|  |  |
| --- | --- |
| **Name** | V.Karuna Sree |
| **Project Title** | Skill Recommender Application 30Gp |

**Assignment 2**

1. Create a Flask App

2. Add the Home page, About Page

3. Add the Bootstrap

4. Add the Sign in page and App the Signup Page + database connectivity

**Program**

from google.colab import drive

drive.mount('/content/drive')

import pandas as pd

import numpy as np

import sklearn as sk

import seaborn as sns

data=pd.read\_csv("/content/Churn\_Modelling Dataset2.csv")

df=data.head(10)

Univariate Analysis

import matplotlib.pyplot as plt

plt.bar (df['Age'],4)

Bivariate Analysis

plt.scatter(df['Age'],df['CreditScore'])

Multivariate analysis

plt.scatter(df['Age'],df['CreditScore'],df['Tenure'])

Descriptive statistics on the dataset

data.describe()

Handling Missing values

data.isnull().sum()

Find the outliers and replace the outliers

Finding Outliners

sns.boxplot(data['Age'])

Replacing outliners

q=data.quantile(q=[0.75,0.5])

iqr=q.iloc[0]-q.iloc[1]

iqr

l=q.iloc[1]-(1.5\*iqr)

l['Age']

u=q.iloc[1]+(1.5\*iqr)

u['Age']

data['Age']=np.where(data['Age']>u['Age'],u['Age'],np.where(data['Age']<l['Age'],l['Age'],data['Age']))

sns.boxplot(data['Age'])

Check for Categorical columns and perform encoding.

df.info()

from sklearn.preprocessing import LabelEncoder

from collections import Counter as count

le=LabelEncoder()

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

data

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

data['Gender']=data['Gender'].replace(['Male','Female'],[0,1])

data

Split the data into dependent and independent variables.

Independent Variables

x=data.iloc[:,0:13]

x

Dependent Variables

y=data['Exited']

y

Scale the independent variables

from sklearn.preprocessing import scale

scale(x)

Split the data into training and testing

from sklearn.model\_selection import train\_test\_split

x\_train,x\_test,y\_train,y\_test=train\_test\_split(x,y,test\_size=0.2)

x\_train

x\_train.shape

y\_train

y\_train.shape

x\_test

x\_test.shape

y\_test

y\_test.shape