Name : Sujitha

College : Aditya College of Engineering

Course : B-tech

Branch : CSE

Year : III

DRIVE : https://drive.google.com/drive/folders/1MmDzKms_f6J1gRPdma-6oqHmt9LAvgxT

Github :github.com/Sujitha749/Projects

PROJECT 1: LOGISTIC REGRESSION

```
#  https://www.kaggle.com/datasets/benroshan/factors-affecting-campus-placement
#1.Create dataframe
#importing the pandas
import pandas as pd
df1 = pd.read_csv('/content/archive (5).zip')
df1
```

	sl_no	gender	ssc_p	ssc_b	hsc_p	hsc_b	hsc_s	degree_p	degree_t	workex	etest_p	specialisation	mba_p	status	salary
0	1	M	67.00	Others	91.00	Others	Commerce	58.00	Sci&Tech	No	55.0	Mkt&HR	58.80	Placed	270000.0
1	2	M	79.33	Central	78.33	Others	Science	77.48	Sci&Tech	Yes	86.5	Mkt&Fin	66.28	Placed	200000.0
2	3	M	65.00	Central	68.00	Central	Arts	64.00	Comm&Mgmt	No	75.0	Mkt&Fin	57.80	Placed	250000.0
3	4	M	56.00	Central	52.00	Central	Science	52.00	Sci&Tech	No	66.0	Mkt&HR	59.43	Not Placed	NaN
4	5	M	85.80	Central	73.60	Central	Commerce	73.30	Comm&Mgmt	No	96.8	Mkt&Fin	55.50	Placed	425000.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
210	211	M	80.60	Others	82.00	Others	Commerce	77.60	Comm&Mgmt	No	91.0	Mkt&Fin	74.49	Placed	400000.0
211	212	M	58.00	Others	60.00	Others	Science	72.00	Sci&Tech	No	74.0	Mkt&Fin	53.62	Placed	275000.0
212	213	M	67.00	Others	67.00	Others	Commerce	73.00	Comm&Mgmt	Yes	59.0	Mkt&Fin	69.72	Placed	295000.0
213	214	F	74.00	Others	66.00	Others	Commerce	58.00	Comm&Mgmt	No	70.0	Mkt&HR	60.23	Placed	204000.0
214	215	M	62.00	Central	58.00	Others	Science	53.00	Comm&Mgmt	No	89.0	Mkt&HR	60.22	Not Placed	NaN

215 rows × 15 columns

```
#Reading th information from the dataframe
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 215 entries, 0 to 214
Data columns (total 15 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   sl_no           215 non-null    int64
 1   gender          215 non-null    object
 2   ssc_p           215 non-null    float64
 3   ssc_b           215 non-null    object
 4   hsc_p           215 non-null    float64
 5   hsc_b           215 non-null    object
```

```
 6  hsc_s          215 non-null    object
 7  degree_p       215 non-null    float64
 8  degree_t       215 non-null    object
 9  workex         215 non-null    object
 10 etest_p        215 non-null    float64
 11 specialisation 215 non-null    object
 12 mba_p          215 non-null    float64
 13 status         215 non-null    object
 14 salary         148 non-null    float64
dtypes: float64(6), int64(1), object(8)
memory usage: 25.3+ KB
```

```
#Shape of dataframe
df1.shape
```

```
(215, 15)
```

```
df1.size
3225
```

```
#Exploratary Data Analysis-EDA
#isnull is used to find the null values
df1.isnull().sum()
```

```
sl_no 0

gender 0

ssc_p 0

ssc_b 0

hsc_p 0

hsc_b 0

hsc_s 0

degree_p 0

degree_t 0

workex 0

etest_p 0

specialisation 0

mba_p 0 s

tatus 0

salary 67 dtype: int64
```

```
df1=df1.fillna(value=0)
df1
```

	sl_no	gender	ssc_p	ssc_b	hsc_p	hsc_b	hsc_s	degree_p	degree_t	workex	etest_p	specialisation	mba_p	status	salary
0	1	M	67.00	Others	91.00	Others	Commerce	58.00	Sci&Tech	No	55.0	Mkt&HR	58.80	Placed	270000.0
1	2	M	79.33	Central	78.33	Others	Science	77.48	Sci&Tech	Yes	86.5	Mkt&Fin	66.28	Placed	200000.0
2	3	M	65.00	Central	68.00	Central	Arts	64.00	Comm&Mgmt	No	75.0	Mkt&Fin	57.80	Placed	250000.0
3	4	M	56.00	Central	52.00	Central	Science	52.00	Sci&Tech	No	66.0	Mkt&HR	59.43	Not Placed	0.0
4	5	M	85.80	Central	73.60	Central	Commerce	73.30	Comm&Mgmt	No	96.8	Mkt&Fin	55.50	Placed	425000.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
210	211	M	80.60	Others	82.00	Others	Commerce	77.60	Comm&Mgmt	No	91.0	Mkt&Fin	74.49	Placed	400000.0
211	212	M	58.00	Others	60.00	Others	Science	72.00	Sci&Tech	No	74.0	Mkt&Fin	53.62	Placed	275000.0
212	213	M	67.00	Others	67.00	Others	Commerce	73.00	Comm&Mgmt	Yes	59.0	Mkt&Fin	69.72	Placed	295000.0
213	214	F	74.00	Others	66.00	Others	Commerce	58.00	Comm&Mgmt	No	70.0	Mkt&HR	60.23	Placed	204000.0
214	215	M	62.00	Central	58.00	Others	Science	53.00	Comm&Mgmt	No	89.0	Mkt&HR	60.22	Not Placed	0.0

215 rows × 15 columns

```
df1.isnull().sum()
```

```
sl_no 0
gender 0
ssc_p 0
ssc_b 0
hsc_p 0
hsc_b 0
```

```
hsc_s 0

degree_p 0

degree_t 0

workex 0

etest_p 0 s

pecialisation 0

mba_p 0

status 0

salary 0

dtype: int64
```

```
#As there is no use of sl.no we have to drop the column
df1 = df1.drop(columns = 'sl_no')
df1
```

	gender	ssc_p	ssc_b	hsc_p	hsc_b	hsc_s	degree_p	degree_t	workex	etest_p	specialisation	mba_p	status	salary
0	M	67.00	Others	91.00	Others	Commerce	58.00	Sci&Tech	No	55.0	Mkt&HR	58.80	Placed	270000.0
1	M	79.33	Central	78.33	Others	Science	77.48	Sci&Tech	Yes	86.5	Mkt&Fin	66.28	Placed	200000.0
2	M	65.00	Central	68.00	Central	Arts	64.00	Comm&Mgmt	No	75.0	Mkt&Fin	57.80	Placed	250000.0
3	M	56.00	Central	52.00	Central	Science	52.00	Sci&Tech	No	66.0	Mkt&HR	59.43	Not Placed	0.0
4	M	85.80	Central	73.60	Central	Commerce	73.30	Comm&Mgmt	No	96.8	Mkt&Fin	55.50	Placed	425000.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
210	M	80.60	Others	82.00	Others	Commerce	77.60	Comm&Mgmt	No	91.0	Mkt&Fin	74.49	Placed	400000.0
211	M	58.00	Others	60.00	Others	Science	72.00	Sci&Tech	No	74.0	Mkt&Fin	53.62	Placed	275000.0
212	M	67.00	Others	67.00	Others	Commerce	73.00	Comm&Mgmt	Yes	59.0	Mkt&Fin	69.72	Placed	295000.0
213	F	74.00	Others	66.00	Others	Commerce	58.00	Comm&Mgmt	No	70.0	Mkt&HR	60.23	Placed	204000.0
214	M	62.00	Central	58.00	Others	Science	53.00	Comm&Mgmt	No	89.0	Mkt&HR	60.22	Not Placed	0.0

215 rows × 14 columns

```
#Replacing the placed with 1
df1['status'] = df1['status'].str.replace('Placed','1')
df1['status']
```

```
0    1
1    1
```

```
2       1
3     Not 1
4       1
    ...
210     1
211     1
212     1
213     1
214   Not 1
Name: status, Length: 215, dtype: object
```

```
#Replace the Not placed with 0
df1['status'] = df1['status'].str.replace('Not 1','0')
df1['status']
```

```
0    1
1    1
2    1
3    0
4    1
   ..
210  1
211  1
212  1
213  1
214  0
Name: status, Length: 215, dtype: object
```

```
#Placed-1
#Not placed-0
#Converting the object datatype of status to int64
df1['status'] = df1['status'].astype('int64')
df1.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 215 entries, 0 to 214
Data columns (total 14 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   gender       215 non-null    object
 1   ssc_p        215 non-null    float64
 2   ssc_b        215 non-null    object
 3   hsc_p        215 non-null    float64
 4   hsc_b        215 non-null    object
 5   hsc_s        215 non-null    object
 6   degree_p     215 non-null    float64
 7   degree_t     215 non-null    object
 8   workex       215 non-null    object
 9   etest_p      215 non-null    float64
```

```
 10  specialisation  215 non-null    object
 11  mba_p           215 non-null    float64
 12  status          215 non-null    int64
 13  salary          215 non-null    float64
dtypes: float64(6), int64(1), object(7)
memory usage: 23.6+ KB
```

CodeText

```
#Extracting only the int and float datatypes from the dataframe
df = df1.select_dtypes(include = ['float64','int64'])
df
```

	ssc_p	hsc_p	degree_p	etest_p	mba_p	status	salary
0	67.00	91.00	58.00	55.0	58.80	1	270000.0
1	79.33	78.33	77.48	86.5	66.28	1	200000.0
2	65.00	68.00	64.00	75.0	57.80	1	250000.0
3	56.00	52.00	52.00	66.0	59.43	0	0.0
4	85.80	73.60	73.30	96.8	55.50	1	425000.0
...	...	...	...	...	...	...	...
210	80.60	82.00	77.60	91.0	74.49	1	400000.0
211	58.00	60.00	72.00	74.0	53.62	1	275000.0
212	67.00	67.00	73.00	59.0	69.72	1	295000.0
213	74.00	66.00	58.00	70.0	60.23	1	204000.0
214	62.00	58.00	53.00	89.0	60.22	0	0.0

215 rows × 7 columns

```
#Number of Students are placed and number of students are not placed
df.groupby(df['status']).size()
```

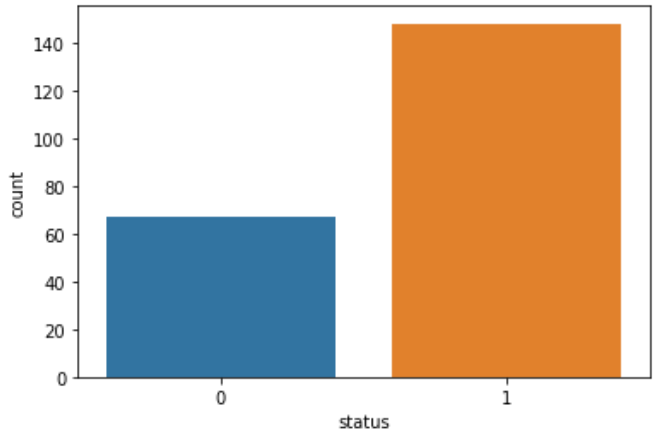
```
status
0    67
1   148
dtype: int64
```

```
#3. Data Visualization

#importing the seaborn library
```

```
import seaborn as sns
sns.countplot(x = 'status',data = df)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fa6b23c7b10>
```



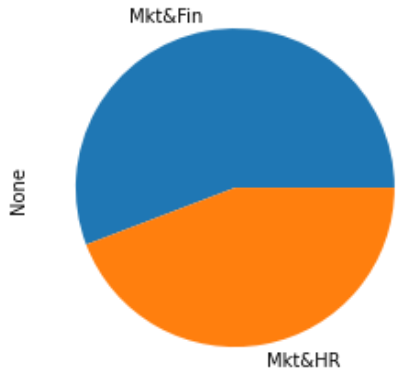

```
#To know no.of students are specilized in marketing & finance and ma
rkrting & HR
df1.groupby(df1['specialisation']).size()
```

specialisation
Mkt&Fin 120
Mkt&HR 95
dtype: int64

[ ]

```
df1.groupby(df1['specialisation']).size().plot(kind='pie')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fa6b1dbd690>
```




```
# To get the all the information of the employee who got the maximum
 salary
df1.loc[df1['salary']==df1['salary'].max()]
```

	gender	ssc_p	ssc_b	hsc_p	hsc_b	hsc_s	degree_p	degree_t	workex	etest_p	specialisation	mba_p	status	salary
119	M	60.8	Central	68.4	Central	Commerce	64.6	Comm&Mgmt	Yes	82.66	Mkt&Fin	64.34	1	940000.0

```
#4.Divide the data into input and output
#Output - person is placed or  not
```

```
x = df.iloc[:,0:5].values
x
```

```
array([[67. , 91. , 58. , 55. , 58.8 ],
       [79.33, 78.33, 77.48, 86.5 , 66.28],
       [65. , 68. , 64. , 75. , 57.8 ],
       ...,
       [67. , 67. , 73. , 59. , 69.72],
       [74. , 66. , 58. , 70. , 60.23],
       [62. , 58. , 53. , 89. , 60.22]])
```

[ ]

```
y = df.iloc[:,5].values
y
```

```
array([1, 1, 1, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 1,
       1, 1, 1, 0, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 1,
       1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0,
       1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0,
       1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 1, 0,
       1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1,
       1, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1,
       1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 1, 1, 0, 1, 0,
       1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1,
       0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 1, 1, 0])
```

[ ]

```
#5.Train and Test variables - train_test_split()
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(x,y,random_state =
0)

# Input data is divided into the training and testing data
print(x.shape)
print(x_train.shape)
print(x_test.shape)
```

```
(215, 5)
(161, 5)
(54, 5)
```

[ ]

```
#output y is divided into the y train and y_test for train model and
 test model
print(y.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(215,)
(161,)
(54,)
```

[ ]

```
#6.There is no need of scaling as they are already scaled
#7.apply Classifier,regressor or clusterer
from sklearn.linear_model import LogisticRegression
model= LogisticRegression()
#8.Fitting the model
model.fit(x_train,y_train)
```

```
LogisticRegression()
```

```
# 9.Predict the output
y_pred = model.predict(x_test)
y_pred
```

```
array([0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1, 0, 1,
       1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0, 1, 1, 1,
       1, 0, 1, 1, 1, 1, 1, 1, 1, 1])
```

[ ]

```
#y_test outputs
y_test
```

```
rray([0, 1, 1, 0, 1, 0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0,
       1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1,
       1, 1, 1, 1, 0, 1, 1, 1, 1, 1])
```

```
#10.Accuracy
from sklearn.metrics import accuracy_score
accuracy_score(y_pred,y_test)*100
```

```
77.77777777777779
```

```
#Here we got the accuracy of the model as 77.777777779%
#INDIVIDUAL PREDICTION
model.predict([[78.09,89.00,65.00,56.1,57.0]])
```

```
array([1])
```

```
#Hence he might be placed

model.predict([[65.09,56.44,55.33,65.22,50.90]])
```

```
array([1])
```

```
#Predicting by giving the values
model.predict([[50.46,54.21,49.00,60.11,52.44]])
```

```
array([0])
```

PROJECT 2: IMAGE PROCESSING

NOW OUR GOAL IS DETECTING THE PERSON'S IS SMILE.

SOURCE:DATA IS COLLECTED FROM THE GITHUB HARCASCADE CLASSIFIER

https://raw.githubusercontent.com/opencv/opencv/master/data/haarcascades/haarcascade_smile.xml

```
# Importing  the opencv library

import cv2

#Reading the data from the image

img= cv2.imread("im1.jpg")

smile_cascade = cv2.CascadeClassifier('haarcascade_smile.xml')

smile = smile_cascade.detectMultiScale(img,1.1,19)


for (x,y,w,h) in smile:

        cv2.rectangle(img, (x, y), (x + h, y + w), (200, 0, 0), 5)

        cv2.imshow('Output', img)


cv2.waitKey(0)

cv2.destroyAllWindows()
```