TASK-2: Keyword-Based Song Classification

Introduction:

This is the report of the code that I have written for the task-2.THis report gives a detailed explanation of all the algorithms and methods that I have used in the task . So lets begin…

Problem statement:

We have given a dataset , which contains the following columns**:**

- song_id
- keyword_1
- keyword_2
- keyword_3
- Genre

Here keyword_1,keyword_2 ,keyword_3 basically represents the instrument , mood and style of the song.
Now we have to extract insights from data and group similar songs and provide actionable insights using the genre.so lets begin…

Explanation :

Requirements :

Extracting the insights from data is not an easy task , it needs a lot of mathematical operations . however python has a lot of libraries for that .\
For this task we need :

Numpy:

This library is for mathematical operations , matrix operations . It provides flexible too.

Pandas:

This library is used for working with the dataset.

Matplotlib.pyplot:

Matplotlib is a library which provides accessibility for plotting the graphs and simulations . pyplot is a collection of functions in it ,which can be used to create subplots and add plot labels and much more…

First , we have to load the dataset using the .read_csv function of pandas
We can drop the “song_id“ column as it has no significant contribution to the data.

Vectorization of keywords:

The main reason to convert the keywords into vectors is that the machines and algorithms are not good at analyzing the words as they are for the numbers.
So now we have to create vectors for each keyword for each song . That gives us three vectors for each song .
There are many popular techniques for converting a word into vectors:

- Bag of words
- Skip gram
- TF-IDF(Term Frequency Inverse Document Frequency)
- Continuous Bag of Words

And many more**…**

However in this task we are going to talk about two methods:

- Bag of words
- TF-IDF

Bag of words:

A bag of words is a representation of text that describes the occurrence of words within a document. We just keep track of word counts and disregard the grammatical details and the word order.In BoW, the corresponding component value for the vector is equal to the frequency of the word in the document.

TF-IDF:

TF-IDF (Term Frequency-Inverse Document Frequency) is a statistical measure used in information retrieval and text mining to evaluate the importance of a word in a document relative to a collection (or corpus) of documents.

In TFIDF , **TF** means term frequency which is given by :

$$TF(t,d) = \frac{number\ of\ times\ t\ appears\ in\ d}{total\ number\ of\ terms\ in\ d}$$

The intuition behind TF is simple: the more a word appears in a document, the more relevant it is to the document's content.

Whereas **IDF** means Inverse-document-frequency which is given by:

$$IDF(t) = log\frac{N}{1+df}$$

While TF tells us about the importance of a term within a document, IDF reveals its significance in the entire corpus. IDF measures how rare or common a term is across all the documents in the collection.

Advantages of TF-IDF over BoW:

TF-IDF assigns weights based on term importance, reducing the impact of commonly used words.BoW treats all words equally, leading to less meaningful representations.TF-IDF gives higher scores to rare words that are more informative for a document.

So due to these pros of TF-IDF over BoW , we use TF-IDF in our task for generating vectors.

Dimensionality reduction:

Now we have three vectors for each song in the dataset , representing its three keywords. But there is a problem: the dimensions of these vectors are large and vectors of high dimensions can't be plotted .Dimensions refer to the number of features in our dataset. so we must reduce the dimensions of the vectors and preserve as much information as possible.for this task we have a method called PCA(Principal Component Analysis)

PCA(Principal Component Analysis):

PCA can tell us which category (or feature) is the most valuable for clustering the data.

PCA finds a new set of dimensions such that all the dimensions are perpendicular and linearly independent to each other. And also order by the importance (variation or the spread they cause in data).

Covariance matrix:

The covariance matrix is the matrix which indicates the level to which the two variables vary together.

$$Cov(X,Y) = \frac{1}{n}\sum (X_i - \bar{X})(Y_i - \bar{Y})^T$$

$$Cov(X,X) = \frac{1}{n}\sum (X_i - \bar{X})(X_i - \bar{X})^T$$

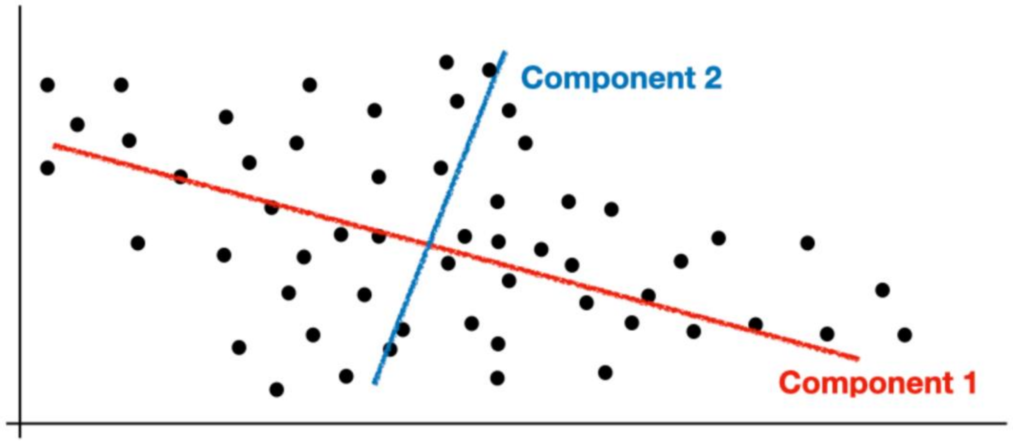
Eigenvectors:

Eigenvectors are the direction pointing vectors. They point in the direction of maximum variation .

Procedure:

- First , calculate the variance and mean for each feature of X (input data).
- Calculate the covariance matrix of X with respect to X i.e *Cov*(X,X).
- Calculate the eigenvectors for the covariance matrix calculated above.
- Sort the eigenvectors according to their respective eigenvalues.
- Choose the first n eigenvectors depending upon the n values.
- These n vectors are our new n dimensions.
- Transform the original data points into these data points via dot product(i.e. Taking projections).

So in this way we can use PCA to reduce the dimensions of a high dimension vector and simultaneously preserving as much information as possible.



Combining the three vectors into one vectors:

We have each song represented by a list of three 2-D vectors which have the information about each keyword.now it is our to combine these vectors into one single 2-d vector which is able to represent the whole song in the dataset.

There are many techniques to embed vectors into one such as:

- **Averaging:**

Compute the element-wise mean of the three vectors.This balances the influence of all keywords.

- **Summation:**
  Compute the element-wise sum of the three vectors.This gives higher importance to stronger keyword signals.
- **Weighted Combination:**
  If some keywords are more important, assign different weights:
  Vfinal=w1*V1+w2*V2+w3*V3
  We can Choose weights {w1,w2,w3} based on relevance.
- **Max Pooling:**
  Take the element-wise maximum:
  Vfinal=max(V1,V2,V3)
  This ensures that the most significant value for each dimension is retained.
  Out of the above methods specified , we are going to select the weighted Combination Method since in our dataset the influence of different keywords on deciding the genre of the song is different.

Weighted Combination:

Since in our analysis we can't use the genres column (i.e. the ground truths), we have to optimize the weights by maximizing the variance(Higher variance means more diverse information from all vectors) and preserving magnitudes(i.e. to ensure the combined vector retains significant values from each individual vector).

Clustering:

Now we have a single two dimensional vector to represent each song in the dataset. We have grouped the similar song vectors to form clusters and also plot on a graph for visualisation .

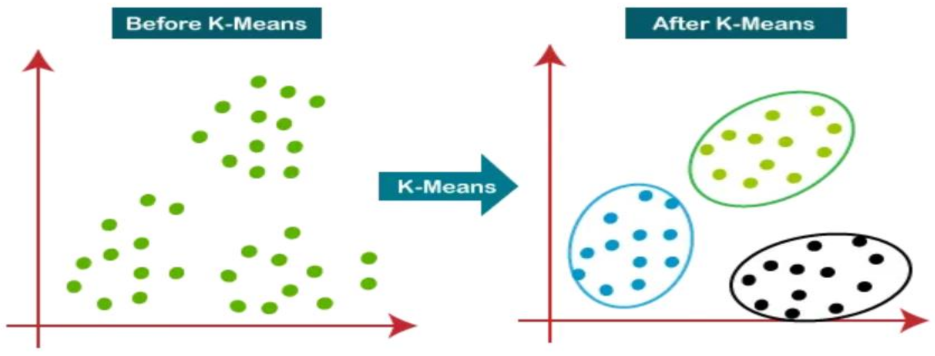
For this clustering we are going to use Kmeans clustering method:

K-mean clustering :

K-Means is an unsupervised machine learning algorithm used for clustering data points into K distinct groups based on their similarity. It works by iteratively minimizing the variance within each cluster.

Iterative optimization:

- First , we will initialized the centroids of clusters randomly
- Iterate until the data points are converged:
  >update the cluster centroids : assign the points to the nearest cluster centroid
  >update the centroids of the clusters: update the centroid to the data center of each cluster



How do we choose the optimal value for the k in K-Means clustering ?
Choosing the optimal value of **k** in K-Means clustering is crucial to obtaining meaningful clusters. Here are some common methods:

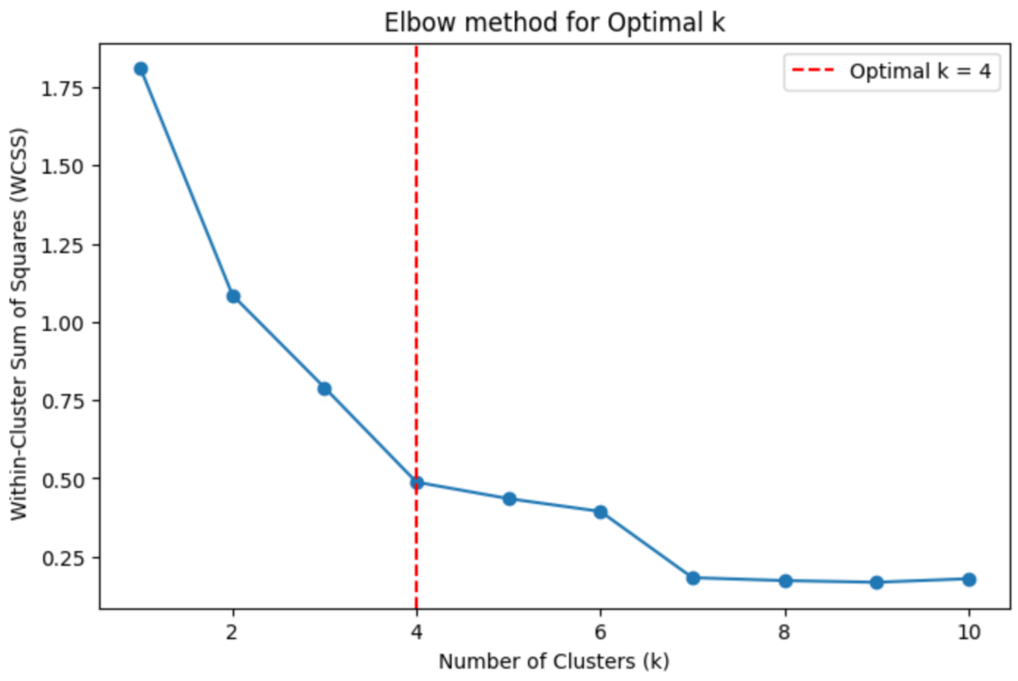
- Elbow method with WCSS
- Elbow method with variance
- Silhouette Score method
- Gap Statistics and many more….

In this task we going task we are going to use the two variations of the elbow method:

Elbow method with WCSS:

In this method we have to calculate the WCSS(Within Cluster Sum of Squares), Which is the average distance between the data points and the centroid of the cluster in each cluster , by varying the value k . Now we have to plot the values in a graph .The optimal k value is where the WCSS starts decreasing at a slower rate, forming an "elbow."

In order to the optimal k value mathematically, we have to draw a line that passes through the first and last data points of the graph, and we have to measure the perpendicular distances of the data points from the line . The data point which has the maximum distance gives us the value for the optimal k.



Elbow method with variance:

In this we have to use the variance instead of WCSS, first we have to calculate the variance in each cluster which is the same as WCSS of the cluster . next we have to calculate the ratio called explained variance which is given by:

Explained Variance=(Total Variance−Within-Cluster Variance/Total Variance)

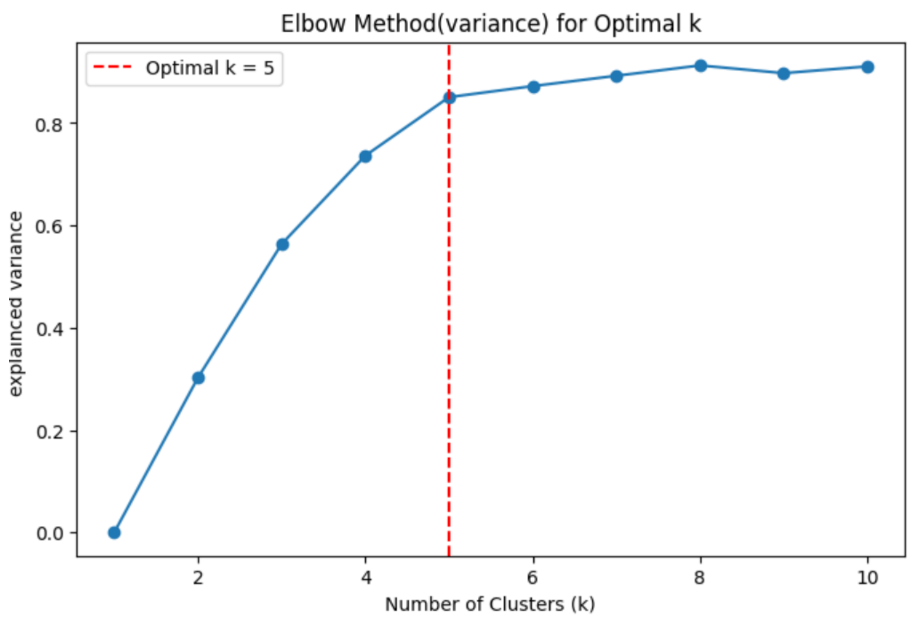
Once we have calculated the variances value for values of K , we have to plot the data points in a graph . The optimal k value is where the Variance starts decreasing at a slower rate, forming an "elbow."

In order to the optimal k value mathematically, we have to draw a line that passes through the first and last data points of the graph, and we have to measure the perpendicular distances of the data points from the line . The data point which has the maximum distance gives us the value for the optimal k.

$$\text{Explained Variance} = \frac{\text{Total Variance} - \text{Within-Cluster Variance}}{\text{Total Variance}}$$

$$TV = \sum_{j=1}^{d} \text{Var}(X_j) \qquad WCV = \sum_{i=1}^{k} \sum_{x \in C_i} ||x - \mu_i||^2$$

- Once we have calculated the optimal value for the K we have to cluster the data points using our K-means clustering.



Accuracy:

Since we have now clustered the data points , we can measure the accuracy of our clustering (how well the data points have clustered) using some metrics such as :

1. Internal Metrics (Intrinsic):

These metrics do not require ground truth labels and evaluate the clustering structure.

- Inertia (Sum of Squared Errors - SSE)
- Silhouette Score
- Davies-Bouldin Index (DBI)

2. External Metrics (Extrinsic)

These metrics are used when ground true labels are available to compare clustering with actual groups.

- Adjusted Rand Index (ARI)
- Normalized Mutual Information (NMI)

Since in our case the usage of the ground truths is not allowed . we have to use one of the intrinsic metrics, and we are going to select Silhouette Score as the metric.

Silhouette Score:

The Silhouette score is a metric used to evaluate how good clustering results are in data clustering. This score is calculated by measuring each data point’s similarity to the cluster it belongs to and how different it is from other clusters.
The Silhouette score ranges from -1 to +1, where, a score of +1 indicates that the data points are well-clustered , a score of zero (0) indicates that the data points are near the edges of the clusters , and finally a score of -1 indicates that the points are not well clustered.
Procedure for calculating the silhouette score :

In order to calculate the silhouette score for a data point , we must calculate two values for the data point one value[a(i)] representing the cohesion and another value representing separation[b(i)]

cohesion: adhesion represents the similarity of the data point to the data points of same cluster

Calculation of cohesion:

Cohesion is calculated by averaging the distances from the data point to other data points within the cluster

Separation: separation represents the variation of the data point with other cluster

Calculation of separation:

Separation is calculated by taking the max of averages of the distances from the data point to the data points of other specific clusters .

Calculation of Silhouette score:

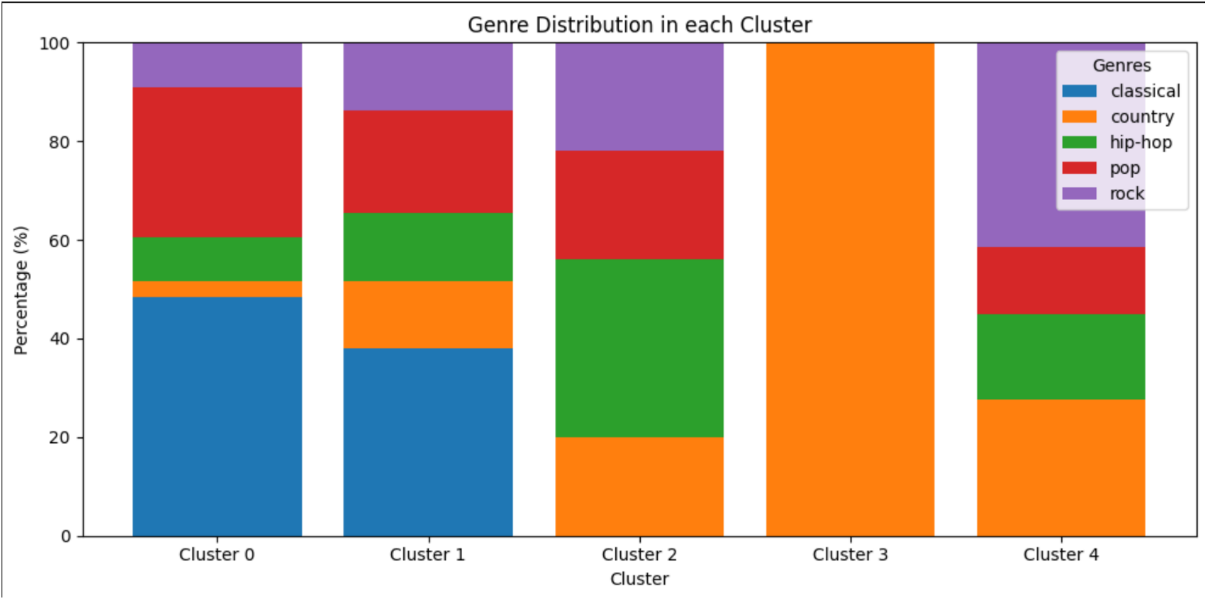
Silhouette score is calculated by dividing the difference of b(i) and a(i) by max(a(i),b(i)).

$$s = \frac{b-a}{\max(a,b)}$$

Analysis of Dataset:

Percentage distribution of music genres in each cluster:

Since we have done our clustering part now we have to plot distributions of the genres in each cluster . By doing so, we can visually represent how well the clusters overlapped with our ground truths .



Prediction:

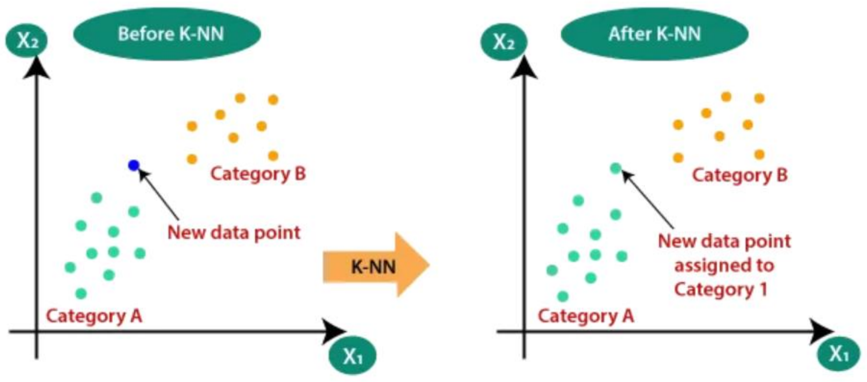
Now that we have trained the model with the given dataset we can predict the genres of some new songs if the keywords are given , for this task we are gonna need a predictive algorithm . In this task , we are gonna use the KNN(K-Nearest Neighbors) algorithm for the prediction part .

KNN:

The K-Nearest Neighbors (KNN) algorithm is a powerful supervised learning algorithm used for classification and regression tasks. It works by finding the K-closest data points to a given test point and making predictions based on the majority class in case of classification.

Procedure:

- Choose the value for K
- Calculate the distance between the test point and all training points(here we are using the euclidean distance).
- Select the first K smallest distances in order to select the K - nearest neighbors.
- In case of classification , assign the data point the most common class among the K neighbors.



Now since our KNN model is ready , we must pass the reduced vectors of the new songs and ground labels into the model and predict the genres for the songs .

Conclusion:

In this report, we explored many techniques to extract insights from the given dataset and effectively group similar songs based on their genres. We have used vectorization techniques like TF-IDF to convert keywords into numerical vectors, enabling us to process and analyze them efficiently. Dimensionality reduction through PCA helped optimize our data for visualization and clustering while preserving essential information.

To group similar songs , we used the K-means clustering algorithm ,using the elbow method to determine the optimal number of clusters we need.We calculated clustering performance using the Silhouette Score metric.Finally, we implemented the K-Nearest Neighbors (KNN) algorithm to predict song genres based on their keywords.

References:

- TF-IDF from Scratch in Python
- An Introduction to Bag of Words (BoW)
- The Math Behind Principal Component Analysis (PCA)
- Build K-Means from Scratch
- The Elbow Method: Finding the Optimal Number of Clusters
- KNN Classifier from scratch
- How to Evaluate the Performance of Clustering Algorithms Using Silhouette Coefficient

Executive Summary :

This report provides an in-depth analysis of the methods used to extract insights from a given dataset containing song information, including keywords representing instruments, mood, and style. The objective was to group similar songs and predict their genres based on keyword patterns.

The approach involved data preprocessing, where unnecessary columns were dropped, and keywords were vectorized using **TF-IDF** for better numerical representation. **Dimensionality reduction** was achieved through **Principal Component Analysis (PCA)** to optimize visualization and clustering.

For **clustering**, the **K-Means algorithm** was applied, with the **elbow method** used to determine the optimal number of clusters. The **Silhouette Score** metric was employed to evaluate clustering accuracy.

In the **prediction phase**, the **K-Nearest Neighbors (KNN) algorithm** was implemented to classify new songs based on their keywords, using Euclidean distance to determine the closest matches.

Overall, this process enabled effective grouping and classification of songs, demonstrating a structured approach to **unsupervised learning (clustering) and supervised learning (prediction)** in the given dataset.