Divya Thomas

Class: CS 677

Date: 4/24/2023

Machine Learning on Song Popularity Characteristics

Overview

For my final course project in CS 677, I chose to use a dataset created and posted publicly on Kaggle by user, M YASSER H. (link: https://www.kaggle.com/datasets/yasserh/song-popularity-dataset) This dataset contains a list of over 12000 songs, both popular and unpopular, in addition to their various characteristics. This project was designed with objective of utilizing these characteristics and machine learning algorithms learned throughout this course to answer the question of whether a song's popularity can be predicted.

The dataset contains numerous continuous features, as described on Spotify's Web-API Documentation (https://developer.spotify.com/documentation/web-api/reference/get-several-audio-features) .

1. Song_duration_ms – duration of a track in milliseconds
2. Acousticness - A confidence measure from 0.0 to 1.0 of whether the track is acoustic.
3. Danceability – how suitable a track is for dancing based on a combination of musical elements including tempo, rhythm stability, beat strength, and overall regularity. A value of 0.0 is least danceable and 1.0 is most danceable
4. Energy - measure from 0.0 to 1.0 and represents a perceptual measure of intensity and activity.
5. Instrumentalness - Predicts whether a track contains no vocals. "Ooh" and "aah" sounds are treated as instrumental in this context. Rap or spoken word tracks are clearly "vocal". The closer the instrumentalness value is to 1.0, the greater likelihood the track contains no vocal content. Values above 0.5 are intended to represent instrumental tracks, but confidence is higher as the value approaches 1.0.
6. Key - The key the track is in. Integers map to pitches using standard Pitch Class notation. E.g. 0 = C, 1 = C♯/D♭, 2 = D, and so on. If no key was detected, the value is -1.
7. Liveness - Detects the presence of an audience in the recording. Higher liveness values represent an increased probability that the track was performed live. A value above 0.8 provides strong likelihood that the track is live.
8. Loudness - The overall loudness of a track in decibels (dB). Loudness values are averaged across the entire track and are useful for comparing relative loudness of tracks. Loudness is the quality of a sound that is the primary psychological correlate of physical strength (amplitude). Values typically range between -60 and 0 db.
9. Audio_mode - Mode indicates the modality (major or minor) of a track, the type of scale from which its melodic content is derived. Major is represented by 1 and minor is 0.
10. Speechiness - Speechiness detects the presence of spoken words in a track. The more exclusively speech-like the recording (e.g. talk show, audio book, poetry), the closer to 1.0 the attribute value.
11. Tempo - The overall estimated tempo of a track in beats per minute (BPM).

12. Time_signature - a notational convention to specify how many beats are in each bar (or measure). The time signature ranges from 3 to 7 indicating time signatures of "3/4", to "7/4".
13. Audio_valence - A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track.
14. Song_popularity – the level of recognition and success this track has received from 0 to 100.

For the purpose of increasing specificity of the analysis of the data, we will be focusing on 5 randomly selected features and testing whether or not they are sufficient to predict song popularity. These features are the following: 'audio_valence', 'acousticness', 'energy', 'instrumentalness', and 'loudness'.

Initial Analysis

In the initial analysis, this data was copied into a dataframe with the additional column 'rating'. Based off of the mean value of song popularity, anything with a popularity score above the mean was given a rating of '+', indicating it was a popular track. Otherwise, unpopular track were given a '-' rating.

Popularity mean: 48.75090446201259

Popularity standard deviation: 20.37878255505516

Following this step, mean and standard deviation of all tested features were calculated this was done for the entire dataset, as well as each class value subset. See results below.

class	audio_valence		acousticness		energy		instrumentalness		loudness	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
+	0.52	0.24	0.26	0.29	0.64	0.22	0.07	0.22	-7.48	4.1
-	0.54	0.25	0.28	0.30	0.64	0.22	0.11	0.26	-7.90	3.91
all	0.53	0.25	0.27	0.3	0.64	0.22	0.09	0.24	-7.68	4.02

It is clear through this data that these values are very similar throughout the dataset, regardless of the popularity rating. It is good to note that the unpopular tracks tend to have higher than average audio_valence, acousticness, and instrumentalness mean values, whereas popular one have lower. Energy seems to be the same for all three, and loudness mean is more negative for unpopular songs than average and less negative for popular songs than average.

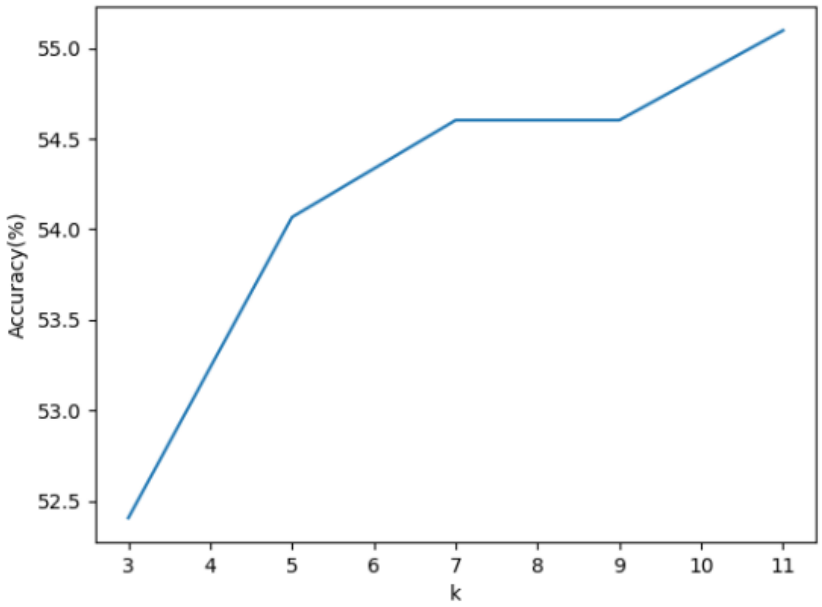
Machine Learning Algorithms

Following the initial analysis on the data, a series of machine learning algorithms were used in order to train and test prediction accuracy on these specified characteristics. All machine learning algorithms were trained on 50% of the total dataset, selected at random. The remaining 50% was used as test data to allow predictions to be made by the classifiers and checked for accuracy.

k-NN Classifier (k – Nearest Neighbors)

The KNN algorithm works by measuring the distance between the new, unseen data point and all the training data points. It then selects the K nearest neighbors, or the K data points with the smallest distance to the new data point, and assigns the new data point to the class that is most common among its K nearest neighbors.

Similar to previous homework problems, this classifier was run with multiple k values first [3, 5, 7, 9, 11] and accuracy values for each k was plotted in the knn_plot.png file (screenshot below). As you can see from the data, the most accurate k value shows to be 11. This will now be the value used from here on out whenever the k-NN classifier is used.



Linear-Kernel SVM Classifier

The linear kernel computes the dot product between the feature vectors of two data points, which can be thought of as a measure of similarity between them. The SVM with a linear kernel tries to find the hyperplane that maximally separates the two classes in this feature space.

Logistic Regression Classifier

The logistic regression algorithm models the probability of the binary outcome using a logistic function, which maps any input value to a value between 0 and 1. The logistic function is a sigmoidal curve that increases rapidly near the center and flattens out towards the ends.

Naïve Bayesian Classifier

This is an algorithm based on Bayes' theorem, which states that the probability of a hypothesis (in this case, a class label) given the observed evidence (in this case, the input features) is proportional to the probability of the evidence given the hypothesis, multiplied by the prior probability of the hypothesis. The "naive" part of Naive Bayes comes from the assumption that the input features are conditionally independent given the class label. This means that the presence or absence of one feature does not affect the probability of the other features appearing together.

Consolidated Classifier

This classifier follows the concept of an ensemble algorithm. It consolidates and compares the predictions from other algorithms to make the most optimal and, in this case, most common, prediction results.

Post-Algorithmic Analysis

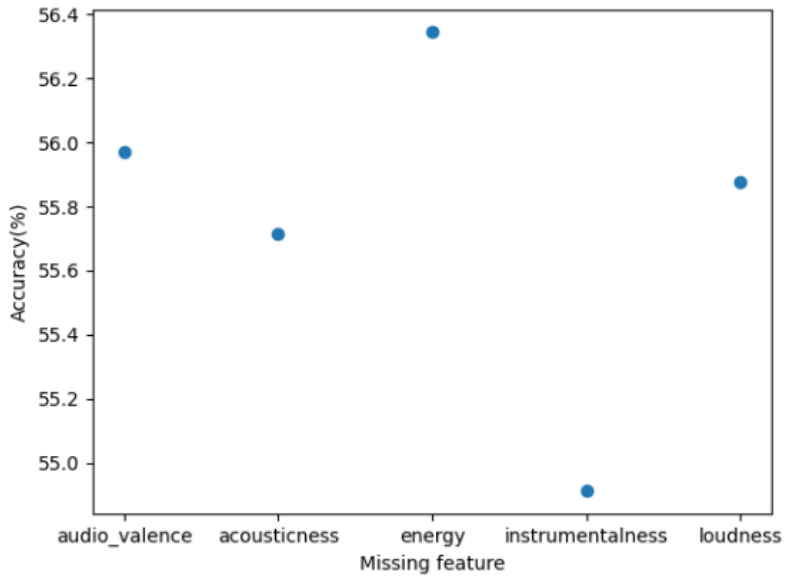
Each classifier was trained and tested on the same spit data and performance measures were calculated as shown in the table below. The values calculated are true positive (TP) counts, false positive (FP) counts, true negative (TN) counts, false negative (FN) counts, accuracy values as a percentage, true positive rate (TPR) , and true negative rate (TNR) values. True positive rate indicates the algorithm's ability to successfully predict positive ('+') values while the true negative rate indicates for successful prediction of negative ('-') values.

Model	TP	FP	TN	FN	Accuracy (%)	TPR	TNR
k-NN	2551	1983	1562	1367	55.11	0.65	0.44
LK	3626	3127	418	292	54.19	0.93	0.12
NB	3473	2873	672	445	55.54	0.89	0.19
LR	3081	2426	1119	837	56.28	0.79	0.32
Cons.	3296	2621	924	622	56.55	0.84	0.26

As shown in the data above, each model used provided an accuracy of less than 60%. Which linear kernel SVM shows the lowest overall accuracy, its revealed to have the highest TPR of the five. It also has the lowest TNR, which is the likely factor for it's overall low accuracy. Contrary to that, the k-NN classifier was the next lowest overall accuracy, but resulted in the highest TNR and lowest TPR of the five. Overall highest accuracy was received by the consolidated model, which held the median TPR and TNR values of the five.

Feature Selection

Additional analysis was made on the results of the consolidated model. In order to determine the impact of the five features used, this process of re-running the model with each feature missing was implemented. The same split on training and testing data was used on each feature to ensure consistency. After removing a feature and running the consolidated model on the updated dataset, the accuracy of the predictions were saved and plotted, as shown below.



Judging from the accuracy chart above (missing_feature.png), it is clear to see that there are definitely some extremes. Energy shows the highest level of accuracy when removed, indicating this feature may have the high impact in bringing the accuracy rate down, and the consolidated algorithm's accuracy would likely benefit from removing it. On the other extreme, we can see that the missing instrumentalness feature brings the accuracy rate significantly lower, indicating that this feature may be a very beneficial factor for accurately predicting popularity of the song.

In the future I would like to improve this algorithm by taking more of the features into account and analyzing their impact, as shown above. Instead of selecting features at random, an enhanced classifier can look into all features which result in lower accuracy rates when missing, like instrumentalness. It may also be beneficial to omit all features resulting in high accuracy rates when missing, like energy.

Bonus: Predictor

After determining the most accurate predictive model, I chose to create an interactive interface for users to input the tested values of a song of their own choice. This predictor used the most accurate consolidated model as well as the entire song dataset as training data on the model. The test data was the values for the 'audio_valence', 'acousticness', 'energy', 'instrumentalness', and 'loudness' as input by the user.