Machine Learning For Robotics
Assignment 1

Maha Qaiser 22i-2348

→ Purpose

Using various audio features and metadata provided in the dataset, understand which features contribute most to a song's popularity and create a regression model that can effectively predict how popular a song will be.

→ Dataset

Domain: Music
Target variable: Popularity (in percentage)
Number of features: 15
Number of records: 1,001,373 (1 million+)
Type of problem: Regression
Link: kaggle.com/datasets/amitanshjoshi/spotify-1million-tracks

→ Features

i. **Year**: the year the song was released.
ii. **Genre**: category or style of the music.
iii. **Danceability**: measure of how suitable the song is for dancing.
iv. **Energy**: measure of the intensity of the song.
v. **Key**: musical key the song is in.
vi. **Loudness**: overall loudness in decibels.
vii. **Mode**: indicates major or minor tonality of the song.
viii. **Speechiness**: presence of spoken words.
ix. **Acousticness**: measure of whether the song is acoustic.
x. **Instrumentalness**: whether the song contains no vocals.
xi. **Liveness**: detects audience presence in the song.
xii. **Valence**: measure of the song's positivity.
xiii. **Tempo**: speed of the song in beats per minute.
xiv. **Duration**: length of the song in milliseconds.
xv. **Time signature**: number of beats per bar in the song's rhythm.

**Note: I did not consider these columns from the dataset: Unnamed: 0, artist_name, track_name, track_id because they do not contain useful numerical or categorical data that could improve any model's predicting performance.*

→ **Numerical Features**

i. Year
ii. Danceability
iii. Energy
iv. Loudness
v. Speechiness
vi. Acousticness
vii. Instrumentalness
viii. Liveness
ix. Valence
x. Tempo
xi. Duration

→ **Categorical Features**

i. Genre (Text)
ii. Key
iii. Mode
iv. Time Signature

→ **Exploratory Data Analysis Summary**

- Removed rows where popularity was 0, as these were negatively impacting the dataset. This reduced the dataset from 1,159,764 to 1,001,373 rows.
- Created histograms for numerical features to understand their distributions. Normal distribution: Danceability and Tempo, while others were skewed.
- Created scatter plots and a correlation matrix alongside a heatmap to identify relationships between features, focusing on the correlation with the target variable Popularity.
- Created boxplots to identify outliers in the dataset, and later identified them using the interquartile range method.
- No missing values were found in the dataset,
- Important features:

- Year: Moderate positive correlation with Popularity. More recent songs tend to be more popular.
- Danceability: Weak positive correlation with Popularity. Danceable songs may be more engaging than others.
- Loudness: Weak positive correlation with Popularity. Loudness is often associated with energetic music but does not determine popularity alone.

→ **Preprocessing Steps**

- Data cleaning: Removed rows with popularity equal to 0.
- Dropped unnecessary columns: Unnamed: 0, artist_name, track_name, track_name
- Feature engineering:
  - Numerical features: popularity, year, danceability, energy, loudness, speechiness, acousticness, instrumentalness, liveness, valence, tempo, and duration_ms.
  - Categorical features: genre, key, mode, and time_signature
- Applied One-Hot Encoding to genre, key, and time_signature. Mode was already in binary format.
- Scaled numerical features:
  - MinMax Scaling: danceability and tempo
  - Standard Scaling: year, energy, loudness, speechiness, acousticness, instrumentalness, liveness, valence, and duration_ms
- Balanced classes: Created equal-sized groups to avoid the issue of imbalanced classes, which can lead to biased model performance.
- Data splitting: split the data to 80% training and 20% testing to use for model training.

→ **Stratified Sampling**

- Approach:
  - To ensure proportional representation of different popularity levels, the dataset was divided using stratified sampling with bins based on popularity_cat.
  - pd.qcut() was used to bin popularity into three equal-sized categories: low, medium, and high.

    - StratifiedShuffleSplit was applied to maintain the original distribution while splitting the dataset into 80% training and 20% test.
- Justification:
    - In datasets with an imbalance between categories, a random split could lead to underrepresentation of certain categories in the training or test set.
    - Stratified sampling ensures that the proportions of each category in the training and test sets match the original dataset, preventing bias and improving model generalization.
- Outcome:
    - The category distributions in the original dataset, training set, and test set are nearly identical.
    - This confirms that stratification worked correctly, ensuring a representative dataset split despite there being less data for more or ‘high’ popularity songs.

→ **Model Selection and Training**

- Stratified sampling with 80% training and 20% testing.
- Used root mean squared error and r squared score for evaluating model performance.
- Linear Regression: instant and just solves a linear system.
- Decision Tree: fast and splits the data.
- Gradient Boosting: Slow and boosts trees sequentially so it’s hard to parallelise.

→ **Fine-Tuning Process**

- Used Grid Search to find the optimal hyperparameters for the best-performing models.
- Applied k-fold cross-validation to ensure that the model’s performance was consistent across different parts of the data.

→ **Tuning and Model Performance**

- Lower RMSE: After hyperparameter tuning, the model achieved a lower RMSE, indicating better predictive accuracy.
- Better Generalization: The optimized model performed better on validation data, reducing overfitting compared to the previous model.

- Optimal Parameter Selection: The best hyperparameters helped balance model complexity and performance, improving efficiency.
- Trade-offs: While tuning increased training time, it significantly enhanced the model's predictive capability.

→ **Model Performance**

	Linear Regressor	**Gradient Boosting Regressor**	**Decision Tree Regressor**
RMSE	10.56883380113163	11.204624073616623	13.23848911695792
R^2 score	0.5144871928261849	0.4543161084927272	0.23823081794803447
Cross-validation RMSE score (mean)	10.718864	11.62107242	14.017569
Mean RMSE	10.561927406058897	11.207068377776853	13.283369596532145
Std Dev of RMSE	0.015285906665499801	0.015982819225682843	0.024059924127718714

→ **Final Conclusions and Best Model**

- The best model was the Linear Regressor.
- Features contributing more to popularity than others were: year, danceability, and loudness.
- This model can be used by music platforms to predict the popularity of new songs and market them accordingly.

**Final note: Due to RAM limitations on both my laptop and Colab, I am only able to train two models instead of three at one time. So the code for the third model is commented out. I hope this will be sufficient for evaluation. Thank you! :D*

Video:
https://drive.google.com/file/d/1ICVq-ifkTeL1efpukn_Sxkl8yEXJxg6U/view?usp=sharing