**WEEK 2**

**AIM:**

Perform various different machine learning algorithms like regression, decision tree, random forests, etc and differentiate between the models and analyse their performances on a specific dataset.

**DESCRIPTION:**

**Linear regression**: Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.

Mathematically, we can represent a linear regression as:

y= a0+a1x+ ε

**Here,**

Y= Dependent Variable (Target Variable)  
X= Independent Variable (predictor Variable)  
a0= intercept of the line (Gives an additional degree of freedom)  
a1 = Linear regression coefficient (scale factor to each input value).  
ε = random error

**Decision Tree:** A decision tree is a flowchart-like tree structure where an internal node represents feature or attribute), the branch represents a decision rule, and each leaf node represents the outcome. It learns to partition on the basis of the attribute value. It partitions the tree in recursively manner call recursive partitioning. This flowchart-like structure helps in decision making.

**The basic idea behind any decision tree algorithm is as follows:**

Select the best attribute using Attribute Selection Measures (ASM) to split the records.

Make that attribute a decision node and breaks the dataset into smaller subsets.

Starts tree building by repeating this process recursively for each child until one of the conditions will match:

All the tuples belong to the same attribute value.

There are no more remaining attributes.

There are no more instances.

**Random Forest:**

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of **ensemble learning*,****which is a process of*combining multiple classifiers to solve a complex problem and to improve the performance of the model.

The Working process can be explained in the below steps and diagram:

**Step-1:** Select random K data points from the training set.

**Step-2:** Build the decision trees associated with the selected data points (Subsets).

**Step-3:** Choose the number N for decision trees that you want to build.

**Step-4:** Repeat Step 1 & 2.

**Step-5:** For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

**CODE:**

**import numpy as np**

**import pandas as pd**

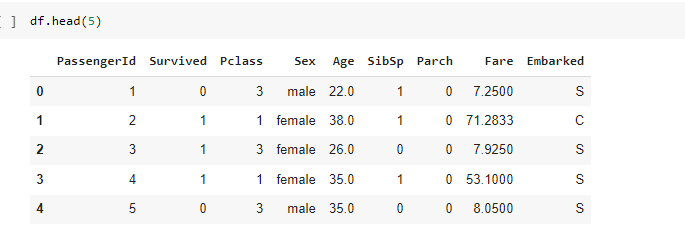
**import seaborn as sns**

**import matplotlib.pyplot as plt**

**df = pd.read\_csv("/content/train.csv")**

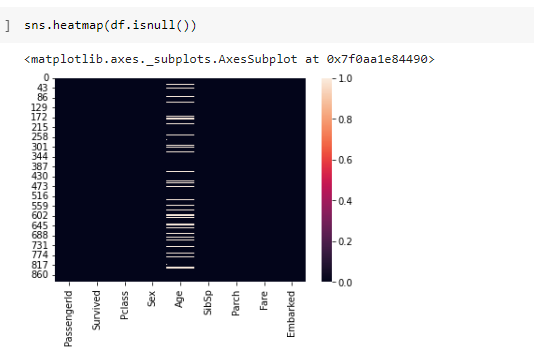
**df = df.drop(['Name','Ticket','Cabin'], axis=1)**

**df.head(5)**

****

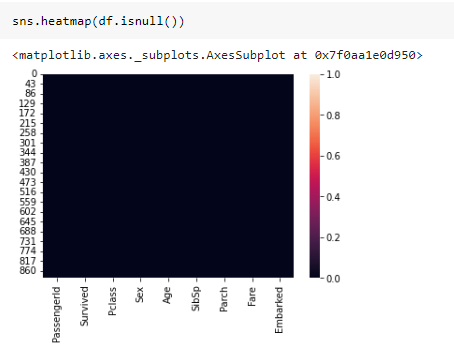
**### Checking for Null Values**

**sns.heatmap(df.isnull())**

****

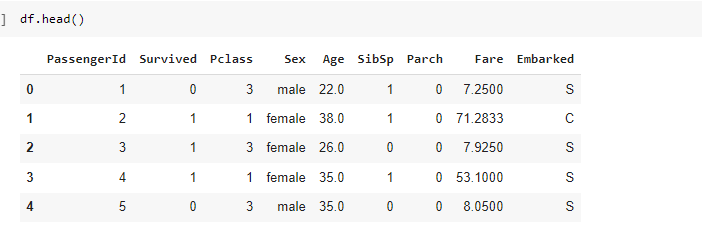
**df['Age'] = df['Age'].interpolate()**

**sns.heatmap(df.isnull())**

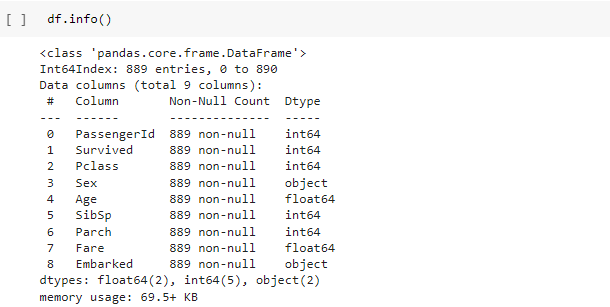
****

**df = df.dropna()**

**df.head()**

****

**df.info()**

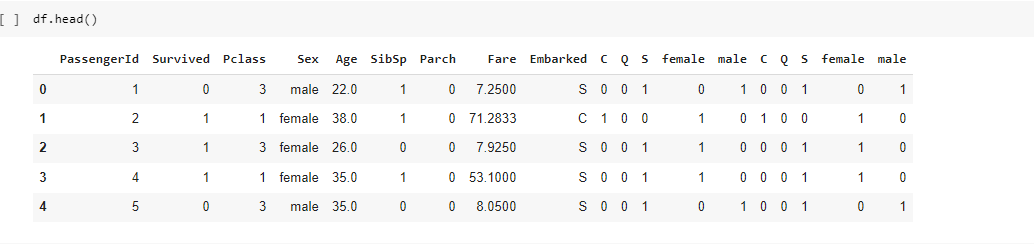
****

**Embarked = pd.get\_dummies(df['Embarked'])**

**Sex = pd.get\_dummies(df['Sex'])**

**df = pd.concat((df,Embarked,Sex), axis=1)**

**df.head()**

****

**df = df.drop(['Sex','Embarked'],axis=1)**

**x = df.values**

**y = df['Survived'].values**

**x = np.delete(x,1,axis=1)**

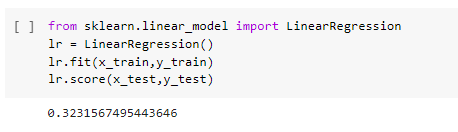
**### Linear Regression**

**from sklearn.linear\_model import LinearRegression**

**lr = LinearRegression()**

**lr.fit(x\_train,y\_train)**

**lr.score(x\_test,y\_test)**

****

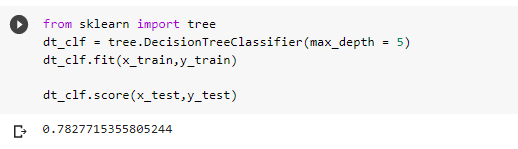
**### Decision Tree Classifier**

**from sklearn import tree**

**dt\_clf = tree.DecisionTreeClassifier(max\_depth = 5)**

**dt\_clf.fit(x\_train,y\_train)**

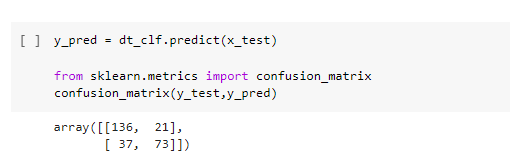
**dt\_clf.score(x\_test,y\_test)**

****

**y\_pred = dt\_clf.predict(x\_test)**

**from sklearn.metrics import confusion\_matrix**

**confusion\_matrix(y\_test,y\_pred)**

****

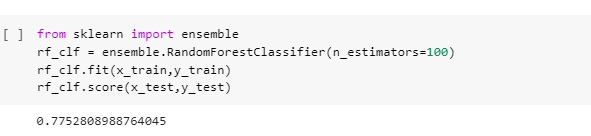
**### Random Forest Classifier**

**from sklearn import ensemble**

**rf\_clf = ensemble.RandomForestClassifier(n\_estimators=100)**

**rf\_clf.fit(x\_train,y\_train)**

**rf\_clf.score(x\_test,y\_test)**

****