resulting in features with a mean of 0 and a standard deviation of 1.
   * This approach helps to reduce the effect of outliers and prevents features with large ranges from dominating the model.

A computer code with text

Description automatically generated with medium confidence

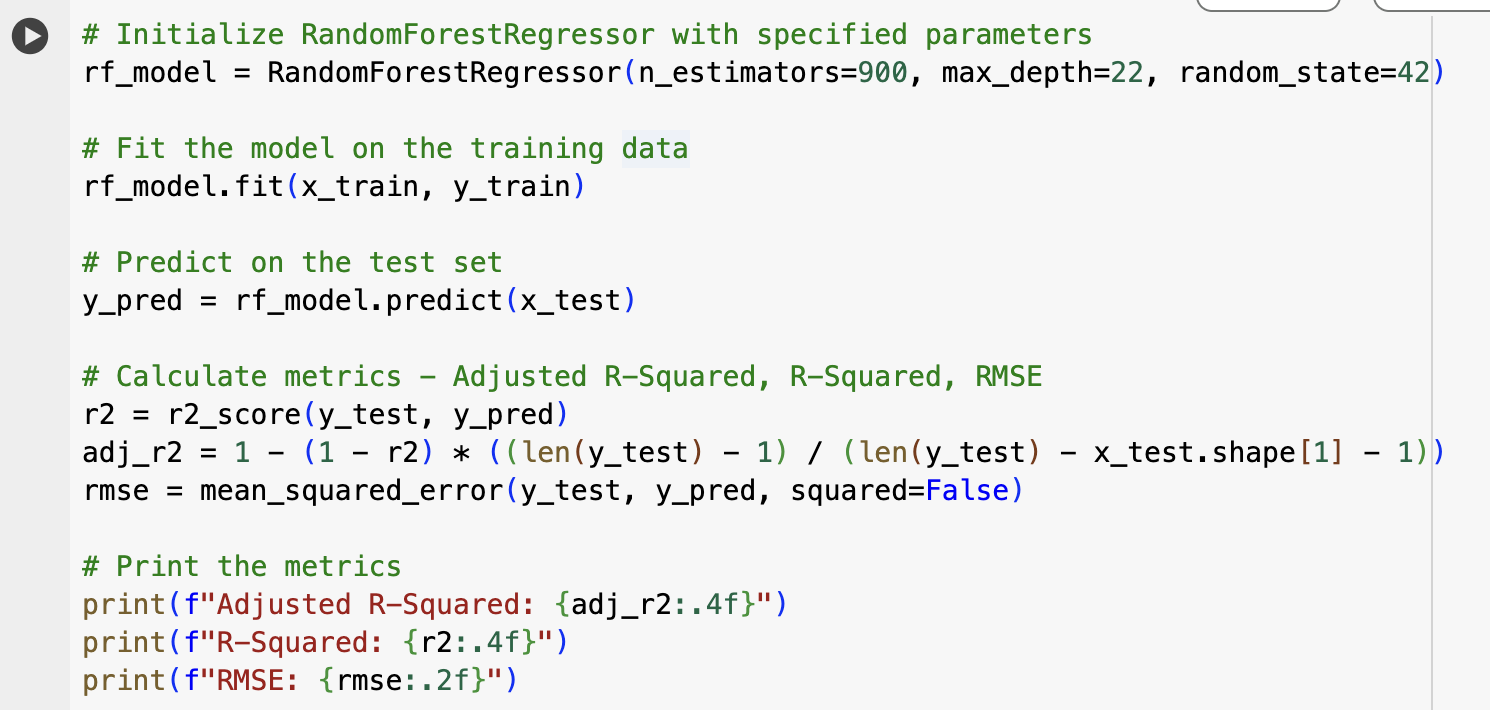
**Part 4: Model Selection and Training**

1. **Split the Data**
   * I split the dataset into training and testing sets (80/20 split).
   * This can be seen in the 3rd hashtag which shows test size that is 20% and training which is 80%.
   * I also chose to not use all of the columns for the feature variables and decided on a limit number. I decided to omit columns like artist name and track name because I didn’t think that a name of a song itself could be highly correlated with the streams which was the target variable.

A screenshot of a computer code

Description automatically generated

1. **Choose a Machine Learning Algorithm**
   * To start off, I chose Random Forest Regressor.
   * As the target variable is streams which is numerical data, it made sense to use regression analysis.
   * Some of the other algorithms I tested were: gradient boosting, linear regression, xgboost, and neural network.
2. **Train the Model**
   * I trained the model using the training dataset.
   * For Random Forest: The number of estimators was set to 900, which allows for a robust ensemble of decision trees that can effectively capture complex relationships in the data. The maximum depth of 22 was chosen to strike a balance between model complexity and risk of overfitting, enabling the model to generalize well to new data. Finally, a random state of 42 was specified to ensure reproducibility of the results. These hyperparameters work in concert to enable the model to learn a accurate and generalizable representation of the data.



**Part 5: Model Evaluation and Analysis**

1. **Performance Evaluation**
   * I evaluate the model performance using appropriate metrics like adjusted r-squared, r-squared, and RMSE as they are relevant for regression. A black text on a white background

     Description automatically generated
2. **Compare Models**
   * Here I compare the chosen model with other models to find the best-performing model. I used the same steps for each model with relevant parameters that the model called for.
   * Random Forest Regressor: A black text on a white background

     Description automatically generated
   * Gradient Boosting Regressor: A black text with black numbers

     Description automatically generated
   * Linear Regressor: A black text on a white background

     Description automatically generated
   * XGBoost: A black text with black numbers

     Description automatically generated
   * Neural Network Regression: A black text with black numbers

     Description automatically generated
   * The results for the models above indicated that Random Forest and XGBoost are the best models to use as they had the highest adjusted R-squared value as well lower RMSE value. This indicates that these two models are suitable for predicting the next hit in streams.