**PROJECT REPORT**

**TOPIC** : Demonstration of music recommendation based on age and gender using decision tree classification

**ABSTRACT:**

The dataset consists of objects with age, gender and class features which describe each user. The class refers to the genre of music that is preferred by the user. Using the dataset, we train a model based on decision tree classification which relates and classifies the preferred genre to the age and gender of the user. Using this model we will be able to predict the preferred genre using the features said above. The dataset consists of 5 different classes namely ‘Acoustic’, ’Classical’, ‘Dance’, ‘Hip-Hop’ and ‘Jazz’. The platform and language used to build the model is ‘Jupyter Notebook’ and Python 3.0 respectively. The process is elaborated below.

**PROCEDURE:**

Steps to implement in our model:

1. Importing The Data
2. Preparing The Data/Cleaning The Data
3. Predicting And Learning
4. Calculating The Accuracy
5. Visualizing Our Decision Tree

In this project we are assuming our own data set and it is as follows:

Men between age of 20-25 like Hip-Hop,

Men between age of 25-30 like Jazz,

Men above the age of 30 like Classical,

Women between the age of 20-25 like Dance

Women between the age of 25-30 like Acoustic

Women above the age of 30 like Classical

Let’s visualize this dataset in an excel sheet of a csv file format.



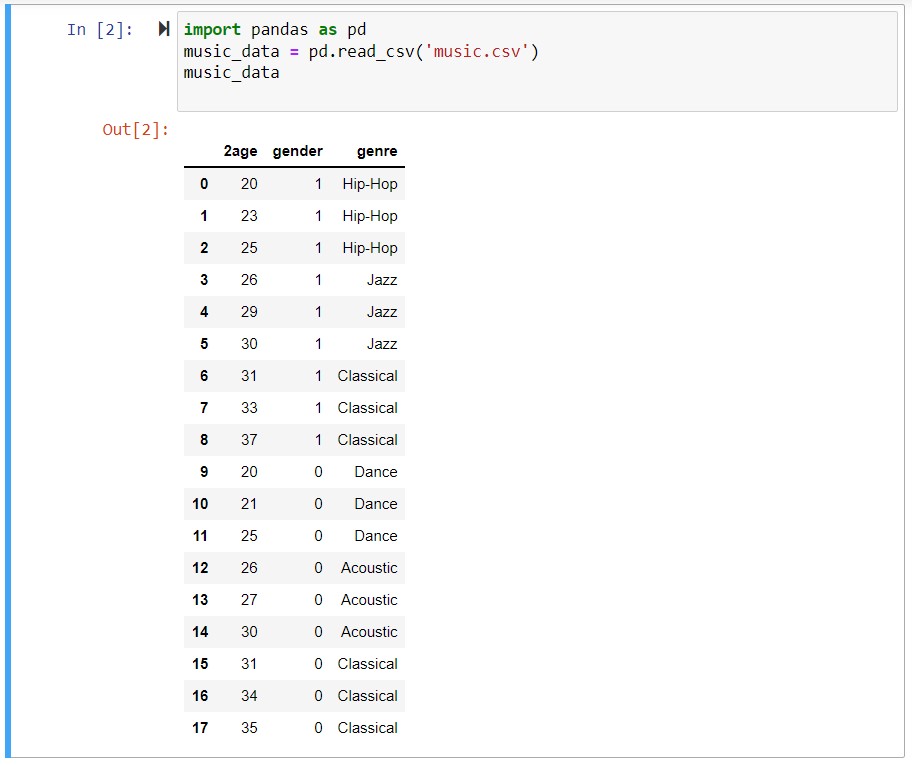
In the above data set, 1 refers to Male and 0 refers to Female.

**Step1: Importing the Data**

The musical data set was imported into the jupyter notebook using ‘pandas’ and saved into a variable called ‘music\_data’.



The data then read is output as follows :



**Step2: Preparing the Data/Cleaning the Data**

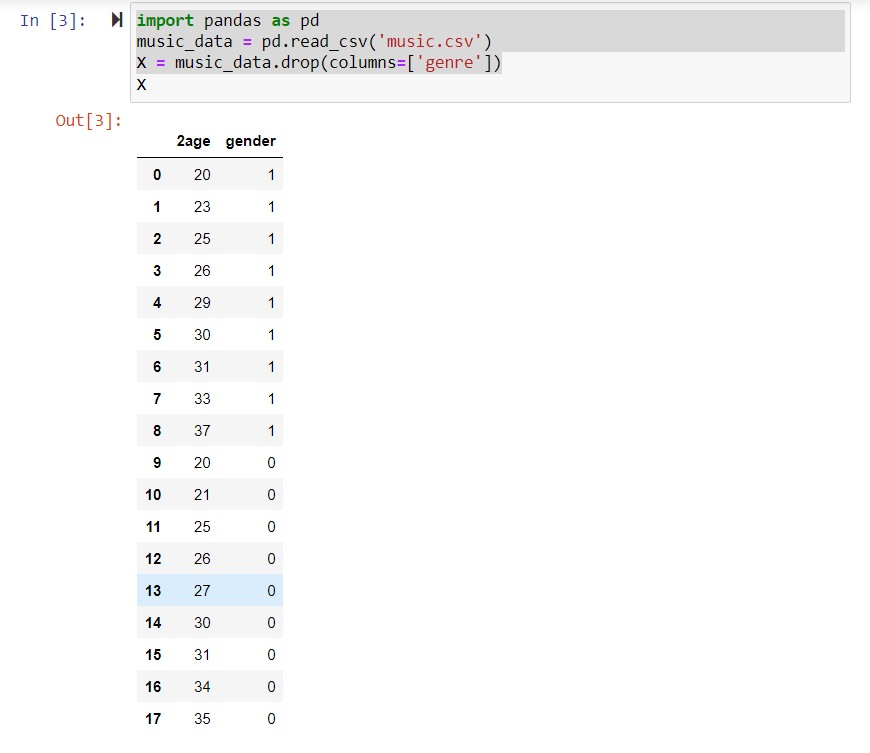
This part involves tasks such as removing duplicate null values from the data set and so on. In the data that we have chosen, there is no duplicates as all rows have values for all columns so no null values are there but there is one thing we need to do that is split this data set into two separate data sets.

* One with the 1st 2 columns i.e. ‘2age’ and ‘gender’ which we refer as input set
* The other with the ‘genre ‘ we refer as Output set.

So here we train the model by giving 2 two separate data sets i.e. Input and Output Data set. The genre column contains the predictions so we are telling in our model about that. Basically if we have a 20year old male we know that they like Hip-Hop.

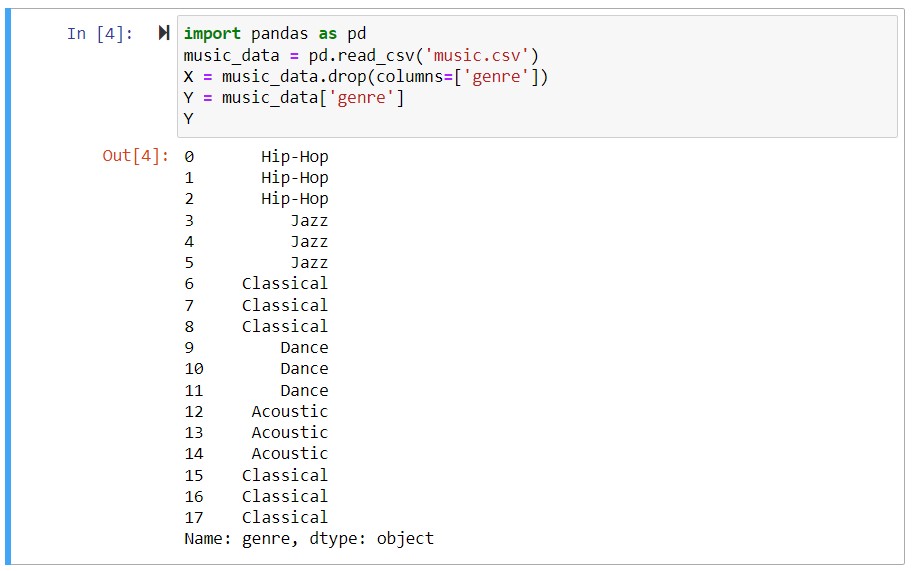
Once we train our model we expect it to predict the Preferred genre of any new data given based on the age and gender. Eg: We have a new user who is 21 years old and is a male what genre he likes? This collection of data from new user is not present in the data set considered. So expectation is that the model should predict itself.

That is the reason we separate this data set into 2 sets i.e. Input and Output.



So what we did here is we create a new data set but without ‘genre’ column. By convention we use ‘X’ to represent that Input data set and when we Run the input cell we get the following output as shown above without the ‘genre’ column.

Hence we can say that the column ‘age’ and ‘gender’ are considered here as Input set.



So from the above image ‘Y’ is considered as the output and installed the data set ‘genre’ to Y so once we type ‘Y’ and run it we will be basically storing the ‘genre’ column in Y which is the Output set.

We have now prepared our data, next we create a model using decision tree algorithm.

**Step3: Predicting and learning:**

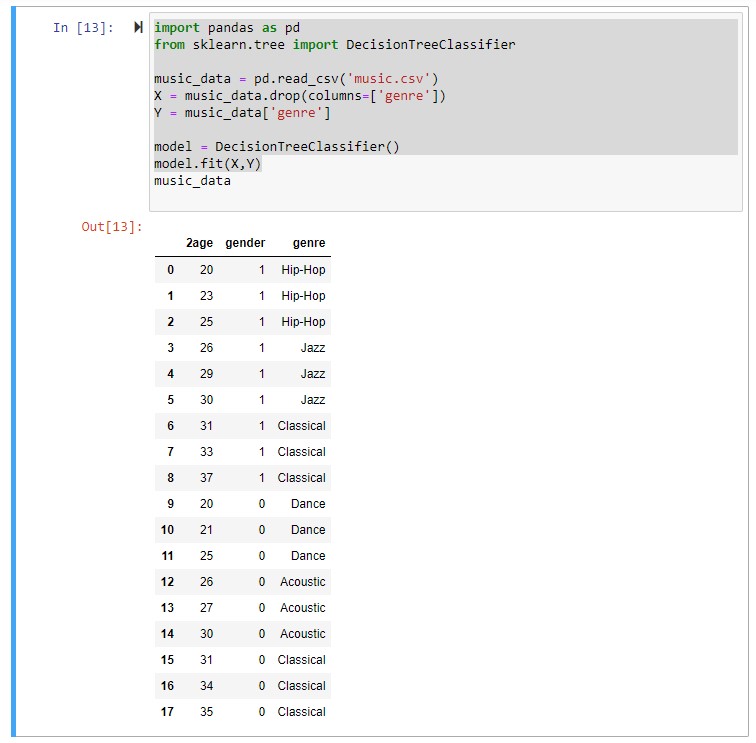
We use Decision Tree algorithm for this model which are already implemented in libraries called ‘Scikit learn’.

What is Decision Tree?

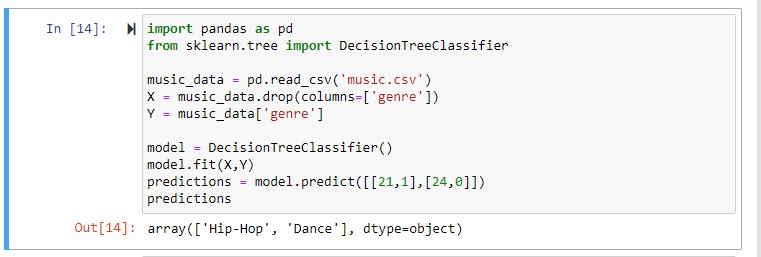
A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. It is one way to display an algorithm that only contains conditional control statements.

The above definition means that the data set will be classified in the form of a tree which starts with the parent node and ends with the Child nodes that are completely classified. At each node a condition is assumed which splits the parent node into child nodes in the best possible way. The function of split is to reduce the randomness of the nodes. The nodes are split until randomness becomes 0. We measure randomness using ‘gini’ impurity, information gain, and entropy based on information theory.

The data set was fit into the model ‘decisiontreeclassifier’ based on ‘scikitlearn’ and RUN.



After the model was completely prepared and trained we fit in another model which predicts the class based on necessary features i.e. age and gender.



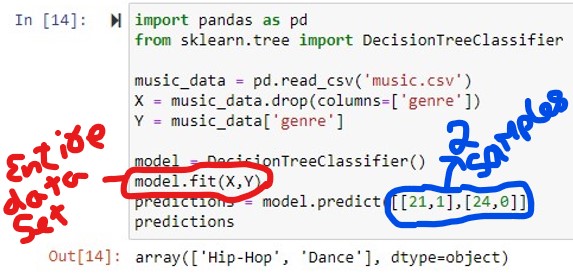


The results are shown and compared as above.

**Step4: Calculating The Accuracy**

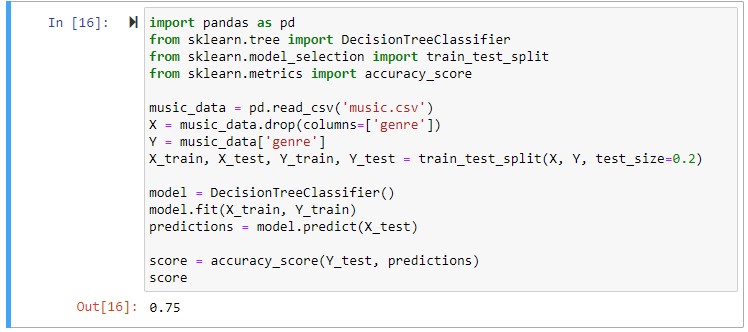
To measure the accuracy for the model, we split the data set. One for ‘training’ and one for ‘testing’.

Right now we are passing the entire data set. So for training model we are using 2 samples for making predictions.



Generally it is 70-80% of data for training & 20-30% for testing. Then we can pass a data-set for testing and we’ll get predictions in the test set based on that we can calculate the accuracy.

So;



As you can see It’s showing 0.75(75%).

Now we run it;



Again If I run I get



That is not enough to calculate the accuracy of a model.

Again I run it



Another Run



Another one



Another Run



Another one



Another one



Another run



Every model has a feature called accuracy score which determines the precision of the model. The higher the accuracy score the more precise your model is. The data set is read into variables X and Y and it is split into train and test sets. In this model, we give a feature called test size which determines the size of the test and train sets. This means that out of the entire data set only a small portion will be taken to train the model and the rest will be used to test it. Here, we have given the test size as 0.2. This means that out of the 18 objects in the dataset, 20% of it is considered as the test set. This means only 4 objects are taken but which 4 of these objects are modelled as train sets is completely random. This is the reason why upon different iterations different accuracy scores are shown. This gives a more complete approach and understanding of the said model. For example, in our first iteration, we see that the accuracy score is 1. If the accuracy score is 1, it means that the model is 100% error free and this is possible only when all the objects already belongs to the same class in our dataset. Here, the class ‘Classical’ has six objects and randomly all four of the objects considered belong to the same class hence determining the accuracy to be 1. Similarly different combinations of the tests it will give you different accuracy model evidently seen as 0.75, 0.5 and so forth.

**Step5: Visualizing our Decision Tree:**

Decision Tree is the easiest to understand. **Here we will export out model in visual format to see how this model is classified. [NOTE: The model is exported as dot file and viewed with visual studio code]**



At line3: we import a function called tree which stimulates the goal to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features. At line9: we use ‘tree.export\_graphviz’ which takes many parameters.

1st parameter is out\_file which we set it to music-recommender.dot format that is a graph description language.

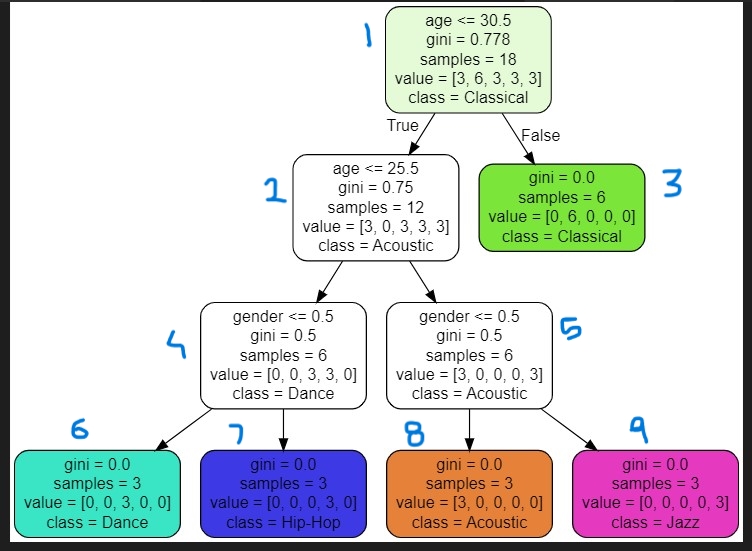
2nd parameter is feature\_names by setting it to an array of 2 strings ‘age’ & ‘gender’ which are the features or the columns of our data set. So they are the properties or features of our data.

3rd parameter is class\_names by setting to list of classes or labels we have in our Output data set ‘(Y)’ like Hip-Hop, Jazz, Classical and so on. So this ‘Y’ data includes all genre or all the classes of our data but they are repeated a few times in our data set. So we call ‘class\_names= Y.sortedunique()’ which returns a unique list of classes that is sorted alphabetically.

We now use visual studio code and in a new window drag and drop the dot file.

We will see a dot format which is a textural language that describes the graph.

**CONCLUSION:**



This is exactly how our model works. We start with the parent node which consists of all 18 objects and 5 classes. We now find the feature to split the parent node into 2 child nodes. The impurity of each node is measured by gini index where pure nodes have a gini index of 0. The feature ‘Age <= 30.5’ is considered here, because it’s the best feature split. This condition splits the parent node into 2 child nodes where one of them is a pure node i.e all object belong to the same class. This pure node cannot be split further. Similarly in all further splits we choose the feature such that the child node have the least gini impurity. The node 2 has multiple classes and needs to be split further. In a similar fashion the features ‘Age <=25.5’ splits the nodes further and features ‘Gender <= 0.5’ splits it further down until all the nodes are pure. As we finally see, the last nodes of different classes all have gini value of 0, stating that they all are pure nodes.