```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df=pd.read_csv('/content/Job_Placement_Data.csv')
df
```

	gender	ssc_percentage	ssc_board	hsc_percentage	hsc_board	hsc_subject	degree_percentage	undergrad_degree
0	M	67.00	Others	91.00	Others	Commerce	58.00	Sci&Tech
1	M	79.33	Central	78.33	Others	Science	77.48	Sci&Tech
2	M	65.00	Central	68.00	Central	Arts	64.00	Comm&Mgmt
3	M	56.00	Central	52.00	Central	Science	52.00	Sci&Tech
4	M	85.80	Central	73.60	Central	Commerce	73.30	Comm&Mgmt
...	...	...	...	...	...	...	...	..
210	M	80.60	Others	82.00	Others	Commerce	77.60	Comm&Mgmt
211	M	58.00	Others	60.00	Others	Science	72.00	Sci&Tech
212	M	67.00	Others	67.00	Others	Commerce	73.00	Comm&Mgmt
213	F	74.00	Others	66.00	Others	Commerce	58.00	Comm&Mgmt
214	M	62.00	Central	58.00	Others	Science	53.00	Comm&Mgmt

215 rows × 13 columns

```
gender                 0
ssc_percentage         0
ssc_board              0
hsc_percentage         0
hsc_board              0
hsc_subject            0
degree_percentage      0
undergrad_degree       0
work_experience        0
emp_test_percentage    0
specialisation         0
mba_percent            0
status                 0
dtype: int64
```

```
print(df.columns)
```

```
Index(['gender', 'ssc_percentage', 'ssc_board', 'hsc_percentage', 'hsc_board',
       'hsc_subject', 'degree_percentage', 'undergrad_degree',
       'work_experience', 'emp_test_percentage', 'specialisation',
       'mba_percent', 'status'],
      dtype='object')
```

```
df.head()
```

	gender	ssc_percentage	ssc_board	hsc_percentage	hsc_board	hsc_subject	degree_percentage	undergrad_degree
0	M	67.00	Others	91.00	Others	Commerce	58.00	Sci&Tech
1	M	79.33	Central	78.33	Others	Science	77.48	Sci&Tech
2	M	65.00	Central	68.00	Central	Arts	64.00	Comm&Mgmt
3	M	56.00	Central	52.00	Central	Science	52.00	Sci&Tech
4	M	85.80	Central	73.60	Central	Commerce	73.30	Comm&Mgmt

```
df.tail()
```

	gender	ssc_percentage	ssc_board	hsc_percentage	hsc_board	hsc_subject	degree_percentage	undergrad_degree
210	M	80.6	Others	82.0	Others	Commerce	77.6	Comm&Mgmt
211	M	58.0	Others	60.0	Others	Science	72.0	Sci&Tech
212	M	67.0	Others	67.0	Others	Commerce	73.0	Comm&Mgmt
213	F	74.0	Others	66.0	Others	Commerce	58.0	Comm&Mgmt
214	M	62.0	Central	58.0	Others	Science	53.0	Comm&Mgmt

```
df.shape
```

```
(215, 13)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 215 entries, 0 to 214
Data columns (total 13 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   gender                215 non-null    object
 1   ssc_percentage        215 non-null    float64
 2   ssc_board             215 non-null    object
 2   ssc_board           215 non-null    object
 3   hsc_percentage      215 non-null    float64
 4   hsc_board           215 non-null    object
 5   hsc_subject         215 non-null    object
 6   degree_percentage   215 non-null    float64
 7   undergrad_degree    215 non-null    object
 8   work_experience     215 non-null    object
 9   emp_test_percentage 215 non-null    float64
 10  specialisation      215 non-null    object
 11  mba_percent         215 non-null    float64
 12  status              215 non-null    object
dtypes: float64(5), object(8)
memory usage: 22.0+ KB
```

```
df.describe()
```

	ssc_percentage	hsc_percentage	degree_percentage	emp_test_percentage	mba_percent
count	215.000000	215.000000	215.000000	215.000000	215.000000
mean	67.303395	66.333163	66.370186	72.100558	62.278186
std	10.827205	10.897509	7.358743	13.275956	5.833385
min	40.890000	37.000000	50.000000	50.000000	51.210000
25%	60.600000	60.900000	61.000000	60.000000	57.945000
50%	67.000000	65.000000	66.000000	71.000000	62.000000
75%	75.700000	73.000000	72.000000	83.500000	66.255000
max	89.400000	97.700000	91.000000	98.000000	77.890000

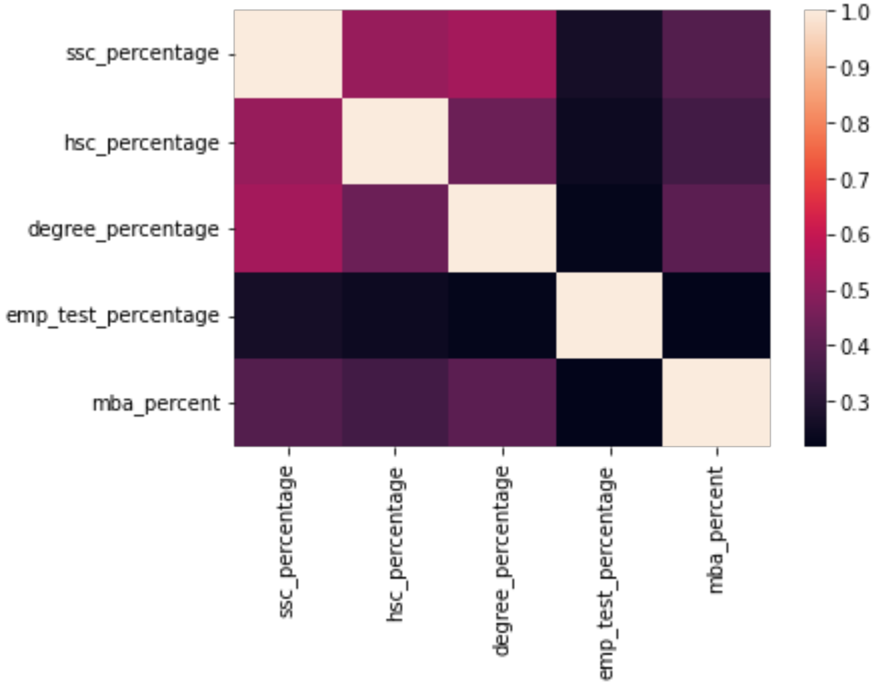
```
df.corr()
```

	ssc_percentage	hsc_percentage	degree_percentage	emp_test_percentage	mba_percent
ssc_percentage	1.000000	0.511472	0.538404	0.261993	0.388478

hsc_percentage	0.511472	1.000000	0.434206	0.245113	0.354823
degree_percentage	0.538404	0.434206	1.000000	0.224470	0.402364
emp_test_percentage	0.261993	0.245113	0.224470	1.000000	0.218055
mba_percent	0.388478	0.354823	0.402364	0.218055	1.000000

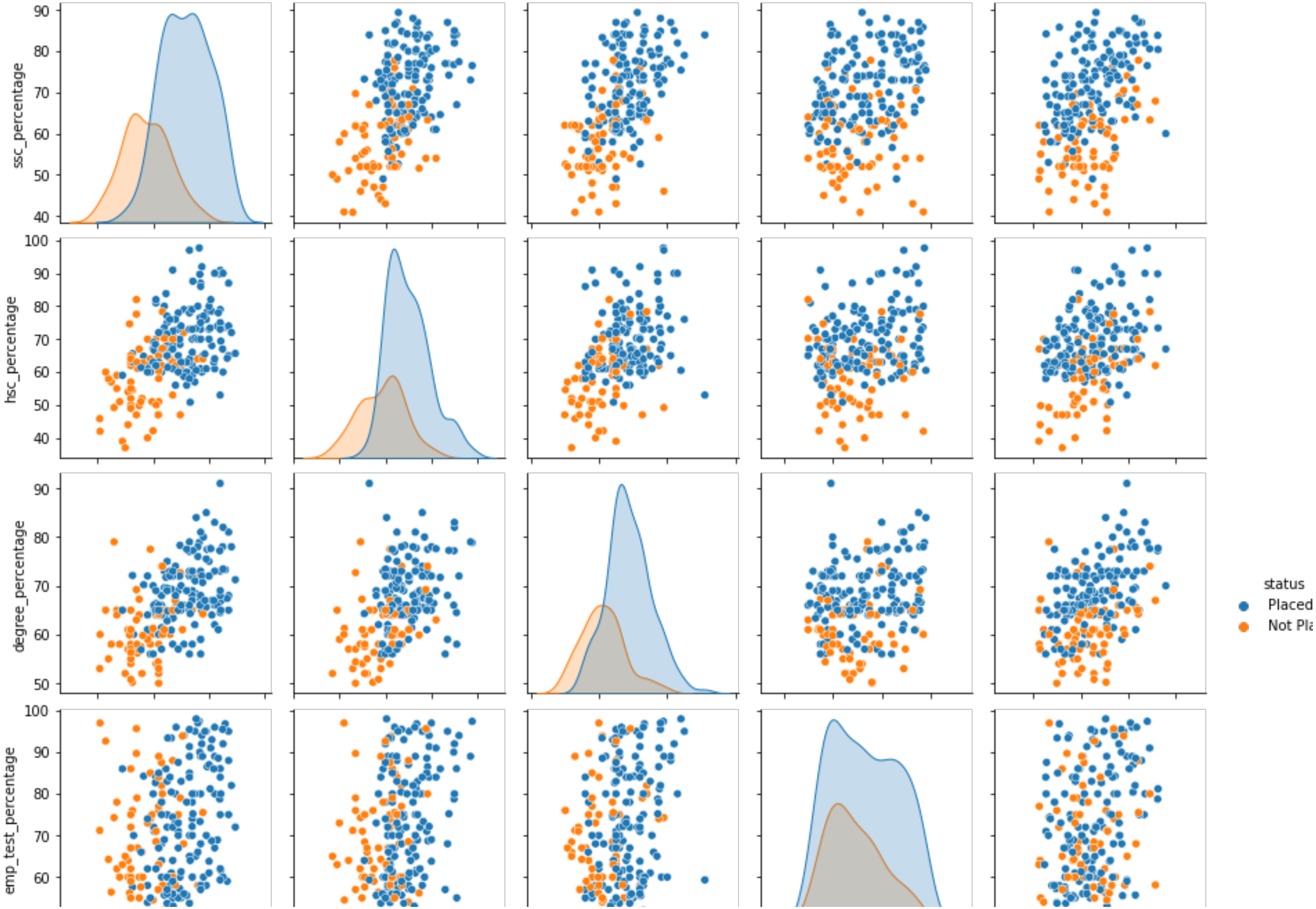
```
import seaborn as sns
sns.heatmap(df.corr())
```

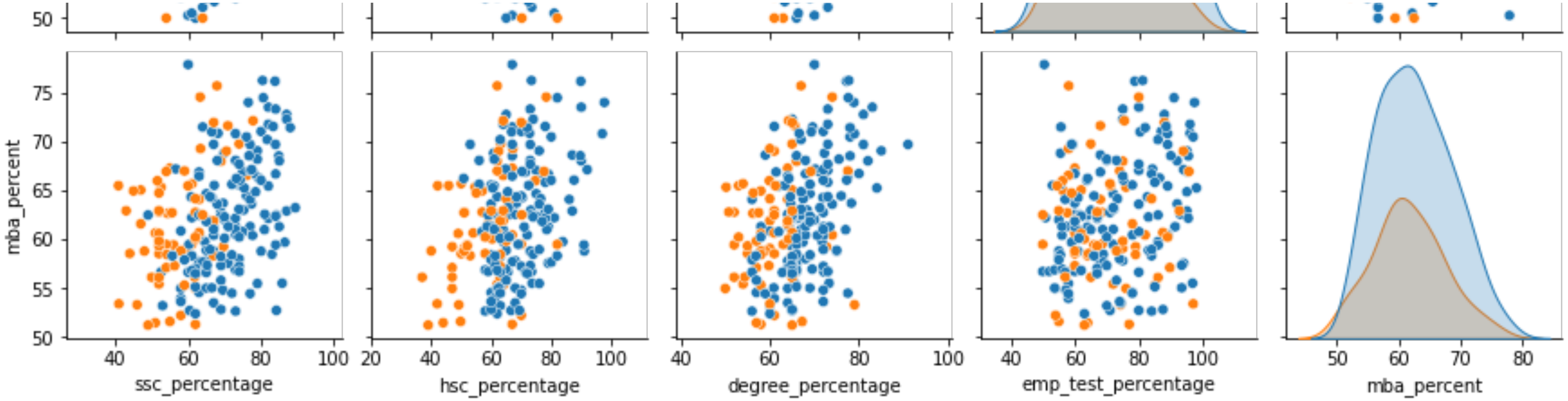
```
<AxesSubplot:>
```



```
sns.pairplot(df,hue='status')
```

```
<seaborn.axisgrid.PairGrid at 0x7fc512f8f940>
```


ssc_percentage
90
80
70
60
50
40
hsc_percentage
100
90
80
70
60
50
40
degree_percentage
90
80
70
60
50
emp_test_percentage
100
90
80
70
60
status
Placed
Not Pla

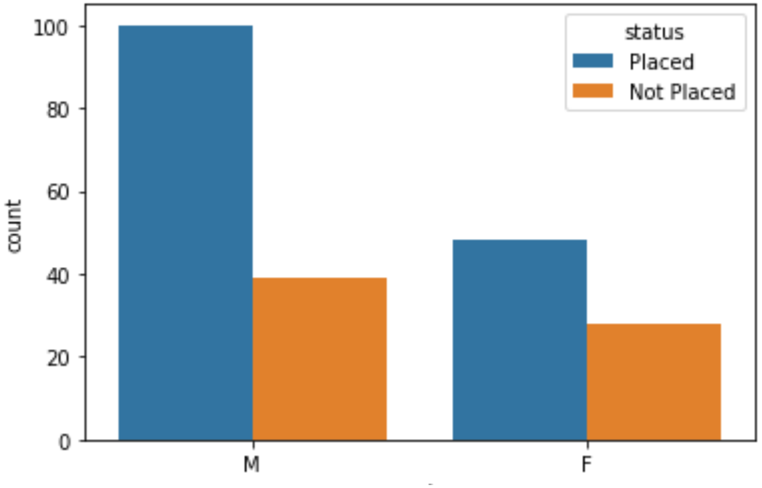


```
df['gender'].value_counts()
```

```
M    139
F     76
Name: gender, dtype: int64
```

```
sns.countplot(data=df,x='gender',hue='status')
```

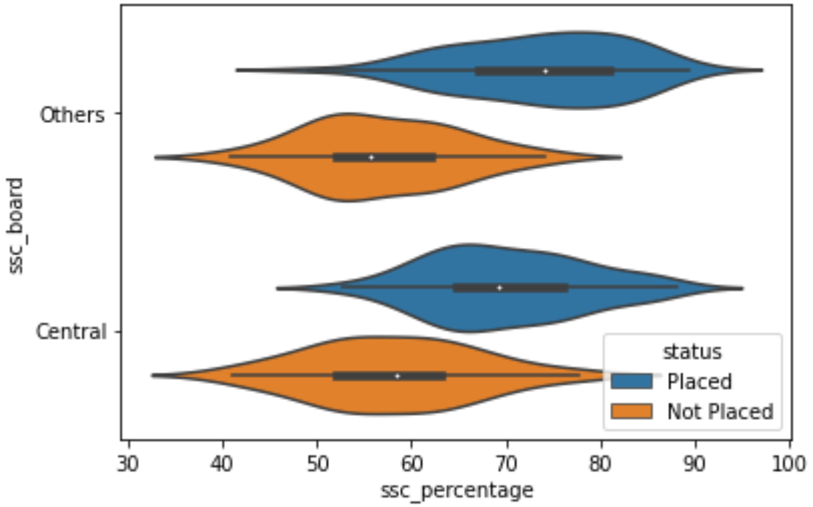
```
<AxesSubplot:xlabel='gender', ylabel='count'>
```



gender

```
sns.violinplot(x='ssc_percentage',y='ssc_board',hue='status',data=df)
```

```
<AxesSubplot:xlabel='ssc_percentage', ylabel='ssc_board'>
```

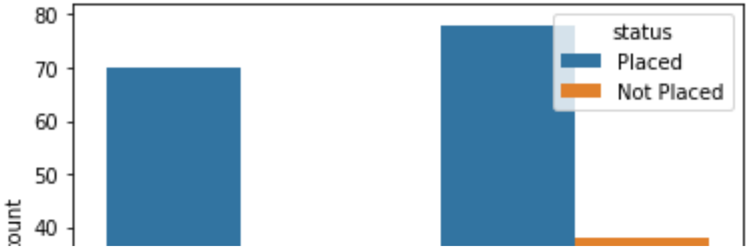


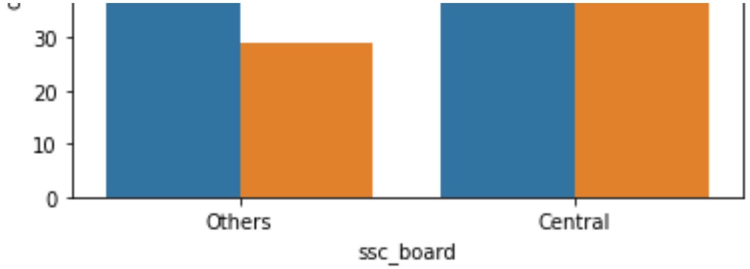
```
df['ssc_board'].value_counts()
```

```
Central    116
Others      99
Name: ssc_board, dtype: int64
```

```
sns.countplot(data=df,x='ssc_board',hue='status')
```

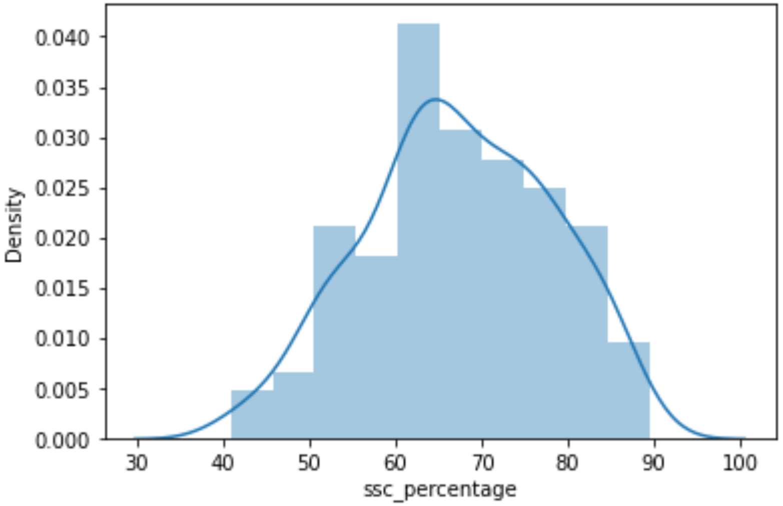
```
<AxesSubplot:xlabel='ssc_board', ylabel='count'>
```





```
sns.distplot(df['ssc_percentage'])
```

```
/usr/local/lib/python3.8/dist-packages/seaborn/distributions.py:2619: FutureWarning: `distplot` is a deprecated f
  warnings.warn(msg, FutureWarning)
<AxesSubplot:xlabel='ssc_percentage', ylabel='Density'>
```

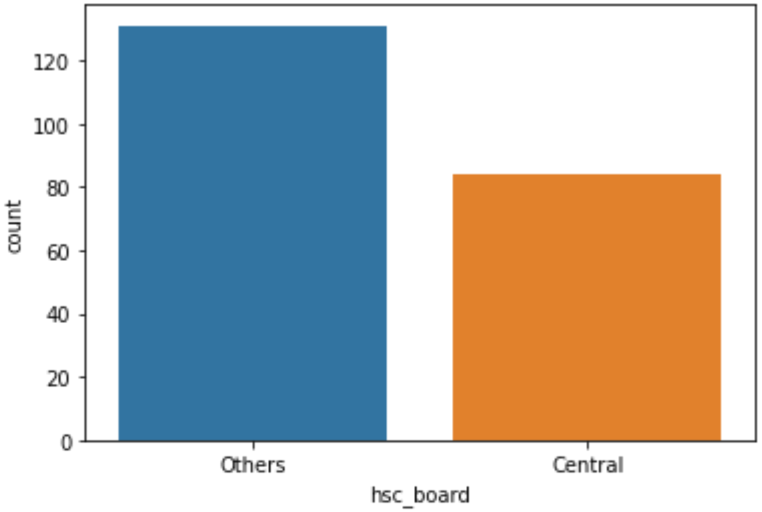


```
df['hsc_board'].value_counts()
```

```
Others     131
Central     84
Name: hsc_board, dtype: int64
```

```
sns.countplot(x='hsc_board',data=df)
```

```
<AxesSubplot:xlabel='hsc_board', ylabel='count'>
```

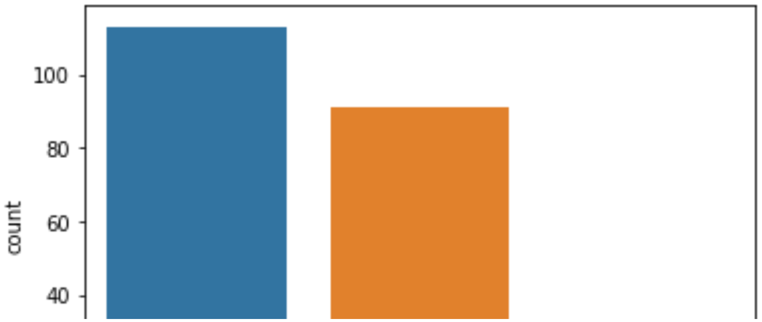


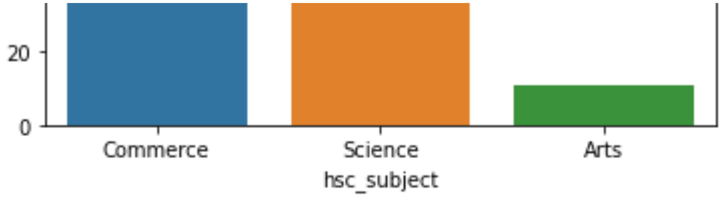
```
df['hsc_subject'].value_counts()
```

```
Commerce    113
Science      91
Arts         11
Name: hsc_subject, dtype: int64
```

```
sns.countplot(x='hsc_subject',data=df)
```

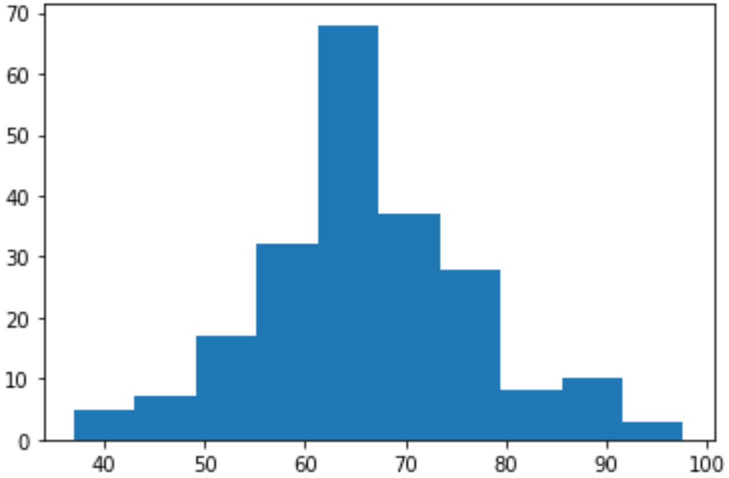
```
<AxesSubplot:xlabel='hsc_subject', ylabel='count'>
```





```
plt.hist(df['hsc_percentage'])
```

```
(array([ 5.,  7., 17., 32., 68., 37., 28.,  8., 10.,  3.]),
 array([37.  , 43.07, 49.14, 55.21, 61.28, 67.35, 73.42, 79.49, 85.56,
        91.63, 97.7 ]),
 <BarContainer object of 10 artists>)
```

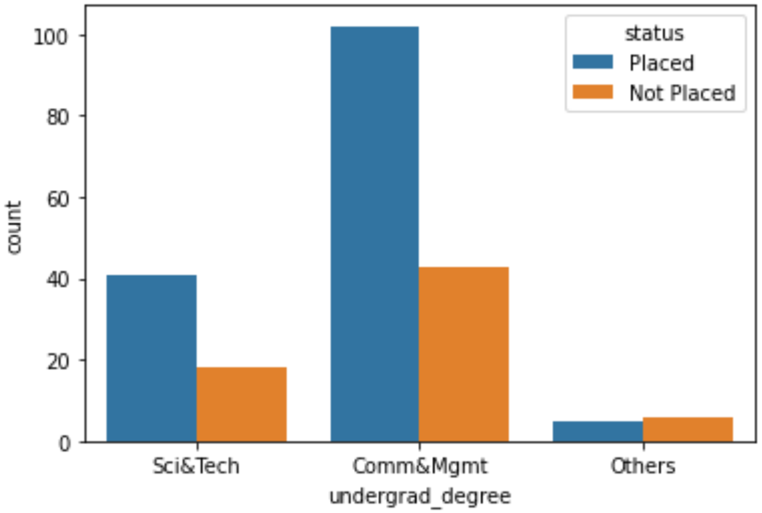


```
df['undergrad_degree'].value_counts()
```

```
Comm&Mgmt    145
Sci&Tech      59
Others        11
Name: undergrad_degree, dtype: int64
```

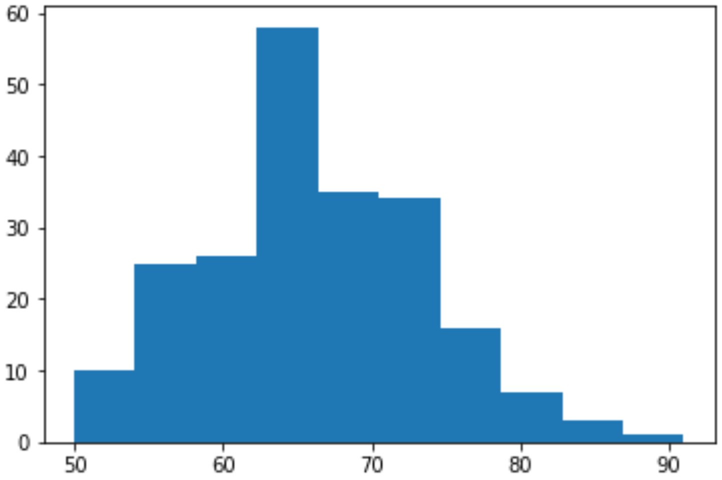
```
sns.countplot(x='undergrad_degree',data=df,hue='status')
```

```
<AxesSubplot:xlabel='undergrad_degree', ylabel='count'>
```



```
plt.hist(df['degree_percentage'])
```

```
(array([10., 25., 26., 58., 35., 34., 16.,  7.,  3.,  1.]),
 array([50. , 54.1, 58.2, 62.3, 66.4, 70.5, 74.6, 78.7, 82.8, 86.9, 91. ]),
 <BarContainer object of 10 artists>)
```

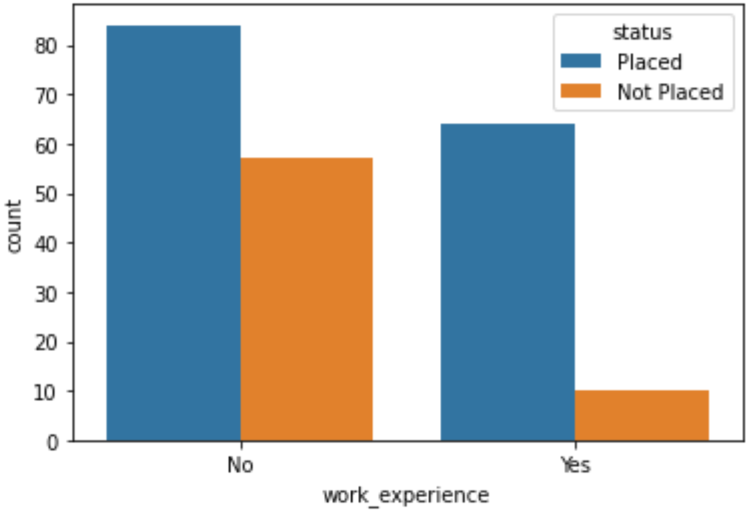


```
df['work_experience'].value_counts()
```

```
No     141
Yes     74
Name: work_experience, dtype: int64
```

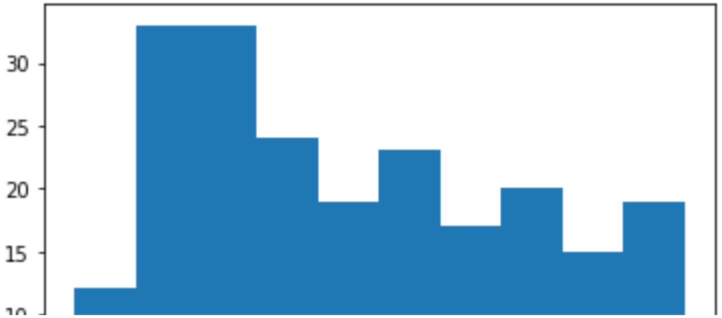
```
sns.countplot(x='work_experience',data=df,hue='status')
```

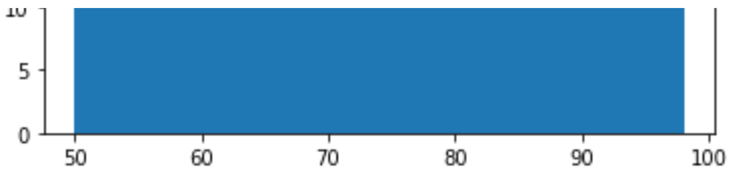
```
<AxesSubplot:xlabel='work_experience', ylabel='count'>
```



```
plt.hist(df['emp_test_percentage'])
```

```
(array([12., 33., 33., 24., 19., 23., 17., 20., 15., 19.]),
 array([50. , 54.8, 59.6, 64.4, 69.2, 74. , 78.8, 83.6, 88.4, 93.2, 98. ]),
 <BarContainer object of 10 artists>)
```



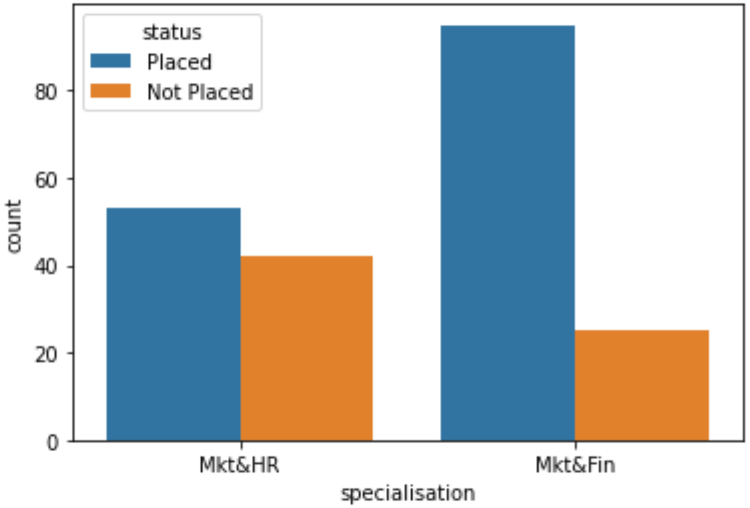


```
df['specialisation'].value_counts()
```

```
Mkt&Fin    120
Mkt&HR      95
Name: specialisation, dtype: int64
```

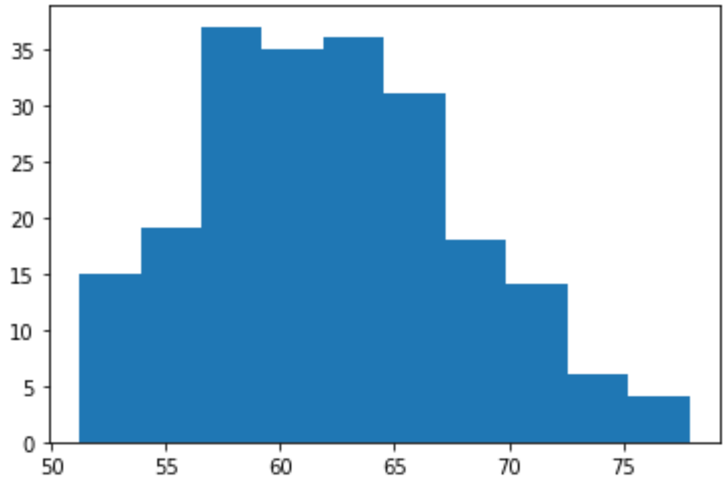
```
sns.countplot(x='specialisation',data=df,hue='status')
```

```
<AxesSubplot:xlabel='specialisation', ylabel='count'>
```



```
plt.hist(df['mba_percent'])
```

```
(array([15., 19., 37., 35., 36., 31., 18., 14.,  6.,  4.]),
 array([51.21 , 53.878, 56.546, 59.214, 61.882, 64.55 , 67.218, 69.886,
        72.554, 75.222, 77.89 ]),
 <BarContainer object of 10 artists>)
```



```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
df['specialisation']=le.fit_transform(df['specialisation'])
df['undergrad_degree']=le.fit_transform(df['undergrad_degree'])
df['hsc_subject']=le.fit_transform(df['hsc_subject'])
df['hsc_board']=le.fit_transform(df['hsc_board'])
df['ssc_board']=le.fit_transform(df['ssc_board'])
df['gender']=le.fit_transform(df['gender'])
df['work_experience']=le.fit_transform(df['work_experience'])
```

```
df
```

	gender	ssc_percentage	ssc_board	hsc_percentage	hsc_board	hsc_subject	degree_percentage	undergrad_degree
0	1	67.00	1	91.00	1	1	58.00	2
1	1	79.33	0	78.33	1	2	77.48	2
2	1	65.00	0	68.00	0	0	64.00	0

3	1	56.00	0	52.00	0	2	52.00	2
4	1	85.80	0	73.60	0	1	73.30	0
...	...	...	...	...	...	...	...	..
210	1	80.60	1	82.00	1	1	77.60	0
211	1	58.00	1	60.00	1	2	72.00	2
212	1	67.00	1	67.00	1	1	73.00	0
213	0	74.00	1	66.00	1	1	58.00	0
214	1	62.00	0	58.00	1	2	53.00	0

215 rows × 13 columns

```
df.dtypes
```

```
gender                   int64
ssc_percentage         float64
ssc_board                int64
hsc_percentage         float64
hsc_board                int64
hsc_subject              int64
degree_percentage      float64
undergrad_degree         int64
work_experience          int64
emp_test_percentage    float64
specialisation           int64
mba_percent            float64
status                  object
dtype: object
```

```
x= df.drop('status',axis=1)
y= df[['status']]
x
```

	gender	ssc_percentage	ssc_board	hsc_percentage	hsc_board	hsc_subject	degree_percentage	undergrad_degree
0	1	67.00	1	91.00	1	1	58.00	2
1	1	79.33	0	78.33	1	2	77.48	2
2	1	65.00	0	68.00	0	0	64.00	0
3	1	56.00	0	52.00	0	2	52.00	2
4	1	85.80	0	73.60	0	1	73.30	0
...	...	...	...	...	...	...	...	..
210	1	80.60	1	82.00	1	1	77.60	0
211	1	58.00	1	60.00	1	2	72.00	2
212	1	67.00	1	67.00	1	1	73.00	0
213	0	74.00	1	66.00	1	1	58.00	0
214	1	62.00	0	58.00	1	2	53.00	0

215 rows × 12 columns

```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.30,random_state=42)
x_test
```

	gender	ssc_percentage	ssc_board	hsc_percentage	hsc_board	hsc_subject	degree_percentage	undergrad_degree

200	1	69.0	1	60.0	1	1	65.00	0
212	1	67.0	1	67.0	1	1	73.00	0
138	0	82.0	1	64.0	1	2	73.00	2
176	0	59.0	0	60.0	1	1	56.00	0
15	0	65.0	0	75.0	0	1	69.00	0
...	...	...	...	...	...	...	...	..
68	0	69.7	0	47.0	0	1	72.70	2
5	1	55.0	1	49.8	1	2	67.25	2
136	0	47.0	0	59.0	0	0	64.00	0
56	1	63.0	1	71.4	1	1	61.40	0
100	0	45.0	1	57.0	1	1	58.00	0

65 rows × 12 columns

```
from sklearn.preprocessing import StandardScaler
scalar=StandardScaler()
scalar.fit(x_train)
x_train=scalar.transform(x_train)
x_test=scalar.transform(x_test)
```

```
#KNN ALGORITHM
from sklearn.model_selection import GridSearchCV
from sklearn.neighbors import KNeighborsClassifier
cls1=KNeighborsClassifier()
parametrs={'n_neighbors':[3,5,7,9,11,13,15,17,19,21],'weights':['uniform','distance']}
```

```
clf=GridSearchCV(cls1,parametrs,cv=10,scoring='accuracy')  #cv=cross vallidation
clf.fit(x_train,y_train)
```

```
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
```

```
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
```

```
print(clf.best_params_)
```

```
{'n_neighbors': 17, 'weights': 'distance'}
```

```
clf2=KNeighborsClassifier(n_neighbors=21,weights='distance')
clf2.fit(x_train,y_train)
y_pred1=clf2.predict(x_test)
y_pred1
```

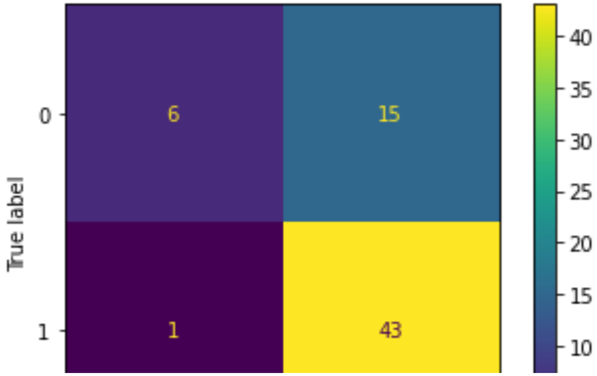
```
/usr/local/lib/python3.8/dist-packages/sklearn/neighbors/_classification.py:198: DataConversionWarning: A column-
  return self._fit(X, y)
array(['Placed', 'Placed', 'Placed', 'Not Placed', 'Placed', 'Not Placed',
       'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Placed', 'Not Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Placed', 'Placed', 'Not Placed'], dtype=object)
```

```
from sklearn.metrics import accuracy_score,classification_report,ConfusionMatrixDisplay,confusion_matrix
report=classification_report(y_test,y_pred1)
cunf_mat1=confusion_matrix(y_test,y_pred1)
cm1=ConfusionMatrixDisplay(cunf_mat1)
cm1.plot()
score_knn=accuracy_score(y_test,y_pred1)
print('accuracy_score :',score_knn)
print('************************************************************************')
print(report)
```

```
accuracy_score : 0.7538461538461538
************************************************************************
              precision    recall  f1-score   support

  Not Placed       0.86      0.29      0.43        21
      Placed       0.74      0.98      0.84        44

    accuracy                           0.75        65
   macro avg       0.80      0.63      0.64        65
weighted avg       0.78      0.75      0.71        65
```



```
#Naive_bayes
from sklearn.naive_bayes import GaussianNB
model2=GaussianNB()
model2.fit(x_train,y_train)
y_pred2=model2.predict(x_test)
y_pred2
```

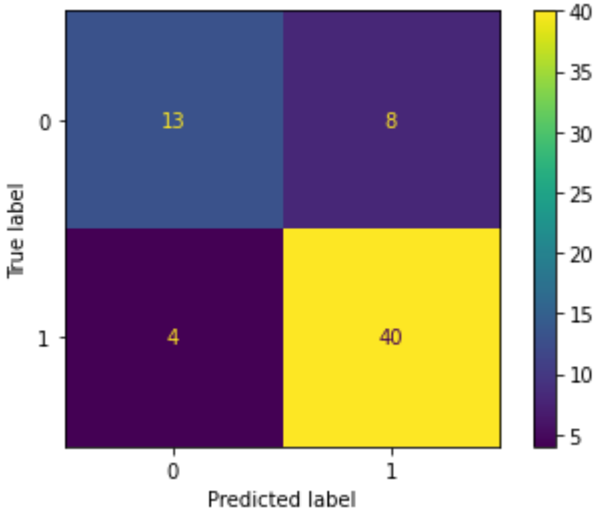
```
/usr/local/lib/python3.8/dist-packages/sklearn/utils/validation.py:993: DataConversionWarning: A column-vector y 
  y = column_or_1d(y, warn=True)
array(['Placed', 'Placed', 'Placed', 'Not Placed', 'Placed', 'Not Placed',
       'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Not Placed', 'Not Placed', 'Placed', 'Not Placed', 'Placed',
       'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Not Placed', 'Not Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Not Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Not Placed', 'Not Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Not Placed', 'Not Placed', 'Placed',
       'Not Placed'], dtype='<U10')
```

```
from sklearn.metrics import accuracy_score,classification_report,ConfusionMatrixDisplay,confusion_matrix
cunf_mat2=confusion_matrix(y_test,y_pred2)
cm2=ConfusionMatrixDisplay(cunf_mat2)
cm2.plot()
report2=classification_report(y_test,y_pred2)
score_naive_bays=accuracy_score(y_test,y_pred2)
print('accuracy_score :',score_naive_bays)
print('*********************************************************')
print(report2)
```

```
accuracy_score : 0.8153846153846154
********************************************************
              precision    recall  f1-score   support

  Not Placed       0.76      0.62      0.68        21
      Placed       0.83      0.91      0.87        44

    accuracy                           0.82        65
   macro avg       0.80      0.76      0.78        65
weighted avg       0.81      0.82      0.81        65
```



```
#SVM
from sklearn.svm import SVC
model3=SVC()
model3.fit(x_train,y_train)
y_pred3=model3.predict(x_test)
y_pred3
```

```
/usr/local/lib/python3.8/dist-packages/sklearn/utils/validation.py:993: DataConversionWarning: A column-vector y
  y = column_or_1d(y, warn=True)
array(['Placed', 'Placed', 'Placed', 'Not Placed', 'Placed', 'Not Placed',
       'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed'
```

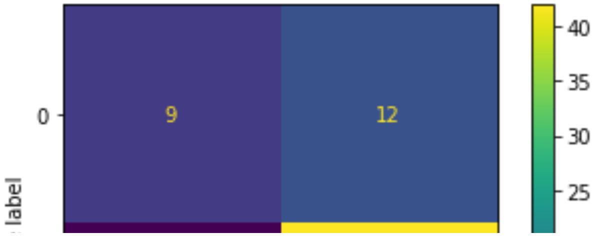
```
         'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
         'Placed', 'Placed', 'Placed', 'Placed', 'Not Placed', 'Not Placed',
         'Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
         'Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
         'Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
         'Placed', 'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed',
         'Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
         'Not Placed', 'Placed', 'Not Placed', 'Placed', 'Placed', 'Placed',
         'Placed', 'Placed', 'Not Placed', 'Placed', 'Not Placed'],
        dtype=object)
```

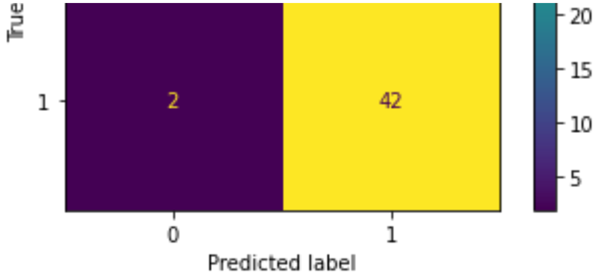
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix,ConfusionMatrixDisplay
score_svm=accuracy_score(y_test,y_pred3)
report3=classification_report(y_test,y_pred3)
cunf_mat3=confusion_matrix(y_test,y_pred3)
cm3=ConfusionMatrixDisplay(cunf_mat3)
cm3.plot()
print('accuracy_score :',score_svm)
print("************************************************************")
print(report3)
```

```
accuracy_score : 0.7846153846153846
************************************************************
              precision    recall  f1-score   support

  Not Placed       0.82      0.43      0.56        21
      Placed       0.78      0.95      0.86        44

    accuracy                           0.78        65
   macro avg       0.80      0.69      0.71        65
weighted avg       0.79      0.78      0.76        65
```





```
#Random_Forest
from sklearn.ensemble import RandomForestClassifier
model4=RandomForestClassifier()
model4.fit(x_train,y_train)
y_pred4=model4.predict(x_test)
y_pred4
```

```
<ipython-input-66-e6c18c1b66b5>:4: DataConversionWarning: A column-vector y was passed when a 1d array was expect
  model4.fit(x_train,y_train)
array(['Placed', 'Placed', 'Placed', 'Not Placed', 'Placed', 'Not Placed',
       'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Not Placed',
       'Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Not Placed', 'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Not Placed', 'Not Placed', 'Placed', 'Not Placed'],
      dtype=object)
```

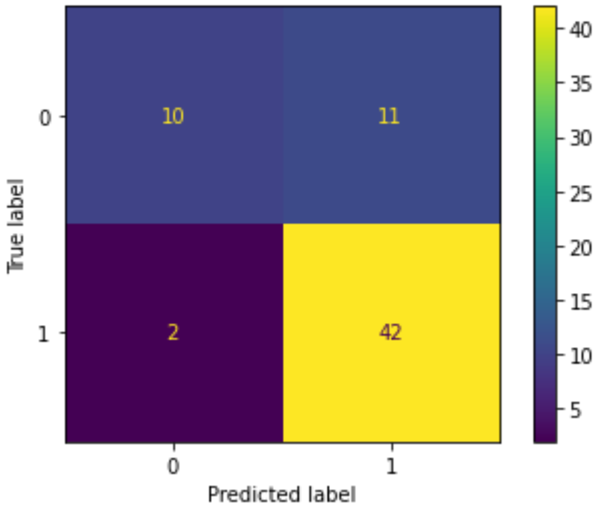
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix,ConfusionMatrixDisplay
score_randomforest=accuracy_score(y_test,y_pred4)
report4=classification_report(y_test,y_pred4)
cunf_mat4=confusion_matrix(y_test,y_pred4)
cm4=ConfusionMatrixDisplay(cunf_mat4)
cm4.plot()
```

```
print('accuracy_score :',score_randomforest)
print('*************************************************')
print(report4)
```

```
accuracy_score : 0.8
*************************************************
              precision    recall  f1-score   support

  Not Placed       0.83      0.48      0.61        21
      Placed       0.79      0.95      0.87        44

    accuracy                           0.80        65
   macro avg       0.81      0.72      0.74        65
weighted avg       0.81      0.80      0.78        65
```



```
#Decision_Tree
from sklearn.tree import DecisionTreeClassifier
model5=DecisionTreeClassifier()
model5.fit(x_train,y_train)
y_pred5=model5.predict(x_test)
y_pred5
```

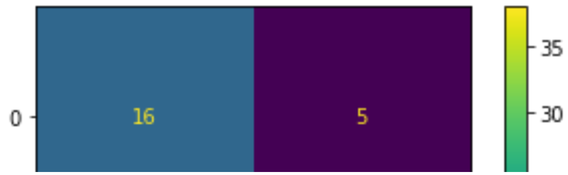
```
array(['Placed', 'Placed', 'Placed', 'Not Placed', 'Placed', 'Not Placed',
array(['Placed', 'Placed', 'Placed', 'Not Placed', 'Placed', 'Not Placed',
       'Not Placed', 'Placed', 'Placed', 'Placed', 'Not Placed', 'Placed',
       'Not Placed', 'Not Placed', 'Placed', 'Not Placed', 'Placed',
       'Placed', 'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Not Placed', 'Not Placed', 'Placed', 'Not Placed', 'Not Placed',
       'Placed', 'Not Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Placed', 'Not Placed', 'Placed', 'Placed',
       'Placed', 'Placed', 'Placed', 'Placed', 'Placed', 'Placed',
       'Not Placed', 'Not Placed', 'Not Placed', 'Not Placed', 'Placed',
       'Placed', 'Placed', 'Placed', 'Not Placed', 'Not Placed',
       'Not Placed', 'Placed', 'Not Placed'], dtype=object)
```

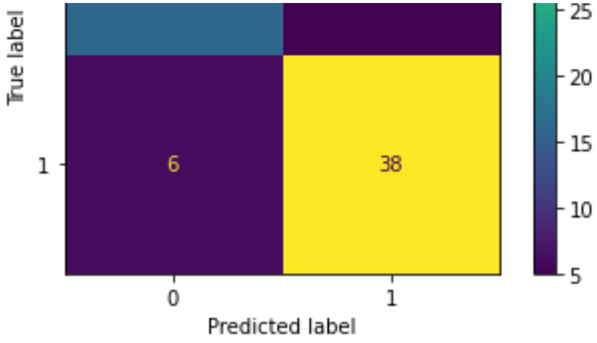
```
from sklearn.metrics import accuracy_score,classification_report,confusion_matrix,ConfusionMatrixDisplay
score_tree=accuracy_score(y_test,y_pred5)
report5=classification_report(y_test,y_pred5)
cunf_mat5=confusion_matrix(y_test,y_pred5)
cm5=ConfusionMatrixDisplay(cunf_mat5)
cm5.plot()
print('Accuracy_Score:',score_tree)
print('*********************************************************')
print(report5)
```

```
Accuracy_Score: 0.8307692307692308
*********************************************************
              precision    recall  f1-score   support

  Not Placed       0.73      0.76      0.74        21
      Placed       0.88      0.86      0.87        44

    accuracy                           0.83        65
   macro avg       0.81      0.81      0.81        65
weighted avg       0.83      0.83      0.83        65
```



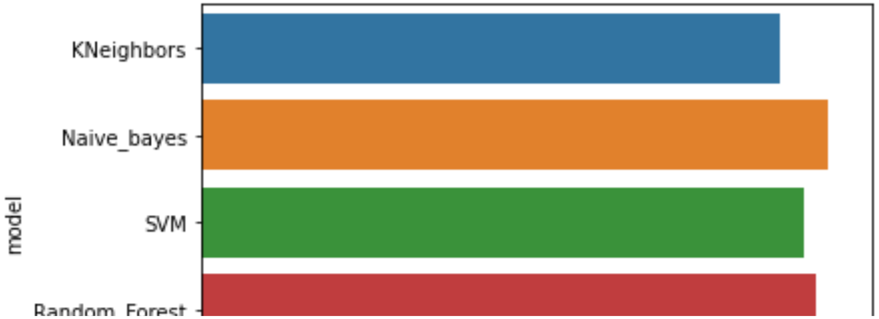


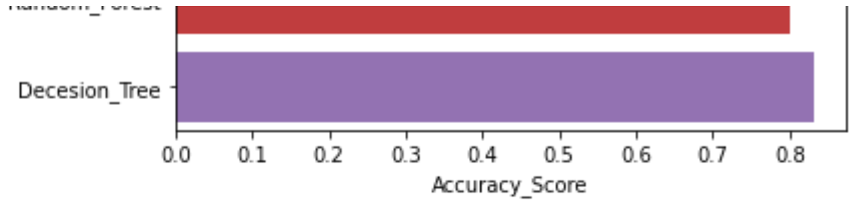
```
modelss=pd.DataFrame({'model':['KNeighbors','Naive_bayes','SVM','Random_Forest','Decesion_Tree'],'Accuracy_Score':[scor
modelss
```

	model	Accuracy_Score
0	KNeighbors	0.753846
1	Naive_bayes	0.815385
2	SVM	0.784615
3	Random_Forest	0.800000
4	Decesion_Tree	0.830769

```
sns.barplot(x='Accuracy_Score',y='model',data=modelss)
```

```
<AxesSubplot:xlabel='Accuracy_Score', ylabel='model'>
```





```
Decision_Tree algorithm gives highest accuracy
```