LAB-2 HARSHINI KANAPARTHI(112687951)

YouTubeLInk

https://youtu.be/EiPsVVyUtfY

DataSet
The dataset chosen for this visualisation project is the audio-analysis of the top 500 most Streamed tracks on Spotify till date. The data was extracted using a custom written script in PHP taking help of the Spotify official REST APIs.
Top 500 Songs playlist used :
https://open.spotify.com/playlist/2YRe7HRKNRvXdJBp9nXFza?si=huyNFAKlR4KsKW3usdqiWg

API for extracting songs from the playlist :
https://developer.spotify.com/documentation/web-api/reference/playlists/get-playlist/

Spotify was generous enough to provide song Analysis data through their API :
https://developer.spotify.com/documentation/web-api/reference/tracks/get-audio-features/

Scope of the dataset
The dataset has good number of variables with each variable contributing varied data points. Also, the dataset seemed appropriate to visualise the musical patterns which go into making a hit song.

Parameters/Attributes
The following variables are used

- Danceability
- Energy
- key note
- loudness (dB)
- Speechiness
- Acousticness
- Instrumentalness
- Liveness
- Valence
- tempo (BPM)
- duration (s)

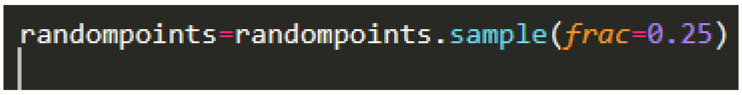
Descriptions for the above parameters can be found in
https://developer.spotify.com/documentation/web-api/reference/tracks/get-audio-features/

To begin with, the dataset has 589 total data points. It contains 11 attributes, out of which 10 are numerical and the other is categorical. Applied Label encoding to the categorical attribute and converted the values into numerical ones.

<u>Task 1: data clustering and decimation (30 points)</u>

Random Sampling

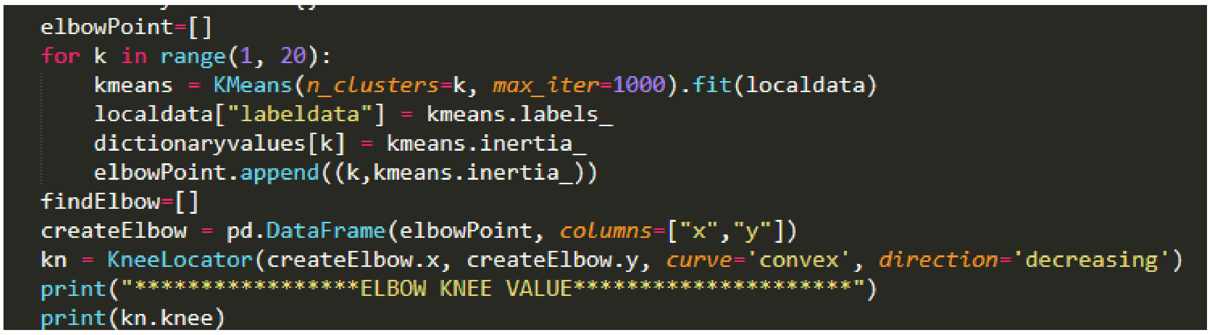
To remove the 75% of data, implemented random sampling on the original dataset. The remaining 25% of data obtained is used.



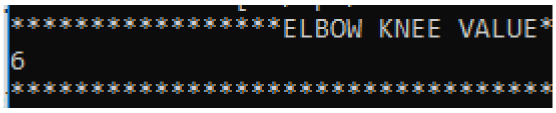
```
randompoints=randompoints.sample(frac=0.25)
```

Stratified Sampling

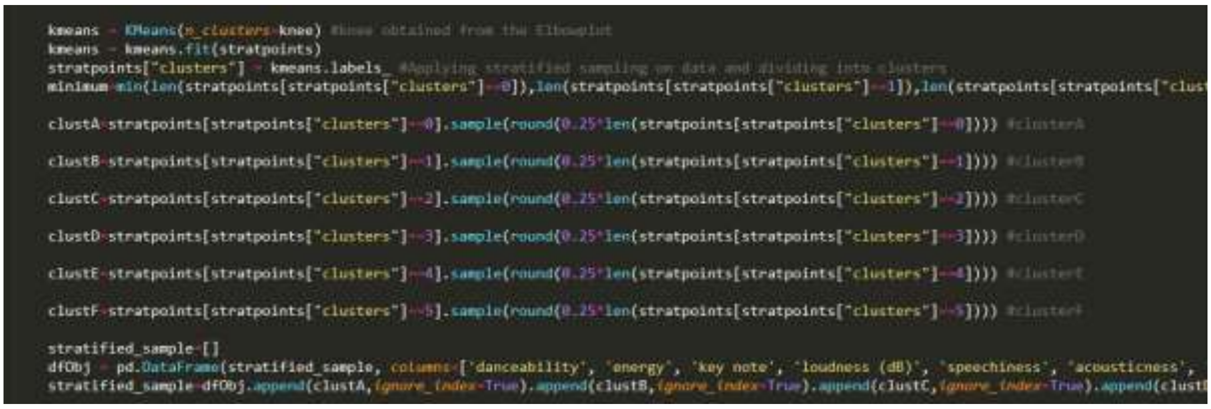
For stratified sampling, plotted graph between distortion and number of clusters. The elbow point(optimized k using elbow) is obtained as **6** which represent the optimal number of clusters. From each of the 6 clusters 25% of data is taken and combined.



```
elbowPoint=[]
for k in range(1, 20):
    kmeans = KMeans(n_clusters=k, max_iter=1000).fit(localdata)
    localdata["labeldata"] = kmeans.labels_
    dictionaryvalues[k] = kmeans.inertia_
    elbowPoint.append((k,kmeans.inertia_))
findElbow=[]
createElbow = pd.DataFrame(elbowPoint, columns=["x","y"])
kn = KneeLocator(createElbow.x, createElbow.y, curve='convex', direction='decreasing')
print("****************ELBOW KNEE VALUE********************")
print(kn.knee)
```



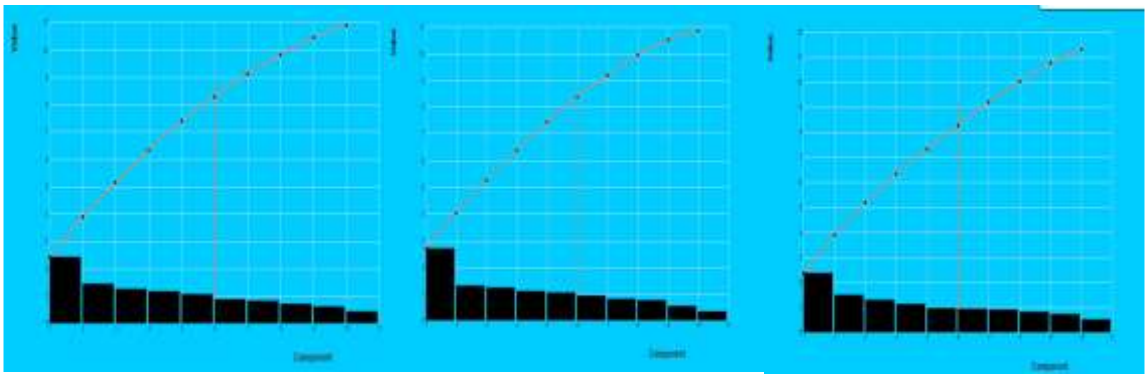
```
****************ELBOW KNEE VALUE*
6
*********************************
```



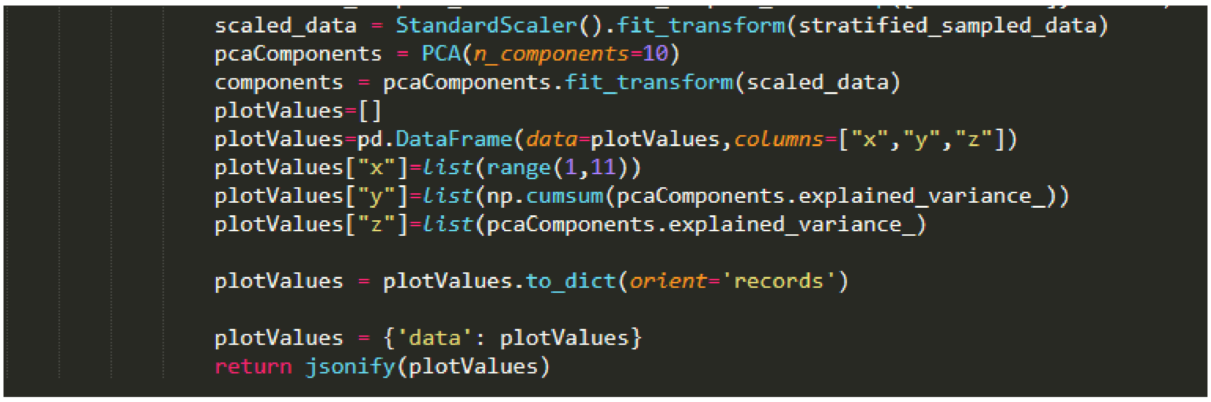
```
kmeans = KMeans(n_clusters=knee) #knee obtained from the Elbowplot
kmeans = kmeans.fit(stratpoints)
stratpoints["clusters"] = kmeans.labels_ #Applying stratified sampling on data and dividing into clusters
minimum=min(len(stratpoints[stratpoints["clusters"]==0]),len(stratpoints[stratpoints["clusters"]==1]),len(stratpoints[stratpoints["clus

clustA=stratpoints[stratpoints["clusters"]==0].sample(round(0.25*len(stratpoints[stratpoints["clusters"]==0]))) #clusterA

clustB=stratpoints[stratpoints["clusters"]==1].sample(round(0.25*len(stratpoints[stratpoints["clusters"]==1]))) #clusterB

clustC=stratpoints[stratpoints["clusters"]==2].sample(round(0.25*len(stratpoints[stratpoints["clusters"]==2]))) #clusterC

clustD=stratpoints[stratpoints["clusters"]==3].sample(round(0.25*len(stratpoints[stratpoints["clusters"]==3]))) #clusterD

clustE=stratpoints[stratpoints["clusters"]==4].sample(round(0.25*len(stratpoints[stratpoints["clusters"]==4]))) #clusterE

clustF=stratpoints[stratpoints["clusters"]==5].sample(round(0.25*len(stratpoints[stratpoints["clusters"]==5]))) #clusterF

stratified_sample=[]
dfObj = pd.DataFrame(stratified_sample, columns=['danceability', 'energy', 'key note', 'loudness (dB)', 'speechiness', 'acousticness',
stratified_sample=dfObj.append(clustA,ignore_index=True).append(clustB,ignore_index=True).append(clustC,ignore_index=True).append(clust
```

<u>Task 2: dimension reduction on both org and 2 types of reduced data (30)</u>

The scree plots are plotted using d3.js. Used the PCA component till which we capture at least 75% of the information. The line plots in d3 are used to mark/indicate the intrinsic dimensionality.



After going through the data and plots generated, Same component has captured atleast 75% of the variance. The scree plots before/after sampling to assess the bias introduced are almost similar to each other. Hence the bias introduced isn’t very high.



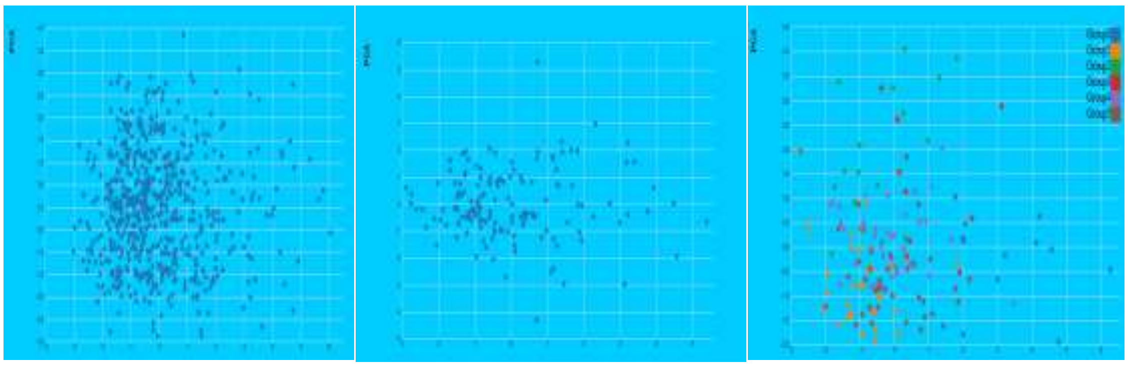
```
                scaled_data = StandardScaler().fit_transform(stratified_sampled_data)
                pcaComponents = PCA(n_components=10)
                components = pcaComponents.fit_transform(scaled_data)
                plotValues=[]
                plotValues=pd.DataFrame(data=plotValues,columns=["x","y","z"])
                plotValues["x"]=list(range(1,11))
                plotValues["y"]=list(np.cumsum(pcaComponents.explained_variance_))
                plotValues["z"]=list(pcaComponents.explained_variance_)

                plotValues = plotValues.to_dict(orient='records')

                plotValues = {'data': plotValues}
                return jsonify(plotValues)
```

Sum Of Squared Loadings of the pca components on each of the attributes is determined and the values are sorted. The top 3 values are selected. 'danceability', 'speechiness' and 'energy' are the three attributes with highest PCA loadings

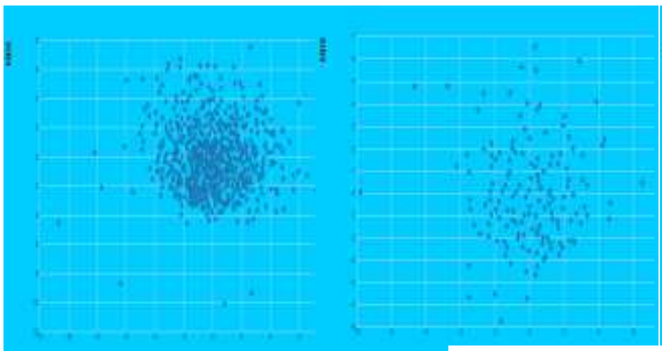
<u>Task 3: visualization of both original and 2 types of reduced data (40 points)</u>

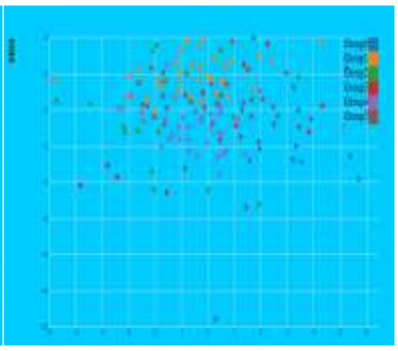
Used the sklearn library and applied PCA. The n_components is chosen as 2 and obtained the top 2 PCA vectors. A 2D scatter plot is drawn for Original, Random and stratified data.

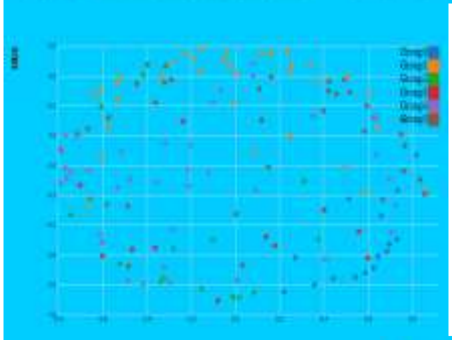
2D PCA Scatter Plot

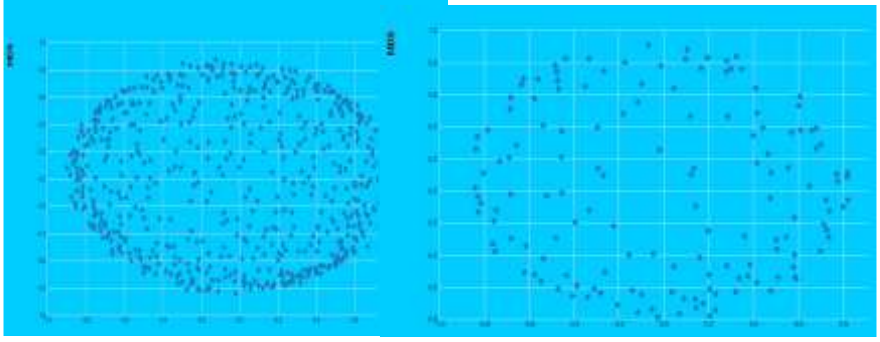


2D MDS Euclidean Scatter Plot



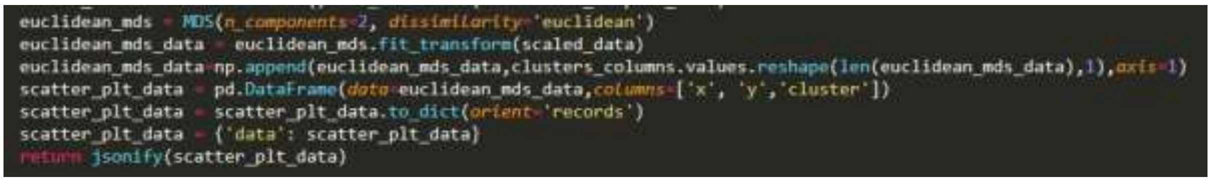




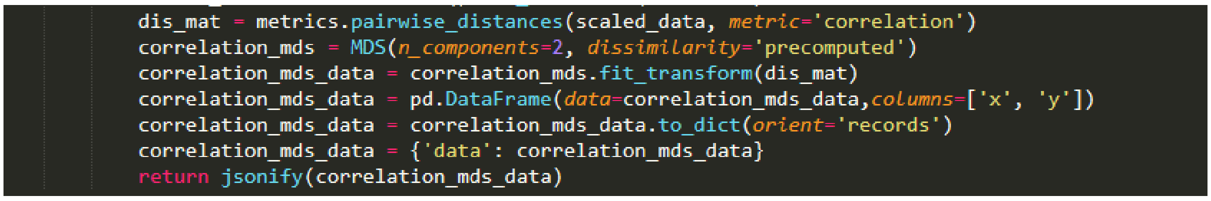


2D MDS Correlation Scatter Plot

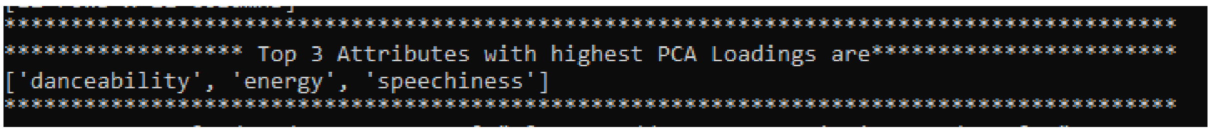
MDS attempts to preserve pairwise distances. It constructs a configuration of n points in Euclidian space by using the information about the distances between the n patterns. It is a means of visualizing the level of similarity of individual cases of a dataset. Visualized The data via MDS (Euclidian & correlation distance) in 2D scatterplots. Used the MDS function by choosing n_components=2 and dissimilarity as 'euclidean' and ‘correlation’.



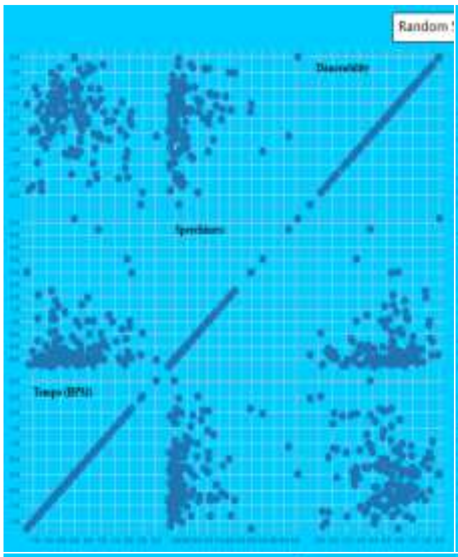
```
euclidean_mds = MDS(n_components=2, dissimilarity='euclidean')
euclidean_mds_data = euclidean_mds.fit_transform(scaled_data)
euclidean_mds_data=np.append(euclidean_mds_data,clusters_columns.values.reshape(len(euclidean_mds_data),1),axis=1)
scatter_plt_data = pd.DataFrame(data=euclidean_mds_data,columns=['x', 'y','cluster'])
scatter_plt_data = scatter_plt_data.to_dict(orient='records')
scatter_plt_data = {'data': scatter_plt_data}
return jsonify(scatter_plt_data)
```

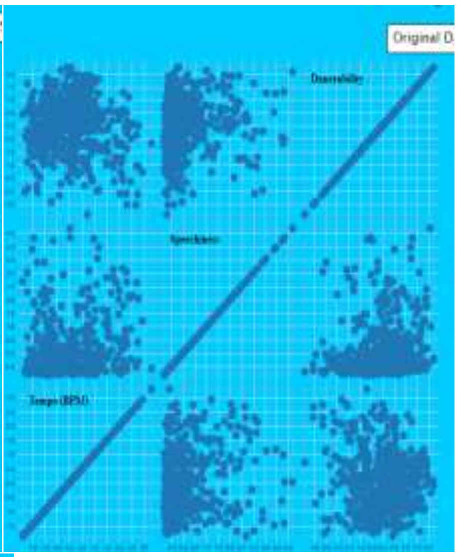


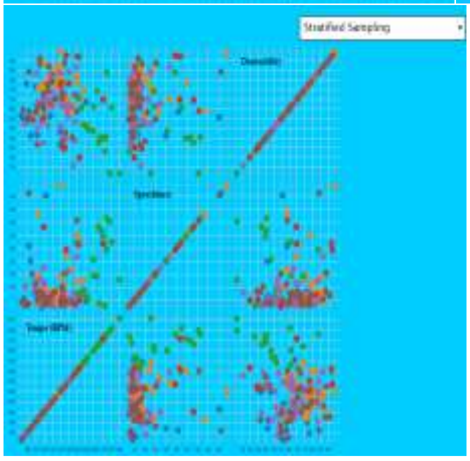
```
dis_mat = metrics.pairwise_distances(scaled_data, metric='correlation')
correlation_mds = MDS(n_components=2, dissimilarity='precomputed')
correlation_mds_data = correlation_mds.fit_transform(dis_mat)
correlation_mds_data = pd.DataFrame(data=correlation_mds_data,columns=['x', 'y'])
correlation_mds_data = correlation_mds_data.to_dict(orient='records')
correlation_mds_data = {'data': correlation_mds_data}
return jsonify(correlation_mds_data)
```


```
**************************************************************************************
***************** Top 3 Attributes with highest PCA Loadings are***********************
['danceability', 'energy', 'speechiness']
**************************************************************************************
```

Obtained the three highest PCA loaded attributes and visualized them with scatterplot matrix. Visualized both original and the reduced data.







In terms of Visualization, Scree plot are used to find the top components with maximum variance. The bias is identified using the the scatter plots which are plotted on both original and reduced data. All the plots helps us to find the difference between original and reduced data. A little bias is introduced after the sampling. The difference can be observed from their plots.