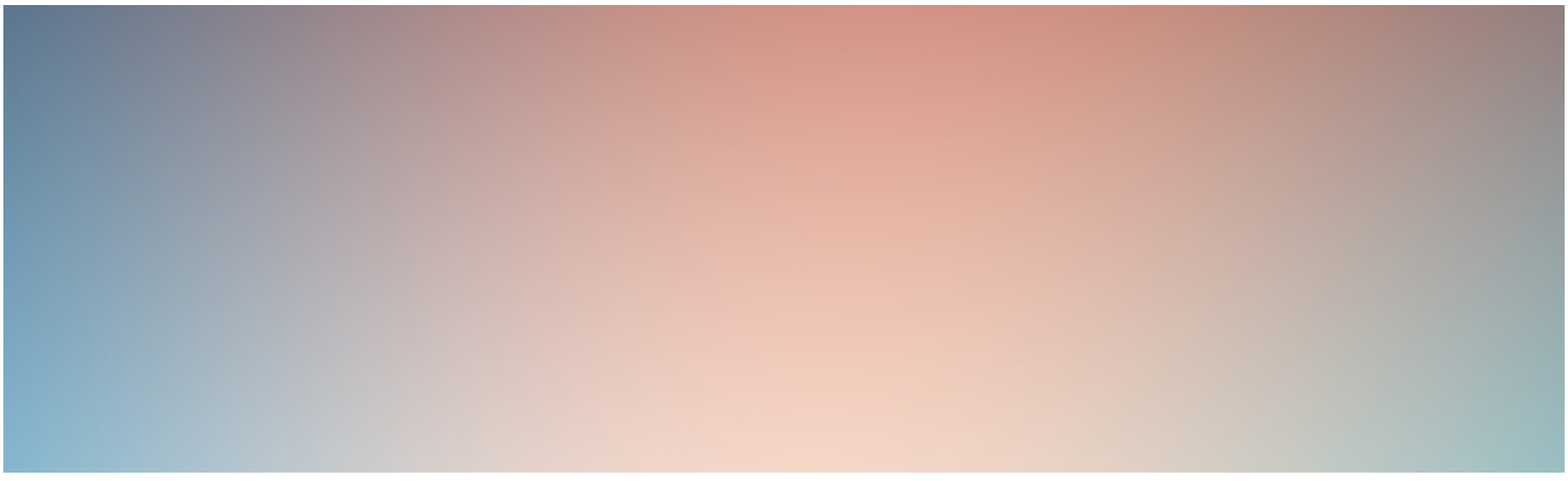


PROJECT REPORT

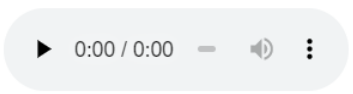
PROJECT: MUSIC GENRE CLASSIFICATION

DIVYANSHI SINGH BORA (B20EE018)

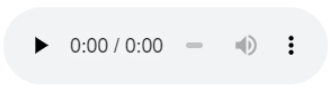
GHELANI SHUBHAM BHAVESHBHAI (B20EE019)

- QUESTION: To automatically classify different musical genres from audio files.
- ***About the dataset:***
  - The input provided is an audio file representing a particular genre of music ('pop', 'blues', 'hip hop', 'classical', 'disco', 'metal', 'country', 'blues', 'reggae', 'jazz', 'rock').
  - We are also provided with the spectrogram of each audio file provided in the dataset.
  - Here below is a sample audio file of each genre provided in the dataset.

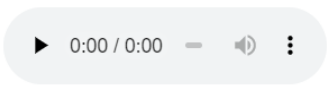
- genre: blue



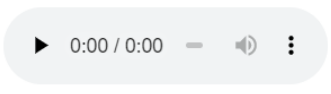
- genre: classical



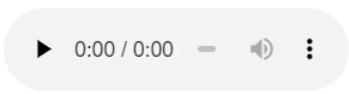
- genre: country



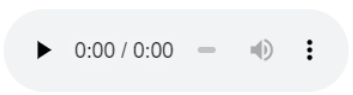
- genre: disco



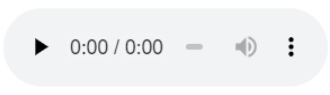
- genre: hip hop



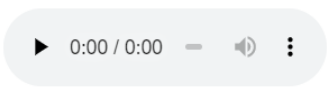
- genre: jazz



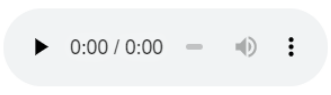
- genre: metal



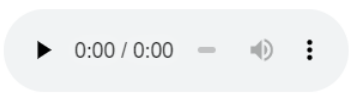
- genre: pop



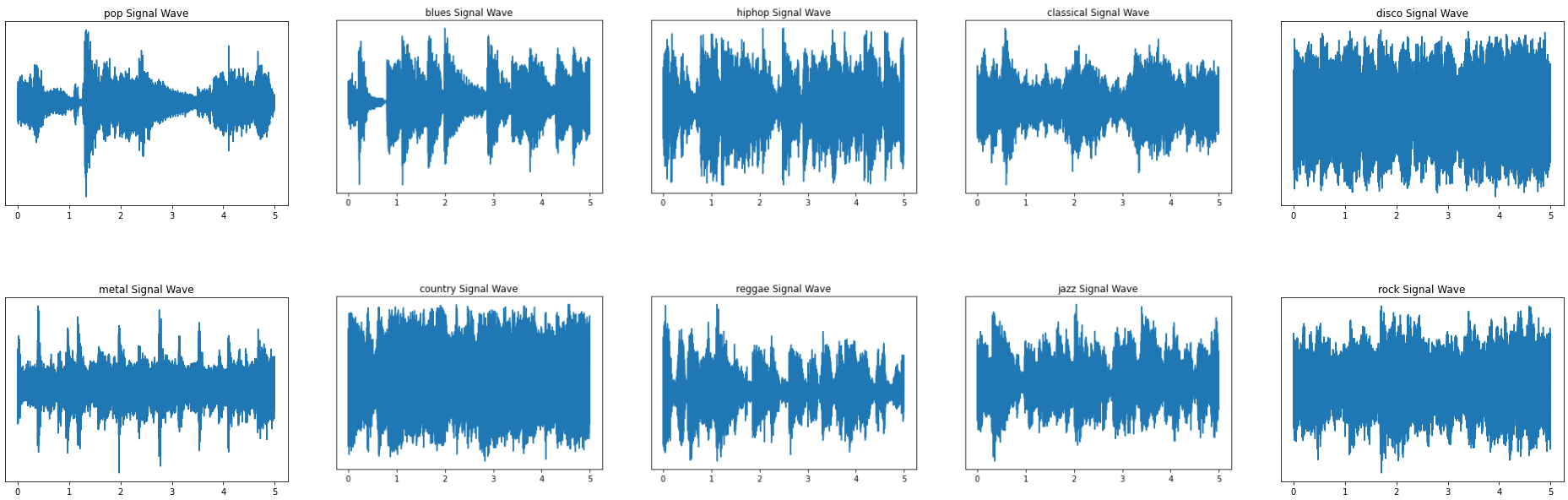
- genre: reggae



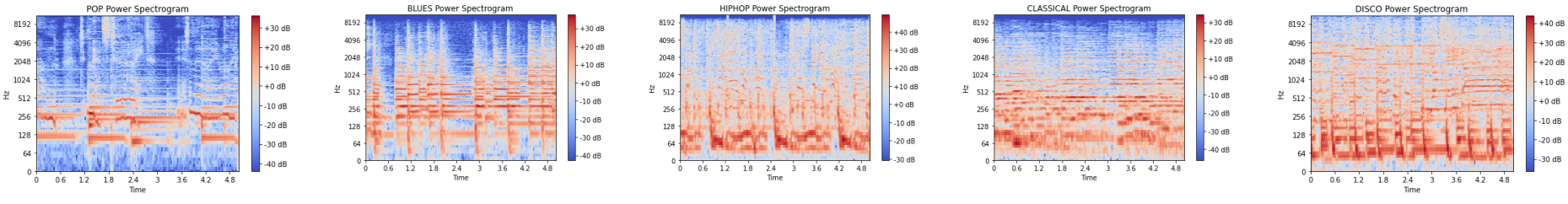
- genre: rock

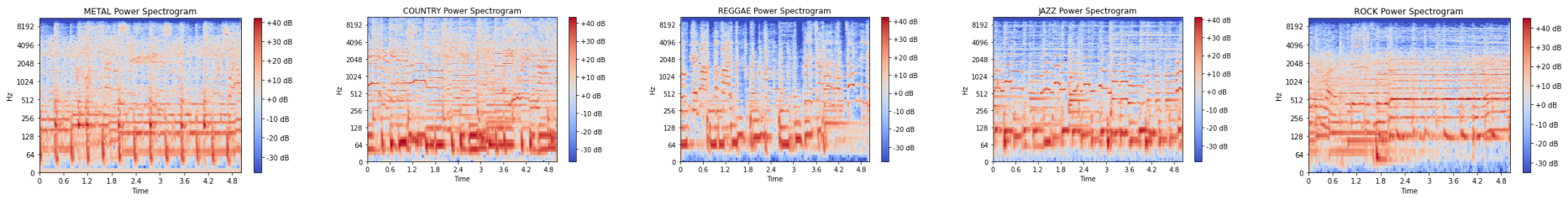


- ***Visualization of the dataset:***
  - The given dataset is visualized with the help of the library librosa into a graph format so as to see that each genre yield a very different and distinctive graph which has characteristic of its own.
  - Graphs also help us to distinguish each genre.



- ***Visualization of the dataset:***
  - The dataset is visualized in a power spectrogram so as to study each genre in detail and get a better understanding.
  - A spectrogram is a visual way of representing the signal strength, or “loudness”, of a signal over time at various frequencies present in a particular waveform.





- ***Data extraction:***
  - To extract features and compile them into a dataset from the given audio files.
  - All the features from the audio files have been extracted with the help of the library librosa.
  - Features such as:

```
'chroma_stft_mean', 'rms_mean', 'spectral_centroid_mean',
'spectral_bandwidth_mean', 'rolloff_mean', 'zero_crossing_rate_mean',
'tempo', 'mfcc1_mean', 'mfcc2_mean', 'mfcc3_mean', 'mfcc4_mean',
'mfcc5_mean', 'mfcc6_mean', 'mfcc7_mean', 'mfcc8_mean', 'mfcc9_mean',
'mfcc10_mean', 'mfcc11_mean', 'mfcc12_mean', 'mfcc13_mean',
'mfcc14_mean', 'mfcc15_mean', 'mfcc16_mean', 'mfcc17_mean',
'mfcc18_mean', 'mfcc19_mean', 'mfcc20_mean'
```

  - After extracting all the feature they are appended in an array which late on converted into a dataframe.
  - Then after having the dataframe we did save it in the drive in .csv format.
  - A sample dataset extracted for an audio of blue genre is shown below:

features extracted from the audio file given:

index	chroma_stft_mean	rms_mean	spectral_centroid_mean	spectral_bandwidth_mean	rolloff_mean	zero_crossing_rate_mean	tempo	mfcc1_mean	mfcc2_mea
0	0\.3875771164894104	0\.1291273534297943	1996\.327664393569	2097\.080527547891	4227\.935352015661	0\.0894592371622534	123\.046875	-116\.60436248779297	109\.25070190

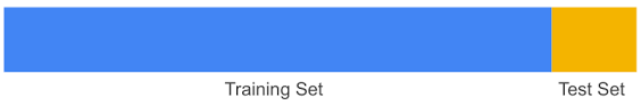
- ***Preprocessing our dataset:***
  - Our dataset is preprocessed with the help of the library label encoder.
  - This encodes our ordinal data items into numerical data by encoding them or assigning a unique integer value to each unique data.
  - Label Encoding also refers to converting the labels into a numeric form so as to convert them into the machine-readable form.

```
'pop':0,
'blues':1,
'hiphop':2,
'classical':3,
'disco':4,
'metal':5,
'country':6,
'blues':7,
'reggae':8,
'jazz':9,
'rock':10
```

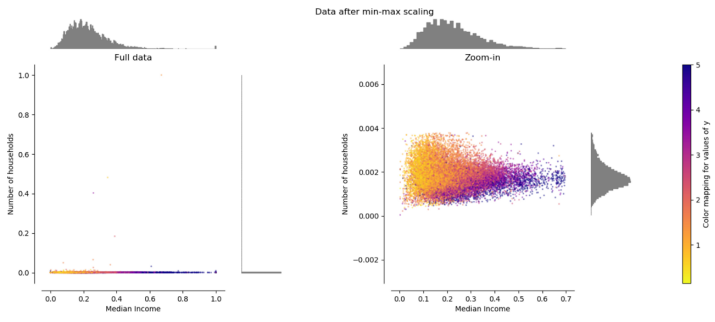
  - It was also necessary to drop some unwanted features from our dataset so as to extract only relevant information from our dataset.
  - The dropped features are:

```
"filename","length","chroma_stft_var","perceptr_var",
"spectral_bandwidth_var", "spectral_centroid_var", "rolloff_var",
"zero_crossing_rate_var","harmony_var","rms_var","harmony_var",
"mfcc1_var","mfcc2_var","mfcc3_var","mfcc4_var", "mfcc5_var",
"mfcc6_var", "mfcc7_var","mfcc8_var","mfcc9_var","mfcc10_var",
"mfcc11_var","mfcc12_var","mfcc13_var","mfcc14_var", "mfcc15_var",
"mfcc16_var", "mfcc17_var","mfcc18_var","mfcc19_var","mfcc20_var",
"harmony_mean", "perceptr_mean"
```

- ***Data splitting:***
  - Splitting our given dataset into testing and training dataset.
  - The dataset is splitted with 30% of the dataset goes to testing data and 70% of the the dataset to the training part.
  - Training data is the initial dataset you use to teach a machine learning application to recognize patterns or perform to your criteria, while testing or validation data is used to evaluate your model's accuracy.



- ***Normalising the dataset:***
  - The dataset is normalised using minmaxscalar.
  - Transform features by scaling each feature to a given range. This estimator scales and translates each feature individually such that it is in the given range on the training set, e.g. between zero and one.
  - MinMaxScaler. For each value in a feature, MinMaxScaler subtracts the minimum value in the feature and then divides by the range. The range is the difference between the original maximum and original minimum. MinMaxScaler preserves the shape of the original distribution.



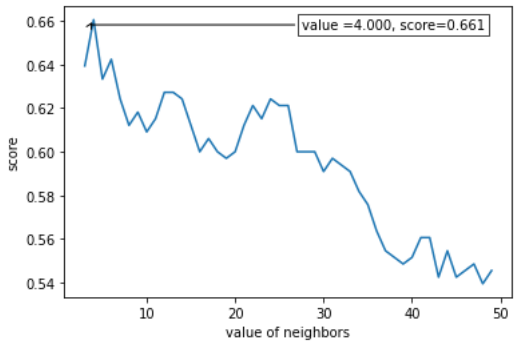
- ***Training and testing the different model:***
  - Model 01: *KNeighborsClassifier*
    - K-nearest neighbors (KNN) algorithm is a simple, supervised machine learning algorithm that can be used to solve both classification and regression problems.
    - The accuracy produced after training and test dataset in this model is:

```
64.24242424242425
```

    - After hyperparameter tuning the accuracy turns out to be:

```
66.06060606060606
```

    - The optimum nearest neighbour value turns out to be : 4



  - Model 02: *RandomForestClassifier*
    - Random forest algorithm can be used for both classifications and regression task. It provides higher accuracy through cross validation. Random forest classifier will handle the missing values and maintain the accuracy of a large proportion of data.
    - The accuracy produced after training and test dataset in this model is:

```
69.39393939393939
```

  - Model 03: *lightgbm*
    - Light GBM is a fast, distributed, high-performance gradient boosting framework based on decision tree algorithm, used for ranking, classification and many other machine learning tasks.
    - The accuracy produced after training and test dataset in this model is:

```
67.27272727272727
```

  - Model 04: *GradientBoostingClassifier*
    - Gradient boosting is a machine learning technique used in regression and classification tasks among others. It gives a prediction model in the form of an ensemble of weak prediction models, which are typically decision trees.
    - The accuracy produced after training and test dataset in this model is:

```
63.63636363636363
```

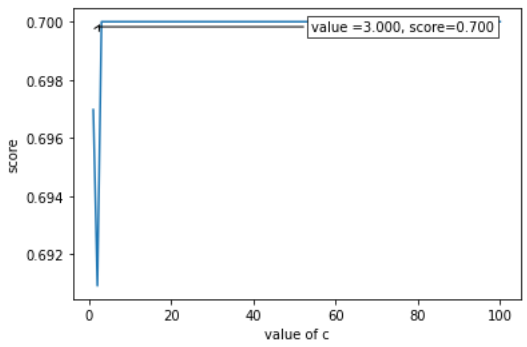
  - Model 05: *support vector machine kernel: quadratic*
    - A new quadratic kernel-free non-linear support vector machine (which is called QSVM) is introduced. The SVM optimization problem can be stated as follows: Maximize the geometrical margin subject to all the training data with a functional margin greater than a constant.
    - The accuracy produced after training and test dataset in this model is:

```
70.0
```

    - After hyperparameter tuning the accuracy turns out to be:

```
70.00
```

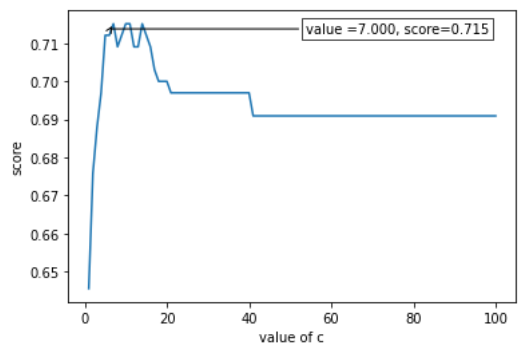
    - The optimum value of C turns out to be : 3



- Model 06: *support vector machine kernel: rbf*
  - RBF Kernel is popular because of its similarity to K-Nearest Neighborhood Algorithm. It has the advantages of K-NN and overcomes the space complexity problem as RBF Kernel Support Vector Machines just needs to store the support vectors during training and not the entire dataset.
  - After hyperparameter tuning the accuracy turns out to be:

```
71.51515151515152
```

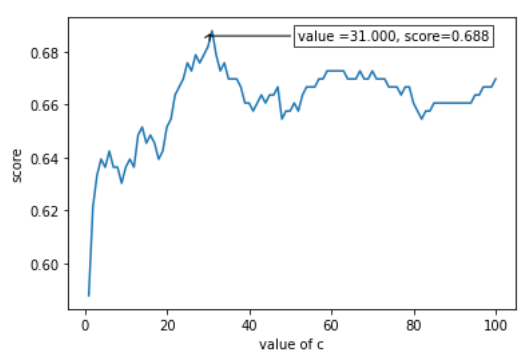
  - The optimum value of C turns out to be : 7



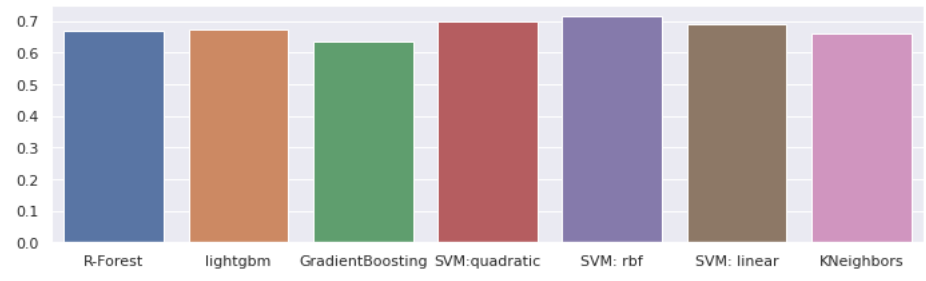
- Model 07: *support vector machine kernel: linear*
  - Linear Kernel is used when the data is Linearly separable, that is, it can be separated using a single Line. It is one of the most common kernels to be used. It is mostly used when there are a Large number of Features in a particular Data Set.
  - After hyperparameter tuning the accuracy turns out to be:

```
68.78787878787878
```

  - The optimum value of C turns out to be : 31

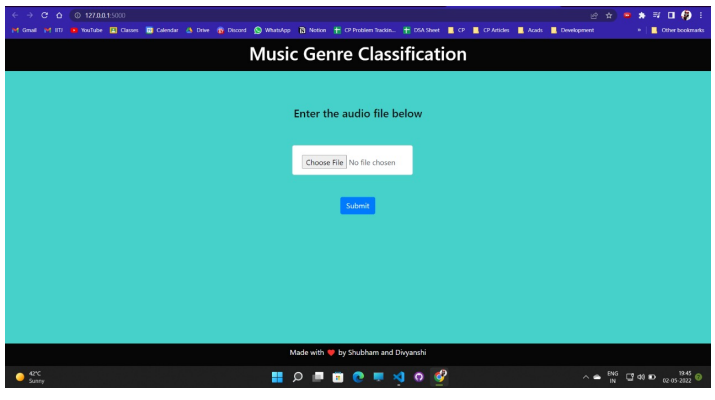


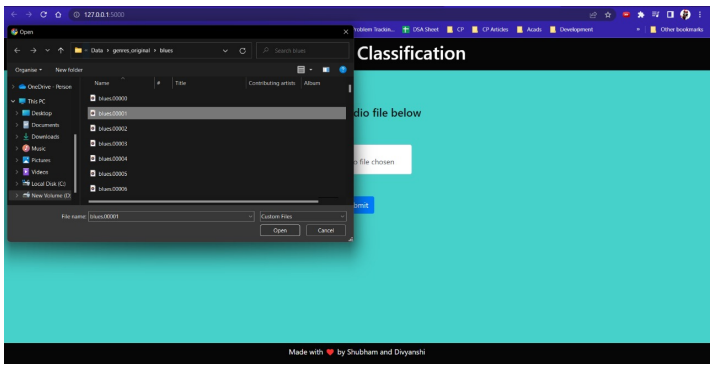
- ***Predicting the best model:***
  - Storing all the accuracies in a dictionary and comparing the accuracy of each model until we find the best model which gives us the maximum accuracy.
  - To visualize the best accuracy we have used seaborn library to get a better understanding of our dataset and the predictions.

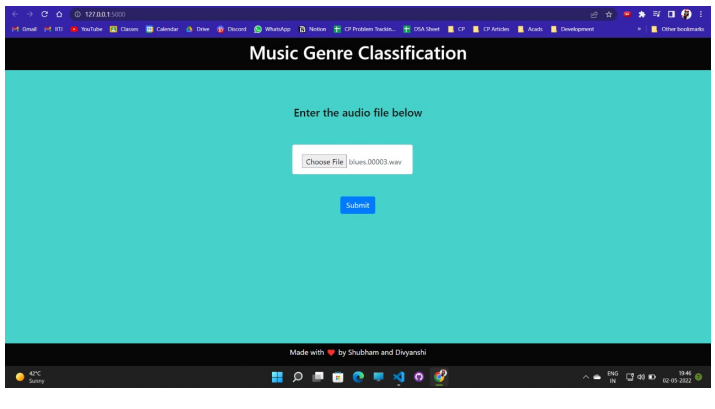


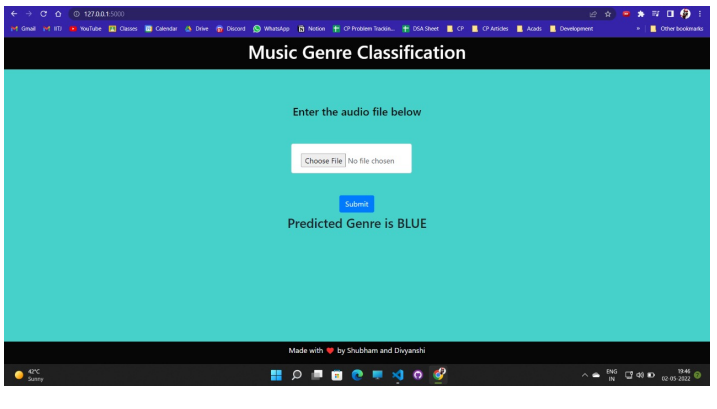
  - Looking carefully at the we can see that svm: rbf has turned out to score the maximum frequency.
  - We finally conclude that our best model is svm: rbf.

- ***Deployment:***
  - A flask API endpoint was created and hosted on a local webserver. The API endpoint accepts a request including a list of flight values and provides an estimated pricing.
  - TeckStacks used - Python, Flask, HTML, CSS
  - There is an option to upload an audio file. Users can upload music files preferably in mp3 format and then you can press submit button. The genre of the music will be predicted and the results will be shown below the submit button.
  - In the backend part of the code, the first features will be extracted out of the music audio file that is given by user. Features are then converted into CSV format and preprocessed so that it can be used accordingly for predicting the results. After the data is ready it will be used to predict the result using the machine learning model that we have trained.
  - Thus after this, you can see the results on the web app.
  - Screenshots of deployment work are shown below :









- ***Contributions:***
  - *DIVYANSHI SINGH BORA:*
    - extraction of dataset from the given audio files and converting it into .csv file.
    - plotting the graphs of hyperparameter tuning with maximum score.
    - visualization of the dataset and accuracy.
    - making of the report.
  - *GHELANI SHUBHAM BHAVESHBHAI:*
    - Preprocessing our dataset obtained.
    - Training and testing of the model.
    - Deployment
- ***Reference:***
  - Sklearn library
  - Data Visualization using Python for Machine Learning and Data science|by sanat|towards data science
  - Cross validation in machine learning | geeks for geeks
  - https://scikit-learn.org/stable/modules/grid_search.html
  - https://analyticsindiamag.com/guide-to-hyperparameters-tuning-using-gridsearchcv-and-randomizedsearchcv