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 Privacy 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, these will be removed.

For this project; Khoi implemented Logistic Regression, and Decision Trees; Airi implemented Naive Bayes, and Random Forest. We also compared our models with existing models from the scikit-learn package; namely Ridge Regression, Stochastic Gradient Descent, K-Nearest Neighbors, and Support Vector Machines.

2. Overview and Preliminary 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.

Principal Component Analysis (PCA) is a technique in statistics and data analysis for reducing the dimensions of a dataset, finding patterns, and visualizing relationships. It works by identifying the principal components that explain the most variation in the data, and projecting the data onto these components. Several sources validate the effectiveness of PCA in various applications, such as genetics, finance, image processing, and ecology. Some notable references include the works of (Jolliffe, 2002), (Hotelling, 1933), and (Pearson, 1901). We were able to reduce our numeric features into 2 principal components.

Looking at the Figure 1, 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 classify 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 first comparing the Accuracy value, then the values of Precision, Recall, and produce ROC curves for each algorithm, should the accuracy be equal.

4. Experimental Results and Analysis

4.1. Khoi's Experiments

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.

Decision Tree is a machine learning algorithm used 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. This is a measure of the impurity or randomness of a set of examples used in decision trees. It measures the probability that two randomly chosen examples from the set have different class labels. The aim is to find the split that results in the lowest Gini impurity of the resulting subsets. With that being said, in (Breiman et al., 1984), the author mentioned that decision tree can achieve high accuracy, but can be unstable due to small changes in the training data.

The results for these algorithms will be included in table 1.

4.2. Airi's Experiments

Airi implemented Naive Bayes and Random Forest algorithms.

Naive Bayes is a group of probabilistic machine learning algorithms based on applying Bayes' theorem with the assumption of independence between features. This assumes that the presence or absence of one feature does not affect the presence or absence of any other feature within a class, and calculates the probabilities of each class label given the values of the features from the input. The class with the highest probability is then selected as the predicted class label for the given input. In a study conducted by (Singh et al., 2017), it is stated that Naive Bayes performed better on nominal data, while its performance on numerical and mixed data was dropped. Te majority of our data is numerical, meaning the performance of our model was not as good as other models such as K-NN or Random Forest as shown in table 1.

Random Forest is an ensemble learning method that combines multiple decision trees to improve performance and reduce over-fitting. Airi was able to build her model based on Khoi's Decision Tree code. In a study conducted by (Tangirala, 2020), it was found that both Gini impurity or Information gain provided similar performances regardless of whether the data set was balanced or unbalanced. However, the Gini index outperformed information gain in terms of accuracy on certain datasets, while information gain exhibited better computational time efficiency.

In addition, according to (Singh et al., 2017), Random Forest performed better on numerical data, while its performance dropped for nominal and mixed data. Even though our dataset is mixed data, our Random Forest model still achieved an accuracy of over 98% .

For this model, Airi used the Gini index as the criterion for tree splitting, varying the number of decision trees from 10 to 100 with depth = 10 and selecting the best performance. The maximum accuracy was achieved when the number of trees was set to 30.

The results for these algorithms will be included in table 1.

4.3. Hypothesis 1

Table 1 shows our results.

	Log Reg	Naive Bayes	Decision Tree	Random Forest
Accuracy	97.6%	97.45%	98.07%	98.61%
Precision	96.19%	97.58%	97.02%	98.63%
Recall	97.22%	96.88%	97.64	98.35%
Runtime(ms)	677.4	65.41	12789.01	270066.1

Table 1. Results Of Our Experiments

From the above results, we can see that our experiments matched the expectations of hypothesis 1; all four of our models performed with accuracy better than 90%.

4.4. Hypothesis 2

For this hypothesis, we first compared the accuracy of each algorithm, then the area under the curve (AUC), and finally, the running time. The algorithm with the higher accuracy is generally considered to be better. If the accuracy is the same, then a higher AUC indicates a better algorithm. Finally, if both accuracy and AUC are the same, then the faster algorithm is the better one.

Looking at table 1, we can see that we only needed to compare accuracy. This means that Random Forest performed best, followed by Decision Tree, then Logistic Regression, and finally Naive Bayes.

Overall, the four algorithms were ranked as expected.

5. Related Work Reviews

We both compared our models with built-in models from the scikit-learn python package. Khoi compared Logistic Re-

gression with Ridge Reggrsion, Decision Tree with Stochastic Gradient Descent.

Ridge regression is a regularized regression method that addresses the issue of overfitting by adding a penalty term to the loss function. According to (Hastie et al., 2009), ridge regression can improve prediction accuracy when the number of predictor variables is large and highly correlated. With our dataset, ridge regression has similar, but slightly lower accuracy, however, it achieved higher precision, and lower recall when compared to Logistics Regression.

Stochastic gradient descent (SGD) is an iterative optimization algorithm commonly used for training neural networks and other machine learning models. Compared to batch gradient descent, which updates the model weights based on the entire training dataset, SGD updates the weights based on small batches of data, leading to faster convergence. In the paper by (Zhang et al., 2017), the author stated SGD can achieve higher accuracy and faster convergence compared to other optimization methods for deep learning. When compared to the current decision tree model (with depth 5), SGD performed worse, with lower accuracy, precision, and recall.

For Airi, she compared her models with K-Nearest Neighbors, and Support Vector Machine (SVM)

K-nearest neighbors is a non-parametric classification and regression method that uses the k-nearest data points in the training set to make predictions. In the research by (Cover & Hart, 1967), it was found that KNN can achieve high classification accuracy but may suffer from the curse of dimensionality when the number of predictors is high. Addtionally, the study by (Jadhav & Channe, 2016) found that K-NN performed better on datasets with smaller numbers of features and larger numbers of instances. Finally, according to (Singh et al., 2017), K-NN performed better on mixed data, while its performances dropped on numerical and nominal data.

Our dataset has only 11 features and a large number of instances, consisting of both numerical and categorical data, our K-NN model performed better than Naive Bayes but did not reach the level of performance achieved by the Random Forest model.

In Airi's case, she compared K-NN model with different numbers of the nearest neighbors from 1 to 100 and obtain the best number of k value.

Support vector machines (SVM) are popular supervised learning methods that seeks to find the hyperplane that maximally separates the different classes in the input space. (Cortes & Vapnik, 1995) stated that SVM can achieve higher classification accuracy than other methods such as neural networks and decision trees, especially in high-dimensional spaces.

With our data, SVM performed worse than Random Forest, with a lower accuracy, precision, and recall.

Table 2 shows the results of these scikit-learn models:

	Ridge Reg	SGD	KNN	SVM
Accuracy	97.29%	97.6%	98.14%	97.83%
Precision	97.16%	96.58%	98.12%	97.78%
Recall	95.29%	96.79%	97.85%	97.51%

Table 2. Comparison Models' Results

6. Conclusion and Future Work

In conclusion, throughout the project, we were able to implement our own algorithms for predicting a user's preference for a particular song on Spotify. All of our algorithms achieved more than 90% accuracy, and they were ranked as hypothesized; with Random Forest performing best, followed by Decision Tree, Logistic Regression, and Naive Bayes.

In the future, we would like to implement these algorithms using a larger dataset. Spotify also allows users to request their lifetime streaming history, and in Khoi's case, this would result in a much larger dataset, as he has been using the platform for a while. Furthermore, we found that the performance of these algorithms varied depending on the type of data used through the research papers. The data set we used consists of mostly continuous variables, so we would like to find out how their performances change if we add more or remove categorical variables.

7. Relevant Figures

This section includes any figures relevant to the project. It has the PCA results, and the ROC curves for our model, as well as a comparison of performance for the Random Forest algorithm

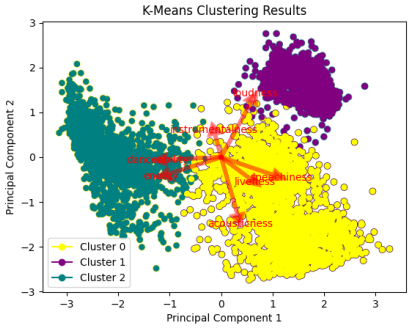

Figure 1. Clustering PCA Results

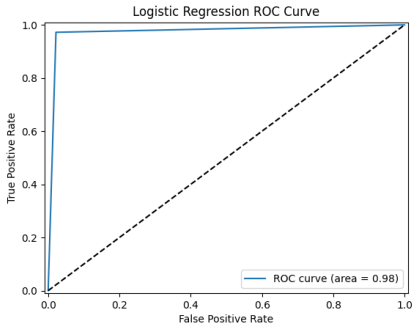

Figure 2. Logistic Regression ROC

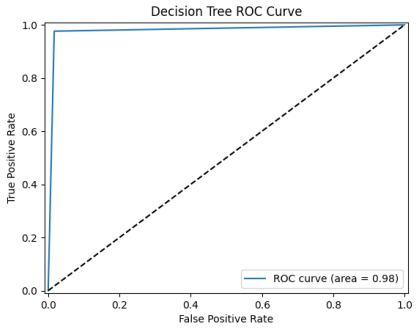

Figure 3. Decision Tree ROC

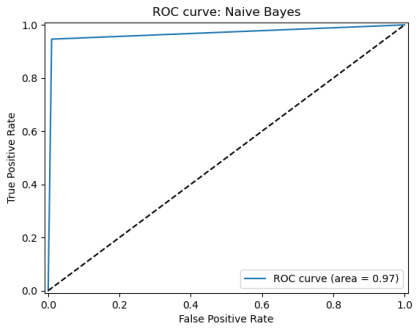

Figure 4. Naive Bayes ROC

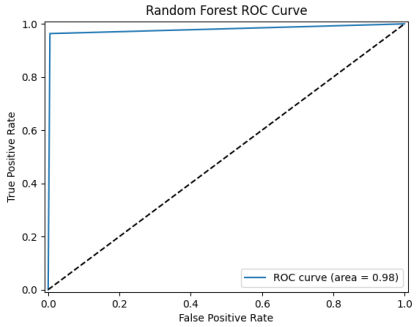

Figure 5. Random Forest ROC

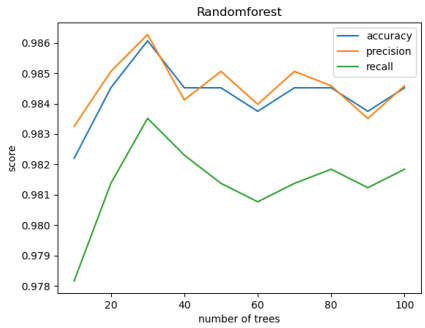

Figure 6. Result of Random Forest

References

Breiman, L., Friedman, J., Olshen, R. & Stone, C. (1984). *Classification and Regression Trees*. Chapman and Hall.

Cortes, C. & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20(3)*, 273–297.

Cover, T. & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, *13(1)*, 21–27.

Hastie, T., Tibshirani, R. & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, *24(6)*, 417–441.

Jadhav, S. D. & Channe, H. (2016). Comparative study of k-nn, naive bayes and decision tree classification techniques. *International Journal of Science and Research (IJSR)*, *5(1)*, 1842–1845.

Jolliffe, I. T. (2002). *Principal Component Analysis*. Springer.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, *2(11)*, 559–572.

Singh, A., Halgamuge, M. N. & Lakshmiganthan, R. (2017). Impact of different data types on classifier performance of random forest, naive bayes, and k-nearest neighbors algorithms. *International Journal of Advanced Computer Science and Applications*, *8(12)*.

Tangirala, S. (2020). Evaluating the impact of gini index and information gain on classification using decision tree classifier algorithm. *International Journal of Advanced Computer Science and Applications*, *11(2)*, 612–619.

Zhang, C., Bengio, S., Hardt, M., Recht, B. & Vinyals, O. (2017). Understanding deep learning requires rethinking generalization. *arXiv preprint arXiv:1611.03530*.