National College of Ireland

Configuration Manual

MSc Research Project
Data Analytics

Naveenreddy Konreddy
X22170219

School of Computing
National College of Ireland

Supervisor: Furqan Rustam

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: ……. ……Naveenreddy Konreddy……

Student ID: ……x22170219……

Programme: ……Data Analytics…… **Year:** ……2023……..

Module: ……Research Project…….……

Lecturer: ……Furqan Rustam……

Submission Due Date: ……14/12/2023……

Project Title: Enhancing music recommendations with meachine learning techniques

Word Count: ……672…… **Page Count:** ……10…….……

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ……**NAVEENREDDY KONREDDY**……

Date: ……14/12/2023……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Naveenreddy Konreddy
X22170219

1. Introduction

This manual demonstrates all the instructions on setting up and executing the code for the code implementation of enhancing music recommendations using neural networks on a large-scale dataset. The application is implemented in Python and incorporates advanced machine learning and neural network method approaches. The following sections guides through the necessary requirements configurations and tools.

2. System Specification

The classification recommendation system has been developed on these following hardware configurations:

- Process: Intel i7 generation
- Operating System: Windows 11 (Home)
- Ram: 16 GB (DDR4)
- Stroage Hard Drive: 512GB (SSD)

3. Softwares Used:

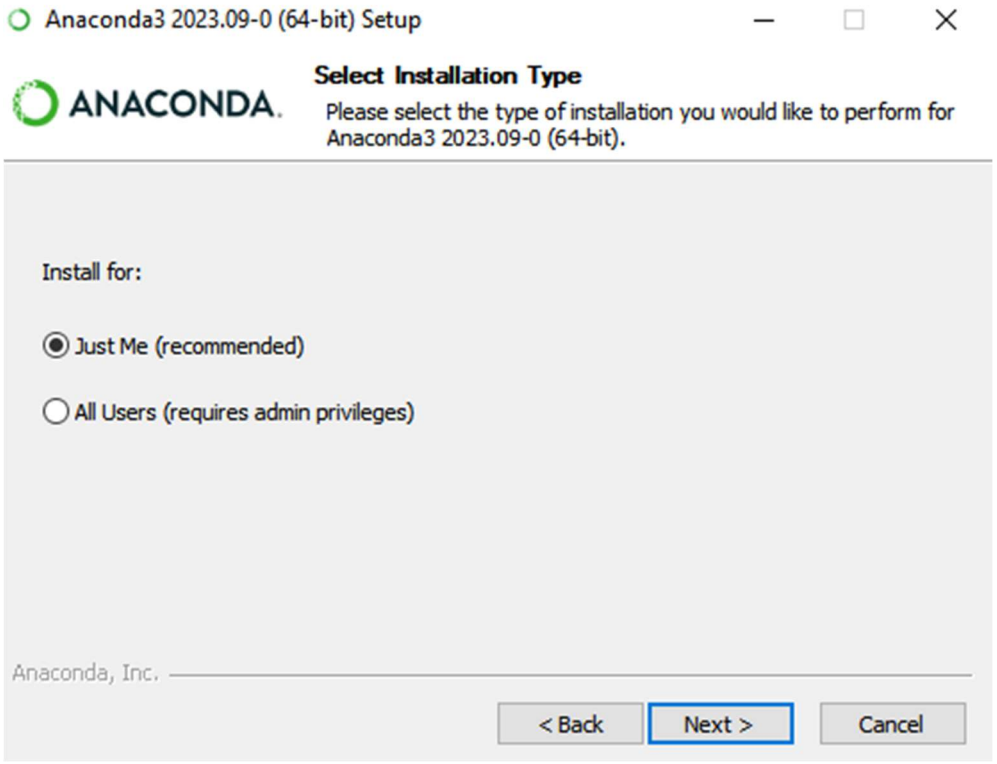
The following tools which are required to use and development for music recommendation system:

- Anaconda
- Tensorflow and Keras
- Pandas
- Numpy
- Matplotlib
- Seaborn
- Sklearn
- Jupyter

4. Installation of the Software:

- First download the anaconda from the their official website and then start installing to the operation system website: https://www.anaconda.com

- Chosen it for (Just Me) and then clicked on Next until the installation get started.



- Creates the new virtual environment for the purpose of the our application (Music Recommendation System)

```
PS D:\new\assignment_left\Music Recomendation System> virtualenv music_recommendation_evn
created virtual environment CPython3.11.4.final.0-64 in 430ms
  creator CPython3Windows(dest=D:\new\assignment_left\Music Recomendation System\music_recommendation_evn, clear=False, no_vcs
_ignore=False, global=False)
  seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy, app_data_dir=C:\Users\rohit\AppDat
a\Local\pypa\virtualenv)
    added seed packages: pip==23.3.1, setuptools==68.2.2, wheel==0.41.3
  activators BashActivator,BatchActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator
PS D:\new\assignment_left\Music Recomendation System>
```

- Activate the new virtual environment and install the required packages to make the our research would get done by necessary packages.

```
PS D:\new\assignment_left\Music Recomendation System> & "d:/new/assignment_left/Music Recomendation System/music_recommendatio
n_evn/Scripts/Activate.ps1"
(music_recommendation_evn) PS D:\new\assignment_left\Music Recomendation System> pip install pandas
Collecting pandas
  Downloading pandas-2.1.4-cp311-cp311-win_amd64.whl.metadata (18 kB)
Collecting numpy<2,>=1.23.2 (from pandas)
  Downloading numpy-1.26.2-cp311-cp311-win_amd64.whl.metadata (61 kB)
     ---------------------------------------- 61.2/61.2 kB 3.2 MB/s eta 0:00:00
Collecting python-dateutil>=2.8.2 (from pandas)
  Downloading python_dateutil-2.8.2-py2.py3-none-any.whl (247 kB)
     ---------------------------------------- 247.7/247.7 kB 7.7 MB/s eta 0:00:00
Collecting pytz>=2020.1 (from pandas)
  Downloading pytz-2023.3.post1-py2.py3-none-any.whl.metadata (22 kB)
Collecting tzdata>=2022.1 (from pandas)
  Downloading tzdata-2023.3-py2.py3-none-any.whl (341 kB)
     ---------------------------------------- 341.8/341.8 kB 7.1 MB/s eta 0:00:00
Collecting six>=1.5 (from python-dateutil>=2.8.2->pandas)
  Downloading six-1.16.0-py2.py3-none-any.whl (11 kB)
Downloading pandas-2.1.4-cp311-cp311-win_amd64.whl (10.6 MB)
     ---------------------------------------- 10.6/10.6 MB 16.0 MB/s eta 0:00:00
Downloading numpy-1.26.2-cp311-cp311-win_amd64.whl (15.8 MB)
     ---------------------------------------- 15.8/15.8 MB 14.2 MB/s eta 0:00:00
Downloading pytz-2023.3.post1-py2.py3-none-any.whl (502 kB)
     ---------------------------------------- 502.5/502.5 kB 7.9 MB/s eta 0:00:00
Installing collected packages: pytz, tzdata, six, numpy, python-dateutil, pandas
```

5. Source of Dataset

Gather the large-scale music track dataset which would be suitable for training for machine learning as well as neural networks models. Datasets would be like Million Song Dataset or others available on platforms where but I used the Kaggle to chose the dataset.

6. Code Execution

Open the jupyter noteboook to start developing or modifying the .ipynb (Integrated Python Notebook) for the task from the beginning loading the dataset to evaluating the models.

Execution to Run the File:

- Loading of the dataset

1. Load the dataset for Music Recommendation in the form of dataframes with the help of Pandas.

With these DataFrames loaded, you can perform various operations such as data analysis, visualization, and comparison between different scenarios' energy data.

In [1]:
```
#imports the pandas library to handle the datasets for creating the dataframe an
import pandas as pd
```

In [2]:
```
#  The data in this DataFrame represents song data
song_dataFrame = pd.read_csv("data.csv")
```

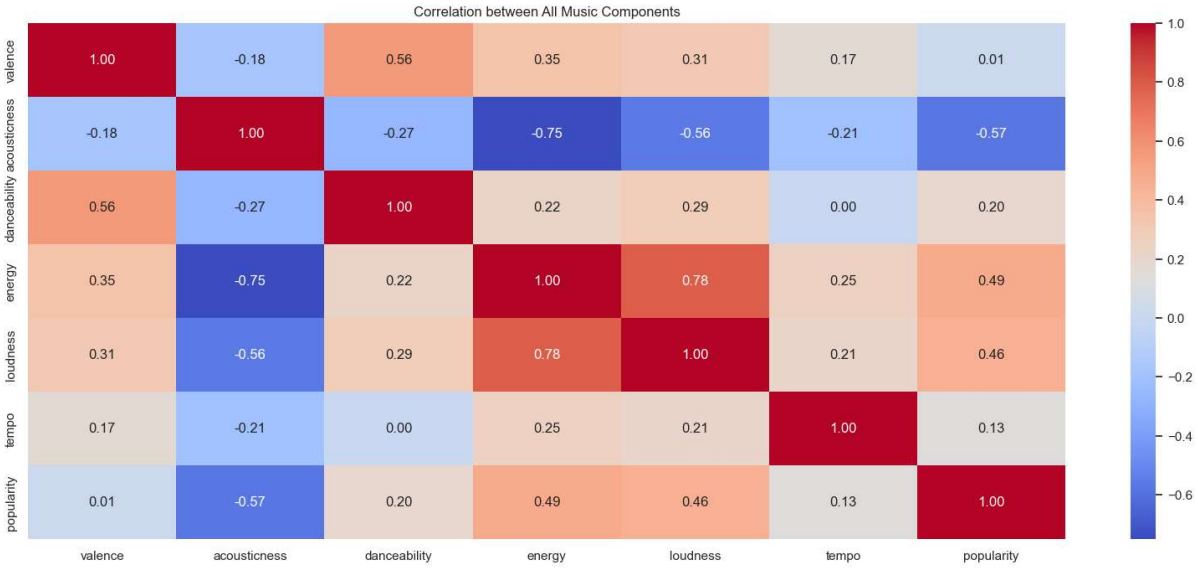
Parameters of Datasets:

Column Name	Description
valence	Positivity or happiness of the track (0.0 to 1.0)
year	Release year of the music track
acousticness	Level of acoustic sound in the track (0.0 to 1.0)
artists	Names of the artists who performed or contributed to the track
danceability	Suitability of the track for dancing (0.0 to 1.0)
duration_ms	Duration of the track in milliseconds
energy	Energy level of the track (0.0 to 1.0)
explicit	Indication of explicit or mature content (0 or 1)
id	Unique identifier for the track
instrumentalness	Proportion of instrumental music in the track (0.0 to 1.0)
key	Musical key of the track
liveness	Indicates if the track was recorded with a live audience (0.0 to 1.0)
loudness	Overall loudness of the track in decibels (dB)
mode	Modality of the track (major or minor)
name	Name or title of the music track
popularity	Popularity of the track (numerical value)
release_date	Date on which the track was released
speechiness	Presence of spoken words or speech in the track (0.0 to 1.0)
tempo	Tempo of the track in beats per minute (BPM)

- Preprocesing of the dataset

```
In [14]: # Remove columns with all NaN values if any lefted
         song_dataFrame = song_dataFrame.dropna(axis=1, how='all')

In [15]: # Remove any duplicate rows
         song_dataFrame = song_dataFrame.drop_duplicates()

In [16]: missing_values = sum(song_dataFrame.isnull().sum())
         print("Missing Values:\n", missing_values)

         Missing Values:
          0

In [17]: # Create a feature for artist popularity
         artist_popularity = song_dataFrame.groupby('artists')['popularity'].transform('m
         song_dataFrame['artist_popularity'] = artist_popularity
```

- Exploratory Data Analysis (EDA)




```
In [21]: #correlation matrix for the all attributes in the dataframe
         correlation_matrix
```

Out[21]:

	valence	acousticness	danceability	energy	loudness	tempo	popularity
valence	1.000000	-0.184101	0.558946	0.353876	0.313512	0.171689	0.014200
acousticness	-0.184101	1.000000	-0.266852	-0.749393	-0.561696	-0.207120	-0.573162
danceability	0.558946	-0.266852	1.000000	0.221967	0.285057	0.001801	0.199606
energy	0.353876	-0.749393	0.221967	1.000000	0.782362	0.250865	0.485005
loudness	0.313512	-0.561696	0.285057	0.782362	1.000000	0.209774	0.457051
tempo	0.171689	-0.207120	0.001801	0.250865	0.209774	1.000000	0.133310
popularity	0.014200	-0.573162	0.199606	0.485005	0.457051	0.133310	1.000000

- Model Selection which contains the feature selection, splitting of dataset and model initialization and model training.

In [30]:
```
#import the train_test_split module for the splitting of data
from sklearn.model_selection import train_test_split
```

In [31]:
```
song_dataFrame.columns
```

Out[31]:
```
Index(['valence', 'year', 'acousticness', 'artists', 'danceability',
       'duration_ms', 'energy', 'explicit', 'id', 'instrumentalness', 'key',
       'liveness', 'loudness', 'mode', 'name', 'popularity', 'release_date',
       'speechiness', 'tempo', 'artist_popularity', 'duration_category'],
      dtype='object')
```

In [32]:
```
# feature attributes denotes to (X) and target to (y)
X = song_dataFrame[['valence', 'year', 'acousticness', 'danceability',
       'duration_ms', 'energy', 'instrumentalness', 'key',
       'liveness', 'loudness', 'mode', 'popularity',
       'speechiness', 'tempo']]  # Exclude the target column
y = song_dataFrame['explicit']
```

In [33]:
```
# Split the data into training and testing sets into 80:20 ration
# 80% percent dataset is denoted to training dataset
# 20% percent will be testing dataset
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_
```

In [38]:
```
# Build the Random Forest model
model1 = RandomForestClassifier(random_state=42)
```

In [39]:
```
# Build the Decision Tree model
model2 = DecisionTreeClassifier()
```

In [40]:
```
# Build the Gradient Boosting model
model3 = GradientBoostingClassifier()
```

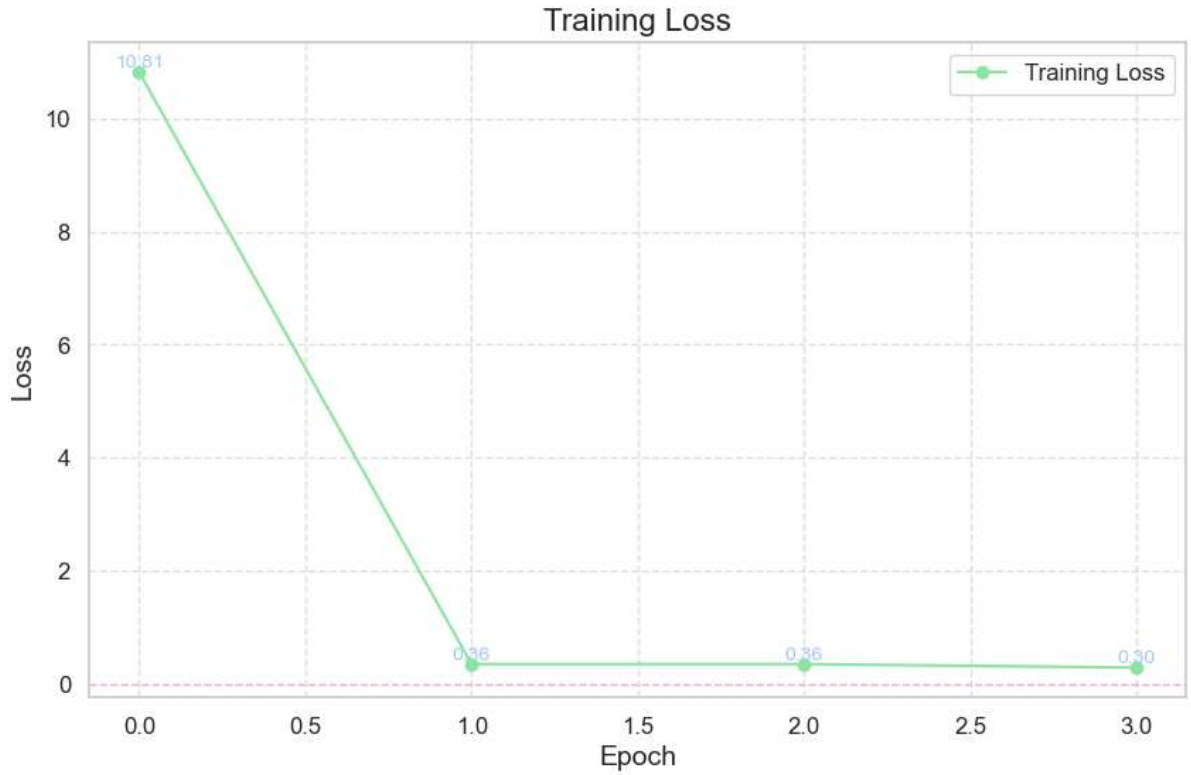
In [41]:
```
# Build the SVM (Support Vector Machines)
model4 = SVC()
```

In [42]:
```
#feature columns of the training data
input_shape = X_train.shape[1]
num_classes = 2  # The number of classes
```

In [43]:
```
# Build the Recurrent Neural Networks (RNN) Model
model5 = RNNClassifier(input_shape, num_classes=num_classes, learning_rate=0.01,
```

```
# Train the neural network
model5.fit(X_train, y_train)
```

```
Epoch 1/4
3414/3414 [==============================] - 4s 1ms/step - loss: 10.8126 - accu
racy: 0.9137 - val_loss: 0.2846 - val_accuracy: 0.9181
Epoch 2/4
3414/3414 [==============================] - 4s 1ms/step - loss: 0.3590 - accur
acy: 0.9152 - val_loss: 0.2842 - val_accuracy: 0.9181
Epoch 3/4
3414/3414 [==============================] - 4s 1ms/step - loss: 0.3591 - accur
acy: 0.9152 - val_loss: 0.2834 - val_accuracy: 0.9181
Epoch 4/4
3414/3414 [==============================] - 4s 1ms/step - loss: 0.2951 - accur
acy: 0.9153 - val_loss: 0.2835 - val_accuracy: 0.9181
```



- Model Evaluation

```
In [51]: print("Accuracy of Random Forest: {:.2f}%".format(accuracy_score(y_test,y_pred)*100))
         print("Accuracy of Support Vector Machine (SVM): {:.2f}%".format(accuracy_score(y_test,y_pred4)*100))
         print("Accuracy of Decision Tree: {:.2f}%".format(accuracy_score(y_test,y_pred2)*100))
         print("Accuracy of Gradient Boosting: {:.2f}%".format(accuracy_score(y_test,y_pred3)*100))
         print("Accuracy of Recurrent Neural Network (RNN): {:.4f}%".format(accuracy_score(y_test,y_pred5)*100))
```

```
Accuracy of Random Forest: 95.62%
Accuracy of Support Vector Machine (SVM): 91.38%
Accuracy of Decision Tree: 93.03%
Accuracy of Gradient Boosting: 95.12%
Accuracy of Recurrent Neural Network (RNN): 91.3803%
```

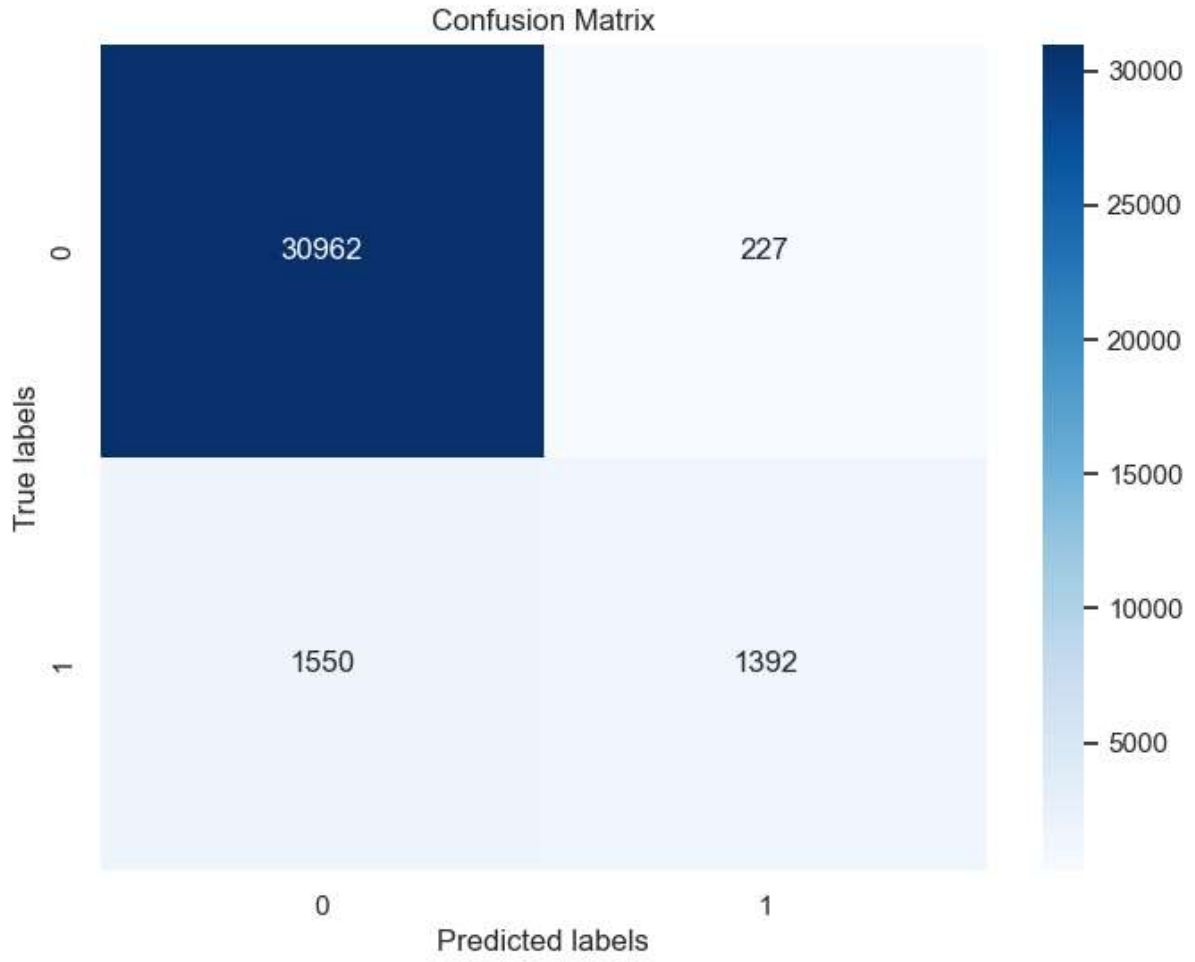
This below illustration contains the classification report for all models includes the Neural Network (RNN) for the user perferences to its (Explicit).

Model Name		precision	recall	f1-score	support
Random Forest	0	0.964461	0.988457	0.976312	31189.000000
	1	0.833795	0.613868	0.707126	2942.000000
	accuracy	0.956169	0.956169	0.956169	0.956169
	macro avg	0.899128	0.801163	0.841719	34131.000000
	weighted avg	0.953198	0.956169	0.953109	34131.000000
Decision Tree	0	0.963779	0.959761	0.961766	31189.000000
	1	0.591471	0.617607	0.604257	2942.000000
	accuracy	0.930269	0.930269	0.930269	0.930269
	macro avg	0.777625	0.788684	0.783011	34131.000000
	weighted avg	0.931687	0.930269	0.930950	34131.000000
Gradient Boosting	0	0.963978	0.983295	0.973541	31189.000000
	1	0.775140	0.610469	0.683020	2942.000000
	accuracy	0.951159	0.951159	0.951159	0.951159
	macro avg	0.869559	0.796882	0.828280	34131.000000
	weighted avg	0.947701	0.951159	0.948499	34131.000000
Support Vector Machine (SVM)	0	0.913803	1.000000	0.954960	31189.000000
	1	0.000000	0.000000	0.000000	2942.000000
	accuracy	0.913803	0.913803	0.913803	0.913803
	macro avg	0.456901	0.500000	0.477480	34131.000000
	weighted avg	0.835035	0.913803	0.872645	34131.000000
Recurrent Neural Network (RNN)	0	0.913803	1.000000	0.954960	31189.000000
	1	0.000000	0.000000	0.000000	2942.000000
	accuracy	0.913803	0.913803	0.913803	0.913803
	macro avg	0.456901	0.500000	0.477480	34131.000000
	weighted avg	0.835035	0.913803	0.872645	34131.000000

Classification Report For Average Prediction:

	precision	recall	f1-score	support
0	0.95	0.99	0.97	31189
1	0.86	0.47	0.61	2942
accuracy			0.95	34131
macro avg	0.91	0.73	0.79	34131
weighted avg	0.94	0.95	0.94	34131

Confusion matrix after averaging the prediction of the models



This configure manual provides as a comprehensive exploration for configuring the installation the required softwares or tools to implementation of the code for understanding the music recommendation system using machine learning and neural networks on a large-scale dataset.

References

Anaconda: https://docs.anaconda.com/free/anaconda/install/windows/
Kaggle Dataset Source: https://www.kaggle.com/