C.NIVEDHAN

G.PULLAIH COLLEGE OF ENGINEERING

3RD YEAR-EEE

```
# Major project-1

# Choose any dataset of your choice and apply a suitable CLASSIFIER/REGRESSOR
# importing necessary libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

```
#1.Take the data and create dataframe
df = pd.read_csv("/content/archive (4).zip")
df
```

	X	Y
0	1	3.888889
1	2	4.555556
2	3	5.222222
3	4	5.888889
4	5	6.555556
...	...	...
295	296	200.555556
296	297	201.222222
297	298	201.888889
298	299	1.888889
299	300	1.888889

300 rows × 2 columns

```
df.head#Check the head()
```

```
<bound method NDFrame.head of          X           Y
0      1    3.888889
1      2    4.555556
2      3    5.222222
3      4    5.888889
4      5    6.555556
..   ...         ...
295  296  200.555556
296  297  201.222222
297  298  201.888889
298  299    1.888889
299  300    1.888889

[300 rows x 2 columns]>
```

```
df.info()#Check the info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 300 entries, 0 to 299
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   X       300 non-null    int64
 1   Y       300 non-null    float64
dtypes: float64(1), int64(1)
memory usage: 4.8 KB
```

```
type(df)# Check the type()
```

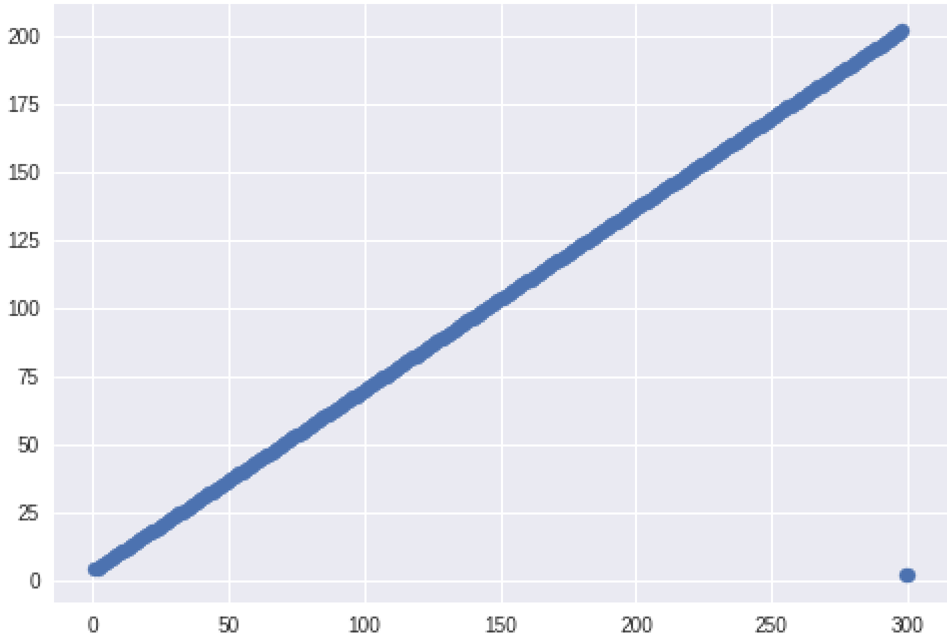
```
pandas.core.frame.DataFrame
```

```
# Search for NAN values
df.isnull().sum()
```

```
X    0
Y    0
dtype: int64
```

```
#3.Data Visualisation - creation of Graphs
plt.scatter(df['X'],df['Y'])
```

```
<matplotlib.collections.PathCollection at 0x7fdefb63c910>
```




```
# From the above graph we can see come outliers at right corner
#for more accuracy we have to remove them
# so there are 298 and 299 rows have really small values compared to the previous ones
print(df.tail())
df.drop([298, 299], inplace=True)
```

```
X = df["X"].to_numpy().reshape(-1, 1)
Y = df["Y"].to_numpy().reshape(-1, 1)
```

```
          X           Y
295  296  200.555556
296  297  201.222222
297  298  201.888889
298  299    1.888889
299  300    1.888889
```

```
# TRAIN and TEST VARIABLES
#sklearn.model_selection - package , train_test_split - library
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test = train_test_split(X,Y, random_state = 0)

#Whatever data splitting /data allocation happens to the xtrain,x_test,ytrain,ytest va
#By default the training variables get 75 % and testing variables get 25%
```

```
print(X.shape) # 298 rows,1 column
print(X_train.shape) # 199 rows,1 column
print(X_test.shape)# 99 rows,1 column
```

```
(298, 1)
(199, 1)
(99, 1)
```

```
print(y.shape) # 298 rows and 1 col
print(y_train.shape) # 223 rows and 1 cols(75 %)
print(y_test.shape) #75rows and 1 col(25%)
```

```
(298, 1)
(223, 1)
(75, 1)
```

```
#SCALING or NORMALISATION -DONE ONLY FOR INPUTS
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler()
x_train = scaler.fit_transform(x_train)
x_test = scaler.fit_transform(x_test)
```

```
#.RUN a CLASSIFIER/REGRESSOR/CLUSTERER
from sklearn.linear_model import LinearRegression
model = LinearRegression()
```

```
#.MODEL FITTING
model.fit(x_train,y_train)
```

```
    LinearRegression()

#9.PREDICT THE OUTPUT
y_pred = model.predict(x_test)#By taking the input testing data , we predict the outpu
y_pred #PREDICTED VALUES
```

```
           [158.64687977],
           [196.49162863],
           [ 43.76103501],
           [103.23135465],
           [162.0258752 ],
           [136.3455099 ],
           [ 60.65601218],
           [ 35.65144597],
           [ 82.95738205],
           [168.10806698],
           [189.73363776],
           [ 43.08523592],
           [  5.91628615],
           [ 99.17656013],
           [144.45509895],
           [128.91171995],
           [149.86149164],
           [122.82952817],
           [  3.88888889],
           [159.99847794],
           [150.53729073],
           [127.56012178],
           [ 23.4870624 ],
           [151.8888889 ],
           [121.47792999],
           [192.43683411],
           [  5.24048706],
           [ 30.92085236],
           [ 70.1171994 ],
           [157.97108068],
           [116.0715373 ],
           [ 68.76560122],
           [ 37.67884323],
           [191.08523594],
           [113.36834095],

           [ 18.08066971],
           [ 45.11263318],
           [201.22222224],
           [155.26788433],
           [ 91.06697109],
           [ 40.38203958],
           [197.16742772],
           [191.76103502],
           [ 55.24961949],
           [ 73.49619483],
           [114.04414004],
           [148.50989347],
           [ 14.02587519],
           [ 31.59665145],
           [112.69254187],
           [152.56468799],
           [119.45053273],
           [ 18.7564688 ],
```

```
       [141.7519026 ],
       [174.19025877],
       [ 25.51445967],
       [ 62.68340944]])
```

```
y_test #ACTUAL VALUES
```

```
       [159.8888889 ],
       [197.2222222 ],
       [ 46.55555556],
       [105.2222222 ],
       [163.2222222 ],
       [137.8888889 ],
       [ 63.22222222],
       [ 38.55555556],
       [ 85.22222222],
       [169.2222222 ],
       [190.5555556 ],
       [ 45.88888889],
       [  9.22222222],
       [101.2222222 ],
       [145.8888889 ],
       [130.5555556 ],
       [151.2222222 ],
       [124.5555556 ],
       [  7.22222222],
       [161.2222222 ],
       [151.8888889 ],
       [129.2222222 ],
       [ 26.55555556],
       [153.2222222 ],
       [123.2222222 ],
       [193.2222222 ],
       [  8.55555556],
       [ 33.88888889],
       [ 72.55555556],
       [159.2222222 ],
       [117.8888889 ],
       [ 71.22222222],
       [ 40.55555556],
       [191.8888889 ],

       [115.2222222 ],
       [ 21.22222222],
       [ 47.88888889],
       [201.8888889 ],
       [156.5555556 ],
       [ 93.22222222],
       [ 43.22222222],
       [197.8888889 ],
       [192.5555556 ],
       [ 57.88888889],
       [ 75.88888889],
       [115.8888889 ],
       [149.8888889 ],
       [ 17.22222222],
       [ 34.55555556],
       [114.5555556 ],
       [153.8888889 ],
       [121.2222222 ],
```

```
       [ 21.88888889],
       [143.2222222 ],
       [175.2222222 ],
       [ 28.55555556],
       [ 65.22222222]])
```

```
print(x_train[10]) #these are scaled/normalised values
```

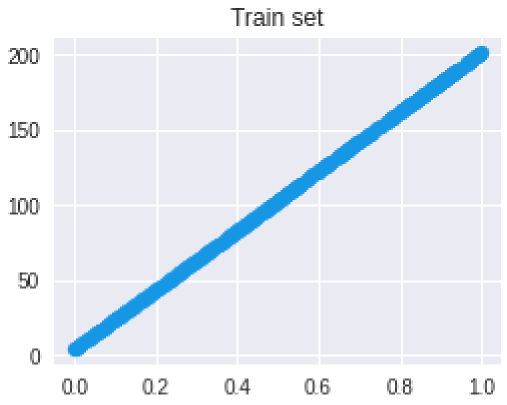
```
[0.91554054]
```

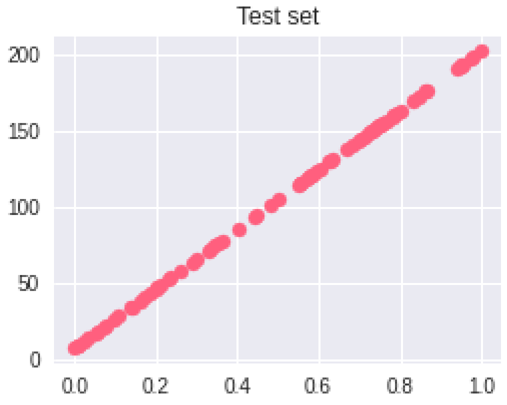
```
#INDIVIDUAL PREDICTION
model.predict([x_train[10]])
```

```
array([[184.55555557]])
```

```
# Print Standardized sets

plt.figure(figsize=(4,3))
plt.title("Train set")
plt.scatter(x_train, y_train, c="#1597E5")

plt.figure(figsize=(4,3))
plt.title("Test set")
plt.scatter(x_test, y_test, c="#FF5F7E")
```

```
<matplotlib.collections.PathCollection at 0x7fdeee84ff50>
```





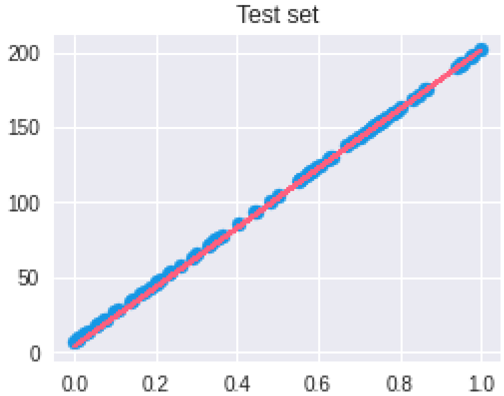

```
# Train our model

from sklearn.linear_model import LinearRegression
from sklearn.metrics import r2_score
```

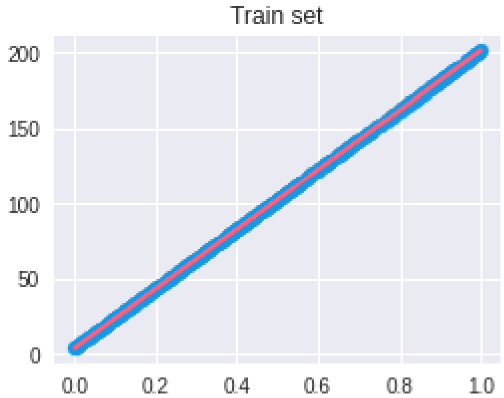
```
from sklearn.metrics import mean_squared_error

model = LinearRegression().fit(x_train, y_train)
y_pred_test = model.predict(x_test)
y_pred_train = model.predict(x_train)
# Evaluate our model

print(f"R2 Score for Test set: ", r2_score(y_test, y_pred_test))
print(f"MSE for Test set: ", mean_squared_error(y_test, y_pred_test))

print(f"R2 Score for Train set: ", r2_score(y_train, y_pred_train))
print(f"MSE for Train set: ", mean_squared_error(y_train, y_pred_train))
```

```
R2 Score for Test set:  0.9986425280384631
MSE for Test set:  4.5909178863626305
R2 Score for Train set:  1.0
MSE for Train set:  3.864880498330617e-16
```

```
# Plot out predictions

plt.figure(figsize=(4,3))
plt.title("Test set")
plt.scatter(x_test, y_test, c="#1597E5")
plt.plot(x_test, y_pred_test, c="#FF5F7E")

plt.figure(figsize=(4,3))
plt.title("Train set")
plt.scatter(x_train, y_train, c="#1597E5")
plt.plot(x_train, y_pred_train, c="#FF5F7E")
```

```
[<matplotlib.lines.Line2D at 0x7fdefe3daed0>]
```





```
# As we can see the red line perfectly fits our test and train data
```


✓ 0s completed at 2:13 PM

```
# MAJOR-PROJECT 2
# Choose any dataset of your choice and apply K Means Clustering

#Import Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
#Get the Data
#Read in the College_Data file using read_csv.

df=pd.read_csv("/content/College.csv",index_col=0)
df
```

```
              Private  Apps  Accept  Enroll  Top10perc  Top25perc  F.Undergrad  P
```

```
# Check the head of the data
df.head
```

```
<bound method NDFrame.head of                               Private   Apps  
Accept  Enroll  Top10perc  \
Abilene Christian University      Yes   1660    1232     721         23   
Adelphi University                Yes   2186    1924     512         16   
Adrian College                    Yes   1428    1097     336         22   
Agnes Scott College               Yes    417     349     137         60   
Alaska Pacific University         Yes    193     146      55         16   
...                               ...    ...     ...     ...        ...   
Worcester State College            No   2197    1515     543          4   
Xavier University                 Yes   1959    1805     695         24   
Xavier University of Louisiana    Yes   2097    1915     695         34   
Yale University                   Yes  10705    2453    1317         95   
York College of Pennsylvania      Yes   2989    1855     691         28   

                                Top25perc  F.Undergrad  P.Undergrad  Outstate  \
Abilene Christian University           52         2885          537      7440   
Adelphi University                     29         2683         1227     12280   
Adrian College                         50         1036           99     11250   
Agnes Scott College                    89          510           63     12960   
Alaska Pacific University              44          249          869      7560   
...                                   ...          ...          ...       ...   
Worcester State College                26         3089         2029      6797   
Xavier University                      47         2849         1107     11520   
Xavier University of Louisiana         61         2793          166      6900   
Yale University                        99         5217           83     19840   
York College of Pennsylvania           63         2988         1726      4990   

                                Room.Board  Books  Personal  PhD  Terminal  \
Abilene Christian University          3300    450      2200   70        78   
Adelphi University                    6450    750      1500   29        30   
Adrian College                        3750    400      1165   53        66   
Agnes Scott College                   5450    450       875   92        97   
Alaska Pacific University             4120    800      1500   76        72   
...                                    ...    ...       ...  ...       ...   
Worcester State College               3900    500      1200   60        60   
Xavier University                     4960    600      1250   73        75   
Xavier University of Louisiana        4200    617       781   67        75   
Yale University                       6510    630      2115   96        96   
York College of Pennsylvania          3560    500      1250   75        75   

                                S.F.Ratio  perc.alumni  Expend  Grad.Rate  
Abilene Christian University         18.1           12    7041         60  
Adelphi University                   12.2           16   10527         56  
Adrian College                       12.9           30    8735         54  
Agnes Scott College                   7.7           37   19016         59  
Alaska Pacific University            11.9            2   10922         15  
...                                   ...          ...     ...        ...  
Worcester State College              21.0           14    4469         40  
Xavier University                    13.3           31    9189         83  
Xavier University of Louisiana       14.4           20    8323         49  
Yale University                       5.8           49   40386         99  
York College of Pennsylvania         18.1           28    4509         99  
```

```
[777 rows x 18 columns]>
```

```
# Check the info()
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 777 entries, Abilene Christian University to York College of Pennsylvania
Data columns (total 18 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   Private      777 non-null    object 
 1   Apps         777 non-null    int64  
 2   Accept       777 non-null    int64  
 3   Enroll       777 non-null    int64  
 4   Top10perc    777 non-null    int64  
 5   Top25perc    777 non-null    int64  
 6   F.Undergrad  777 non-null    int64  
 7   P.Undergrad  777 non-null    int64  
 8   Outstate     777 non-null    int64  
 9   Room.Board   777 non-null    int64  
 10  Books        777 non-null    int64  
 11  Personal     777 non-null    int64  
 12  PhD          777 non-null    int64  
 13  Terminal     777 non-null    int64  
 14  S.F.Ratio    777 non-null    float64
 15  perc.alumni  777 non-null    int64  
 16  Expend       777 non-null    int64  
 17  Grad.Rate    777 non-null    int64  
dtypes: float64(1), int64(16), object(1)
memory usage: 115.3+ KB
```

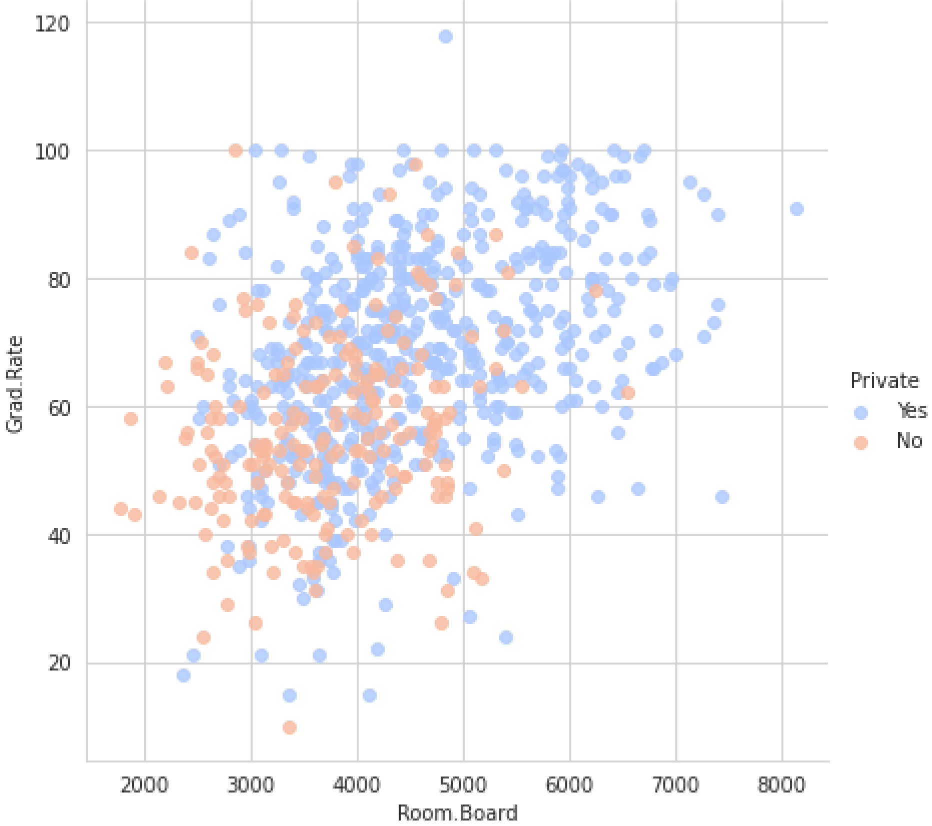
```
#Check the describe()
df.describe()
```

	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad
count	777.000000	777.000000	777.000000	777.000000	777.000000	777.000000
mean	3001.638353	2018.804376	779.972973	27.558559	55.796654	3699.907336
std	3870.201484	2451.113971	929.176190	17.640364	19.804778	4850.420531
min	81.000000	72.000000	35.000000	1.000000	9.000000	139.000000
25%	776.000000	604.000000	242.000000	15.000000	41.000000	992.000000
50%	1558.000000	1110.000000	434.000000	23.000000	54.000000	1707.000000
75%	3624.000000	2424.000000	902.000000	35.000000	69.000000	4005.000000
max	48094.000000	26330.000000	6392.000000	96.000000	100.000000	31643.000000

```
#Creating a scatterplot of Grad.Rate versus Room.Boarding where the points are colored
```

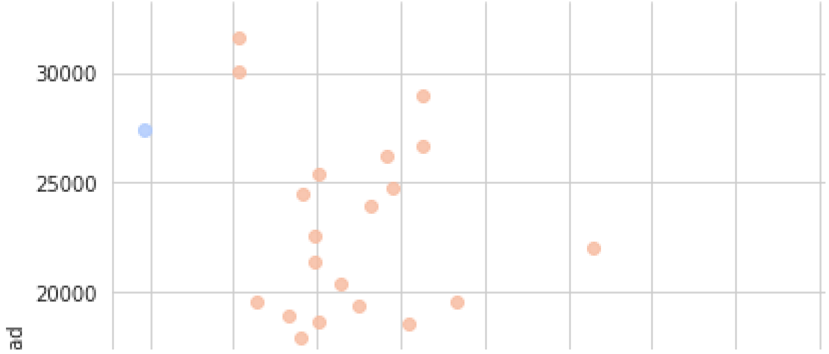
```
sns.set_style('whitegrid')
sns.lmplot('Room.Board','Grad.Rate',data=df, hue='Private',
           palette='coolwarm',size=6,aspect=1,fit_reg=False)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning:
  FutureWarning
/usr/local/lib/python3.7/dist-packages/seaborn/regression.py:581: UserWarning: Tl
  warnings.warn(msg, UserWarning)
<seaborn.axisgrid.FacetGrid at 0x7fc66928e550>
```



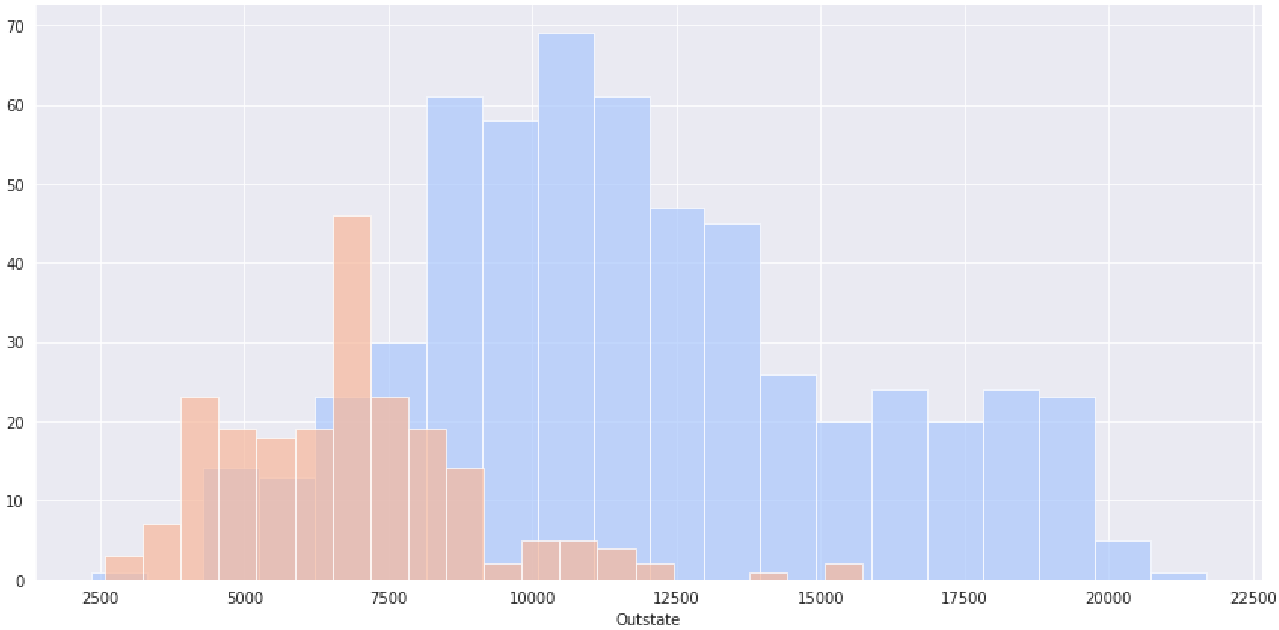

```
# Creating a scatterplot of F.Undergrad versus Outstate where the points are colored b

sns.set_style('whitegrid')
sns.lmplot('Outstate','F.Undergrad',data=df, hue='Private',
           palette='coolwarm',size=6,aspect=1,fit_reg=False)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators.py:43: FutureWarning:
  FutureWarning
/usr/local/lib/python3.7/dist-packages/seaborn/regression.py:581: UserWarning: Th
  warnings.warn(msg, UserWarning)
<seaborn.axisgrid.FacetGrid at 0x7fc6691d9d50>
```



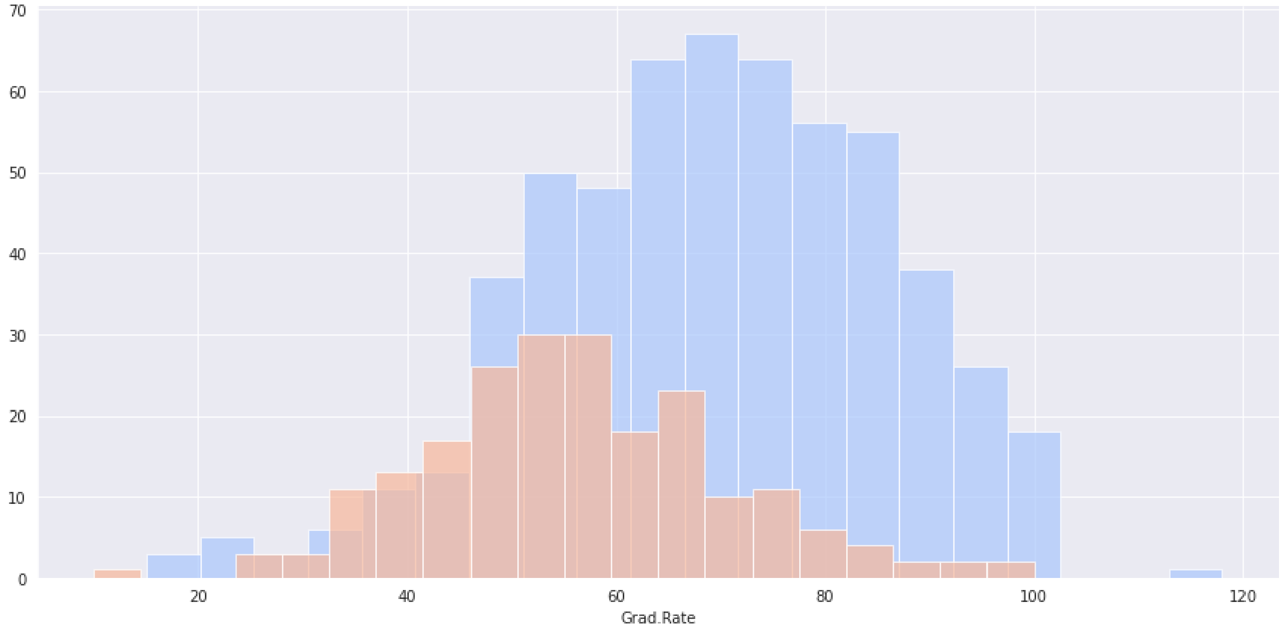

```
#Creating a stacked histogram showing Out of State Tuition based on the Private column

sns.set_style('darkgrid')
g = sns.FacetGrid(df,hue="Private",palette='coolwarm',size=6,aspect=2)
g = g.map(plt.hist,'Outstate',bins=20,alpha=0.7)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:337: UserWarning: The
  warnings.warn(msg, UserWarning)
```




```
# Creating a similar histogram for the Grad.Rate column.

sns.set_style('darkgrid')
g = sns.FacetGrid(df,hue="Private",palette='coolwarm',size=6,aspect=2)
g = g.map(plt.hist,'Grad.Rate',bins=20,alpha=0.7)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:337: UserWarning: The
  warnings.warn(msg, UserWarning)
```




```
# Noticing how there seems to be a private school with a graduation rate of higher tha

df[df['Grad.Rate'] > 100]
```

	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Unc
Cazenovia College	Yes	3847	3433	527	9	35	1010	

```
# lets Set that school's graduation rate to 100 so it makes sense.

df['Grad.Rate']['Cazenovia College'] = 100
df[df['Grad.Rate'] > 100]
```

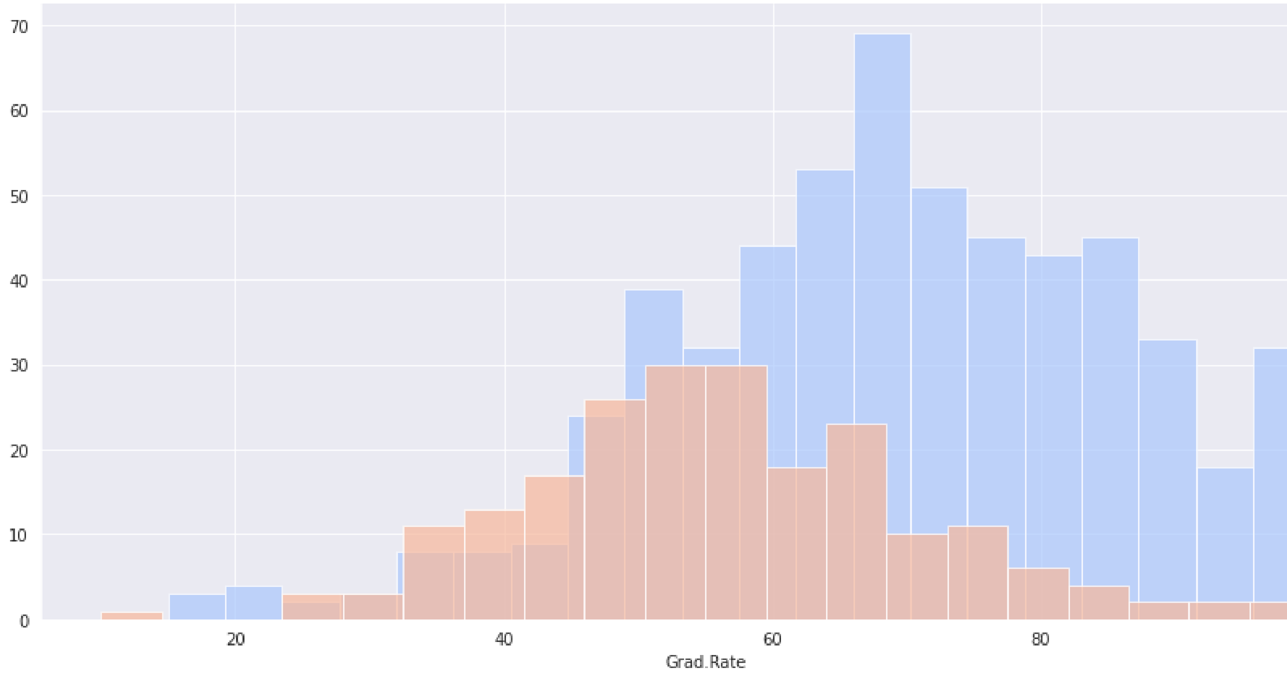
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: SettingWithCopyWa
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
  This is separate from the ipykernel package so we can avoid doing imports until
```

Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Undergrad	O

```
sns.set_style('darkgrid')
g = sns.FacetGrid(df,hue="Private",palette='coolwarm',size=6,aspect=2)
```

```
g = g.map(plt.hist,'Grad.Rate',bins=20,alpha=0.7)
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/axisgrid.py:337: UserWarning: The
  warnings.warn(msg, UserWarning)
```




```
# K Means Cluster Creation
# Import KMeans from SciKit Learn

from sklearn.cluster import KMeans


#Creating an instance of a K Means model with 2 clusters.

kmeans=KMeans(n_clusters=2)


#Fitting the model to all the data except for the Private label.

kmeans.fit(df.drop('Private',axis=1))
```

```
KMeans(n_clusters=2)
```

```
kmeans.cluster_centers_
```

```
array([[1.81323468e+03, 1.28716592e+03, 4.91044843e+02, 2.53094170e+01,
        5.34708520e+01, 2.18854858e+03, 5.95458894e+02, 1.03957085e+04,
        4.31136472e+03, 5.41982063e+02, 1.28033632e+03, 7.04424514e+01,
        7.78251121e+01, 1.40997010e+01, 2.31748879e+01, 8.93204634e+03,
        6.50926756e+01],
       [1.03631389e+04, 6.55089815e+03, 2.56972222e+03, 4.14907407e+01,
        7.02037037e+01, 1.30619352e+04, 2.46486111e+03, 1.07191759e+04,
        4.64347222e+03, 5.95212963e+02, 1.71420370e+03, 8.63981481e+01,
```

```
        9.13333333e+01, 1.40277778e+01, 2.00740741e+01, 1.41705000e+04,
        6.75925926e+01]])
```

```
# Create a new column for df called 'Cluster', which is a 1 for a Private school, and

def converter(cluster):
    if cluster=='Yes':
        return 1
    else:
        return 0

df['Cluster'] = df['Private'].apply(converter)

df.head()
```

	Private	Apps	Accept	Enroll	Top10perc	Top25perc	F.Undergrad	P.Und
Abilene Christian University	Yes	1660	1232	721	23	52	2885	
Adelphi University	Yes	2186	1924	512	16	29	2683	
Adrian College	Yes	1428	1097	336	22	50	1036	
Agnes Scott College	Yes	417	349	137	60	89	510	
Alaska Pacific University	Yes	193	146	55	16	44	249	

```
# Create a confusion matrix and classification report to see how well the Kmeans clust

from sklearn.metrics import confusion_matrix,classification_report
print(confusion_matrix(df['Cluster'],kmeans.labels_))
print(classification_report(df['Cluster'],kmeans.labels_))
```

```
[[138  74]
 [531  34]]
              precision    recall  f1-score   support

           0       0.21      0.65      0.31       212
           1       0.31      0.06      0.10       565

    accuracy                           0.22       777
   macro avg       0.26      0.36      0.21       777
```

```
    weighted avg       0.29      0.22      0.16       777
```

```
# its the final report
```

Colab paid products - Cancel contracts here

0s completed at 2:24 PM