5rfcuhliq

February 5, 2025

1 Machine Learning Part 2

[2]:
```
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import FunctionTransformer
from mlxtend.feature_selection import SequentialFeatureSelector
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression


data = pd.read_csv("/Users/ratnadeepgurav/Desktop/AIDS/Study for carrer␣
 ↪material/WSCUBE_Data_Analyst/ML/loan.csv")
data
```

[2]:
```
      Loan_ID  Gender Married Dependents     Education Self_Employed  \
0    LP001002    Male      No          0      Graduate            No
1    LP001003    Male     Yes          1      Graduate            No
2    LP001005    Male     Yes          0      Graduate           Yes
3    LP001006    Male     Yes          0  Not Graduate            No
4    LP001008    Male      No          0      Graduate            No
..        …     …     …        …           …           …
609  LP002978  Female      No          0      Graduate            No
610  LP002979    Male     Yes         3+      Graduate            No
611  LP002983    Male     Yes          1      Graduate            No
612  LP002984    Male     Yes          2      Graduate            No
613  LP002990  Female      No          0      Graduate           Yes

     ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  \
0               5849                0.0         NaN             360.0
1               4583             1508.0       128.0             360.0
2               3000                0.0        66.0             360.0
3               2583             2358.0       120.0             360.0
4               6000                0.0       141.0             360.0
```

```
..               …                …         …               …
609             2900                0.0        71.0             360.0
610             4106                0.0        40.0             180.0
611             8072              240.0       253.0             360.0
612             7583                0.0       187.0             360.0
613             4583                0.0       133.0             360.0

     Credit_History Property_Area Loan_Status
0                1.0         Urban           Y
1                1.0         Rural           N
2                1.0         Urban           Y
3                1.0         Urban           Y
4                1.0         Urban           Y
..              …           …         …
609              1.0         Rural           Y
610              1.0         Rural           Y
611              1.0         Urban           Y
612              1.0         Urban           Y
613              0.0     Semiurban           N

[614 rows x 13 columns]
```

2 Feature Scaling Normalization

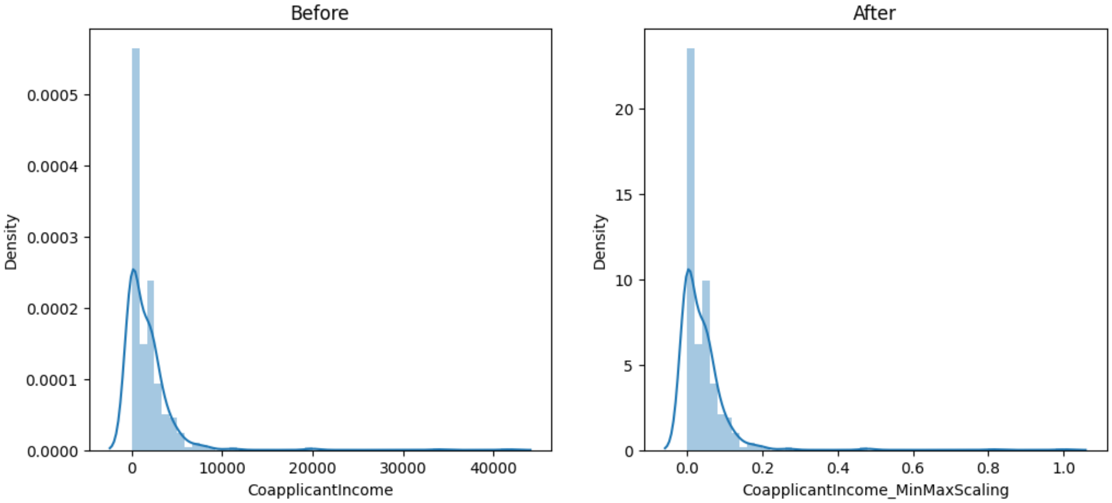
[75]:
```
ms = MinMaxScaler()

ms.fit(data[["CoapplicantIncome"]])

data['CoapplicantIncome_MinMaxScaling'] = ms.
 ↪transform(data[['CoapplicantIncome']])

data.head(3)
```

[75]:
```
    Loan_ID Gender Married Dependents Education Self_Employed  \
0  LP001002    Male      No          0  Graduate            No
1  LP001003    Male     Yes          1  Graduate            No
2  LP001005    Male     Yes          0  Graduate           Yes

   ApplicantIncome  CoapplicantIncome  LoanAmount  Loan_Amount_Term  \
0              5849                0.0         NaN             360.0
1              4583             1508.0       128.0             360.0
2              3000                0.0        66.0             360.0

   Credit_History Property_Area Loan_Status  CoapplicantIncome_MinMaxScaling
0             1.0         Urban           Y                         0.000000
1             1.0         Rural           N                         0.036192
```

```
2              1.0          Urban           Y                             0.000000
```

[76]:
```
plt.figure(figsize=(12, 5))

# Before
plt.subplot(1, 2, 1)
plt.title("Before")
sns.distplot(data["CoapplicantIncome"], kde=True)

# After
plt.subplot(1, 2, 2)
plt.title("After")
sns.distplot(data["CoapplicantIncome_MinMaxScaling"], kde=True)

plt.show()
```

```
/var/folders/c_/rbrshmgx64b9ch2skklfhfbw0000gn/T/ipykernel_1549/2589777655.py:6:
UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data["CoapplicantIncome"], kde=True)
/var/folders/c_/rbrshmgx64b9ch2skklfhfbw0000gn/T/ipykernel_1549/2589777655.py:11
: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data["CoapplicantIncome_MinMaxScaling"], kde=True)
```



3 Handle Duplicate Data

[77]:
```
data.shape
```

[77]:
```
(614, 14)
```

[27]:
```
data.drop_duplicates(inplace = True)
```

[30]:
```
data.shape
```

[30]:
```
(614, 14)
```

4 Replace Data Types

[34]:
```
data.isnull().sum()
```

[34]:
```
Loan_ID                      0
Gender                      13
Married                      3
Dependents                  15
Education                    0
Self_Employed               32
ApplicantIncome              0
CoapplicantIncome            0
LoanAmount                  22
Loan_Amount_Term            14
Credit_History              50
Property_Area                0
```

```
Loan_Status                     0
CoapplicantIncome_MinMaxScaling     0
dtype: int64
```

[40]:
```
data['Dependents'].value_counts()
```

[40]:
```
Dependents
0     360
1     102
2     101
3+     51
Name: count, dtype: int64
```

[57]:
```
data["Dependents"].fillna(data["Dependents"].mode()[0],inplace = True)

data["Dependents"].replace("3+","3",inplace = True)

print("\n\n Values in data :- \n\n",data["Dependents"].value_counts(),"\n\n")

data["Dependents"] = data["Dependents"].astype("int64")

data.info()
```

```


 Values in data :-

 Dependents
0    360
1    102
2    101
3     51
Name: count, dtype: int64


<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 14 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   Loan_ID                         614 non-null    object
 1   Gender                          601 non-null    object
 2   Married                         611 non-null    object
 3   Dependents                      614 non-null    int64
 4   Education                       614 non-null    object
 5   Self_Employed                   582 non-null    object
 6   ApplicantIncome                 614 non-null    int64
 7   CoapplicantIncome               614 non-null    float64
 8   LoanAmount                      592 non-null    float64
 9   Loan_Amount_Term                600 non-null    float64
 10  Credit_History                  564 non-null    float64
 11  Property_Area                   614 non-null    object
 12  Loan_Status                     614 non-null    object
 13  CoapplicantIncome_MinMaxScaling 614 non-null    float64
dtypes: float64(5), int64(2), object(7)
memory usage: 67.3+ KB
```

```
/var/folders/c_/rbrshmgx64b9ch2skklfhfbw0000gn/T/ipykernel_1549/3797479550.py:1:
FutureWarning: A value is trying to be set on a copy of a DataFrame or Series
through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work
because the intermediate object on which we are setting values always behaves as
a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using
'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value)
instead, to perform the operation inplace on the original object.


  data["Dependents"].fillna(data["Dependents"].mode()[0],inplace = True)
/var/folders/c_/rbrshmgx64b9ch2skklfhfbw0000gn/T/ipykernel_1549/3797479550.py:3:
FutureWarning: A value is trying to be set on a copy of a DataFrame or Series
through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work
because the intermediate object on which we are setting values always behaves as
a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using
'df.method({col: value}, inplace=True)' or df[col] = df[col].method(value)
instead, to perform the operation inplace on the original object.


  data["Dependents"].replace("3+","3",inplace = True)
```

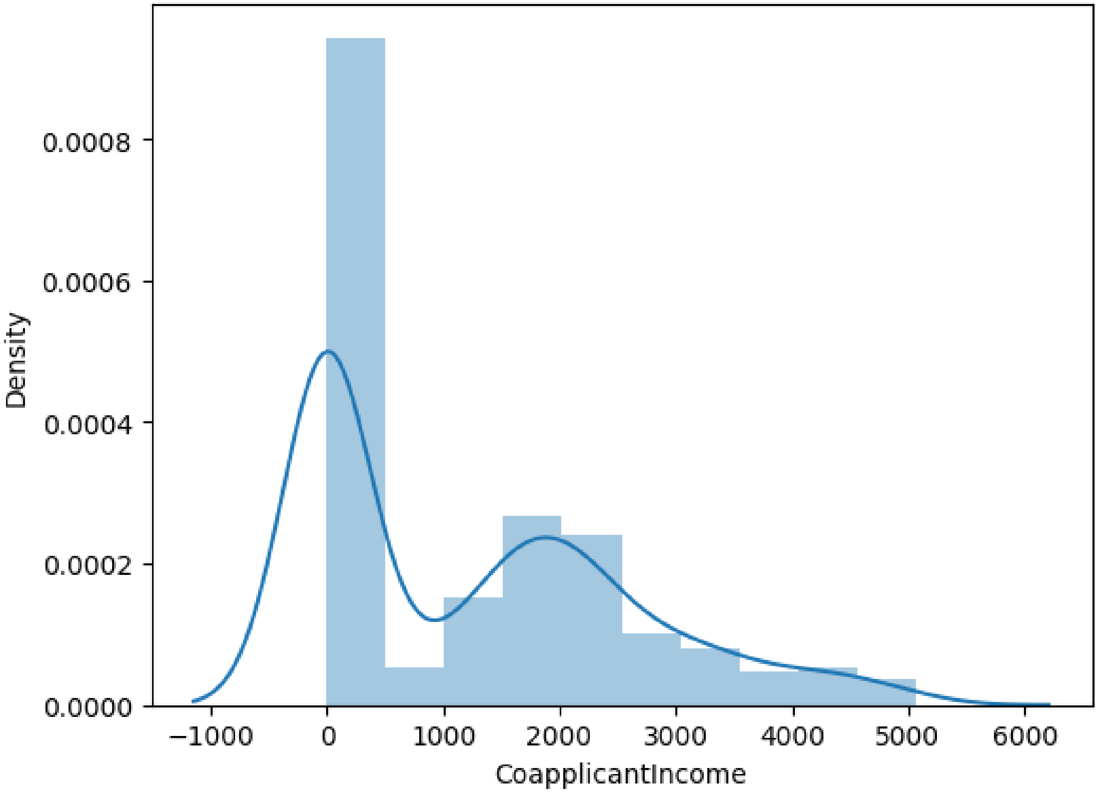
5 Function Transformer (Convert Non-distrubute to distrubute)

[91]:

```
sns.distplot(data['CoapplicantIncome'])
plt.show()
```

```
/var/folders/c_/rbrshmgx64b9ch2skklfhfbw0000gn/T/ipykernel_1549/1526187818.py:1:
UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.
```

```
Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data['CoapplicantIncome'])
```



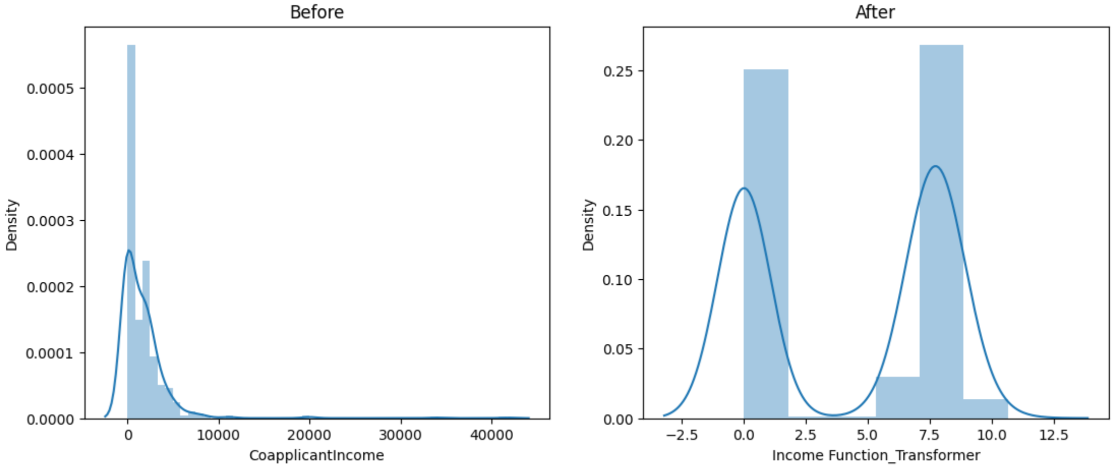
[90]:
```
q1 = data['CoapplicantIncome'].quantile(0.25)
q3 = data['CoapplicantIncome'].quantile(0.75)

IQR = q3 - q1

min_range = q1 - (1.25 * IQR)
max_range = q3 + (1.25 * IQR)

min_range,max_range

data = data[data["CoapplicantIncome"]<max_range]
```

6 Use Function transformer

[103]:

```
ft = FunctionTransformer(func = np.log1p)

ft.fit(data[["CoapplicantIncome"]])
data["Income Function_Transformer"] =  ft.transform(data[["CoapplicantIncome"]])

plt.figure(figsize = (13,5))

plt.subplot(1,2,1)
sns.distplot(data["CoapplicantIncome"])
plt.title("Before")


plt.subplot(1,2,2)
sns.distplot(data["Income Function_Transformer"])
plt.title("After")

plt.show()
```

```
/var/folders/c_/rbrshmgx64b9ch2skklfhfbw0000gn/T/ipykernel_1549/3596817702.py:9:
UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data["CoapplicantIncome"])
/var/folders/c_/rbrshmgx64b9ch2skklfhfbw0000gn/T/ipykernel_1549/3596817702.py:14
: UserWarning:

`distplot` is a deprecated function and will be removed in seaborn v0.14.0.

Please adapt your code to use either `displot` (a figure-level function with
similar flexibility) or `histplot` (an axes-level function for histograms).

For a guide to updating your code to use the new functions, please see
https://gist.github.com/mwaskom/de44147ed2974457ad6372750bbe5751

  sns.distplot(data["Income Function_Transformer"])
```



7 Feature Selection Technique

8 Forword Elimination

[15]:

```
data1 = pd.read_csv("/Users/ratnadeepgurav/Desktop/AIDS/Study for carrer␣
 ↪material/WSCUBE_Data_Analyst/ML/diabetes.csv")

x = data1.iloc[:,:-1]
y = data1["Outcome"]

print(x.shape)

lr = LogisticRegression()

fs = SequentialFeatureSelector(lr,k_features = 5 , forward = True)
fs.fit(x,y)

print("\n\n Features :- ",fs.feature_names,"\n\n")
print("\n\n K - Features :-",fs.k_feature_names_,"\n\n")
print("\n\n K - Score :- ",fs.k_score_,"\n\n")
```

```
(768, 8)


 Features :-  ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness',
'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age']
```

```
K - Features :- ('Pregnancies', 'Glucose', 'Insulin', 'BMI', 'Age')
```

```
K - Score :-  0.7708768355827178
```

9 Regression Analysis (Supervised Learning)

[6]:

```
data_clg = pd.read_csv("/Users/ratnadeepgurav/Desktop/AIDS/Study for carrer␣
 ↪material/WSCUBE_Data_Analyst/ML/placement.csv")

data_clg.drop(columns = ["iq"],inplace = True)

data_clg.head(3)
```

[6]:

```
   Unnamed: 0  cgpa  Package
0           0   6.8     3.12
1           1   5.9     2.00
2           2   5.3     1.80
```

[8]:

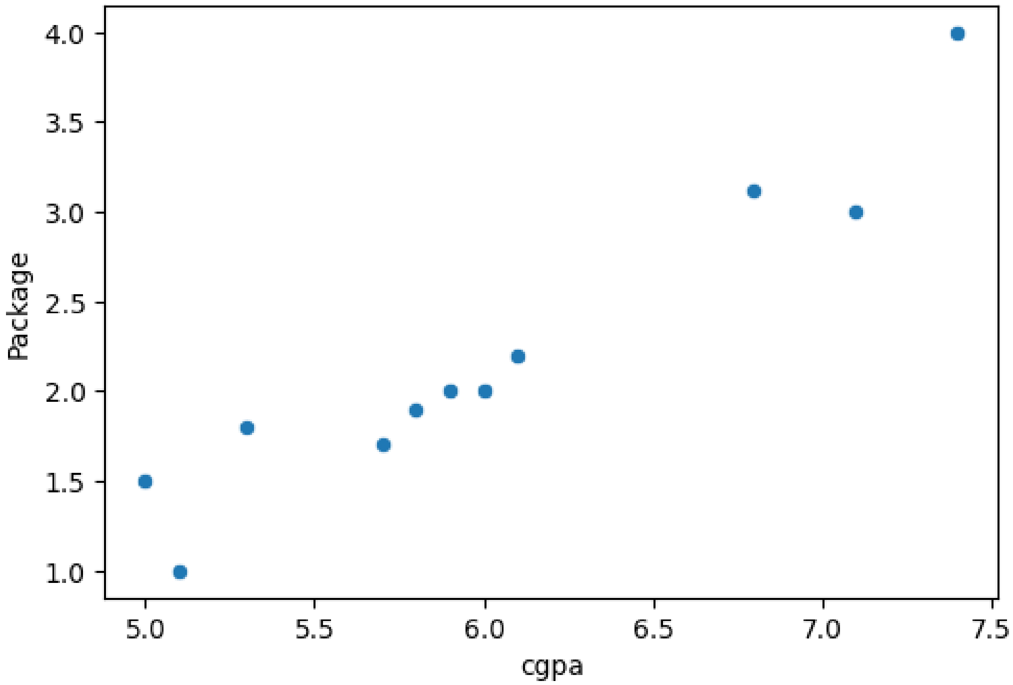
```
data_clg.isnull().sum()
```

[8]:

```
Unnamed: 0    0
cgpa          0
Package       0
dtype: int64
```

[13]:

```
plt.figure(figsize = (6,4))
sns.scatterplot(x = "cgpa", y = "Package",data = data_clg)
plt.show()
```



[63]:

```
x1 = data_clg[["cgpa"]]
y1 = data_clg["Package"]

x_train,x_test,y_train,y_test = train_test_split(x1,y1,test_size=0.
 ↪2,random_state = 42)


lr = LinearRegression()

lr.fit(x_train,y_train)

lr.predict([[5.3]])

lr.score(x_test,y_test)*100
```

```
/opt/homebrew/Cellar/jupyterlab/4.2.3/libexec/lib/python3.12/site-
packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid
feature names, but LinearRegression was fitted with feature names
  warnings.warn(
```

[63]:

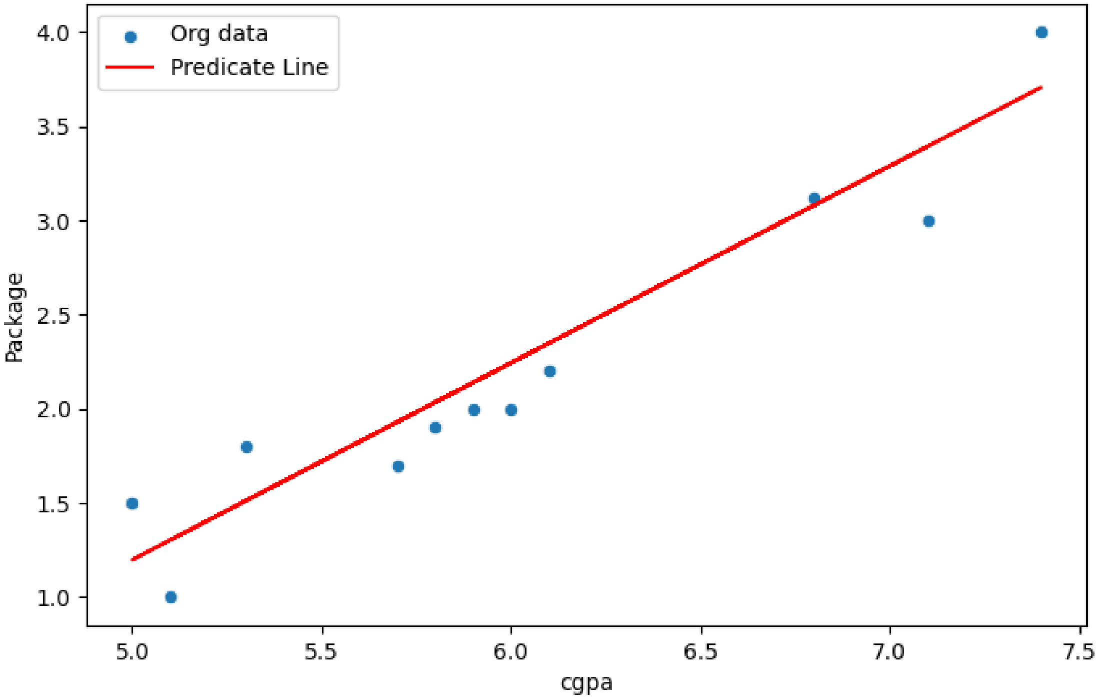
```
91.32265824841723
```

[37]:

```
y_pred = lr.predict(x1)
```

[44]:

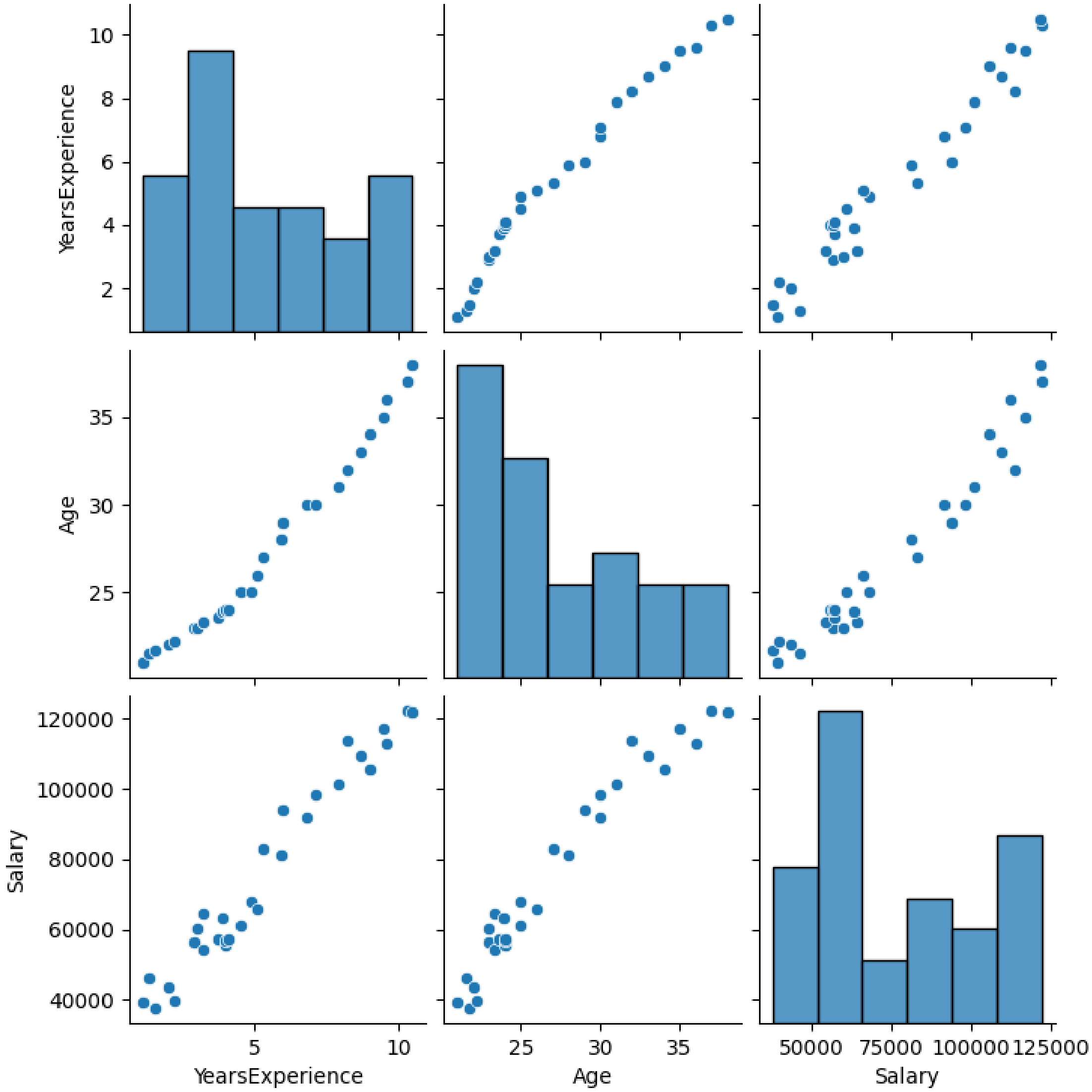
```
plt.figure(figsize = (8,5))
sns.scatterplot(x = "cgpa", y = "Package",data = data_clg)
plt.plot(data_clg["cgpa"],y_pred,c = "red")
plt.legend(["Org data","Predicate Line"])
plt.show()
```



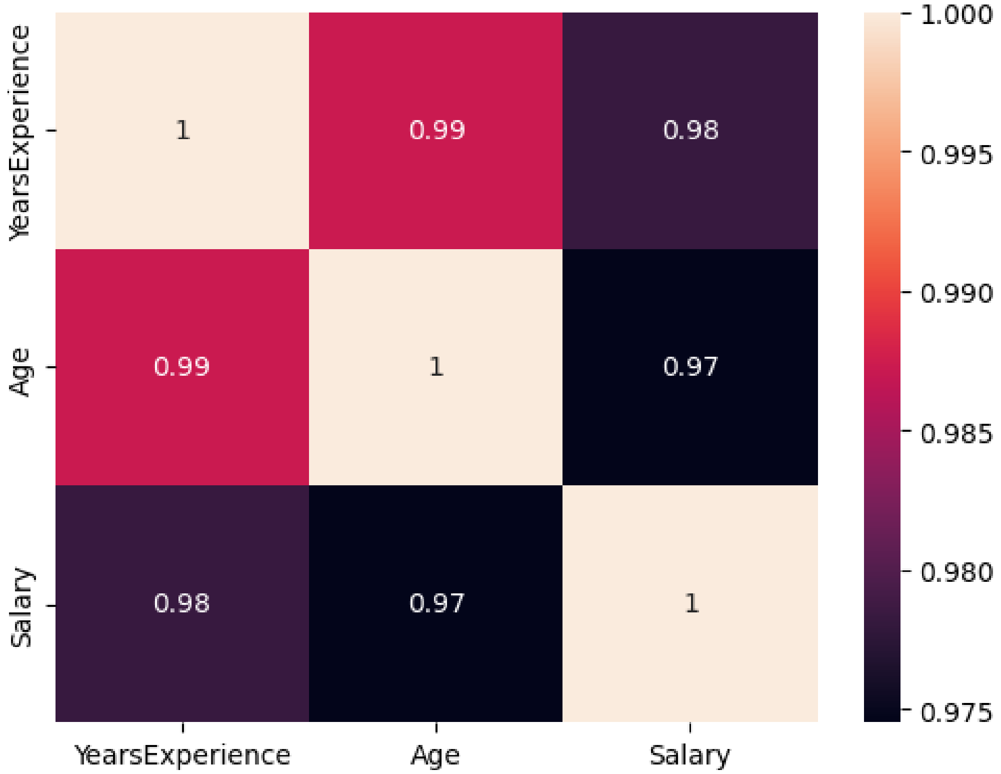
10 Multiple Linear Regression

[3]:

```
data3 = pd.read_csv("/Users/ratnadeepgurav/Desktop/AIDS/Study for carrer␣
 ↪material/WSCUBE_Data_Analyst/ML/Salary_Data.csv")

data3.isnull().sum()

sns.pairplot(data = data3)
plt.show()
```



[4]:
```
sns.heatmap(data=data3.corr(),annot = True)
plt.show()
```



```
[5]: x3 = data3.iloc[:,:-1]
     y3 = data3["Salary"]

     x_train,x_test,y_train,y_test = train_test_split(x3,y3,test_size=0.
      ↪2,random_state = 42)

     lr = LinearRegression()

     lr.fit(x_train,y_train)

     lr.score(x_test,y_test)*100

     lr.predict(x3)
```

```
[5]: array([ 38675.56314937,  40935.75217728,  42425.68560928,  45637.01545868,
             47126.94889069,  52598.467768  ,  53086.6826187 ,  54833.367916  ,
             54833.367916  ,  58044.6977654 ,  59791.38306271,  60536.34977871,
             60536.34977871,  61024.56462941,  65544.94268522,  67497.80208802,
             71041.75044243,  74585.69879684,  80082.50655405,  83138.24005776,
             89611.47751637,  91076.12206847,  97549.35952708, 101581.52273219,
            106590.1156387 , 110622.27884381, 115630.87175032, 118686.60525402,
            124671.62786194, 128215.57621634])
```

[ ]:

```
```