———🕮———A picture containing picture frame, rectangle, frame

Description automatically generated

**VIET NAM NATIONAL UNIVERSITY HO CHI MINH CITY**

**UNIVERSITY OF INFORMATION TECHNOLOGY**

**INFORMATION SYSTEM FACULTY**



**SUBJECT: DATA MINING**

**FINAL PROJECT REPORT**

**TOPIC:**

**PREDICT THE MUSIC GENRE OF A SONG**

**Lecturer**: Mrs Cao Thi Nhan

Mr Vu Minh Sang

**Class:** IS252.M21.HTCL

**Group:** Team 11

**Student performance**:

Bui Duc Duy 20521228

Nguyen Thi Cam Van 20522145

Ho Bao An 20520876

Luu Thao Linh 20521532

**Ho Chi Minh City, June 2023**

**VIET NAM NATIONAL UNIVERSITY HO CHI MINH CITY**

**UNIVERSITY OF INFORMATION TECHNOLOGY**

**INFORMATION SYSTEM FACULTY**

———🕮———



**SUBJECT: DATA MINING**

**FINAL PROJECT REPORT**

**TOPIC:**

**PREDICT THE MUSIC GENRE OF A SONG**

**Lecturer**: Mrs Cao Thi Nhan

Mr Vu Minh Sang

**Class:** IS252.M21.HTCL

**Group:** Team 11

**Student performance**:

Bui Duc Duy 20521228

Nguyen Thi Cam Van 20522145

Ho Bao An 20520876

Luu Thao Linh 20521532

**Ho Chi Minh City, June 2023**

# TEACHER’S COMMENTS

…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

……………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

…………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

# ACKNOWLEDGEMENT

In fact, there is no success that is not tied to the support and help, whether more or less, directly or indirectly from others. With the deepest gratitude, first of all, our group would like to express our sincere thanks to the teachers of the University of Information Technology - Vietnam National University, Ho Chi Minh City and the teachers of the Faculty of Information Systems helped the group to have the basic knowledge as a basis to carry out this topic.

In particular, our team would like to express our sincere thanks to Mrs. Cao Thi Nhan - theoretical lecturer and Mr. Vu Minh Sang -practical lecturer of Data Mining who wholeheartedly helped, directlyinstructed and guided the group throughout the process a project.

Thanks to that, we have gained a lot of useful knowledge in applying as well as project-making skills. Without the guidance and teachings of the teacher, our group thinks this project of the group would be very difficult to complete. Once again, I sincerely thank teacher. In addition, for the project to be completed, it is impossible to thank the people who did it, thank you to the team members who worked hard and completed the task on schedule.

**Finally, thank you to all the team members who worked at their best to complete their thesis well. Sincerely thank!**

# TABLE OF CONTENTS

[**TEACHER’S COMMENTS 3**](#_heading=h.gjdgxs)

[**ACKNOWLEDGEMENT 4**](#_heading=h.30j0zll)

[**TABLE OF CONTENTS 5**](#_heading=h.1fob9te)

[**CHAPTER I : INTRODUCTION 7**](#_heading=h.3znysh7)

[1.1 Problems 7](#_heading=h.2et92p0)

[1.2 Project goal 7](#_heading=h.tyjcwt)

[1.3 Developer tools & Technology 7](#_heading=h.3dy6vkm)

[**CHAPTER II: DATA PREPROCESSING 9**](#_heading=h.1t3h5sf)

[2.1 Description of original data 9](#_heading=h.4d34og8)

[2.1.1 Data Sources 9](#_heading=h.2s8eyo1)

[2.1.2 Data file 9](#_heading=h.17dp8vu)

[2.1.3 Attribute number and value 9](#_heading=h.3rdcrjn)

[2.1.4 Statistics of attribute values 9](#_heading=h.26in1rg)

[2.1.5 Subclass Number 15](#_heading=h.lnxbz9)

[2.2 Data preprocessing 15](#_heading=h.35nkun2)

[2.2.1 Import Library 15](#_heading=h.1ksv4uv)

[2.2.2 Import Dataset 16](#_heading=h.44sinio)

[2.2.3 Check data type, information 16](#_heading=h.2jxsxqh)

[2.2.4 Overview of the data 17](#_heading=h.z337ya)

[2.2.5 Description of dataset information 18](#_heading=h.3j2qqm3)

[2.2.6 Handle the null data 19](#_heading=h.1y810tw)

[2.2.7 Check the balance data of the predictor attribute 20](#_heading=h.4i7ojhp)

[2.2.8 Data preprocessing and visualization with the attributes one by one 20](#_heading=h.2xcytpi)

[**CHAPTER III: ALGORITHMS AND EXPERIMENTS 48**](#_heading=h.1ci93xb)

[3.1 Algorithm used 48](#_heading=h.3whwml4)

[3.1.1 Decision Tree 48](#_heading=h.2bn6wsx)

[3.1.2 Random Forest 50](#_heading=h.qsh70q)

[3.1.3 Naive Bayes 54](#_heading=h.3as4poj)

[3.1.4 K-Nearest Neighbors 58](#_heading=h.1pxezwc)

[3.1.5 Support Vector Machine 60](#_heading=h.49x2ik5)

[3.2 Experiments on Jupyter Notebook 63](#_heading=h.2p2csry)

[3.2.1 Replace categorical attribute values ​​with numerical 63](#_heading=h.147n2zr)

[3.2.2 Split the decision property column to a separate column 63](#_heading=h.3o7alnk)

[3.2.3 Separating train and test data 63](#_heading=h.23ckvvd)

[3.2.4 Decision Trees Algorithm 64](#_heading=h.ihv636)

[3.2.5 Random Forest algorithm 66](#_heading=h.32hioqz)

[3.2.6 Naive Bayes Algorithm 68](#_heading=h.1hmsyys)

[3.2.7 K-Nearest Neighbors Algorithm (KNN) 70](#_heading=h.41mghml)

[3.2.8 Support Vector Machine 72](#_heading=h.2grqrue)

[3.2.9 Comparison, Evaluation 74](#_heading=h.vx1227)

[**CHAPTER IV: PREDICTIVE SOFTWARE 77**](#_heading=h.3fwokq0)

[4.1 Software overview 77](#_heading=h.1v1yuxt)

[4.1.1 Algorithms used 77](#_heading=h.4f1mdlm)

[4.1.2 Properties used to make predictions 77](#_heading=h.2u6wntf)

[4.1.3 Interface and Testing 78](#_heading=h.19c6y18)

[4.2 Software code 80](#_heading=h.3tbugp1)

[4.2.1 Interface section code 80](#_heading=h.28h4qwu)

[4.2.2 Processing section code 84](#_heading=h.nmf14n)

[**CHAPTER V: CONCLUSION 87**](#_heading=h.37m2jsg)

[5.1 Advantages and limitations of each algorithm 87](#_heading=h.1mrcu09)

[5.1.1 Decision Tree 87](#_heading=h.46r0co2)

[5.1.2 Random Forest 88](#_heading=h.2lwamvv)

[5.1.3 Naïve Bayes 89](#_heading=h.111kx3o)

[5.1.4 K–Nearest Neighbors 90](#_heading=h.3l18frh)

[5.1.5 Support Vector Machine 91](#_heading=h.206ipza)

[**REFERENCES 93**](#_heading=h.4k668n3)

# CHAPTER I : INTRODUCTION

## 1.1 Problems

Music is everywhere around us. The impact of music on human beings cannot be expressed in words. Through its perennial journey, the face of music has seen a lot of diversity and change in its form. The world around us is not homogenous and there exists a vast diversity in the lifestyle of people, their thinking and their culture. With this diversity it is apparent that music too will be different and will have diversity that matches to the interests of different people. Keeping this diversity in mind there is tremendous scope for audio streaming and media services companies to automatically classify the genre of a particular song in order to cater similar songs to potential listeners. With the current boom in Data Science practices and predictive modelling and with the availability of structured/unstructured data online such tasks have become convenient and are at the forefront of large companies.

In this online assesment the given task at hand aims to categorize songs into their respective categories viz. (Electronic, Rap, Hip-Hop, Pop) based on some certain parameters or features given in a tabular data format on the basis of which we classify which category the songs belong to.

The first and foremost task would be to import all the necessary libraries.

## 1.2 Project goal

- Building a data system on natural language, using machine learning to train machines to make highly reliable predictions and information for humans.

- Music genre prediction helps recommend songs, albums or playlists based on genres that the user has liked before. This helps users discover new music that might match their taste

## 1.3 Developer tools & Technology

**In the process of implementation, the group used a number of software for researching and developing the topic:**

* Information collection and analysis using the python library and programming language.
* Data sources: [Prediction of music genre | Kaggle](https://www.kaggle.com/datasets/vicsuperman/prediction-of-music-genre)

All of the above software is installed and used by the team on Microsoft Windows 10 operating system. The compatibility of the above software with other operating systems is not within the scope of this study.

# CHAPTER II: DATA PREPROCESSING

## 2.1 Description of original data

### 2.1.1 Data Sources

**Author**: VICSUPERMAN

### 2.1.2 Data file

**Total data rows**: 50005

### 2.1.3 Attribute number and value

**Total columns**: 18

**Dataset characteristics**: Multivariable

**Attribute number characteristics**: Characters, real numbers, integers

**Lost value**: None

### 2.1.4 Statistics of attribute values

**Symbol**: # -number, -character

Sources: [Prediction of music genre | Kaggle](https://www.kaggle.com/datasets/vicsuperman/prediction-of-music-genre)

A screenshot of a graph

Description automatically generated with low confidence

A screenshot of a graph

Description automatically generated with low confidence

A screenshot of a graph

Description automatically generated with low confidence

A screenshot of a graph

Description automatically generated with low confidence

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a graph

Description automatically generated with low confidence

A screenshot of a computer

Description automatically generated with medium confidence

A screenshot of a graph

Description automatically generated with medium confidence

A screenshot of a graph

Description automatically generated with low confidence

**Attribute statistics table:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| STT | Attribute | Attribute meaning | Value of property | Average value | Median value | Mode |
| 1 | **instance\_id** | Serial number of the song |  |  |  |  |
| 2 | **artist\_name** | Name of the artist |  |  |  | Empty\_field |
| 3 | **track\_name** | Title of the song |  |  |  | Home |
| 4 | **popularity** | Score assigned to the song | From [0-100] | 44.220 | 48.0 | 52.0 |
| 5 | **acousticness** | Acoustic a song | From  [0.0 – 1.0] | 0.3063 | 0.0171 | 0.995 |
| 6 | **danceability** | Danceability | From  [0.0 – 1.0] | 0.558 | 0.591 | 0.529 |
| 7 | **duration\_ms** | Duration in milliseconds |  | 221252.602 | 219360.0 | -1.0 |
| 8 | **energy** | Energetic the song | From  [0 - 1] | 0.5997 | 0.674 | 0.805 |
| 9 | **instrumentalness** | The amount of vocals | From  [0.0 – 1.0] | 0.1816 | 0.000159 | 0.0 |
| 10 | **key** | Group of pitches or scale |  |  |  | G |
| 11 | **liveness** | Probability that the song was recorded with a live audience |  | 0.1938 | 0.33 | 0.11 |
| 12 | **loudness** | Loud the song |  | -9.1337 | -6.74 | -5.443 |
| 13 | **mode** | Major and Minor scales that the song |  |  |  | Major |
| 14 | **speechiness** | Presence of spoken words in a track |  | 0.0935 | 0.26899 | 0.0332 |
| 15 | **tempo** | Speed at which the song is being played |  |  |  | ? |
| 16 | **obtained\_date** | The date at which the song metadata was retrieved |  |  |  | 4-Apr |
| 17 | **valence** | The musical positiveness conveyed by a track | From  [0.0 – 1.0] | 0.4562 | 0.505 | 0.3379 |
| 18 | **music\_genre** | The actual category to which the song belongs |  |  |  | Electronic |

### 2.1.5 Subclass Number

Subclass Attributes: **instance\_id, popularity, acousticness, danceability, duration\_ms, energy, instrumentalness, liveness, loudness, speechiness, valence.**

## 2.2 Data preprocessing

**Purpose**:

* DataTransformation
* Data Collection
* Data visualization and comments

### 2.2.1 Import Library

A picture containing text, font, screenshot

Description automatically generated

### 2.2.2 Import Dataset

A screenshot of a computer

Description automatically generated with low confidence

### 2.2.3 Check data type, information

A picture containing text, screenshot, software

Description automatically generated

*A screenshot of a computer

Description automatically generated*

### 2.2.4 Overview of the data

- Data is a combination of string and integer values.

**- instance\_id :** Serial number of the song in the dataset.

**- artist\_name** : Name of the artist of the song.

**- track\_name** : Title of the song.

**- popularity** : An arbitrary score assigned to the song in the range of 0-100 with 100 being most popular and 0 being least.

**- acousticness** : This value describes how acoustic a song is. A score of 1.0 means the song is most likely to be an acoustic one.

**- danceability** : Danceability describes how suitable a track is for dancing based on a combination of musical elements. A value of 0.0 is least danceable and 1.0 is most danceable.

**- duration\_ms** : Is the duration in milliseconds of the song.

**- energy** : Represents how energetic the song is. The range of this field is between [0-1] with 1 being song with highest energy and 0 with lowest.

**- instrumentalness** : This value represents the amount of vocals in the song. The closer it is to 1.0, the more instrumental the song is.

**- key** : Key of a piece is the group of pitches, or scale, that forms the basis of a music composition.

**- liveness** : This value describes the probability that the song was recorded with a live audience.

**- loudness** : Column representing how loud the song is.

**- mode :** Major and Minor scales that the song is based upon.

**- speechiness** : Speechiness detects the presence of spoken words in a track.

**- tempo** : Speed at which the song is being played.

**- obtained\_date** : The date at which the song metadata was retrieved.

**- valence** : A measure from 0.0 to 1.0 describing the musical positiveness conveyed by a track. Tracks with high valence sound more positive.

**- music\_genre** : The actual category to which the song belongs. This is our target variable.

### 2.2.5 Description of dataset information

A screenshot of a data

Description automatically generated with low confidence

A screenshot of a computer

Description automatically generated with low confidence

We can see that there are 17 features and one label column (music\_genre). Out of the features, 12 are numerical (one of which, tempo, is missclassified and will be dealt with later), and 5 are categorical.

We can also already see hints to hidden missing values in 3 features **('tempo', 'artist\_name' and 'duration\_ms').** Those will be dealt with shortly one by one.

### 2.2.6 Handle the null data

Check the null dataA picture containing text, screenshot, font, number

Description automatically generated

There are only 5 rows that contain NaN values. We’ll remove themA screenshot of a computer

Description automatically generated with medium confidence

### 2.2.7 Check the balance data of the predictor attribute

A picture containing text, screenshot, software

Description automatically generated

There are 10 different genres with equal distribution (balanced data). This means the accuracy score will be a good metric to use.

### 2.2.8 Data preprocessing and visualization with the attributes one by one

**1. Instance\_id column**

This is just an index. We’ll drop it.



**2. artist\_name column**

Print out the unique values

A picture containing text, font, line, screenshot

Description automatically generated

Attribute description

A screenshot of a computer

Description automatically generated with medium confidence

Find the missing data

A screenshot of a computer

Description automatically generated with medium confidence

Print out the percentage of data that is missed

A screenshot of a computer

Description automatically generated with medium confidence

Nearly 5% of the observations are missing the artist's names **(marked as 'empty\_field'**), but these entries are still valid otherwise. we will not drop these observations.

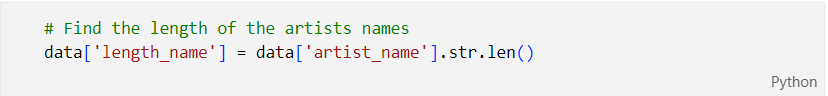
A picture containing text, font, screenshot

Description automatically generated

For the entries that do contain an artist's name, it seems that a song that comes from a particular artist has an ~80% chance of belonging to one specific genre.

However, in it's current form it's not helpful for classifying songs from artists outside the training set. We'll need to extract more general features, starting with the simplest - name length.

Find the length of the **artists names.**

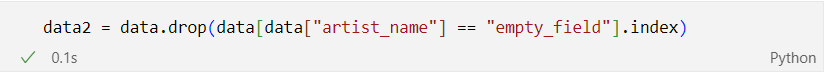


Statistics the data between the two columns are artist\_name and music\_genre.

A screenshot of a computer

Description automatically generated with low confidence

Remove the empty field data



Get top 20 artists

A screenshot of a computer

Description automatically generated

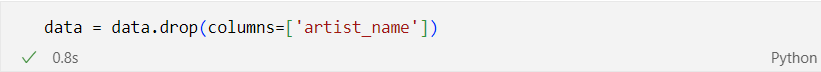
**artist\_name** holds information about the singers' name. The code line below proves the dataset contains information about 6863 artists. In general, this feature could be used for predicting music genre - no one expects to see Beatles in folk charts, or Mozart - in rock top 20s.

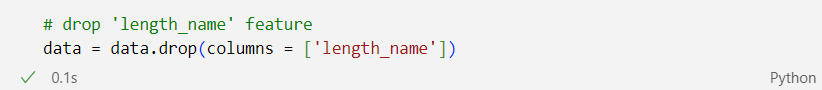
Draw barh chart

A picture containing text, screenshot, font, number

Description automatically generated

It seems the dataset was compiled by Japanese authors or in Japan since several artists in top20 are from the Land of the Rising Sun. Furthermore, many composers (e.g., Mozart, Beethoven, etc.) also found their place in this list. Now, to avoid large number of features, the artist\_name and length\_name is removed.





**3. track\_name column**

Attribute description

A picture containing text, font, screenshot

Description automatically generated

**track\_name** contains information about title of the song. The line of code below proves that the dataset contains information about 41699 songs because it has too many unique attributes and is not helpful for classification, otherwise it will cause an error for the algorithm so I will remove it.



**4. popularity column**

See the unique attributes

A picture containing text, screenshot, font, number

Description automatically generated

Has 100 unique properties that range from 0 – 99.

Next, let's see the distribution of the attributes in the dataset

A picture containing text, font, screenshot

Description automatically generated

A picture containing diagram, text, screenshot, plot

Description automatically generated

Attribute description

A screenshot of a computer

Description automatically generated with medium confidence

The histogram shows the distribution of the data's values

A picture containing text, font, line, screenshot

Description automatically generated

A picture containing text, screenshot, colorfulness, rectangle

Description automatically generated

This feature shows a nice spread of distributions for the different genres. Could definitely be useful for classification.

Rap, Hip-Hop and Rock seem to be the most popular genres, while Anime, Blues and Classical are the least popular. The other 4 genres are somewhere in between.

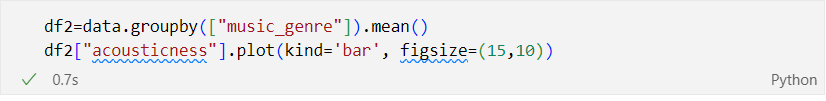
**5. acousticness column**

Statistics the data of acousticness

A screenshot of a computer

Description automatically generated with medium confidence

The histogram shows the distribution of the data's values

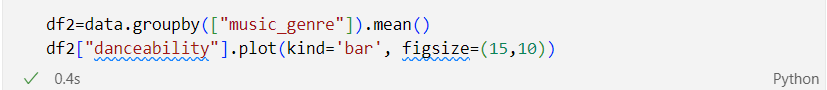


A picture containing text, screenshot, diagram, plot

Description automatically generated

Interestingly, classical and jazz music has a higher percentage of acousticness than the rest.

**6. danceability column**



A picture containing text, screenshot, plot, font

Description automatically generated

Classical music sticks out again, but Rap and Hip-Hop can also be distinguished from the rest (they seem to go together often).

**7. duration\_ms column**

Statistics the data between the two columns are duration\_ms and music\_genre.

A screenshot of a computer

Description automatically generated with medium confidence

-1.0 is obvously not a valid time measurement. These are missing values.

A screenshot of a computer

Description automatically generated with medium confidence

Nearly 10% of entries are missing the duration and comparing with the statistics most of the tracks are about the same which is not valid for classification so we will proceed to remove the feature



**8. energy column**

The histogram shows the distribution of the data's values

A picture containing text, font, line, white

Description automatically generated

A picture containing diagram, colorfulness, square, rectangle

Description automatically generated

As usual, classical music stands out (and Jazz to a much lesser degree). Rap and Hip-Hop still match each other.

**9. instrumentalness column**

See the distribution of the data's values.

A screenshot of a computer program

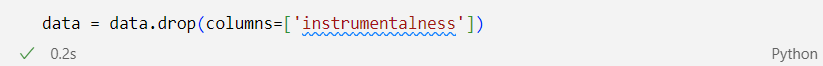
Description automatically generated with low confidence

The histogram shows the distribution of the data's values

A screen shot of a graph

Description automatically generated with medium confidence

Such a large number of 0.0 entries likely indicates missing values rather than real data points, we won't fill in missing values. Instead, we'll discard this feature entirely.



**10. key column**

Check the unique attributes

A picture containing text, font, line, screenshot

Description automatically generated

The histogram shows the distribution of the data's values

A picture containing text, font, screenshot, line

Description automatically generated

A picture containing text, screenshot, plot, diagram

Description automatically generated

Different genres have noticeably different spreads.

Draw a histogram showing the distribution of key attribute values with music\_genre

A close-up of a computer code

Description automatically generated with low confidence

A picture containing screenshot, colorfulness, plot

Description automatically generated

* We can see with some genres like electronic, rap, hiphop, key = C# is the main
* Country and blues, key = D or key = G is the majority
* Key = D# is quite low in all genres
* The rest of the distribution is quite even
* We'll keep this feature, but use encode to make it useful.

A picture containing text, font, line, screenshot

Description automatically generated

Data before encoder

A screenshot of a computer

Description automatically generated with medium confidence

Data after encoder

A screenshot of a computer

Description automatically generated with medium confidence

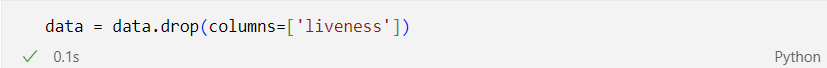
**11. liveness column**

The histogram shows the distribution of the data's values

A screen shot of a graph

Description automatically generated with medium confidence

The distributions seem similarly skewed for all genres, so this feature will likely not contribute much to the model. We'll discard this feature entirely.



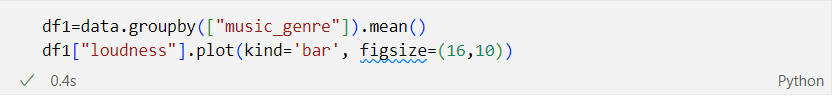
**12. loudness column**

Attribute description

A screenshot of a computer

Description automatically generated with medium confidence

The histogram shows the distribution of the data's values



A picture containing text, screenshot, line, rectangle

Description automatically generated

As usual, classical music is far from the rest, with Jazz (and Blues) also differing from the rest somewhat.

**13. mode column**

Check the unique values

A picture containing text, font, line, white

Description automatically generated

The histogram shows the distribution of the data's values

A close-up of a computer code

Description automatically generated with low confidence

A picture containing text, screenshot, plot, diagram

Description automatically generated

All genres seem to have a prefererence for the "Major" mode, but to different degrees. It is the most pronounced in the Country genre. We'll use this feature after encoding.

A black text on a white background

Description automatically generated with low confidence

Data before encoder

A picture containing text, font, white, line

Description automatically generated

Data after encoder

A picture containing text, screenshot, font, line

Description automatically generated

**14. speechiness column**

The lineplot histogram shows the distribution of the data's values

A picture containing text, font, line, screenshot

Description automatically generated

A picture containing text, screenshot, colorfulness, diagram

Description automatically generated

This feature should contribute especially to identifying Hip-Hop and Rap.

**15. tempo column**

Statistics the data between the two columns are tempo and music\_genre

A screenshot of a computer

Description automatically generated

This feature should be numeric. The "?" is a missing value.

A screenshot of a computer

Description automatically generated with low confidence

Replace "?" with np.nan and correctly classify the feature.

A picture containing text, font, line, screenshot

Description automatically generated

The boxplot histogram shows the distribution of the data's values

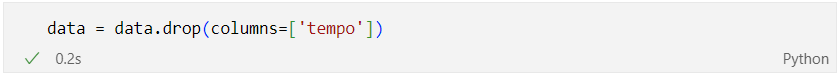
A picture containing text, font, screenshot, line

Description automatically generated

A picture containing diagram, square, rectangle, colorfulness

Description automatically generated

The variation between genres is not great so we'll drop the feature.



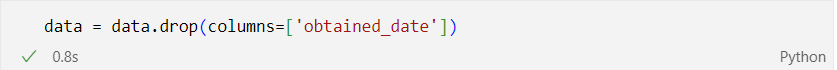
**16. obtained\_date column**

Check the unique values

A picture containing text, font, line, white

Description automatically generated

Only gives the 4 dates at which the data was obtained. Not useful to us, so we'll drop it.



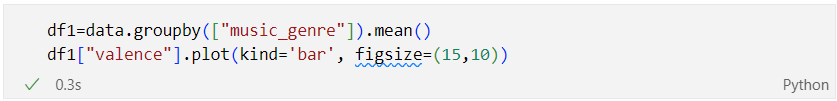
**17. valence column**

Statistics the data between the two columns are valence and music\_genre

A screenshot of a graph

Description automatically generated with low confidence

The bar histogram shows the distribution of the data's values



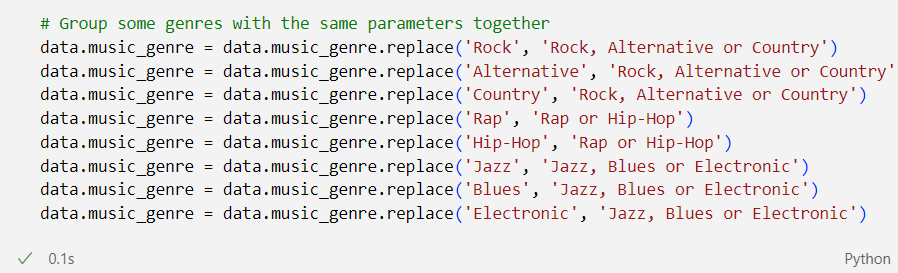
A picture containing text, screenshot, plot, diagram

Description automatically generated

Again, only classical music truly stands out from the rest.

**18. music\_genre column**

Group some genres with the same parameters together



Review properties after grouping

A screenshot of a computer

Description automatically generated with medium confidence

# CHAPTER III: ALGORITHMS AND EXPERIMENTS

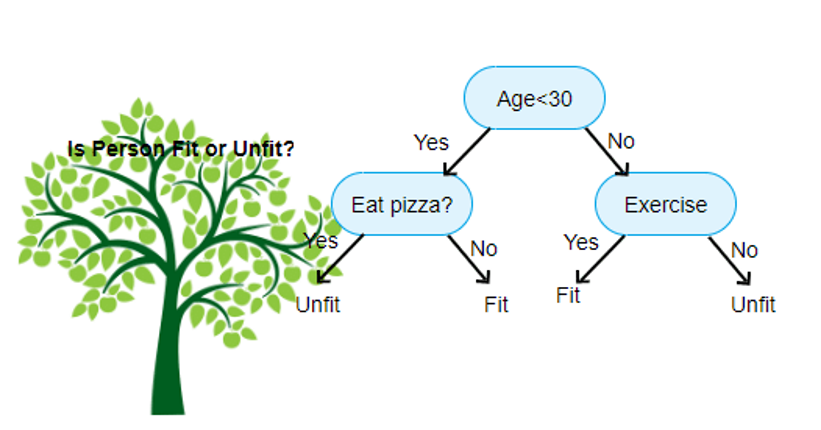
## 3.1 Algorithm used

### 3.1.1 Decision Tree

**- A decision tree is a tree structure such that:**

* Each node in the network corresponds to a test on an attribute.
* Each branch represents the test result.
* Leaf nodes represent classes or class distributions.
* The highest node in the tree is the root node.

**- Decision tree shap:**



**- Basic strategy:**

* Start from single node showing all samples.
* If the samples belong to the same class, the node becomes a leaf node and is labeled with that class.
* In contrast, using the attribute measure to select the attribute will best separate the samples into classes.
* A branch is created for each value of the selected attribute and the samples are partitioned by use the same process recursively to create a decision tree.
* The process ends only if any of the following conditions are true.

- All templates for a given node belong to the same class

- There are no more attributes that the sample can rely on for further partitioning

- No samples left at node

- ID3 is an algorithm used in decision trees. This algorithm uses information gain to build a decision tree. The largest Information Gain attribute will be selected as the root node

**- Information Gain:**

A picture containing font, text, white, line

Description automatically generated

- **Amount of information needed to classify an element in S based on attribute A: InfoA(S)**

A picture containing font, text, white, typography

Description automatically generated

- **Information gain is the difference between the original Info(S) information value (before partitioning) and the new InfoA(S) information value (after partitioning with A**)

A black text on a white background

Description automatically generated with medium confidence

**CART: Unlike ID3 which uses Information Cain formula, Cart algorithm uses Gini formula. The attribute with the smallest Gini value will be the root node**

**- Gini index of the set S:**

A picture containing font, text, white, graphics

Description automatically generated

**P(j|S) is the frequency of j in S**

**- Gini of attribute:**

**A picture containing font, white, text, symbol

Description automatically generated**

**In case: ni is the number of samples in note I, n is the number of samples in note A**

### 3.1.2 Random Forest

#### 3.1.2.1 Random Forest explain

**Random Forest** is one of the most popular and commonly used algorithms by Data Scientists. **Random forest is a Supervised Machine Learning Algorithm that is used widely in Classification and Regression problems**. It builds decision trees on different samples and takes their majority vote for classification and average in case of regression.

One of the most important features of the Random Forest Algorithm is that it can handle the data set containing continuous variables, as in the case of regression, and categorical variables, as in the case of classification. It performs better for classification and regression tasks.

* **Steps Involved in Random Forest Algorithm**

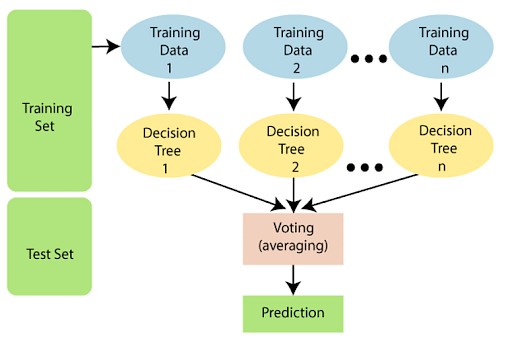
**Step 1:** In the Random forest model, a subset of data points and a subset of features is selected for constructing each decision tree. Simply put, n random records and m features are taken from the data set having k number of records.

**Step 2**: Individual decision trees are constructed for each sample.

**Step 3**: Each decision tree will generate an output.

**Step 4**: Final output is considered based on Majority Voting or Averaging for Classification and regression, respectively.

#### 3.1.2.2 Random Forests Algorithm



**The following steps explain the working Random Forest Algorithm:**

**Step 1:** Select random samples from a given data or training set.

**Step 2**: This algorithm will construct a decision tree for every training data.

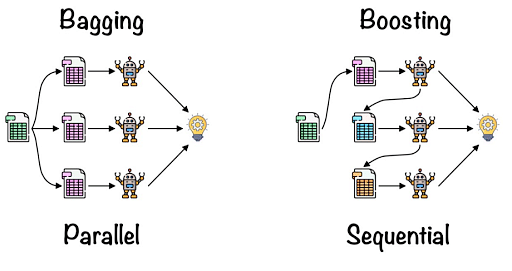
**Step 3**: Voting will take place by averaging the decision tree.

**Step 4**: Finally, select the most voted prediction result as the final prediction result.

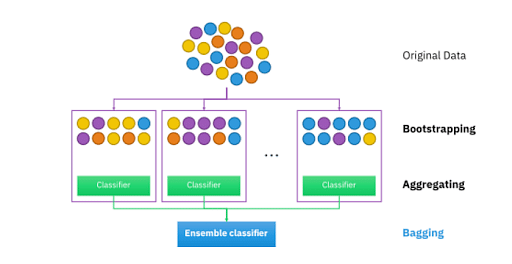
**This combination of multiple models is called Ensemble. Ensemble uses two methods:**

- **Bagging:** Creating a different training subset from sample training data with replacement is called Bagging. The final output is based on majority voting.

- **Boosting:** Combing weak learners into strong learners by creating sequential models such that the final model has the highest accuracy is called Boosting. Example: ADA BOOST, XG BOOST.



BaggingBagging is also known as Bootstrap Aggregation used by random forest. The process begins with any original random data. After arranging, it is organised into samples known as Bootstrap Sample. This process is known as Bootstrapping.Further, the models are trained individually, yielding different results known as Aggregation. In the last step, all the results are combined, and the generated output is based on majority voting. This step is known as Bagging and is done using an Ensemble Classifier.



* **Essential Features of Random Forest**

- **Miscellany:** Each tree has a unique attribute, variety and features concerning other trees. Not all trees are the same.

- **Immune to the curse of dimensionality**: Since a tree is a conceptual idea, it requires no features to be considered. Hence, the feature space is reduced.

- **Parallelization:** We can fully use the CPU to build random forests since each tree is created autonomously from different data and features.

- **Train-Test split:** In a Random Forest, we don’t have to differentiate the data for train and test because the decision tree never sees 30% of the data.

**- Stability:** The final result is based on Bagging, meaning the result is based on majority voting or average.

* **How Random Forest is applied?**

**Random Forest has a wide range of applications across various domains due to its versatility and robustness.**

- **Classification Problems**: Random Forest is often used for classification tasks, such as spam detection, sentiment analysis, customer churn prediction, disease diagnosis, and image classification. Its ability to handle both numerical and categorical features makes it suitable for diverse datasets.

- **Regression Problems:** Random Forest can be applied to regression tasks, including predicting housing prices, stock market trends, energy consumption, and demand forecasting. It can capture complex non-linear relationships between input variables and the target variable.

**- Feature Importance**: Random Forest provides a measure of feature importance, which can be utilized for feature selection in data preprocessing. This information helps identify the most relevant features for prediction and can guide feature engineering efforts.

**- Anomaly Detection**: Random Forest can be used for anomaly detection by training on normal data and identifying instances that deviate significantly from the learned patterns. This is useful in fraud detection, network intrusion detection, and detecting unusual behaviors in various domains.

**- Ensemble Learning**: Random Forest is a type of ensemble learning method, which combines multiple models to improve overall performance. It can be used as a base learner in ensemble techniques such as bagging, boosting, and stacking to further enhance predictive accuracy.

**- Imputation of Missing Values**: Random Forest can handle missing data effectively. It can be used to impute missing values in a dataset by using the available features to predict missing values, making it valuable for data preprocessing tasks.

**- Recommender Systems**: Random Forest can be employed in building recommendation systems to suggest relevant products, movies, or content to users based on their preferences and behavior.

**- Bioinformatics and Genomics**: Random Forest finds applications in analyzing DNA sequences, gene expression data, and protein-protein interactions. It can be used for tasks like gene expression classification, protein structure prediction, and identifying disease biomarkers.

### 3.1.3 Naive Bayes

#### 3.1.3.1 Bayes Theorem

- **Bayes' Theorem (Bayes' Theorem)** **is a mathematical theorem that calculates the probability of a random event A, given that the related event B has occurred**.

- This theorem is named after the 18th century English mathematician Thomas Bayes.

- This is one of the extremely useful tools, a close friend of Data Scientists who work in data science.

- Bayes theorem allows to calculate the probability of a random event A given that related event B has occurred. This probability is denoted P(A|B) and read as “the probability of A if there is B”. This quantity is called conditional probability or posterior probability because it is derived from a given value of B or depends on that value.

- **According to Bayes' theorem, the probability that A occurs when B is known will depend on 3 factors:**

* **The probability that A occurs on its own, regardless of B**. It is denoted by P(A) and read as the probability of A. This is called the marginal probability or a priori probability, it is “a priori” ” in the sense that it is not interested in any information about B.
* **Probability of occurring B on its own, regardless of A**. It is denoted by P(B) and read as “probability of B”. This quantity is also called a normalizing constant because it is always the same, regardless of the event A is trying to know.

- **Probability of B happening when A is known**. It is denoted by P(B|A) and read as “probability of B if there is A”. This quantity is called the likelihood that B will occur, given that A has occurred. Pay attention not to confuse the probability that B will occur when A is known and the probability that A will occur when B is known.

- **We can restate it with the following formula: The probability that A and B occur at the same time is:**

P(A,B) = P(A) P(B)

- **In case:**

* P(A)P(A) is the probability of a distinct A occurring.
* P(B)P(B) is the probability that B occurs separately.

- **If A and B are two related events, and the probability that event B occurs is greater than 0,** we can define the probability that A will occur, given that B occurs as follows:

A picture containing font, white, text, logo

Description automatically generated

- **Bayes theorem is based on the definition of conditional probability above, expressed in the form of a formula as follows**:

A picture containing font, white, text, symbol

Description automatically generated

**The symbol ¬A is not A (or A's complement). We have P(A)+P(¬A) = 1**

**From there: P(B) =P(B,A) + P(B,¬A) = P(B∣A)P(A) + P(B∣¬A)P(¬A)**

**Bayes' theorem is written in variant form as follows:**

A picture containing text, font, white, line

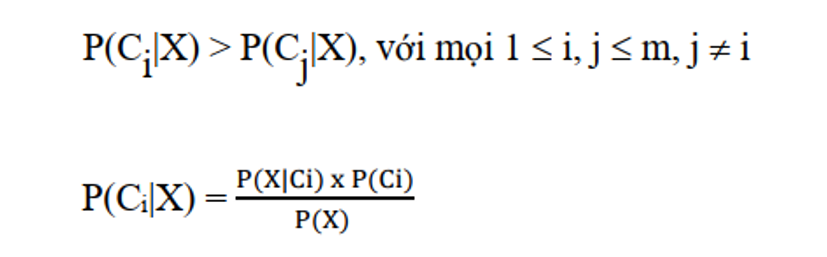
Description automatically generated

#### 3.1.3.2 naïve Bayes classification algorithm

- **Naive Bayes Classification (NBC) is a classification algorithm based on probability calculation applying Bayes theorem. This algorithm belongs to the group of supervised learning algorithms.**

- Each data sample is represented by X=(x1, x2,..., xn) with attributes A1, A2,..., An

- Grades C1, C2, ..., Cm. Given an unknown sample X.

- Subclassing Naive Bayes will determine that X belongs to class Ci if and only if:  


* **According to Bayes' theorem**

Since P(X) is constant for all classes, **only the maximum P(X|Ci) x P(Ci) is needed. If P(Ci) is not known, we need to assume P(C1)=P(C2)=...= P(Cm)** and **we will maximize P(X|Ci). Otherwise, we maximize P(X|Ci) x P(Ci)**

However, the problem of calculating P(X|Ci) is impossible!

**Admit Naive: assume attribute independence**

A picture containing font, text, symbol, white

Description automatically generated

**It is possible to approximate P(x1|Ci),…,P(xn|Ci) from the training samples.**

If Ak is a qualitative attribute, then P(xk|Ci) = sik/si where sik is the number of training samples of Ci with the value xk for Ak and si is the number of samples belonging to class Ci

If Ak is continuous, then it is assumed to have a Gaussian distribution:

A picture containing text, font, handwriting, white

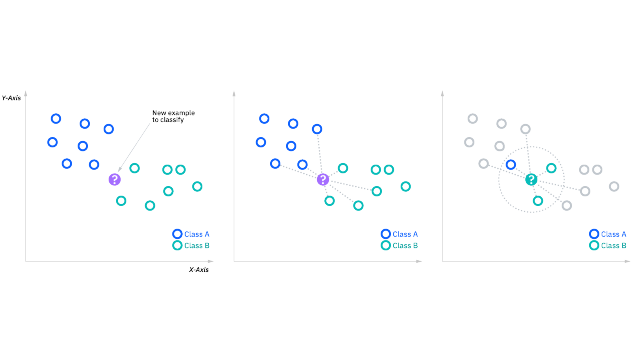
Description automatically generated

### 3.1.4 K-Nearest Neighbors

#### 3.1.4.1 K-Nearest Neighbors explain

The k-nearest neighbors algorithm, also known as KNN or k-NN, is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point. While it can be used for either regression or classification problems, it is typically used as a classification algorithm, working off the assumption that similar points can be found near one another.

For classification problems, a class label is assigned on the basis of a majority vote—i.e. the label that is most frequently represented around a given data point is used. While this is technically considered “plurality voting”, the term, “majority vote” is more commonly used in literature. The distinction between these terminologies is that “majority voting” technically requires a majority of greater than 50%, which primarily works when there are only two categories. When you have multiple classes—e.g. four categories, you don’t necessarily need 50% of the vote to make a conclusion about a class; you could assign a class label with a vote of greater than 25%.



#### 3.1.4.2 K-nearest neighbors Algorithm

**Compute KNN: distance metrics**

To recap, the goal of the k-nearest neighbor algorithm is to identify the nearest neighbors of a given query point, so that we can assign a class label to that point. In order to do this, KNN has a few requirements.

**- Determine your distance metrics**

In order to determine which data points are closest to a given query point, the distance between the query point and the other data points will need to be calculated. These distance metrics help to form decision boundaries, which partitions query points into different regions. You commonly will see decision boundaries visualized with Voronoi diagrams.

**While there are several distance measures that you can choose from, this article will only cover the following**:

**- Euclidean distance (p=2):** This is the most commonly used distance measure, and it is limited to real-valued vectors. Using the below formula, it measures a straight line between the query point and the other point being measured.

A picture containing font, white, handwriting, text

Description automatically generated

**- Manhattan distance (p=1)**: This is also another popular distance metric, which measures the absolute value between two points. It is also referred to as taxicab distance or city block distance as it is commonly visualized with a grid, illustrating how one might navigate from one address to another via city streets.

A picture containing font, text, white, typography

Description automatically generated

**- Minkowski distance**: This distance measure is the generalized form of Euclidean and Manhattan distance metrics. The parameter, p, in the formula below, allows for the creation of other distance metrics. Euclidean distance is represented by this formula when p is equal to two, and Manhattan distance is denoted with p equal to one.

**A picture containing text, font, white, diagram

Description automatically generated**

**- Hamming distance:** This technique is used typically used with Boolean or string vectors, identifying the points where the vectors do not match. As a result, it has also been referred to as the overlap metric. This can be represented with the following formula:

**A picture containing text, font, diagram, white

Description automatically generated**

### 3.1.5 Support Vector Machine

#### 3.1.5.1 Support Vector Machine explain

**Support Vector Machine (SVM**) is a robust classification and regression technique that maximizes the predictive accuracy of a model without overfitting the training data. **SVM is particularly suited to analyzing data with very large numbers (for example, thousands) of predictor fields.**

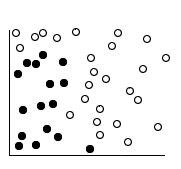
SVM has applications in many disciplines, including customer relationship management (CRM), facial and other image recognition, bioinformatics, text mining concept extraction, intrusion detection, protein structure prediction, and voice and speech recognition.

* **How Support Vector Machine Models works?**

SVM works by mapping data to a high-dimensional feature space so that data points can be categorized, even when the data are not otherwise linearly separable. A separator between the categories is found, then the data are transformed in such a way that the separator could be drawn as a hyperplane. Following this, characteristics of new data can be used to predict the group to which a new record should belong.

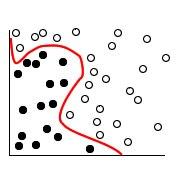
For example, consider the following figure, in which the data points fall into two different categories.

**Figure 1. Data with a preliminary model**



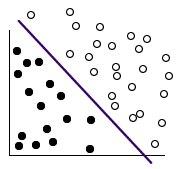
The two categories can be separated with a curve, as shown in the following figure.

**Figure 2. Data with separator added**



After the transformation, the boundary between the two categories can be defined by a hyperplane, as shown in the following figure.

**Figure 3. Transformed data**



The mathematical function used for the transformation is known as the **kernel** function.

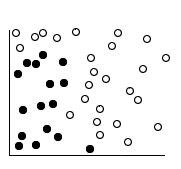
A linear kernel function is recommended when linear separation of the data is straightforward. In other cases, one of the other functions should be used. You will need to experiment with the different functions to obtain the best model in each case, as they each use different algorithms and parameters.

* **How Support Vector Machine Models works?**

SVM works by mapping data to a high-dimensional feature space so that data points can be categorized, even when the data are not otherwise linearly separable. A separator between the categories is found, then the data are transformed in such a way that the separator could be drawn as a hyperplane. Following this, characteristics of new data can be used to predict the group to which a new record should belong.

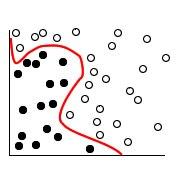
For example, consider the following figure, in which the data points fall into two different categories.

**Figure 1. Data with a preliminary model**



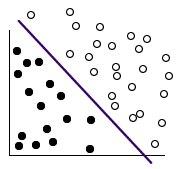
The two categories can be separated with a curve, as shown in the following figure.

**Figure 2. Data with separator added**



After the transformation, the boundary between the two categories can be defined by a hyperplane, as shown in the following figure.

**Figure 3. Transformed data**



The mathematical function used for the transformation is known as the **kernel** function.

A linear kernel function is recommended when linear separation of the data is straightforward. In other cases, one of the other functions should be used. You will need to experiment with the different functions to obtain the best model in each case, as they each use different algorithms and parameters.

## 3.2 Experiments on Jupyter Notebook

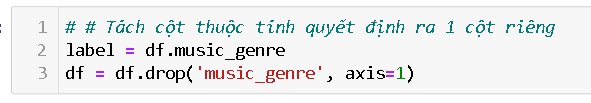
* Graph, count, and view label ratios to get an overview of a song's musical genre.
* Build decision properties, with the decision property as music\_genre.
* Pre-processing before putting in train.

### 3.2.1 Replace categorical attribute values ​​with numerical

A picture containing text, font, screenshot, line

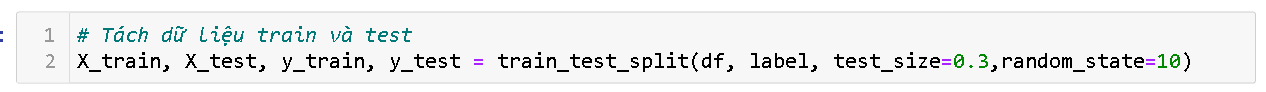
Description automatically generated

### 3.2.2 Split the decision property column to a separate column



### 3.2.3 Separating train and test data

Train data accounts for 70%, test accounts for 30%



### 3.2.4 Decision Trees Algorithm

A screenshot of a computer code

Description automatically generated with medium confidence

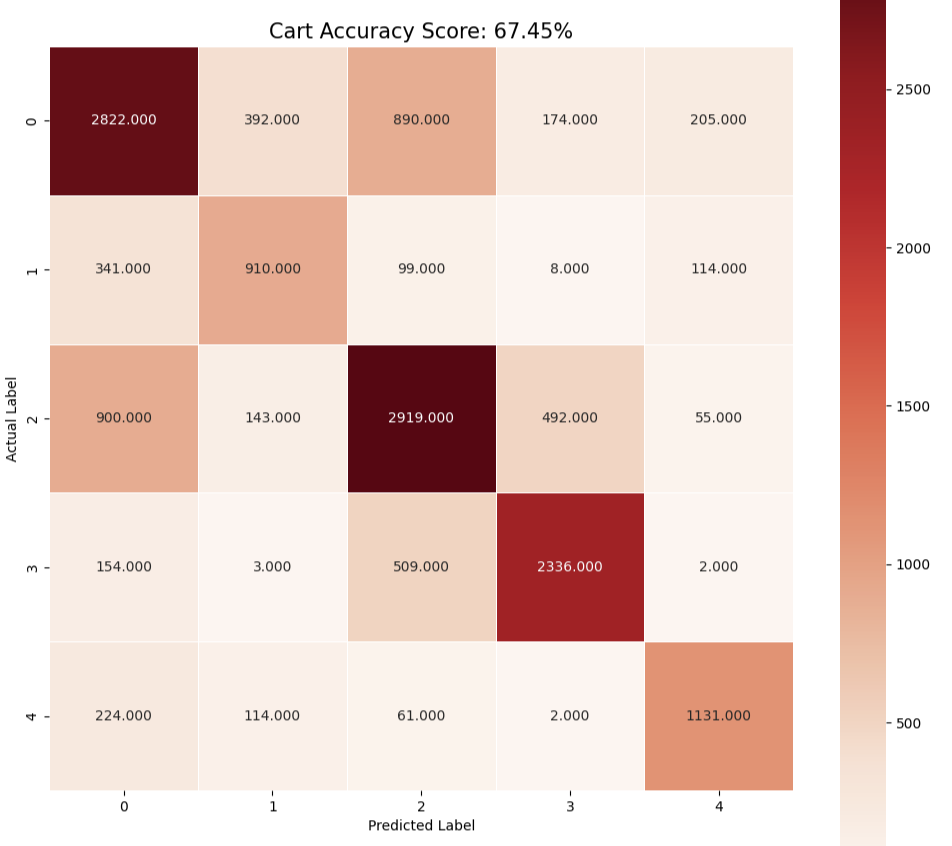
A picture containing text, screenshot, font, number

Description automatically generated

* Algorithm accuracy: 67.45%
* Algorithm runtime: 0.378s

A screenshot of a computer program

Description automatically generated with medium confidence



Through the confused matrix of the Decision Tree algorithm picture tissue (CART), we know:

* Precision of the algorithmic Picture tissue: 67.45%

### 3.2.5 Random Forest algorithm

A screenshot of a computer code

Description automatically generated with medium confidence

A picture containing text, screenshot, font, number

Description automatically generated

* Algorithm accuracy: 76.9%
* Algorithm runtime: 12.3143s

A screenshot of a computer program

Description automatically generated with low confidence

A picture containing text, screenshot, diagram, number

Description automatically generated

Through the confused matrix of random forest algorithm picture tissue, we know:

* Precision of the algorithmic Picture tissue: 67.45%

### 3.2.6 Naive Bayes Algorithm

A screenshot of a computer code

Description automatically generated with low confidence

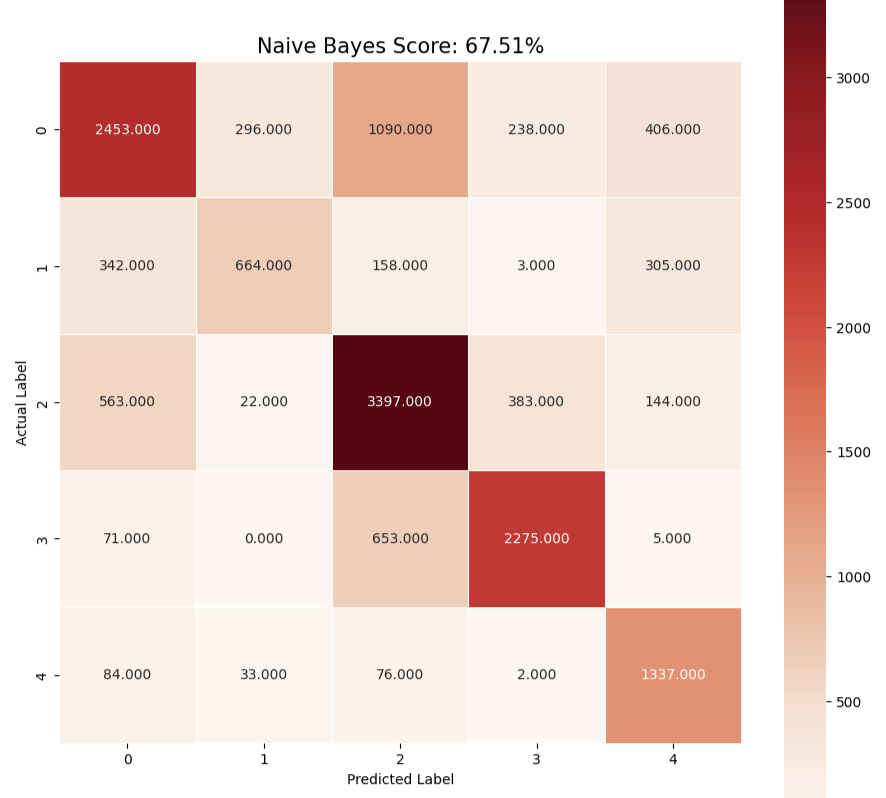
A picture containing text, screenshot, font, number

Description automatically generated

* Algorithm accuracy: 67.51%
* Algorithm runtime: 0.039s

A screenshot of a computer program

Description automatically generated with medium confidence



Through the confused matrix of the Naive Bayes algorithmic Picture tissue, we learn :

* Precision of the algorithmic Picture tissue: 67.51%

### 3.2.7 K-Nearest Neighbors Algorithm (KNN)

A screenshot of a computer code

Description automatically generated with low confidence

A picture containing text, screenshot, font, number

Description automatically generated

* Algorithm accuracy: 63.77%
* Algorithm runtime: 6.5999s

A picture containing text, screenshot, font

Description automatically generated

A screenshot of a graph

Description automatically generated with low confidence

Through the confusion matrix of the KNN algorithm Picture model, we know:

* Precision of the algorithmic Picture tissue: 63.77%

### 3.2.8 Support Vector Machine

A screenshot of a computer program

Description automatically generated with low confidence

A picture containing text, screenshot, font, number

Description automatically generated

* Algorithm accuracy: 71.65%
* Algorithm runtime: 153.1277s

A screenshot of a computer code

Description automatically generated with low confidence

A picture containing text, screenshot, diagram, number

Description automatically generated

Through the confused matrix of Support Vector Machine algorithmic Picture tissue, we learn .

* Precision of the algorithmic Picture tissue: 71.65%

### 3.2.9 Comparison, Evaluation

*Use the BarPlot graph to get an overview of runtime and accuracy between algorithms.*

**Draw a chart comparing the running time of algorithms**

A screenshot of a computer program

Description automatically generated with low confidence

A screenshot of a computer

Description automatically generated with medium confidence

**Conclusion on the runtime chart:**

* The Naïve Bayes algorithm is the algorithm with the fastest running time for datasets. With only 0.039s.
* SVM algorithm is slowest with 153.1277s.
* Draw a chart comparing the accuracy of algorithms:

A picture containing text, screenshot, font, number

Description automatically generated

A picture containing screenshot, text, rectangle, colorfulness

Description automatically generated

**Conclusion on the accuracy chart:**

* The algorithms all give very high accuracy results, balanced with each other, most of which is 67~72% accuracy.
* The K-Nearest Neighbors algorithm has the lowest accuracy of the five algorithms, with an accuracy 63.77%.

**A screenshot of a computer code

Description automatically generated with medium confidence**

# CHAPTER IV: PREDICTIVE SOFTWARE

## 4.1 Software overview

### 4.1.1 Algorithms used

Based on the results obtained in the previous section, the team decided to use random forest algorithm for this software. According to the comparison results, this algorithm, although it has a bad speed, but it gives the highest accuracy.

### 4.1.2 Properties used to make predictions

A screenshot of a computer

Description automatically generated with medium confidence

A picture containing text, screenshot, font

Description automatically generated

A screenshot of a computer

Description automatically generated with low confidence

According to the important attribute classification results, I will take 4 attributes **'popularity', 'danceability', 'loudness', 'speechiness'** to make predictions.

### 4.1.3 Interface and Testing

#### 4.1.3.1 Interface

A screenshot of a computer

Description automatically generated with medium confidence

#### 4.1.3.2 Testing

Test Dataset

**A screenshot of a computer

Description automatically generated with low confidence**

The software results come out in line with the original data.

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated with medium confidence**

## 4.2 Software code

### 4.2.1 Interface section code

A screenshot of a computer program

Description automatically generated with medium confidence

A picture containing text, screenshot, font

Description automatically generated

A screenshot of a computer code

Description automatically generated with low confidence

A screenshot of a computer code

Description automatically generated with medium confidence

A screenshot of a computer program

Description automatically generated with medium confidence

A picture containing text, screenshot, font, number

Description automatically generated

A screenshot of a computer code

Description automatically generated with low confidence

A screenshot of a computer program

Description automatically generated with medium confidence

A screenshot of a computer program

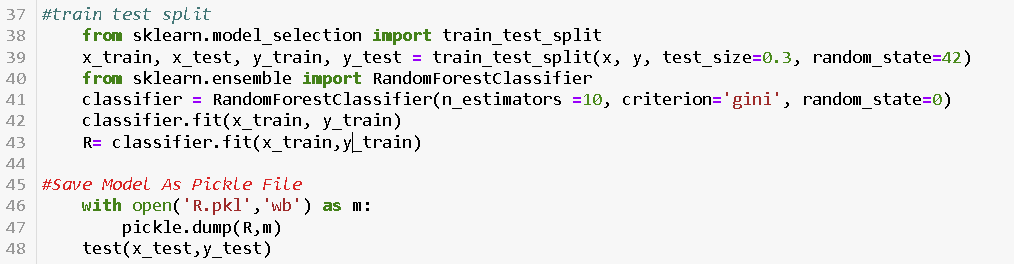
Description automatically generated with low confidence

### 4.2.2 Processing section code

A picture containing text, screenshot, font

Description automatically generated





A screen shot of a computer code

Description automatically generated with low confidence

A picture containing text, screenshot, font, line

Description automatically generated

# CHAPTER V: CONCLUSION

## 5.1 Advantages and limitations of each algorithm

### 5.1.1 Decision Tree

* **Advantages**

- The algorithm is simple, intuitive, not too complicated to understand the first time.

- The training dataset doesn't have to be too large to build an analytical model.

- Some decision tree algorithms are capable of processing missing data and faulty data without applying methods such as "imputing missing values" or removing. Less affected by the exception data.

- There is no need to make initial assumptions about the laws of distribution as in statistics, and as a result the results of the analysis obtained are the most objective, "natural".

- It can help us classify data objects according to multi-layered, multi-class classifications, especially if the target variable is a complex quantitative distortion.

- Can be applied flexibly to target variables, target variables.

- Delivers highly accurate forecast results, easy to implement, fast in training, no need to switch variables.

- Easy to interpret or explain to listeners, viewers who want to understand the results of analysis but have no knowledge of data science.

- Articulate the connection between variables and data attributes in the most intuitive way.

- In addition to economics, finance, decision tree algorithms can be applied in the fields of health, agriculture, biology.

* **Limitations**

- The decision tree algorithm works effectively on a simple dataset that has few data variables that relate to each other, and vice versa if applied to complex datasets.

- When applied with complex datasets, many different variables and attributes can lead to overfitting patterns, which are too consistent with training data leading to the problem of not giving accurate classification results when applied to test data, and new data.

- The variance value is high, when there is a small change in the dataset can affect the structure of the model.

- The tree algorithm decides to apply only to classification trees if misclassification can lead to serious mistakes.

- The tree algorithm decides whether it is likely to be "biased" or biased if the dataset is not balanced.

- Training and testing datasets must be perfectly prepared, good quality must be balanced in layers, groups in target variables.

- There is no technical "support" or "reverse query" capability.

### 5.1.2 Random Forest

* **Advantages**

**Random Forest has several advantages that contribute to its popularity and effectiveness in machine learning tasks.**

- **Robustness:** Random Forest is robust against overfitting, which occurs when a model performs well on training data but fails to generalize to new, unseen data. The algorithm's ensemble approach, combined with bootstrapping and feature randomness, helps to reduce overfitting and improve generalization performance.

- **Versatility**: Random Forest can handle a wide range of data types, including numerical and categorical features. It can also handle missing values and outliers without requiring extensive data preprocessing.

- **Feature Importance**: Random Forest provides a measure of feature importance, indicating the relative contribution of each feature to the model's predictions. This information can be used for feature selection, dimensionality reduction, and gaining insights into the underlying data.

- **Non-linearity and Interactions**: Random Forest can capture non-linear relationships between features and the target variable. It can also handle interactions between features, allowing for more complex and expressive modeling capabilities.

- **Scalability:** Random Forest can efficiently handle large datasets with a high number of features. The algorithm's parallelizability makes it suitable for parallel and distributed computing environments, enabling faster training and prediction times.

- **Resistance to Noise**: Random Forest is less affected by noisy data compared to individual decision trees. By aggregating predictions from multiple trees, the impact of noise is mitigated, leading to more reliable results.

* **Limitations:**

- **Model Interpretability:** Random Forest models can be less interpretable compared to individual decision trees. As the algorithm combines multiple trees, understanding the exact decision-making process can be more challenging.

**- Computationally Intensive:** Training a Random Forest model can be computationally expensive, especially for large datasets with numerous features. The algorithm constructs multiple decision trees, which increases the overall training time and memory requirements.

**- Memory Usage:** Random Forest models consume more memory compared to simpler algorithms. As each decision tree is stored separately, the memory footprint of the model can become substantial, particularly when dealing with a large number of trees or complex datasets.

**- Overfitting in Noisy Data:** While Random Forest is generally robust to overfitting, it can still be affected by noise or outliers in the data. Noisy features can lead to overfitting, where individual trees capture spurious patterns in the data**.**

**- Biased Class Distribution:** Random Forest may struggle with imbalanced class distributions, where one class is significantly more prevalent than others. The algorithm can have a bias towards the majority class, resulting in poorer performance for minority classes.

**- Training Time Sensitivity:** Random Forest training can be sensitive to the choice of hyperparameters, such as the number of trees or the depth of each tree. Finding the optimal set of hyperparameters can require some experimentation and tuning.

### 5.1.3 Naïve Bayes

* **Advantages**

- Independent assumptions: works well for multiple problems/data domains and applications.

- Simple but good enough to solve many problems such as text layering, spam filtering

- Easy to use and fast when it comes to guessing the label of test data. It's pretty good in multi-class prediction (test later).

- When assuming that the features of the data are independent of each other, Naive Bayes runs better than other algorithms such as logistic regression and also needs less data.

- Allows the succession of prior knowledge and obeserved data.

- It is good that there is a numerical difference between the classification classes.

- Model training (parameter estimation) is easy and fast.

* **Limitations:**

- The accuracy of Naive Bayes compared to other algorithms is not high.

- In the real world, it is almost impossible when the features of test data are independent of each other.

- Problem zero (stated how to solve it above).

- The model is not trained by a strong and rigorous optimization method.

- The parameters of the model are estimates of the probability of single conditions. Do not take into account the interaction between these estimates.

### 5.1.4 K–Nearest Neighbors

* **Advantages**

- **Easy to implement:** Given the algorithm’s simplicity and accuracy, it is one of the first classifiers that a new data scientist will learn.

- **Adapts easily**: As new training samples are added, the algorithm adjusts to account for any new data since all training data is stored into memory.

- **Few hyperparameters**: KNN only requires a k value and a distance metric, which is low when compared to other machine learning algorithms.

* **Limitations**

- **Does not scale well:** Since KNN is a lazy algorithm, it takes up more memory and data storage compared to other classifiers. This can be costly from both a time and money perspective. More memory and storage will drive up business expenses and more data can take longer to compute. While different data structures, such as Ball-Tree, have been created to address the computational inefficiencies, a different classifier may be ideal depending on the business problem.

- **Curse of dimensionality**: The KNN algorithm tends to fall victim to the curse of dimensionality, which means that it doesn’t perform well with high-dimensional data inputs. This is sometimes also referred to as the [peaking phenomenon](https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.418.6517&rep=rep1&type=pdf) (PDF, 340 MB), where after the algorithm attains the optimal number of features, additional features increases the amount of classification errors, especially when the sample size is smaller.

- **Prone to overfitting**: Due to the “curse of dimensionality”, KNN is also more prone to overfitting. While feature selection and dimensionality reduction techniques are leveraged to prevent this from occurring, the value of k can also impact the model’s behavior. Lower values of k can overfit the data, whereas higher values of k tend to “smooth out” the prediction values since it is averaging the values over a greater area, or neighborhood. However, if the value of k is too high, then it can underfit the data.

### 5.1.5 Support Vector Machine

* **Advantages**

- Effective in high dimensional spaces.

- Still effective in cases where number of dimensions is greater than the number of samples.

- Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.

- Versatile: different [Kernel functions](https://scikit-learn.org/stable/modules/svm.html#svm-kernels) can be specified for the decision function. Common kernels are provided, but it is also possible to specify custom kernels.

* **Limitations**

- If the number of features is much greater than the number of samples, avoid over-fitting in choosing [Kernel functions](https://scikit-learn.org/stable/modules/svm.html#svm-kernels) and regularization term is crucial.

- SVM do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation.

# REFERENCES

**1.** [**https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf**](https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf)

**2.** [**https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/**](https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/)

**3.** [**http://myweb.sabanciuniv.edu/rdehkharghani/files/2016/02/The-Morgan-Kaufmann-Series-in-Data-Management-Systems-Jiawei-Han-Micheline-Kamber-Jian-Pei-Data-Mining.-Concepts-and-Techniques-3rd-Edition-Morgan-Kaufmann-2011.pdf**](http://myweb.sabanciuniv.edu/rdehkharghani/files/2016/02/The-Morgan-Kaufmann-Series-in-Data-Management-Systems-Jiawei-Han-Micheline-Kamber-Jian-Pei-Data-Mining.-Concepts-and-Techniques-3rd-Edition-Morgan-Kaufmann-2011.pdf)

**4.** [**https://www.ibm.com/docs/en/spss-modeler/saas?topic=nodes-support-vector-machine-models**](https://www.ibm.com/docs/en/spss-modeler/saas?topic=nodes-support-vector-machine-models)

**5.** [**https://scikit-learn.org/stable/modules/svm.html**](https://scikit-learn.org/stable/modules/svm.html)