CS5033 - SL Project

Airi Shimamura Khoi Trinh

1. Introduction

For our Supervised Learning project this semester, we want to create a few models to predict a user's preference for a song on the Spotify streaming platform.

Each song on Spotify has their own set of 11 features. These features are: danceability, energy, key, loudness, mode, speechiness, acousticness, instrumentalness, liveness, valence, and tempo.

A user can request their streaming history from Spotify via the P rivacy section of their account. For the purpose of this project, Khoi's one-year streaming history from February 2022 to February 2023 will be used as the 1 class, for songs we like; and about 4200 random songs which Khoi has never heard will be used as the 0 class, for songs we do not like.

The dataset has 56,210 songs in class 1, and 4,170 songs in class 0. However, the songs in class 1 have duplicates, due to the nature of streaming a song multiple times. We expect this number will be much lower once the duplicates are removed.

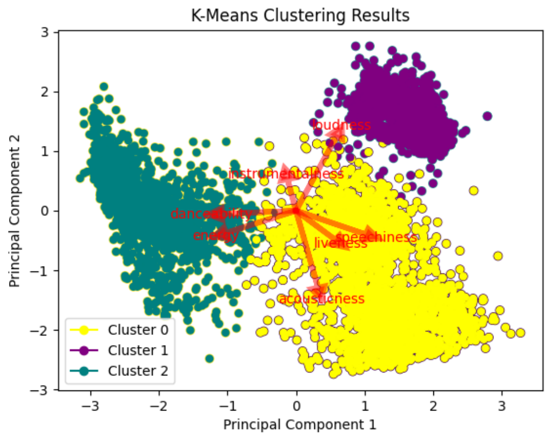

Figure 1. Clustering Results

2. Brief Overview

In this section, some brief overview will be given for the data, as well as some pre-processing and analysis.

One thing we wanted to explore from the data, is how the points are related to one another, through principal component analysis, and clustering. Using k-means clustering, with $k = 3$, we obtained the following figure.

Looking at the figure, we can see that Principal Component 1 separate songs with higher liveness, speechiness, and loudness, and those with lower danceability and energy (these mostly belong to Cluster 0 and 2), and Principal Component 2 separate songs with higher instrumentalness, loudness, and those with lower acousticness (these mostly belong to Cluster 2). Overall, we can see that Cluster 0 has songs that are higher in loudness, speechiness, and liveness. Cluster 1 has songs that are lower in danceability and energy. Cluster 2 has songs high in speechiness, liveness, acousticness, but lower in instrumentalness.

3. Hypotheses

Our first hypothesis is that our supervised learning models will be able to classified these songs as likes or dislikes with at least 90% accuracy.

Our second hypothesis is that out of the four chosen algorithms; Random Forest will give us the best performance; followed by Decision Tree, then Logistic Regression, and finally Naive Bayes. We will evaluate each algorithm's performance by comparing the value of Precision, Recall, and produce ROC curves for each algorithm.

4. Experimental Analysis

4.1. Khoi's Results

For this project, Khoi implemented logistic regression and decision tree.

Logistic Regression is a statistical machine learning algorithm predicting the probability of a target variable by fitting a logistic function to the input features. This optimizes the parameters of the logistic function by minimizing the cost function, which measures the difference between the predicted probabilities and the actual class labels.

For this model, Khoi implemented the following functions:

Sigmoid/Logistic function: $\sigma(z) = \frac{1}{1+e^{-z}}$

Logistic loss function: $logloss = \Sigma_{x,y \in D} - ylog(y') - (1-y)log(1-y')$

Where:

$(x, y) \in D$ is the data set containing labeled instances.

y is the label in a labeled example; in this case, it is either 0 or 1.

y' is the predicted value, given the features in x.

Khoi also implemented the logistic regression function from scratch, and can be seen in the code provided.

Here are the results, with $\alpha = 0.5$ and 1000 iterations.

Accuracy: 97.6%

Precision: 96.19%

Recall: 97.22%

Decision Tree is a machine learning algorithm to predict the class of an input based on its features. This algorithm partitions the input space into increasingly smaller regions based on the values of the input features by selecting the features and test conditions that best separate the training data into the different classes. This process is repeated recursively to create a tree-like structure until a final decision is made.

Similar to the previous model, Khoi implemented this from scratch, and can be seen in the provided code. The criteria Khoi is using to split the node is the Gini impurity. The $build_tree()$ method builds the decision tree recursively by finding the best feature and threshold to split the current node based on the Gini impurity criterion. It then splits the data into left and right subsets based on the chosen feature and threshold, and recursively builds the left and right subtrees until it reaches a stopping criterion, such as reaching the maximum depth of the tree or having only one sample in a node.

Here are the results, with a tree depth of 5:

Accuracy: 98.07%

Precision: 97.02%

Recall: 97.64%

4.2. Airi's Results

5. Future Work

Khoi will continue refining his logistic regression model, as well as work on the decision tree model. The decision tree model will be useful for random forest, as well.

To enhance the performance of the random forest and Naive Bayes models, Airi will continue re-evaluating them and obtain ROC curves. Moreover, she will expand the random forest model by increasing the number of trees and maximum depth for each tree. In addition, she will experiment with two other criteria, namely entropy and chi-squared, to assess their impact on the model's performance.

Finally, as mentioned in the previous section, the Logistic Regression, Naive Bayes, and Random Forest models will need to be redone, to ensure accurate performance.