

## ASSIGNMENT

12/27/2022

Name: **SUMAN SEKHAR MEHER**  
College: **CHANDIGARH UNIVERSITY**

E-mail: [rajameher111@gmail.com](mailto:rajameher111@gmail.com)  
Mobile: 8456833282

### 1. Perform Exploratory Data Analysis (EDA) on Iris dataset.

#### What is Exploratory Data Analysis (EDA) ?

Exploratory Data Analysis (EDA) is a technique by which we can analyze the data with the help of some data visualizations methods. By using this EDA we can perform basic statistical methods to determine the statistical information of the dataset. This technique helps us to deal with the missing values, redundancy, pattern, outliers, etc.

#### Iris Dataset:-

The iris dataset is the collection of information about different iris flowers. In this dataset, the researchers have recorded various features of different iris flowers. Iris dataset having five columns like sepal length, sepal width, petal length, petal width, class.

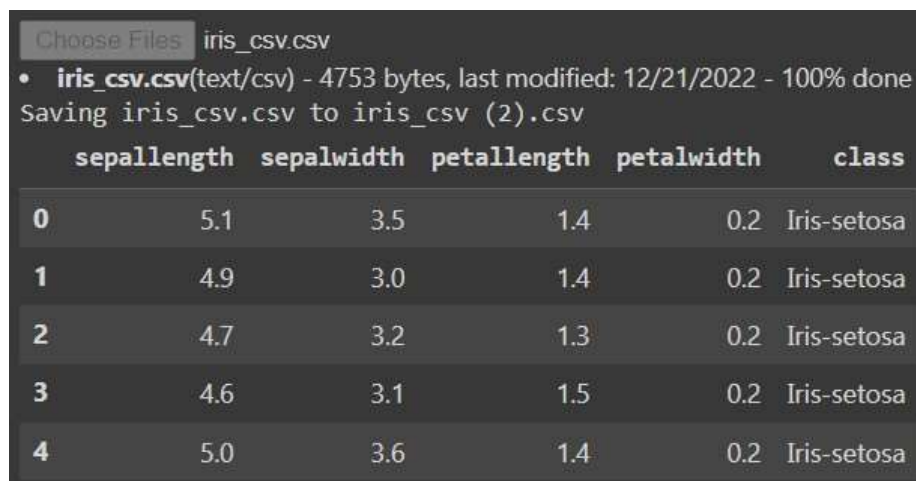
Now, we perform the EDA technique in Iris dataset by using Google Collabs and Python. First, we need to import the required libraries and Iris dataset to perform the EDA in Iris dataset. For this the coding and output is given below,

#### Code:

```
from google.colab import files
import pandas as pd
import io

uploaded = files.upload()
df = pd.read_csv(io.BytesIO(uploaded['iris_csv.csv']))
df.head()
```

#### Output:



The screenshot shows the Google Colab interface. At the top, there is a 'Choose Files' button and a file named 'iris\_csv.csv' is listed. Below this, a message indicates that the file 'iris\_csv.csv' (text/csv) is 4753 bytes and was last modified on 12/21/2022. It also shows the file is being saved as 'iris\_csv (2).csv'. Below the message, a table preview is displayed with the following columns: 'sepal length', 'sepal width', 'petal length', 'petal width', and 'class'. The table shows the first five rows of data, all of which are 'Iris-setosa'.

	sepal length	sepal width	petal length	petal width	class
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

In the above example we import `files` from `google.colabs` for file operations like upload and download of the file, `pandas` for Data analysis methods, `io` for user input operation like we can use the uploaded file further in our program. In this program first we uploaded the iris dataset from local drive and use head() to get upto 5 rows records of the dataset.

Now, we are going to find the shape of the iris dataset using **shape**. To find the shape we write the following code,

Code:     df.shape

Output:

```
(150, 5)
```

From the above we get to know about the shape i.e. the iris dataset having 5 columns & rows of 150.

Now, we will find the details about the attributes and their datatypes by using the **info()** function.

Code:     df.info()

Output:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
#   Column          Non-Null Count  Dtype  
---  -
0   sepallength      150 non-null   float64
1   sepalwidth       150 non-null   float64
2   petallength      150 non-null   float64
3   petalwidth       150 non-null   float64
4   class            150 non-null   object  
dtypes: float64(4), object(1)
memory usage: 6.0+ KB
```

From the above output, we can observe that the dataset have four columns with numeric datatype and one column with categorical values. It display all the basic details about the dataset.

Now, we will perform the basic and common statistical operation on the data values by using **describe()**. By using this function we can get mean, standard deviation, minimum & maximum value, 1<sup>st</sup>, 2<sup>nd</sup> & 3<sup>rd</sup> quantile values for all the columns. This function gives us a efficient information about the distribution of the data. And also it skip the missing values like NA, NaN.

Code:     df.describe()

Output:

	sepal.length	sepal.width	petal.length	petal.width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.054000	3.758667	1.198667
std	0.828066	0.433594	1.764420	0.763161
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

Now, we will use `isnull()` function to check the missing values whether missing values present in the iris dataset or not.

Code: `df.isnull()`

Output:

0	False	False	False	False	False
1	False	False	False	False	False
2	False	False	False	False	False
...	...	...	...	...	...
148	False	False	False	False	False
149	False	False	False	False	False

150 rows x 5 columns

Now, we use `drop_duplicates()` to check whether duplicate values present in the dataset or not.

Code: `data = df.drop_duplicates(subset = "class",)`

`data`

Output:

	sepal.length	sepal.width	petal.length	petal.width	class
0	5.1	3.5	1.4	0.2	Iris-setosa
50	7.0	3.2	4.7	1.4	Iris-versicolor
100	6.3	3.3	6.0	2.5	Iris-virginica

Now, we use **value\_counts()** function to find the total number of records in each class of the dataframe.

Code: `df.value_counts("class")`

Output:

```
class
Iris-setosa      50
Iris-versicolor  50
Iris-virginica   50
dtype: int64
```

## **DATA VISUALIZATION:**

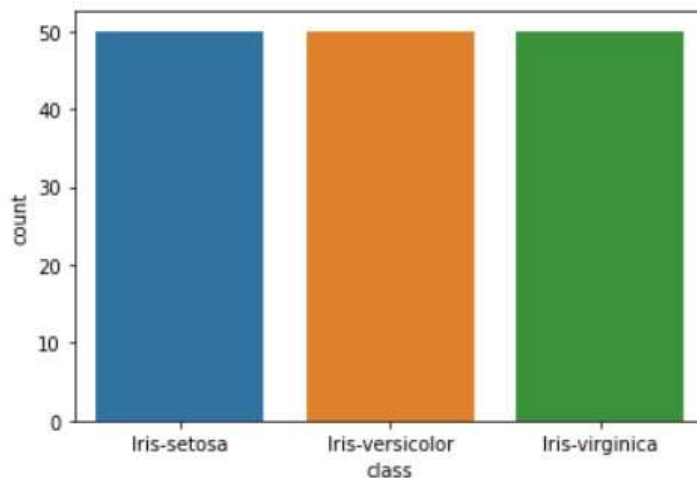
Data visualization is the technique by which we can represent our information and data in graphical ways using graph, histogram, pie chart, bar plot, scatter plot, etc. In python Matplotlib and Seaborn are two libraries those are used in data visualization.

Now , we perform some of the basic graphical representation on the **Iris** dataset.

Code: `import seaborn as sb
import matplotlib.pyplot as plt

sb.countplot(x='class', data=df, )
plt.show()`

Output:

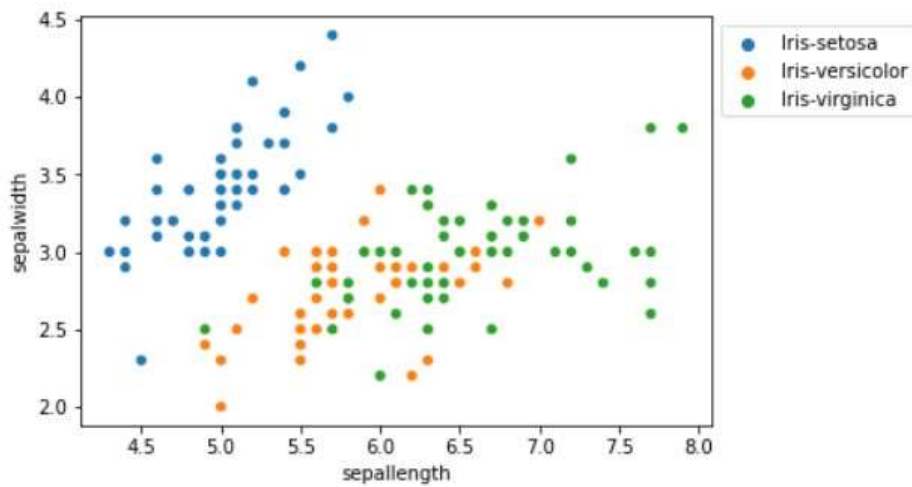


The above graph indicates that the field 'class' have 3 categorical data and having 50 records from each.

Now, we find out the relationship between **Sepal Length & Sepal Width**, also find the relation between **Petal Length & Petal Width** by using Scatter Plot.

Code: `sb.scatterplot(x='sepalwidth', y='sepalwidth', hue='class', data=df, )
plt.legend(bbox_to_anchor=(1, 1), loc=2)
plt.show()`

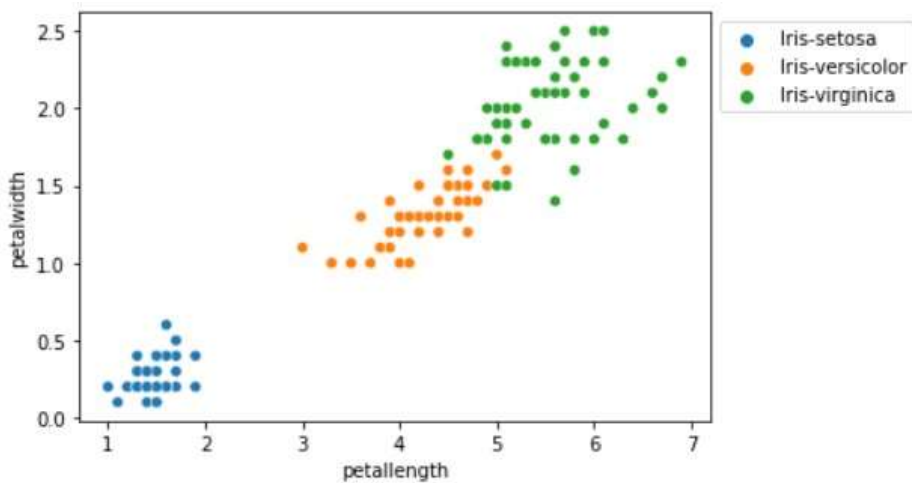
Output:



Code:

```
sb.scatterplot(x='petallength', y='petalwidth', hue='class', data=df, )  
plt.legend(bbox_to_anchor=(1, 1), loc=2)  
plt.show()
```

Output:

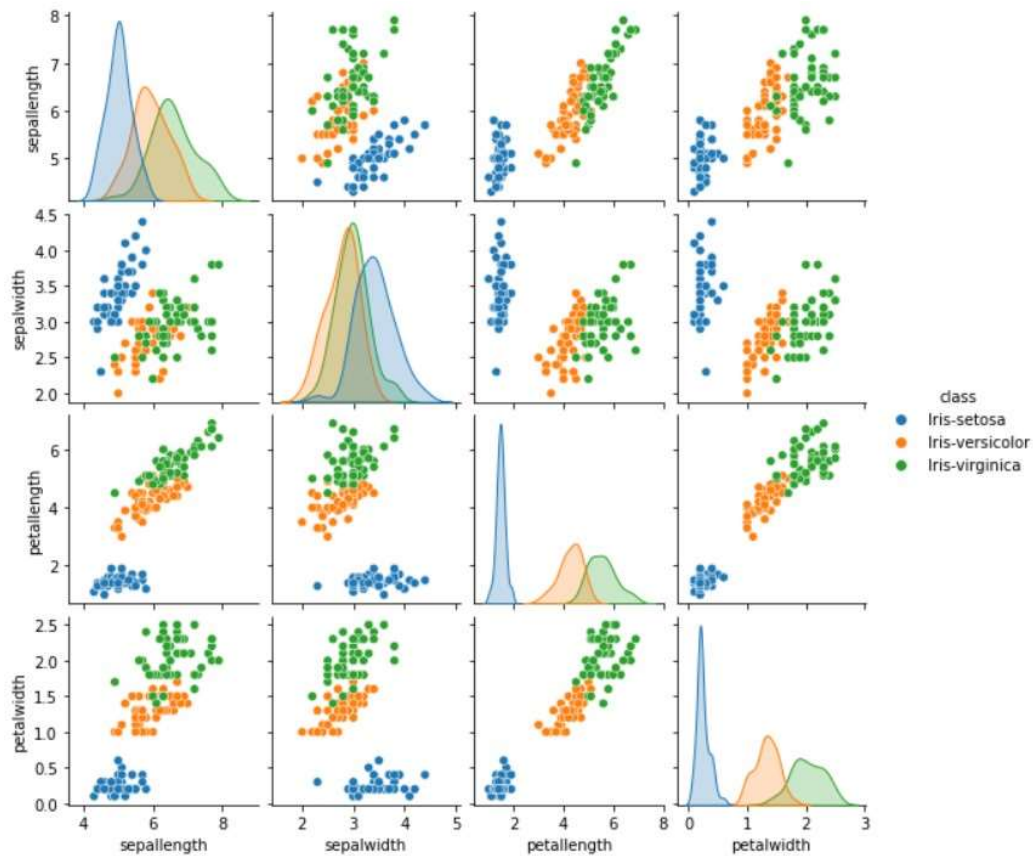


Now, show the relationship among all columns of the given **Iris dataset**. This is done by multivariate analysis using **pairplot()**.

Code:

```
sb.pairplot(df, hue='class', height=2)
```

Output:



### Histogram:

Histogram is a type of graphical representation of data values. It is used to analyze the distribution of data of fields in dataset. It is used for both uni-variate as well as bivariate data analysis.

Code:

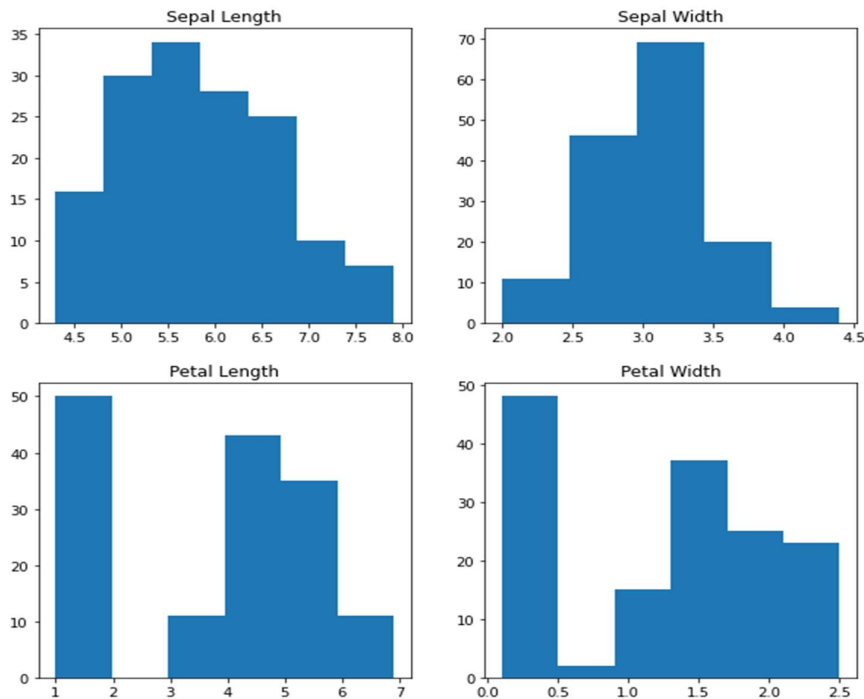
```
fig, axes = plt.subplots(2, 2, figsize=(10,10))
axes[0,0].set_title("Sepal Length")
axes[0,0].hist(df['sepalength'], bins=7)

axes[0,1].set_title("Sepal Width")
axes[0,1].hist(df['sepalwidth'], bins=5);

axes[1,0].set_title("Petal Length")
axes[1,0].hist(df['petallength'], bins=6);

axes[1,1].set_title("Petal Width")
axes[1,1].hist(df['petalwidth'], bins=6);
```

Output:



### Histograms with Distplot Plot:

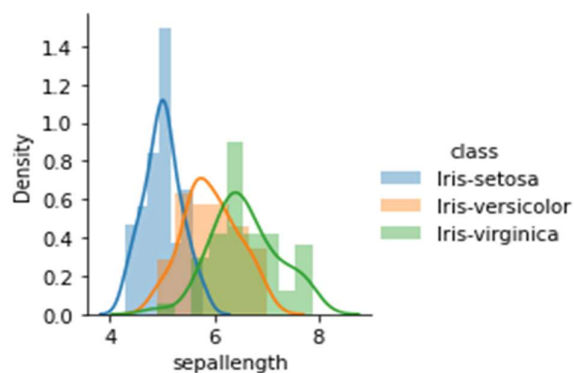
Distplot is a graphical representation used with histogram for univariate data visualization and observations.

Code:

```
plot = sns.FacetGrid(df, hue="Species")
plot.map(sb.distplot, "sepalength").add_legend()

plt.show()
```

Output:



## Heatmap:

Heatmap is a type of graphical representation in where the data values are represented as colors. It is used to show the correlation among all the numeric values present in the data set.

For draw a heatmap we use numpy, matplotlib, seaborn libraries to represents the data values.

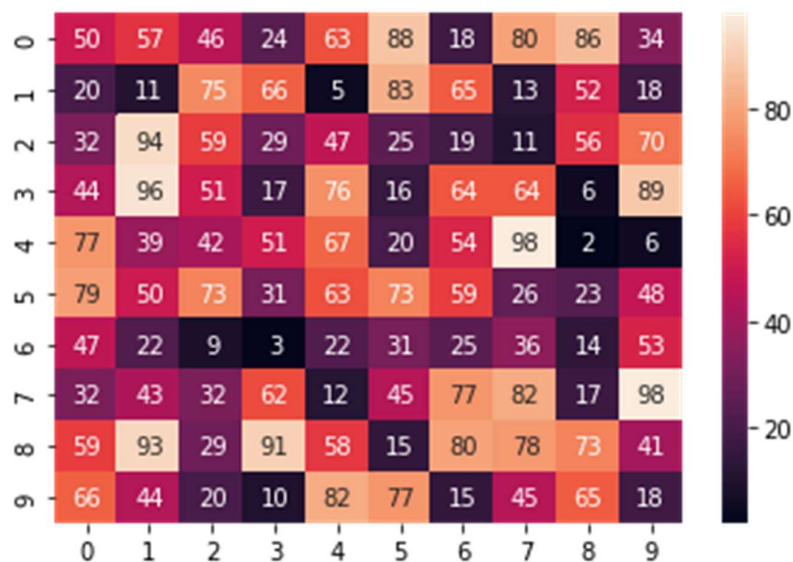
Code:

```
import numpy as np

data = np.random.randint(low = 1, high = 100, size = (10, 10))
sb.heatmap(data = data, annot= True)

plt.show()
```

Output:



## Box Plot:

Box plot is a type of graphical representation of statistical data mainly used to draw the 2<sup>nd</sup> and 3<sup>rd</sup> quartiles. It looks like slider which shows the quartile values and median value in the middle portion of the box.

Code:

```
def graph(y): sb.boxplot(x="class", y=y, data=df)

plt.figure(figsize=(10,10))
plt.subplot(221)
graph('sepalength')

plt.subplot(222)
graph('sepalwidth')

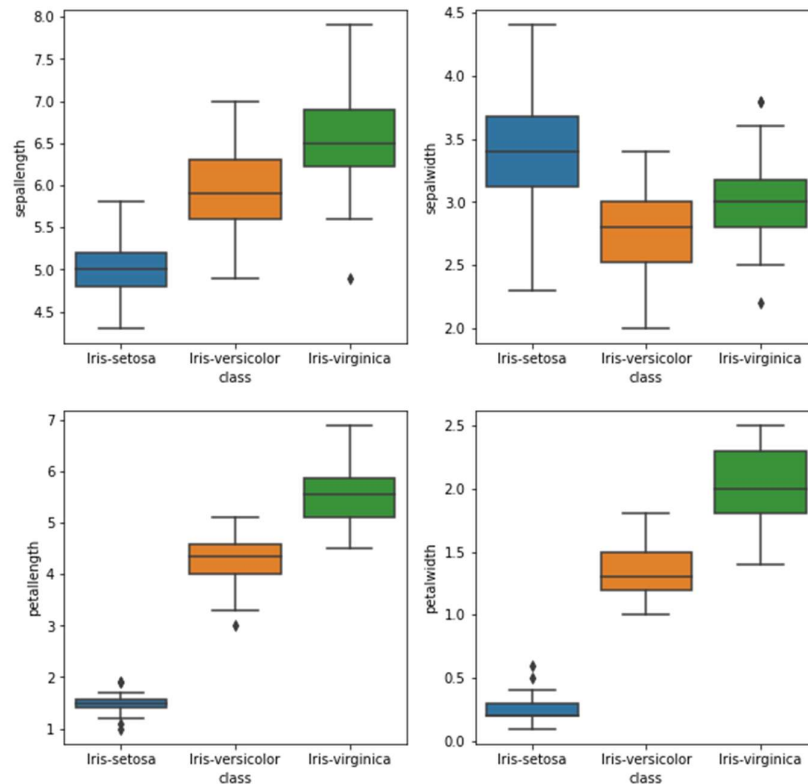
plt.subplot(223)
graph('petallength')
```



```
plt.subplot(224)
graph('petalwidth')

plt.show()
```

Output:



## 2. What is Decision Tree? Draw decision tree by taking the example of Play Tennis.

### Decision Tree:

Decision trees are the most powerful and popular classification and prediction tools. A decision tree is a flowchart-like tree structure where each inner node specifies a test for an attribute, each branch represents the result of the test, and each leaf node contains a class label.

**Building a decision tree:** The tree can be "learned" by splitting the source text into subsets based on attribute value tests. This method is performed recursively for each derived subset, which is known as recursive partitioning. Recursion is complete when a subset of nodes all have the same value of the target variable, or when the split adds no more values to the predictions. Construction of decision tree classifiers is suitable for exploratory knowledge discovery because it does not require domain knowledge or parameter tuning. Decision trees

can handle high-dimensional data. In general, a good result of accuracy determined by decision tree. Decision tree induction is a classic inductive approach for learning classification knowledge.

**Decision tree representation:** A decision tree classifies an instance by reordering the tree from the root to leaf nodes that provide a classification of the instance. As shown in the diagram above, instances are classified by starting at the root node of the tree, testing the attribute specified at that node, and then moving down the attribute's value are corresponded by the decision tree. This process is repeated for the newly created node-rooted subtree. The decision tree in the figure above classifies a particular morning according to its suitability for playing tennis and returns the classification associated with a particular hand.

### Draw decision tree by taking PlayTennis dataset:

To draw the decision tree of PlayTennis dataset, we need to use some libraries and PlayTennis dataset.

Code:

```
from sklearn.preprocessing import LabelEncoder
from google.colab import files
from sklearn import tree
import pandas as pd
import numpy as np
import graphviz
import io

uploaded = files.upload()
pt = pd.read_csv(io.BytesIO(uploaded['PlayTennis.csv']))
pt.head()
```

Output:



The screenshot shows a Jupyter Notebook interface. At the top, there is a button labeled 'Choose Files' and the filename 'PlayTennis.csv'. Below this, a message indicates that the file 'PlayTennis.csv' (text/csv) is 408 bytes, last modified on 12/27/2022, and is 100% done. A second message says 'Saving PlayTennis.csv to PlayTennis (2).csv'. Below the messages, there is a table with 6 columns: outlook, temp, humidity, windy, play, and a play icon. The table contains 5 rows of data, indexed 0 to 4.

	outlook	temp	humidity	windy	play
0	sunny	hot	high	False	no
1	sunny	hot	high	True	no
2	overcast	hot	high	False	yes
3	rainy	mild	high	False	yes
4	rainy	cool	normal	False	yes


The above code imports the basic libraries for the decision tree. It includes programs for upload the PlayTennis dataset for using the data in decision tree in further program. It displayed the overview of dataset with 5 records from starting.

Now, converts all the non-numeric values to numeric values for the analysis and make the decision tree.

Code:

```
result = LabelEncoder()
pt['outlook'] = result.fit_transform(pt['outlook'])
pt['temp'] = result.fit_transform(pt['temp'])
pt['humidity'] = result.fit_transform(pt['humidity'])
pt['windy'] = result.fit_transform(pt['windy'])
pt['play'] = result.fit_transform(pt['play'])
pt
```

Output:



	outlook	temp	humidity	windy	play
0	2	1	0	0	0
1	2	1	0	1	0
2	0	1	0	0	1
3	1	2	0	0	1
4	1	0	1	0	1
5	1	0	1	1	0
6	0	0	1	1	1
7	2	2	0	0	0
8	2	0	1	0	1
9	1	2	1	0	1
10	2	2	1	1	1
11	0	2	0	1	1
12	0	1	1	0	1
13	1	2	0	1	0

Now, split the data into two sets i.e. x & y training set. In this set X contains the training data and y contains the decision data.

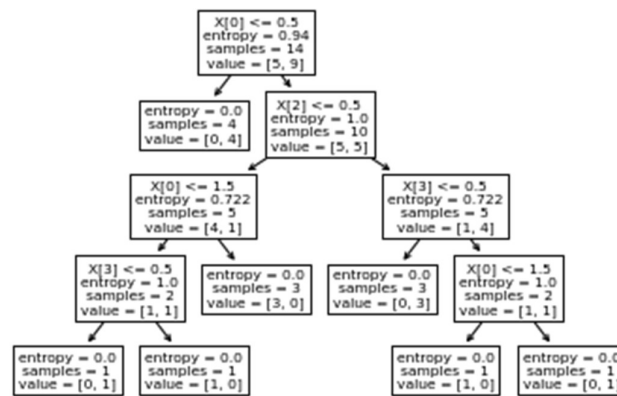
Code:

```
y = pt['play']
X = pt.drop(['play'],axis=1)

clf = tree.DecisionTreeClassifier(criterion = 'entropy')
clf = clf.fit(X, y)

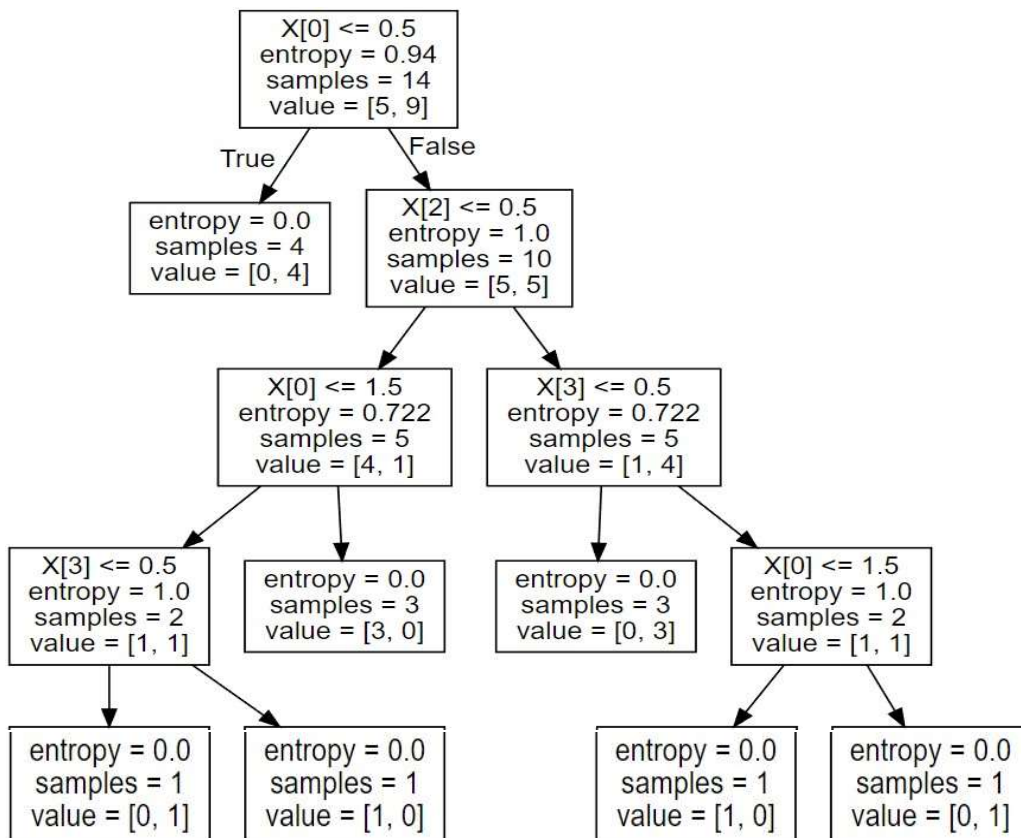
tree.plot_tree(clf)
```

Output:



Code: `dot_data = tree.export_graphviz(clf, out_file=None)`  
`graph = graphviz.Source(dot_data)`  
`graph`

Output:



Here the Decision Tree is clearer and larger in size which is possible by GraphViz. The above graph is the Decision tree for the PlayTennis dataset.

### **3. In k-means or KNN, we use Euclidean distance to calculate them distance between nearest neighbours. Why not Manhattan distance ?**

*We don't use the Manhattan distance, because distances are only calculated horizontally or vertically in Manhattan distance.*

We mainly use Euclidean distance measurement technique to find the distance between consecutive points. Commonly used to find the distance between two real-valued vectors. Euclidean distance is used when you need to calculate the distance of real values like integer, float etc. The problem with this distance measurement technique is that the data must be normalized or standardized by scale. Otherwise, the results will not be favorable.

Manhattan distance is the easiest way to calculate the distance between two points. This distance is often referred to as the taxi distance or the city block distance. You can easily relate this to your daily life. When you start somewhere and arrive at your destination, the Manhattan distance tells you the distance between your starting point and your destination. More mathematically, it computes the absolute value between two points. Calculate the distance exactly like the original path. Do not use diagonal or shortest paths.

### **4. How to test and know whether or not we have overfitting problem?**

Overfitting can be identified by looking at validation metrics such as accuracy and loss. Validation metrics typically increase to a point where they plateau or start declining if the model is overfitting. During an uptrend the model looks for a good fit. If this is achieved, the trend will descend or stagnate.

It's almost impossible to detect overfitting until you test your data. This helps to address the inherent properties of overfitting that cannot be generalized from the dataset. So you can split your data into different subsets to simplify training and testing. The data is split into two main parts. H. Test set and training set.

The training set represents the majority (around 80%) of the data available to train the model. The test set represents a small portion (approximately 20%) of the dataset and is used to test the accuracy of data that has never been exchanged. By segmenting the datasets, we can examine the performance of the model on each dataset, detect when overfitting occurs, and see how the training process works.

To infer the presence of overfitting, we can measure performance using the percentage of accuracy observed on both datasets. If the model performs better on the training dataset than on the test dataset, it means that the model is likely overfitted.

### **5. How is KNN different from k-means clustering?**

- The two most commonly used algorithms in machine learning are k-means clustering and k-nearest neighbor algorithms.
- K-NN performs much better when all data are on the same scale, which is not the case for K-Means.

- In machine learning KNN is a supervised machine learning algorithm, whereas K-mean is a unsupervised algorithm/technique.
- K-NN is a lazy learner, whereas K-Means is an avid learner. Dedicated learners have model fit, which means a training step, while lazy learners do not have a training phase.
- K-NN is a machine learning classification or regression algorithm and K-means is a machine learning clustering algorithm.

## 6. Can you explain the difference between a Test Set and a Validation Set?

### Test set:

- A set of data used for evaluating the performance of a fully specified classifier model.
- A sample of data used to provide an unbiased evaluation of the final model fit on the training data set.
- It ensure it generalizes well to new, invisible data points.
- It is used to obtain an unbiased estimate of the performance of the final model.

### Validation set:

- A set of examples used to tune classifier parameters. B. Choose the number of hidden units in the neural network.
- A sample of data used to provide an unbiased evaluation of the model's fit on the training dataset while optimizing the model's hyperparameters.
- It is responsible for hyperparameter tuning and model selection.
- It is used for optimizing the model parameters.

## 7. How can you avoid overfitting in KNN?

Here are some of the most common overfitting solutions:

- **Cross-validation:** Cross-validation is a powerful safeguard against overfitting. This idea is clever. Generate multiple mini train test splits using the initial training data. After splitting of train test to tune the models.
- **Features Removing:** To prevent overfitting, we can manually improve generalizability by removing irrelevant input features.
- **Regularization:** Regularization refers to various techniques for artificially simplifying a model. How you do this depends on the type of learner you are using.
- **Ensembling:** Ensemble is a machine learning technique for combining predictions from multiple individual models.
- **Train the model again with more data:** It doesn't always work, but the more data you train with, the better your algorithm will be at recognizing signals.
- **Early stopping:** Training a machine learning model iteratively allows you to measure the performance of each iteration of the model.

## 8. What is precision call ?

A classification model's capacity to detect only relevant data items. Mathematically, precision the number of true positives divided by the number of true positive plus the number of false positive.

$$\text{Precision} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$

## 9. Explain How a ROC Curve works.

Receiver Operating Characteristic Curve or ROC curves are a great tool for evaluating the performance of classification models. Measuring the overall quality of a model is not an easy task. The ROC curve is a metric for binary classification problems. There are four outcomes produced by the model those are: true positive, true negative, false positive, and false negative.

- A **true positive** is the result of the model correctly predicting the positive class.
- A **true negative** is the result of the model correctly predicting the negative class.
- A **false Positive** is the result of the model incorrectly predicts the positive class.
- A **false negative** is the result of the model incorrectly predicting the negative class.

The ROC curve visualizes the discriminative power of the classifier at various thresholds. Record the following two parameters:

- True Positive Rate
- False Positive Rate

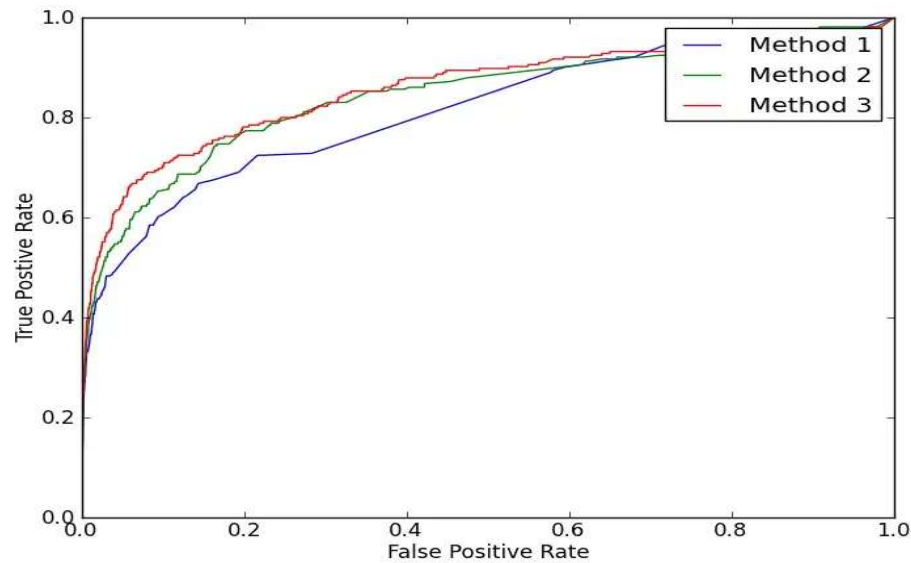
**True positive rate (TPR)** is synonymous with recall and is therefore mathematically represented as:

$$\text{True Positive Rate} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Negative}}$$

Same as previous, **False Positive Rate (FPR)** is mathematically represented as:

$$\text{False Positive Rate} = \frac{\text{False Positive}}{\text{False Positive} + \text{True Negative}}$$

Lowering the classification threshold increases both TPR and FPR as more inputs are classified as true positives. Below is the graph for ROC curve



It can be evaluated as a measure of the performance of three different classifiers at different classification thresholds. Calculating ROC values is very inefficient when you're dealing with millions of data points. One solution to this is called the area under the curve or AUC curve.

## 10. What is Accuracy ?

Accuracy is an evaluation metric that is mainly used for classification model. The accuracy value for the model lies between 0 and 1. If the model gives 0.92 values, it means the model is working with 92% accuracy.

$$\text{Accuracy} = \frac{\text{No.of correct predictions}}{\text{Total Number of predictions}}$$

## 11. What is F1 score ?

F1 is commonly used to measure the performance of binary classification. It combines the precision and recall score by taking the harmonic mean as the average.

$$\frac{2(\text{Precision} * \text{Recall})}{\text{Precision} + \text{Recall}}$$

## 12. What is Recall ?

It is used to measure the model performance in terms of measuring the count of true positive in a correct manner out of all the actual positive values. It is used when the dataset is imbalance.

$$\text{Recall} = \frac{\text{True Positive}}{\text{True Positive} + \text{False Positive}}$$



### 13. What is Confusion matrix and why do we need it?

It is a square matrix which tell how many predictions are true and how many are wrong.

For Example Suppose we have the confusion matrix  $\begin{bmatrix} 3 & 2 \\ 5 & 10 \end{bmatrix}$

This matrix tell that 13 predictions are true and 7 predictions are wrong

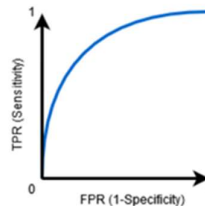
$$\text{Accuracy} = \frac{\text{True Positive} + \text{True Negative}}{\text{True Positive} + \text{False Positive} + \text{False Postitive} + \text{False Negative}}$$

#### NEED OF THE CONFUSION MATRIX

It is used to measure the accuracy and performance of the ML model.

### 14. What do you mean by AUC curve?

AUC stands for Area Under Curve is the measure of the ability of a classifier to distinguish between classes and is used as a summary of the ROC curve. We Can get the higher accuracy level when the AOC curve reached to the higher level.



When  $0.5 < \text{AUC} < 1$ , there is a high chance that the classifier

### 15. What is Precision-Recall Trade off ?

The accuracy-recall curve illustrates the trade-off between precision and recall at various levels. A large area under the curve indicates both high recall and high accuracy, with high precision corresponding to a low false positive rate and high recall corresponding to a low false negative rate.

### 16. What are Decision Tree?

It is a supervised Learning algorithm that can be used to for both classification and regression problem. It is a tree-structured classifier where internal nodes represent the columns of the dataset, branches represent the decision rules and each leaf node represent the outcome.

It is a graphical representation for getting all the possible solutions to a problem/ decision based on a given condition.

#### Algorithm:

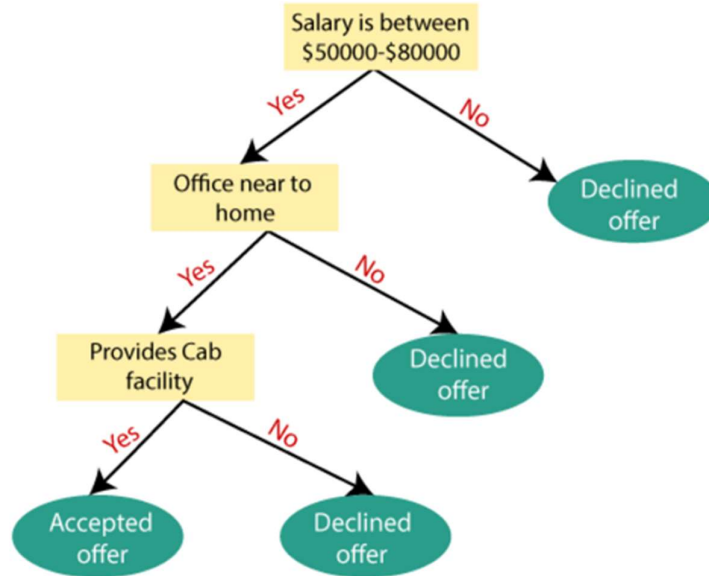
Step-1 Begin the tree with the root node which contain the complete dataset.

Step-2 Using the Attribute Selection Measure, find the best attribute in the dataset (ASM).

Step-3 Divide the S into subsets that contain possible values for the best attribute.

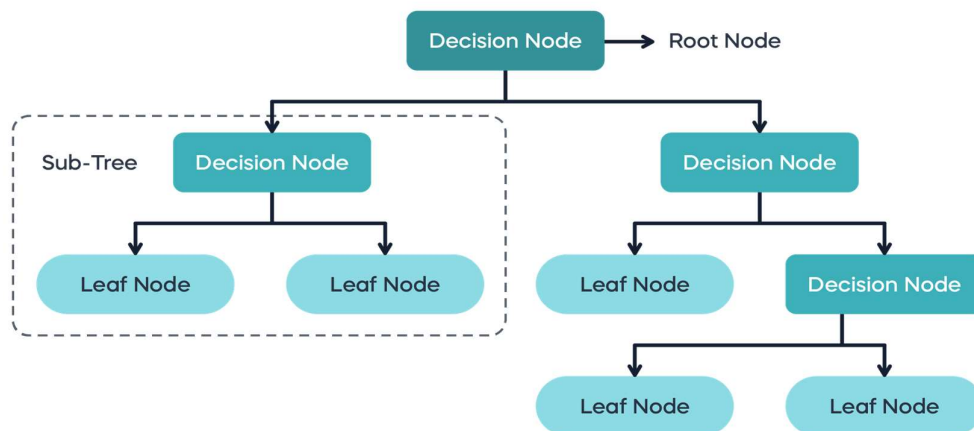
Step-4 Generate the decision Tree node, the tree node with the best characteristic.

Step-5 Recursively make new decision tree using the subsets of the dataset created in Step-6 Continue this process until a stage is reached where you cannot further classify the nodes and called this final node as a leaf node.



## 17. Explain the structure of a Decision Tree.

Decision trees are nonparametric supervised learning algorithms used for both classification and regression tasks. It has a hierarchical tree structure consisting of root nodes, branches, internal nodes, and leaf nodes.



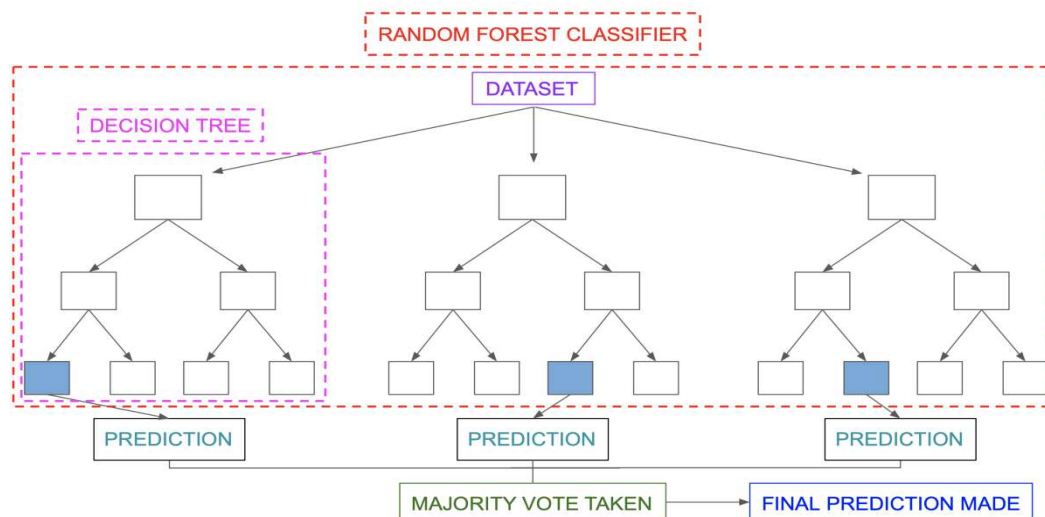
In the above diagram, a decision tree begins with a root node that has no incoming branches. The root node's outgoing branches then feed into the internal nodes, also known as decision nodes. Both node types evaluate the given characteristics to generate homogeneous subsets, which are designated as leaf nodes or terminal nodes. The leaf nodes reflect all of the dataset's conceivable outcomes.

## 18. What are some advantages of using Decision Trees?

- It can be used for both categorical data and numerical data. However, categorical variables are not yet supported by the scikit-learn implementation. Other techniques tend to specialize in analyzing datasets containing only one type of variable.
- It is responsible for multiple output models or problems.
- A situation is observable in the model, the description of that state can be easily explained by Boolean logic. Results from black-box models (such as artificial neural networks) might, on the other hand, be more difficult to comprehend. White box is technique used by decision tree.
- The cost of using a tree is the logarithm of the number of data points used to train the tree.
- Easy to understand and easy to interpret. You can visualize trees.
- It works well even when the underlying model from which the data were derived violates some of the assumptions.
- To validate the model, statistical tests may be utilized. This may explain the model's dependability.

## 19. How is a Random Forest related to Decision Trees?

A random forest is a collection of decision trees whose outcomes aggregate into a final outcome. The ability to limit overfitting without significantly increasing the error due to distortion is what makes these models so powerful. That said, random forests are a powerful modeling technique and much more robust than single decision trees. It aggregates many decision trees to limit overfitting and errors due to bias, thus yielding useful results.



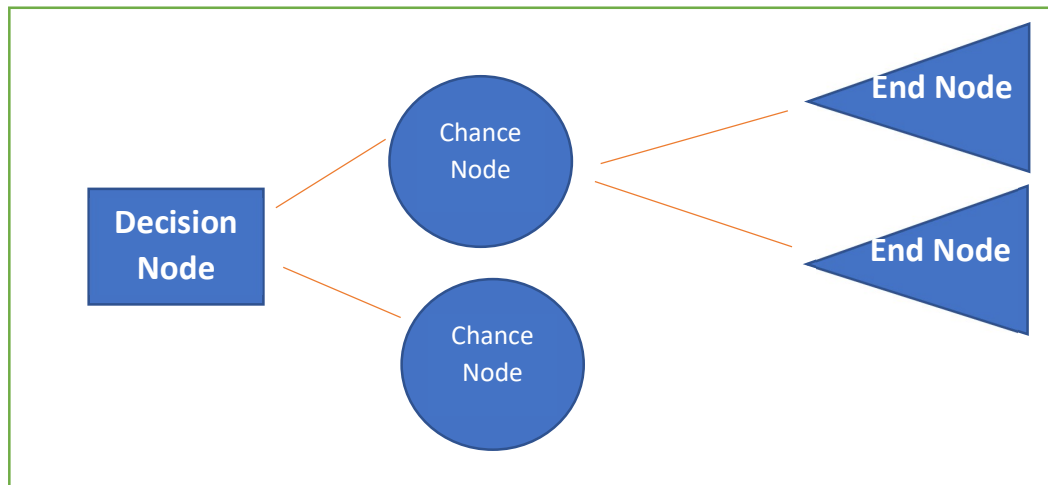
## 20. How are the different nodes of decision trees represented?

A decision tree is consisting of three types of node named as chance node, decision node and end node.

A chance node is represented by circle shape refers to some uncertain results.

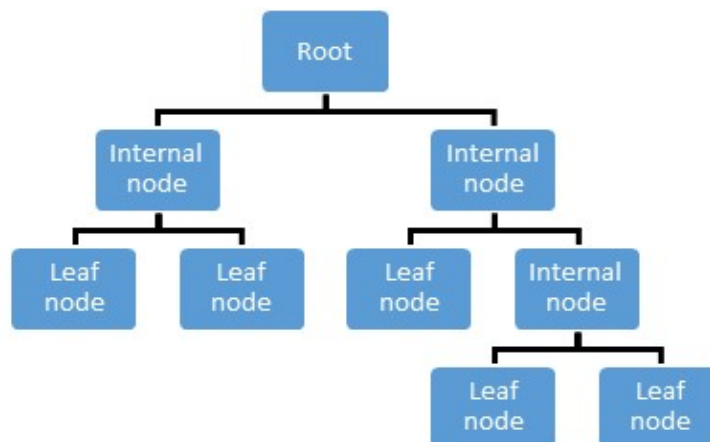
A decision node is represented by rectangle shape indicates the making of decision.

An end node is represented by triangle shows the final output.



## 21. What type of node is considered Pure?

The decision to split at each node is made according to a metric called purity. A node is 100% impure if it is evenly split 50/50, and 100% pure if all the data belongs to a single class. Optimizing the model requires achieving maximum purity and avoiding contamination.



## 22. How would you deal with an Overfitted Decision Tree?

Overfitting is when the model fits the training data perfectly but fails to generalize to the invisible test data. An overfitting condition occurs when the model remembers the noise in the training data and fails to capture important patterns. Perfectly fitted decision trees are good for training data, but not for invisible test data. Overfitted decision tree is avoided by using pre-pruning and post-pruning.

## 23. What are some disadvantages of using Decision Trees and how would you solve them?

**Volatile nature** is one of the limitations of decision trees is that they are very volatile compared to other decision predictors. Small changes in the data can lead to large changes in the structure of the decision tree, which can lead to different results than what the user would get in normal events.

**Decision trees are not very effective at predicting outcomes for continuous variables**, nor are they very effective at making predictions when the main goal is to predict outcomes for continuous variables. This is because decision trees tend to lose information when variables fall into multiple categories.

## 24. What is Gini Index and how is it used in Decision Trees?

The Gini impurity can be viewed as an alternative to the entropy method. Gini impurity is a measure of how often randomly selected items from a set are mislabeled when randomly labeled according to the distribution of labels within the subset.

Mathematically, it can be represented as:

$$Gini = 1 - \sum_{i=1}^k p_i^2$$

Where,  $p_i$  = proportion of a label

The entropy calculation for the Gini method works similarly, but the entropy involves a logarithmic calculation and the Gini impurity involves a quadratic calculation. Prefer the Gini impurity over the entropy because computing the square is cheaper than the logarithmic function.

## 25. How would you define the Stopping Criteria for decision trees?

Usually, if we continue to grow the tree completely until each leaf node corresponds to the lowest impurity, the data will be overfitted. If you stop splitting too early, the error in the training data will not be large enough will cause poor performance. Therefore, avoiding overfitting and underfitting is very important when modeling decision trees, and the following he can be done in two ways:

- a) Setting tree size Constraints
- b) Tree pruning
  - Pre- pruning
  - Post- pruning

## 26. What is Entropy?

Entropy is defined as a measure of the randomness or disorder of information processed in machine learning. In other words, entropy is a machine learning metric that measures the unpredictability or impurity of a system.

Entropy is a measure of the disorder or impurity of information processed in machine learning. The decision tree defines how to choose data partitions. A simple example like tossing a coin can help us understand the term entropy. When you toss a coin, there are two possible outcomes. However, it is difficult to deduce what the exact outcome of a coin toss is because there is no direct relationship between a coin toss and its outcome. The probability of both outcomes is 50%. Entropy would be high in such a scenario.

$p_i \log_2$

$$E(S) = \sum_{i=0}^c -p_i \log_2 p_i$$

## 27. How do we measure the Information?

Information is measured using bits. One bit is equivalent to choosing between two equally probable options. For example, if you know you need to toss a coin but can't see the coin fall, reporting whether the coin came from heads or tails will give you some information.

If there are multiple equally probable choices, the number of bits is equal to the base 2 logarithm of the number of choices. For example, a message is said to contain 4 bits of information if it contains one of the 16 equal probability options. Things get more complicated when the other options aren't equally likely.

## 28. What is the difference between Post-pruning and Pre-pruning?

### Post-pruning:

As the name suggests, pruning is the pruning of trees. Once a tree is created, it can be modified. The CART algorithm repeatedly partitions the data into smaller subsets until these final subsets are homogeneous in terms of the outcome variable. In practice, this often means that each finite subset consists of only one or a few data points. The tree learned the data correctly, but it cannot predict well new data points with little difference.

### Pre-pruning:

Another way to avoid overfitting is to stop the tree-building process early to produce leaves with very small samples. This heuristic is known as early stopping, but is also sometimes referred to as pre-pruning decision trees. Check for cross-validation errors at each step of tree splitting. Stop when the error is not sufficiently reduced. If you stop too early, stopping early may not be enough. The current split may offer little benefit, but this will reduce errors more significantly in subsequent splits. Early pinching and pruning can be used together, separately, or not used at all. Decision trees after pruning are mathematically more rigorous because tree search is at least as good as stopping early. Pre-pruning is a quick fix technique. When used in conjunction with pruning, pruning early can save time.

## 29. Compare Linear Regression and Decision Trees.

### Linear Regression:

Categorical variables cannot be accepted as input variables. The new field should be displayed.

Linear regression will ignore observations with missing values for numeric explanatory variables.

Linear regression is strongly affected by outliers in numeric explanatory variables.

### Decision Tree:

Categorical variables can be accepted as input variables.

It can accept missing items as classes, and you can easily work with the data to develop decision trees.

It is not affected by outliers on the numerical independent variable. Make the class a numeric variable.

## 30. What is the relationship between Information Gain and Information Gain Ratio?

Information gain is the entropy reduction resulting from splitting the set of attributes and finding the best candidate with the maximum value.

Information gain ratio is the ratio of information gain to specific information.

The relationship between Information gain and information gain ratio is:

$$\text{Information Gain Ratio} = \frac{\text{Information Gain}}{\text{Intrinsic Information}}$$

## 31. Compare Decision Trees and k-Nearest Neighbours.

Based on the tests performed, the Decision Tree algorithm produces an average accuracy score of 70.53% with an average processing time of 0.083 seconds. The K-Nearest Neighbor algorithm produces an average accuracy score of 66.36% with an average processing time of 0.03 seconds.

Decision trees can be faster, but ANNs tend to be slower for large datasets because they do not pre-generalize the data, so they scan the entire dataset to make predictions.

Decision trees support automatic interaction of features, but KNNs do not.

## 32. While building Decision Tree how do you choose which attribute to split at each node?

- i. For each split, compute the variance for each child node separately.
- ii. Compute the variance of each split as the weighted average variance of the child nodes.
- iii. Choose the split with the smallest variance.
- iv. Repeat steps (i) to (iii) until you have perfectly uniform nodes.

## 33. How would you compare different Algorithms to build Decision Trees?

## 34. How do you Gradient Boost decision tree?

Gradient boost decision tree is a popular technique which is used in classification and regression for prediction problems solving. Gradient boosting works by repeatedly

developing smaller prediction models, with each model attempting to forecast the error left behind by the preceding model. As a result, the algorithm tends to overfit quite quickly.

### **Algorithm:**

Step-1: Determine the target label's average:

$$\frac{\text{Sum of all target labels values}}{\text{Total number of target label values}}$$

Step-2: Determine the residuals:

$$\text{Residual} = \text{Actual value} - \text{Predicted value}$$

Step-3: Build a decision tree:

we construct a tree with the purpose of predict the residuals, each leaf will carry a Prediction of the residual value.

Step-4: Estimate the targeted label based on all of the trees:

$$\text{Predicted Value} = \text{Average Value} + \text{Learning Rate} \times \text{Residual Predicted value by tree}$$

Step-5: Determine new residuals:

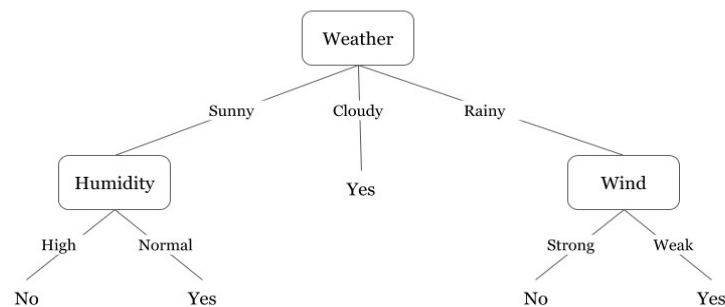
$$\text{Residual} = \text{Actual value} - \text{Predicted value}$$

Step-6: Iterate Step-3 to step-5, until we get the number of iteration equal to the provided number by hyperparameter.

## **35. What are the differences between Decision Trees and Neural Networks?**

### **Decision Tree:**

Decision trees make it easy to follow the natural flow. It is also easy to program for computer systems using IF, THEN, and ELSE statements. You can see that the top node in the tree is the most influential piece of data affecting the model's response variable. These trees are very useful as a modeling technique because they are very easy to understand and provide a visual representation of the data.





## **Neural Network:**

Neural networks are algorithms inspired by biological neural networks. It consists of a connected graph of processor units mimicking neurons. Connections between neurons are so-called weights. Their values are selected in the learning process. The goal of training is to minimize the error between MLP predicted values and actual values. Neural networks are not easy to understand visually. It is very difficult to create computer systems from them, and almost impossible to create descriptions from models. Neural networks can handle binary data better than decision trees, but not categorical values.

