**Report: -**

**Github link: -** <https://github.com/chaitanya9780/ML_Project_Classification_Techniques>

The project is about the analyze the difficulty level of a particular course using classification techniques. So I have used the classification techniques like SVM, KNeighbours, Classifier, Random forest, Decision tree, Gaussian etc. With the help of these algorithms I calculated the difficulty of the course Bachelor of Technology- Master of Technology in Biotechnology (Integrated)

The project contains 4 steps: -

1.Loading the dataset

2.Data Preprocessing

3.Algorithm Implementation

4.Calculating Accuracy

1.Loading The Dataset: - First we import numpy, panadas, matplotlib etc . We load the data in csv format with the help of Pandas

import numpy as np

import pandas as pd

from matplotlib import pyplot as plt

import seaborn as sea

data=pd.read\_csv("C:\\Users\\ADMIN\\Desktop\\ML PROJECT\\project\_data.csv")

print(data)

**2.Data Processing: -**

In this we process the data. This include the handle of various null values with the help of the strategy most frequent. This process includes the removal of the unimportant columns. We also map the data according to the grade. We also plot the data according to our use with the help of the matplotlib.

print('Rows :'+str(data.shape[0]))

print('Columns:'+str(data.shape[1]))

data=data.drop(columns=['Termid','CoureType','Regd No','MHRDName','CA\_1','CA\_2','CA\_3','CA\_4','ProgramType','Direction','ScholarType','Height','Weight','Medium'])

print(data.shape)

print(data.describe())

map1={'O':10,'A+':9,'A':8,'B+':7,'B':6,'C':5,'D':4,'E':3,'F':2}

data['Grade']=data['Grade'].map(map1)

print(data['Grade'])

from sklearn.preprocessing import LabelEncoder

le=LabelEncoder()

data['Course']=le.fit\_transform(data['Course'].values)

data['Gender']=le.fit\_transform(data['Gender'].values)

print(data.isnull().sum())

print(data.info())

print(data.columns)

from sklearn.preprocessing import Imputer

im=Imputer(missing\_values=np.nan,strategy='most\_frequent',axis=0)

data['MTT\_50']=im.fit\_transform(data[['MTT\_50']].values)

data['ETT\_100']=im.fit\_transform(data[['ETT\_100']].values)

data['ETP\_100']=im.fit\_transform(data[['ETP\_100']].values)

data['Course\_Att']=im.fit\_transform(data[['Course\_Att']].values)

print(data.isnull().sum())

print(data.head())

print(data)

data['Gender'].value\_counts().plot(kind='barh')

plt.show()

data['Grade'].value\_counts().plot(kind='barh')

plt.show()

sea.countplot(data.Grade,hue=data['Gender'])

plt.show()

data['MTT\_50'].value\_counts().plot(kind='hist')

plt.show()

data['ETT\_100'].value\_counts().plot(kind='hist')

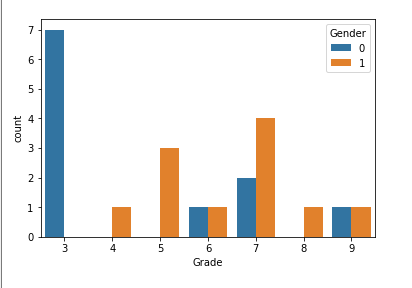
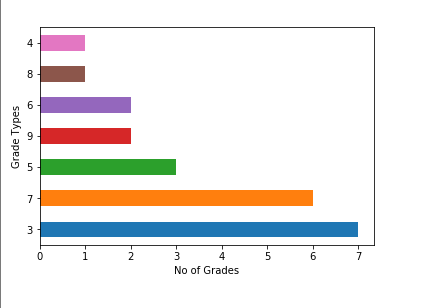
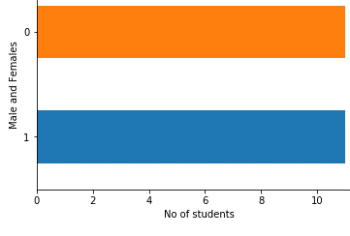
plt.show()

data['ETP\_100'].value\_counts().plot(kind='hist')

plt.show()

data['CA\_100'].value\_counts().plot(kind='hist')

plt.show()



**3.Algorithm Implementation: -**

First, we divide the data into training and testing. Then we will implement all the four algorithms and find the accuracy of each and every algorithm.

from sklearn.svm import SVC

svc=SVC(C=10,kernel='linear')

print(svc)

svc.fit(x\_train,y\_train)

y\_pred=svc.predict(x\_test)

from sklearn.metrics import accuracy\_score

print('Accuracy Score :')

print(accuracy\_score(y\_pred,y\_test))

from sklearn.neighbors import KNeighborsClassifier

knn=KNeighborsClassifier(n\_neighbors=3)

knn.fit(x\_train,y\_train)

y\_pred=knn.predict(x\_test)

print("Accuracy score :")

print(accuracy\_score(y\_test,y\_pred))

from sklearn.naive\_bayes import GaussianNB

gn=GaussianNB()

gn.fit(x\_train,y\_train)

y\_pred=gn.predict(x\_test)

print("Accuracy Score :")

print(accuracy\_score(y\_pred,y\_test))

from sklearn.tree import DecisionTreeClassifier

dt=DecisionTreeClassifier(criterion='gini')

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

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

print("Accuracy Score :")

print(accuracy\_score(y\_pred,y\_test))

# comparing actual response values (y\_test) with predicted response values (y\_pred)

from sklearn.ensemble import RandomForestClassifier

rc=RandomForestClassifier(n\_estimators=100,criterion='gini',max\_features=6)

print(rc)

rc.fit(x\_train,y\_train)

y\_pred=rc.predict(x\_test)

print("Accuracy Score :")

print(accuracy\_score(y\_test,y\_pred))

**4. Accuracy**

In this after calculating the accuracy we the help of the values obtained and then we will divide the data into easy, medium, difficulty. With the help of value obtained we calculate the difficulty of the course

print("Calculating the statistics of dataset provided in order to define the difficulty level of a particular course ..")

max\_grade = np.max(target)

min\_grade = np.min(target)

mean\_grade= np.mean(target)

print("Maximum grade: {}".format(max\_grade))

print("Minimum grade: {}".format(min\_grade))

print("Mean grade: {}".format(mean\_grade))

difference=(mean\_grade)\*10

print("Difficulty level of this course (B-Tech Master of Technology in BioTechnology(Integrated)) whose accuracy value is {} is :".format(difference))

difficulty\_rate=['Easy','Medium','Hard']

if difference<50:

difference=difficulty\_rate[0]

elif difference>50 and difference<75:

difference=difficulty\_rate[1]

else :

difference=difficulty\_rate[2]

print(difference)