```
import pandas as pd

import numpy as np

from sklearn.preprocessing import StandardScaler

from sklearn.linear_model import Ridge,  Lasso, RidgeCV, LassoCV,
ElasticNet, ElasticNetCV, LinearRegression

from sklearn.model_selection import train_test_split

import statsmodels.api as sm

import matplotlib.pyplot as plt

import seaborn as sns

import pickle

pip install pandas-profiling

Looking in indexes: https://pypi.org/simple, https://us-
python.pkg.dev/colab-wheels/public/simple/
Collecting pandas-profiling
  Downloading pandas_profiling-3.6.6-py2.py3-none-any.whl (324 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 324.4/324.4 kB 6.6 MB/s eta
0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 345.9/345.9 kB 32.3 MB/s eta
0:00:00
anylinux_2_17_x86_64.manylinux2014_x86_64.whl (679 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 679.8/679.8 kB 25.0 MB/s eta
0:00:00
ent already satisfied: seaborn<0.13,>=0.10.1 in
/usr/local/lib/python3.9/dist-packages (from ydata-profiling->pandas-
profiling) (0.12.2)
Requirement already satisfied: pandas!=1.4.0,<1.6,>1.1 in
/usr/local/lib/python3.9/dist-packages (from ydata-profiling->pandas-
profiling) (1.5.3)
Requirement already satisfied: numpy<1.24,>=1.16.0 in
/usr/local/lib/python3.9/dist-packages (from ydata-profiling->pandas-
profiling) (1.22.4)
Collecting typeguard<2.14,>=2.13.2
  Downloading typeguard-2.13.3-py3-none-any.whl (17 kB)
Collecting htmlmin==0.1.12
  Downloading htmlmin-0.1.12.tar.gz (19 kB)
  Preparing metadata (setup.py) ... agehash==4.3.1
  Downloading ImageHash-4.3.1-py2.py3-none-any.whl (296 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 296.5/296.5 kB 29.6 MB/s eta
0:00:00
age_path]==0.7.5
  Downloading visions-0.7.5-py3-none-any.whl (102 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 102.7/102.7 kB 11.9 MB/s eta
```

```
0:00:00
atplotlib<3.7,>=3.2
  Downloading matplotlib-3.6.3-cp39-cp39-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl (11.8 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 11.8/11.8 MB 83.2 MB/s eta
0:00:00
anylinux_2_17_x86_64.manylinux2014_x86_64.whl (33.8 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 33.8/33.8 MB 36.6 MB/s eta
0:00:00
ultimethod<1.10,>=1.4
  Downloading multimethod-1.9.1-py3-none-any.whl (10 kB)
Collecting tqdm<4.65,>=4.48.2
  Downloading tqdm-4.64.1-py2.py3-none-any.whl (78 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 78.5/78.5 kB 9.9 MB/s eta
0:00:00
ent already satisfied: statsmodels<0.14,>=0.13.2 in
/usr/local/lib/python3.9/dist-packages (from ydata-profiling->pandas-
profiling) (0.13.5)
Requirement already satisfied: jinja2<3.2,>=2.11.1 in
/usr/local/lib/python3.9/dist-packages (from ydata-profiling->pandas-
profiling) (3.1.2)
Requirement already satisfied: pydantic<1.11,>=1.8.1 in
/usr/local/lib/python3.9/dist-packages (from ydata-profiling->pandas-
profiling) (1.10.7)
Requirement already satisfied: requests<2.29,>=2.24.0 in
/usr/local/lib/python3.9/dist-packages (from ydata-profiling->pandas-
profiling) (2.27.1)
Requirement already satisfied: PyYAML<6.1,>=5.0.0 in
/usr/local/lib/python3.9/dist-packages (from ydata-profiling->pandas-
profiling) (6.0)
Requirement already satisfied: pillow in
/usr/local/lib/python3.9/dist-packages (from imagehash==4.3.1->ydata-
profiling->pandas-profiling) (8.4.0)
Requirement already satisfied: PyWavelets in
/usr/local/lib/python3.9/dist-packages (from imagehash==4.3.1->ydata-
profiling->pandas-profiling) (1.4.1)
Requirement already satisfied: attrs>=19.3.0 in
/usr/local/lib/python3.9/dist-packages (from
visions[type_image_path]==0.7.5->ydata-profiling->pandas-profiling)
(22.2.0)
Collecting tangled-up-in-unicode>=0.0.4
  Downloading tangled_up_in_unicode-0.2.0-py3-none-any.whl (4.7 MB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 4.7/4.7 MB 92.5 MB/s eta
0:00:00
ent already satisfied: networkx>=2.4 in /usr/local/lib/python3.9/dist-
packages (from visions[type_image_path]==0.7.5->ydata-profiling-
>pandas-profiling) (3.1)
Requirement already satisfied: MarkupSafe>=2.0 in
/usr/local/lib/python3.9/dist-packages (from jinja2<3.2,>=2.11.1-
>ydata-profiling->pandas-profiling) (2.1.2)
```

```
Requirement already satisfied: kiwisolver>=1.0.1 in
/usr/local/lib/python3.9/dist-packages (from matplotlib<3.7,>=3.2-
>ydata-profiling->pandas-profiling) (1.4.4)
Requirement already satisfied: pyparsing>=2.2.1 in
/usr/local/lib/python3.9/dist-packages (from matplotlib<3.7,>=3.2-
>ydata-profiling->pandas-profiling) (3.0.9)
Requirement already satisfied: fonttools>=4.22.0 in
/usr/local/lib/python3.9/dist-packages (from matplotlib<3.7,>=3.2-
>ydata-profiling->pandas-profiling) (4.39.3)
Requirement already satisfied: python-dateutil>=2.7 in
/usr/local/lib/python3.9/dist-packages (from matplotlib<3.7,>=3.2-
>ydata-profiling->pandas-profiling) (2.8.2)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.9/dist-packages (from matplotlib<3.7,>=3.2-
>ydata-profiling->pandas-profiling) (23.0)
Requirement already satisfied: contourpy>=1.0.1 in
/usr/local/lib/python3.9/dist-packages (from matplotlib<3.7,>=3.2-
>ydata-profiling->pandas-profiling) (1.0.7)
Requirement already satisfied: cycler>=0.10 in
/usr/local/lib/python3.9/dist-packages (from matplotlib<3.7,>=3.2-
>ydata-profiling->pandas-profiling) (0.11.0)
Requirement already satisfied: pytz>=2020.1 in
/usr/local/lib/python3.9/dist-packages (from pandas!=1.4.0,<1.6,>1.1-
>ydata-profiling->pandas-profiling) (2022.7.1)
Requirement already satisfied: joblib>=0.14.1 in
/usr/local/lib/python3.9/dist-packages (from phik<0.13,>=0.11.1-
>ydata-profiling->pandas-profiling) (1.2.0)
Requirement already satisfied: typing-extensions>=4.2.0 in
/usr/local/lib/python3.9/dist-packages (from pydantic<1.11,>=1.8.1-
>ydata-profiling->pandas-profiling) (4.5.0)
Requirement already satisfied: urllib3<1.27,>=1.21.1 in
/usr/local/lib/python3.9/dist-packages (from requests<2.29,>=2.24.0-
>ydata-profiling->pandas-profiling) (1.26.15)
Requirement already satisfied: idna<4,>=2.5 in
/usr/local/lib/python3.9/dist-packages (from requests<2.29,>=2.24.0-
>ydata-profiling->pandas-profiling) (3.4)
Requirement already satisfied: charset-normalizer~=2.0.0 in
/usr/local/lib/python3.9/dist-packages (from requests<2.29,>=2.24.0-
>ydata-profiling->pandas-profiling) (2.0.12)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.9/dist-packages (from requests<2.29,>=2.24.0-
>ydata-profiling->pandas-profiling) (2022.12.7)
Requirement already satisfied: patsy>=0.5.2 in
/usr/local/lib/python3.9/dist-packages (from
statsmodels<0.14,>=0.13.2->ydata-profiling->pandas-profiling) (0.5.3)
Requirement already satisfied: six in /usr/local/lib/python3.9/dist-
packages (from patsy>=0.5.2->statsmodels<0.14,>=0.13.2->ydata-
profiling->pandas-profiling) (1.16.0)
Building wheels for collected packages: htmlmin
  Building wheel for htmlmin (setup.py) ... lmin: filename=htmlmin-
```

```
0.1.12-py3-none-any.whl size=27096
sha256=e874a0c6babc1d7c189eb5c3c9e7f5771217e21e75f0c32a6bd52e69349bda3
3
  Stored in directory:
/root/.cache/pip/wheels/1d/05/04/c6d7d3b66539d9e659ac6dfe81e2d0fd4c1a8
316cc5a403300
Successfully built htmlmin
Installing collected packages: htmlmin, typeguard, tqdm, tangled-up-
in-unicode, scipy, multimethod, matplotlib, imagehash, visions, phik,
ydata-profiling, pandas-profiling
  Attempting uninstall: tqdm
    Found existing installation: tqdm 4.65.0
    Uninstalling tqdm-4.65.0:
      Successfully uninstalled tqdm-4.65.0
  Attempting uninstall: scipy
    Found existing installation: scipy 1.10.1
    Uninstalling scipy-1.10.1:
      Successfully uninstalled scipy-1.10.1
  Attempting uninstall: matplotlib
    Found existing installation: matplotlib 3.7.1
    Uninstalling matplotlib-3.7.1:
      Successfully uninstalled matplotlib-3.7.1
Successfully installed htmlmin-0.1.12 imagehash-4.3.1 matplotlib-3.6.3
multimethod-1.9.1 pandas-profiling-3.6.6 phik-0.12.3 scipy-1.9.3
tangled-up-in-unicode-0.2.0 tqdm-4.64.1 typeguard-2.13.3 visions-0.7.5
ydata-profiling-4.1.2
```

```
{"pip_warning":{"packages":["matplotlib","mpl_toolkits"]}}
```

```
import pandas_profiling
```

```
<ipython-input-11-6a00893fb3e1>:1: DeprecationWarning: `import
pandas_profiling` is going to be deprecated by April 1st. Please use
`import ydata_profiling` instead.
  import pandas_profiling
```

```
from google.colab import files


uploaded = files.upload()
```

```
<IPython.core.display.HTML object>

Saving Admission_Prediction.csv to Admission_Prediction.csv
```

```
df = pd.read_csv('Admission_Prediction.csv')
```

```
df
```

```
     Serial No.  GRE Score  TOEFL Score  University Rating  SOP  LOR
CGPA  \
0             1      337.0        118.0                4.0  4.5  4.5
```

```
9.65
1             2      324.0        107.0                4.0  4.0  4.5
8.87
2             3        NaN        104.0                3.0  3.0  3.5
8.00
3             4      322.0        110.0                3.0  3.5  2.5
8.67
4             5      314.0        103.0                2.0  2.0  3.0
8.21
..          ...        ...          ...                ...  ...  ...
...
495         496      332.0        108.0                5.0  4.5  4.0
9.02
496         497      337.0        117.0                5.0  5.0  5.0
9.87
497         498      330.0        120.0                5.0  4.5  5.0
9.56
498         499      312.0        103.0                4.0  4.0  5.0
8.43
499         500      327.0        113.0                4.0  4.5  4.5
9.04

     Research  Chance of Admit
0           1              0.92
1           1              0.76
2           1              0.72
3           1              0.80
4           0              0.65
..        ...               ...
495         1              0.87
496         1              0.96
497         1              0.93
498         0              0.73
499         0              0.84

[500 rows x 9 columns]
```

```
from pandas_profiling import ProfileReport

ProfileReport(df, title='Admission Prediction', explorative=True)
```


```
{"model_id":"f6388514c43845fd9a3110160b2dc394","version_major":2,"vers
ion_minor":0}

{"model_id":"2e6e01e52daf43cfa4af67612efa63b1","version_major":2,"vers
ion_minor":0}

{"model_id":"7d33f9356c5041b7b81398cf9f3bbdc0","version_major":2,"vers
ion_minor":0}
```


```
<IPython.core.display.HTML object>
```

```
pf = ProfileReport(df)
```

```
df
```

```
     Serial No.  GRE Score  TOEFL Score  University Rating  SOP  LOR
CGPA  \
0             1      337.0        118.0                4.0  4.5  4.5
9.65
1             2      324.0        107.0                4.0  4.0  4.5
8.87
2             3        NaN        104.0                3.0  3.0  3.5
8.00
3             4      322.0        110.0                3.0  3.5  2.5
8.67
4             5      314.0        103.0                2.0  2.0  3.0
8.21
..          ...        ...          ...                ...  ...  ...
...
495         496      332.0        108.0                5.0  4.5  4.0
9.02
496         497      337.0        117.0                5.0  5.0  5.0
9.87
497         498      330.0        120.0                5.0  4.5  5.0
9.56
498         499      312.0        103.0                4.0  4.0  5.0
8.43
499         500      327.0        113.0                4.0  4.5  4.5
9.04

     Research  Chance of Admit
0           1             0.92
1           1             0.76
2           1             0.72
3           1             0.80
4           0             0.65
..        ...              ...
495         1             0.87
496         1             0.96
497         1             0.93
498         0             0.73
499         0             0.84

[500 rows x 9 columns]
```

```
pf.to_widgets()
```

```
/usr/local/lib/python3.9/dist-packages/pandas_profiling/
profile_report.py:457: UserWarning: Ipywidgets is not yet fully
supported on Google Colab
```

```
(https://github.com/googlecolab/colabtools/issues/60).As an
alternative, you can use the HTML report. See the documentation for
more information.
  warnings.warn(
```


```
{"model_id":"b847027f172b44fa9c115aa7e093bd3f","version_major":2,"vers
ion_minor":0}
```

```
{"model_id":"de8718a35d6344b0a449ed5a78a2f3de","version_major":2,"vers
ion_minor":0}
```

```
{"model_id":"4c6b7507cc6b45ee9ab5e875941110c0","version_major":2,"vers
ion_minor":0}
```


```
pf.to_file(output_file='Advertising.html')
```


```
{"model_id":"1c31749e6f9f4572a8e0dfcdc76c56c4","version_major":2,"vers
ion_minor":0}
```

```
{"model_id":"a117eab392d34d169f841bf11e02bd76","version_major":2,"vers
ion_minor":0}
```


```
pf.to_notebook_iframe()
```

```
<IPython.core.display.HTML object>
```

```
!pip install pyyaml==5.4.1
```

```
Looking in indexes: https://pypi.org/simple, https://us-
python.pkg.dev/colab-wheels/public/simple/
Collecting pyyaml==5.4.1
  Downloading PyYAML-5.4.1-cp39-cp39-manylinux1_x86_64.whl (630 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 630.1/630.1 kB 9.8 MB/s eta
0:00:00
l
  Attempting uninstall: pyyaml
    Found existing installation: PyYAML 6.0
    Uninstalling PyYAML-6.0:
      Successfully uninstalled PyYAML-6.0
Successfully installed pyyaml-5.4.1
```


```
{"pip_warning":{"packages":["yaml"]}}
```


Dealing with Missing Values

```
df['GRE Score'] = df['GRE Score'].fillna(df['GRE Score'].mean())
```

```
df['TOEFL Score'] = df['TOEFL Score'].fillna(df['TOEFL Score'].mean())
```

```
df['University Rating'] = df['University
Rating'].fillna(df['University Rating'].mean())
```

```
df.describe()
```

```
       Serial No.   GRE Score  TOEFL Score  University Rating
SOP  \
count  500.000000  500.000000   500.000000         500.000000
500.000000
mean   250.500000  316.558763   107.187755           3.121649
3.374000
std    144.481833   11.103952     6.051338           1.128802
0.991004
min      1.000000  290.000000    92.000000           1.000000
1.000000
25%    125.750000  309.000000   103.000000           2.000000
2.500000
50%    250.500000  316.558763   107.000000           3.000000
3.500000
75%    375.250000  324.000000   112.000000           4.000000
4.000000
max    500.000000  340.000000   120.000000           5.000000
5.000000

             LOR        CGPA    Research  Chance of Admit
count  500.00000  500.000000  500.000000        500.00000
mean     3.48400    8.576440    0.560000          0.72174
std      0.92545    0.604813    0.496884          0.14114
min      1.00000    6.800000    0.000000          0.34000
25%      3.00000    8.127500    0.000000          0.63000
50%      3.50000    8.560000    1.000000          0.72000
75%      4.00000    9.040000    1.000000          0.82000
max      5.00000    9.920000    1.000000          0.97000
```

```
df.isnull().sum()        #To check the number of missing values
```

```
Serial No.           0
GRE Score            0
TOEFL Score          0
University Rating    0
SOP                  0
LOR                  0
CGPA                 0
Research             0
Chance of Admit      0
dtype: int64
```

```
df.drop(columns=['Serial No.'],inplace=True)  # To delete the column
```

```
df
```

```
      GRE Score  TOEFL Score  University Rating  SOP  LOR  CGPA
Research  \
0    337.000000        118.0                4.0  4.5  4.5  9.65
```

```
1
1    324.000000        107.0                4.0  4.0  4.5  8.87
1
2    316.558763        104.0                3.0  3.0  3.5  8.00
1
3    322.000000        110.0                3.0  3.5  2.5  8.67
1
4    314.000000        103.0                2.0  2.0  3.0  8.21
0
..          ...          ...                ...  ...  ...   ...
...
495  332.000000        108.0                5.0  4.5  4.0  9.02
1
496  337.000000        117.0                5.0  5.0  5.0  9.87
1
497  330.000000        120.0                5.0  4.5  5.0  9.56
1
498  312.000000        103.0                4.0  4.0  5.0  8.43
0
499  327.000000        113.0                4.0  4.5  4.5  9.04
0

     Chance of Admit
0               0.92
1               0.76
2               0.72
3               0.80
4               0.65
..               ...
495             0.87
496             0.96
497             0.93
498             0.73
499             0.84

[500 rows x 8 columns]
```

Building A Model

```
y = df['Chance of Admit']    #y label is the chance of admit because
the model is going to predict the chance od admission of a student
according to his Chance_of_Admit Column

x = df.drop(columns = ['Chance of Admit'])   # dropping the
chance_of_admit column as it is the y label of our model

x
```

```
      GRE Score  TOEFL Score  University Rating  SOP  LOR  CGPA
Research
0    337.000000        118.0                4.0  4.5  4.5  9.65
```

```
1
1    324.000000        107.0                4.0  4.0  4.5  8.87
1
2    316.558763        104.0                3.0  3.0  3.5  8.00
1
3    322.000000        110.0                3.0  3.5  2.5  8.67
1
4    314.000000        103.0                2.0  2.0  3.0  8.21
0
..          ...          ...                ...  ...  ...   ...
...
495  332.000000        108.0                5.0  4.5  4.0  9.02
1
496  337.000000        117.0                5.0  5.0  5.0  9.87
1
497  330.000000        120.0                5.0  4.5  5.0  9.56
1
498  312.000000        103.0                4.0  4.0  5.0  8.43
0
499  327.000000        113.0                4.0  4.5  4.5  9.04
0

[500 rows x 7 columns]
```

```
y
```

```
0      0.92
1      0.76
2      0.72
3      0.80
4      0.65
       ...
495    0.87
496    0.96
497    0.93
498    0.73
499    0.84
Name: Chance of Admit, Length: 500, dtype: float64
```

As the column varies therefore using standard scaler and it will allow the model to understand the feature and label

```
scaler = StandardScaler()
```

```
arr = scaler.fit_transform(x)
```

```
arr
```

```
array([[ 1.84274116e+00,  1.78854223e+00,  7.78905651e-01, ...,
         1.09894429e+00,  1.77680627e+00,  8.86405260e-01],
       [ 6.70814288e-01, -3.10581135e-02,  7.78905651e-01, ...,
```

```
         1.09894429e+00,  4.85859428e-01,  8.86405260e-01],
       [ 5.12433309e-15, -5.27312752e-01, -1.07876604e-01, ...,
         1.73062093e-02, -9.54042814e-01,  8.86405260e-01],
       ...,
       [ 1.21170361e+00,  2.11937866e+00,  1.66568791e+00, ...,
         1.63976333e+00,  1.62785086e+00,  8.86405260e-01],
       [-4.10964364e-01, -6.92730965e-01,  7.78905651e-01, ...,
         1.63976333e+00, -2.42366993e-01, -1.12815215e+00],
       [ 9.41258951e-01,  9.61451165e-01,  7.78905651e-01, ...,
         1.09894429e+00,  7.67219636e-01, -1.12815215e+00]])

df1 = pd.DataFrame(arr)

df1.profile_report()
```


```
{"model_id":"5888a5fdcc4d4cc18745856b42fb0ff6","version_major":2,"vers
ion_minor":0}

{"model_id":"27ed66f479d14a959c8d6e0e4a9d6e30","version_major":2,"vers
ion_minor":0}

{"model_id":"fe3962795bdf4e9e9b2ea9fc2f3c0270","version_major":2,"vers
ion_minor":0}
```


```
<IPython.core.display.HTML object>
```

```
df1
```

```
                0         1         2         3         4         5
6
0    1.842741e+00  1.788542  0.778906  1.137360  1.098944  1.776806
0.886405
1    6.708143e-01 -0.031058  0.778906  0.632315  1.098944  0.485859
0.886405
2    5.124333e-15 -0.527313 -0.107877 -0.377773  0.017306 -0.954043
0.886405
3    4.905178e-01  0.465197 -0.107877  0.127271 -1.064332  0.154847
0.886405
4   -2.306679e-01 -0.692731 -0.994659 -1.387862 -0.523513 -0.606480 -
1.128152
..            ...       ...       ...       ...       ...       ...
...
495  1.392000e+00  0.134360  1.665688  1.137360  0.558125  0.734118
0.886405
496  1.842741e+00  1.623124  1.665688  1.642404  1.639763  2.140919
0.886405
497  1.211704e+00  2.119379  1.665688  1.137360  1.639763  1.627851
0.886405
498 -4.109644e-01 -0.692731  0.778906  0.632315  1.639763 -0.242367 -
1.128152
```

```
499  9.412590e-01  0.961451  0.778906  1.137360  1.098944  0.767220 -
1.128152

[500 rows x 7 columns]
```

```
df1.describe()
```

```
                  0             1             2             3
4  \
count  5.000000e+02  5.000000e+02  5.000000e+02  5.000000e+02
5.000000e+02
mean   4.384049e-15  9.521273e-16  3.979039e-16 -8.526513e-17
4.263256e-17
std    1.001002e+00  1.001002e+00  1.001002e+00  1.001002e+00
1.001002e+00
min   -2.394225e+00 -2.512331e+00 -1.881441e+00 -2.397950e+00 -
2.686789e+00
25%   -6.814090e-01 -6.927310e-01 -9.946589e-01 -8.828175e-01 -
5.235128e-01
50%    5.124333e-15 -3.105811e-02 -1.078766e-01  1.272712e-01
1.730621e-02
75%    6.708143e-01  7.960330e-01  7.789057e-01  6.323155e-01
5.581253e-01
max    2.113186e+00  2.119379e+00  1.665688e+00  1.642404e+00
1.639763e+00

                  5             6
count  5.000000e+02  5.000000e+02
mean   3.119283e-15 -7.815970e-17
std    1.001002e+00  1.001002e+00
min   -2.940115e+00 -1.128152e+00
25%   -7.430227e-01 -1.128152e+00
50%   -2.720919e-02  8.864053e-01
75%    7.672196e-01  8.864053e-01
max    2.223672e+00  8.864053e-01
```

Checking MultiColinearity

Importing variance inflation sctor

```
from statsmodels.stats.outliers_influence import
variance_inflation_factor
```

```
vif_df = pd.DataFrame()
```

```
vif_df['vif'] = [variance_inflation_factor(arr,i) for i in range
(arr.shape[1])]
```

```
vif_df['feature']=x.columns
```

```
vif_df
```

```
        vif            feature
0  4.153268          GRE Score
1  3.792866         TOEFL Score
2  2.508768  University Rating
3  2.775750                SOP
4  2.037308                LOR
5  4.651670               CGPA
6  1.459311           Research
```

```
arr
```

```
array([[ 1.84274116e+00,  1.78854223e+00,  7.78905651e-01, ...,
         1.09894429e+00,  1.77680627e+00,  8.86405260e-01],
       [ 6.70814288e-01, -3.10581135e-02,  7.78905651e-01, ...,
         1.09894429e+00,  4.85859428e-01,  8.86405260e-01],
       [ 5.12433309e-15, -5.27312752e-01, -1.07876604e-01, ...,
         1.73062093e-02, -9.54042814e-01,  8.86405260e-01],
       ...,
       [ 1.21170361e+00,  2.11937866e+00,  1.66568791e+00, ...,
         1.63976333e+00,  1.62785086e+00,  8.86405260e-01],
       [-4.10964364e-01, -6.92730965e-01,  7.78905651e-01, ...,
         1.63976333e+00, -2.42366993e-01, -1.12815215e+00],
       [ 9.41258951e-01,  9.61451165e-01,  7.78905651e-01, ...,
         1.09894429e+00,  7.67219636e-01, -1.12815215e+00]])
```

```
x_train, x_test, y_train, y_test = train_test_split(arr, y, test_size
= 0.25)
```

```
x_train
```

```
array([[ 3.10221404e-01,  6.30614739e-01,  3.93810431e-16, ...,
         1.73062093e-02,  4.85859428e-01,  8.86405260e-01],
       [ 4.00369625e-01, -8.58149178e-01, -1.07876604e-01, ...,
         5.58125251e-01,  7.17567835e-01,  8.86405260e-01],
       [-7.71557248e-01, -8.58149178e-01, -9.94658860e-01, ...,
         1.73062093e-02, -9.87144015e-01,  8.86405260e-01],
       ...,
       [-1.22229835e+00, -1.51982203e+00, -1.88144112e+00, ...,
        -1.06433187e+00, -1.53331383e+00, -1.12815215e+00],
       [-3.20816143e-01, -8.58149178e-01, -1.07876604e-01, ...,
        -5.23512832e-01, -5.07176601e-01, -1.12815215e+00],
       [ 1.57229650e+00,  1.12686938e+00,  7.78905651e-01, ...,
         5.58125251e-01,  1.41269305e+00,  8.86405260e-01]])
```

```
lr = LinearRegression()
```

```
lr.fit(x_train, y_train)
```

```
LinearRegression()
```

```
pickle.dump(lr, open('admission_lr_model.pickle', 'wb'))
```

```
!ls
```

```
 admission_lr_model.pickle   Advertising.html
 Admission_Prediction.csv    sample_data
 advertising.csv     'Spend-20211123T045440Z-001 (1).zip'
```

```
model = pickle.load(open('admission_lr_model.pickle', 'rb'))
```

```
lr.predict([[337.000000,   118.0,     4.0,  4.5,  4.5,  9.65, 1]])
```

```
array([10.42783509])
```

```
test1 = scaler.transform([[337.000000,118.0,     4.0,  4.5,  4.5,  9.65,
     1]])
```

```
/usr/local/lib/python3.9/dist-packages/sklearn/base.py:439:
UserWarning: X does not have valid feature names, but StandardScaler
was fitted with feature names
  warnings.warn(
```

```
model.predict(test1)
```

```
array([0.95014403])
```

```
lr.predict(test1)
```

```
array([0.95014403])
```

Loading Unkown Dataset

```
lr.score(x_test, y_test)
```

```
0.85234261619116
```

```
def adj_r2(x,y):
  r2 = lr.score(x,y)
  n = x.shape[0]
  p = x.shape[1]
  adjusted_r2 = 1-(1-r2)*(n-1)/(n-p-1)
  return adjusted_r2
```

```
adj_r2(x_test, y_test)
```

```
0.8435084137410584
```

Lasso CV

```
lassocv = LassoCV(alphas = None, cv = 10, max_iter = 200000)
```

```
elastic = ElasticNetCV(alphas = None, cv = 10)
```

```
elastic.fit(x_train, y_train)
```

```
ElasticNetCV(cv=10)
```

```
elastic.alpha_
```

```
0.0005086960795153597
```

```
elastic.l1_ratio_
```

```
0.5
```

```
elastic_lr = ElasticNet(alpha = elastic.alpha_, l1_ratio =
elastic.l1_ratio_)
```

```
elastic_lr.fit(x_train, y_train)
```

```
ElasticNet(alpha=0.0005086960795153597)
```

```
elastic_lr.score(x_test, y_test)
```

```
0.8520663417333796
```