In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [2]:
```
x=np.array([95,85,80,70,60])
y=np.array([85,95,70,65,70])
```

In [3]:
```
model= np.polyfit(x, y, 1)
model
```

Out[3]:
```
array([ 0.64383562, 26.78082192])
```

In [4]:
```
predict = np.poly1d(model)
predict(65)
```

Out[4]:
```
68.63013698630137
```

In [5]:
```
y_pred= predict(x)
y_pred
```

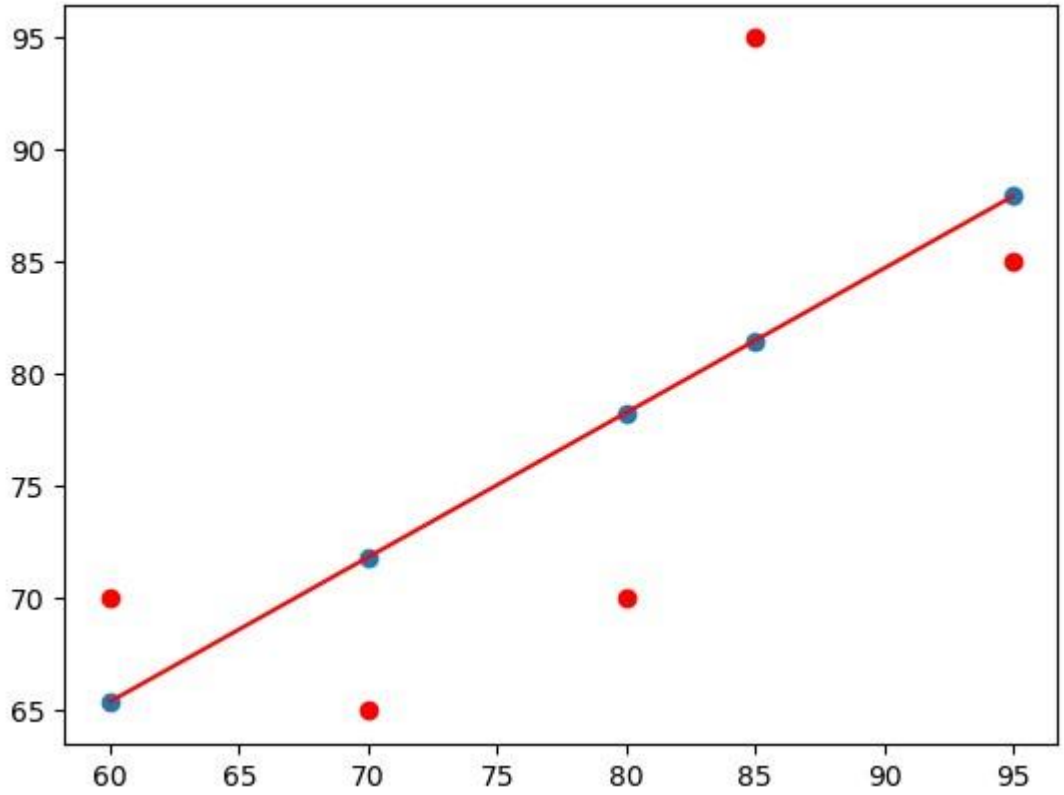
Out[5]:
```
array([87.94520548, 81.50684932, 78.28767123, 71.84931507, 65.4109589 ])
```

In [6]:
```
from sklearn.metrics import r2_score
r2_score(y, y_pred)
```

Out[6]:
```
0.4803218090889326
```

In [7]:
```
y_line = model[1] + model[0]* x
plt.plot(x, y_line, c = 'r')
plt.scatter(x, y_pred)
plt.scatter(x,y,c='r')
```

Out[7]:
```
<matplotlib.collections.PathCollection at 0x2a187f20dd0>
```



In [8]:
```
from sklearn.datasets import load_boston
boston = load_boston()
```

```
---------------------------------------------------------------------------
-
ImportError                               Traceback (most recent call las
t)
Cell In[8], line 1
----> 1 from sklearn.datasets import load_boston
      2 boston = load_boston()

File ~\anaconda3\Lib\site-packages\sklearn\datasets\__init__.py:156, in __
getattr__(name)
    105     if name == "load_boston":
    106         msg = textwrap.dedent(
    107             """
    108             `load_boston` has been removed from scikit-learn since version
 1.2.
    (...)
    154             """
    155         )
--> 156         raise ImportError(msg)
    157     try:
    158         return globals()[name]

ImportError:
`load_boston` has been removed from scikit-learn since version 1.2.

    The Boston housing prices dataset has an ethical problem: as
    investigated in [1], the authors of this dataset engineered a non-
    invertible variable "B" assuming that racial self-segregation had a
    positive impact on house prices [2]. Furthermore the goal of the
    research that led to the creation of this dataset was to study the
    impact of air quality but it did not give adequate demonstration of the
    validity of this assumption.

    The scikit-learn maintainers therefore strongly discourage the use of
    this dataset unless the purpose of the code is to study and educate
    about ethical issues in data science and machine learning.

    In this special case, you can fetch the dataset from the original
    source::

        import pandas as pd
    import numpy as np

        data_url = "http://lib.stat.cmu.edu/datasets/boston"
        raw_df = pd.read_csv(data_url, sep="\s+", skiprows=22, header=None)
    data = np.hstack([raw_df.values[::2, :], raw_df.values[1::2, :2]])
    target = raw_df.values[1::2, 2]

    Alternative datasets include the California housing dataset and the
    Ames housing dataset. You can load the datasets as follows::

        from sklearn.datasets import fetch_california_housing

    housing = fetch_california_housing() for the California

    housing dataset and::

        from sklearn.datasets import fetch_openml
```

```
    housing = fetch_openml(name="house_prices", as_frame=True)
```

for the Ames housing dataset.

[1] M Carlisle.
"Racist data destruction?"
<https://medium.com/@docintangible/racist-data-destruction-113e3eff54a8>

[2] Harrison Jr, David, and Daniel L. Rubinfeld.
"Hedonic housing prices and the demand for clean air." Journal of
environmental economics and management 5.1 (1978): 81-102.
<https://www.researchgate.net/publication/4974606_Hedonic_housing_prices_and_the_demand_for_clean_air>

In [9]:
```
from sklearn.datasets import fetch_california_housing
housing = fetch_california_housing()
```

In [10]:
```
housing
```

Out[10]:

```
{'data': array([[   8.3252    ,   41.        ,    6.98412698, ...,    2.55
555556,
          37.88      , -122.23      ],
       [   8.3014    ,   21.        ,    6.23813708, ...,    2.10984183,
 37.86      , -122.22      ],
       [   7.2574    ,   52.        ,    8.28813559, ...,    2.80225989,
          37.85      , -122.24      ],
       ...,
       [   1.7       ,   17.        ,    5.20554273, ...,    2.3256351 ,
 39.43      , -121.22      ],
       [   1.8672    ,   18.        ,    5.32951289, ...,    2.12320917,
 39.43      , -121.32      ],
       [   2.3886    ,   16.        ,    5.25471698, ...,    2.61698113,
          39.37      , -121.24      ]]),
 'target': array([4.526, 3.585, 3.521, ..., 0.923, 0.847, 0.894]),
 'frame': None,
 'target_names': ['MedHouseVal'],
 'feature_names': ['MedInc',
  'HouseAge',
  'AveRooms',
  'AveBedrms',
  'Population',
  'AveOccup',
  'Latitude',
  'Longitude'],
 'DESCR': '.. _california_housing_dataset:\n\nCalifornia Housing dataset\n
--------------------------\n\n**Data Set Characteristics:**\n\n    :Number
of Instances: 20640\n\n    :Number of Attributes: 8 numeric, predictive at
tributes and the target\n\n    :Attribute Information:\n        - MedInc
median income in block group\n        - HouseAge      median house age in
block group\n        - AveRooms      average number of rooms per household
\n        - AveBedrms     average number of bedrooms per household\n
- Population    block group population\n        - AveOccup      average nu
mber of household members\n        - Latitude      block group latitude\n
- Longitude     block group longitude\n\n    :Missing Attribute Values: No
ne\n\nThis dataset was obtained from the StatLib repository.\nhttps://www.
dcc.fc.up.pt/~ltorgo/Regression/cal_housing.html\n\nThe target variable is
the median house value for California districts,\nexpressed in hundreds of
thousands of dollars ($100,000).\n\nThis dataset was derived from the 1990
U.S. census, using one row per census\nblock group. A block group is the s
mallest geographical unit for which the U.S.\nCensus Bureau publishes samp
le data (a block group typically has a population\nof 600 to 3,000 peopl
e).\n\nA household is a group of people residing within a home. Since the
average\nnumber of rooms and bedrooms in this dataset are provided per hou
sehold, these\ncolumns may take surprisingly large values for block groups
with few households\nand many empty houses, such as vacation resorts.\n\nI
```

In

```
t can be downloaded/loaded using the\n:func:`sklearn.datasets.fetch_califo
rnia_housing` function.\n\n.. topic:: References\n\n    - Pace, R. Kelley
and Ronald Barry, Sparse Spatial Autoregressions,\n      Statistics and Pr
obability Letters, 33 (1997) 291-297\n'}
```

[11]:
```
df = pd.DataFrame(housing.data, columns= housing.feature_names)
df
```

Out[11]:

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitud
0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.2
1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.2
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.2
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.2
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.2
...	...	...	...	...	...	...	...	.
20635	1.5603	25.0	5.045455	1.133333	845.0	2.560606	39.48	-
	121.0							
20636	2.5568	18.0	6.114035	1.315789	356.0	3.122807	39.49	-
	121.2							
20637	1.7000	17.0	5.205543	1.120092	1007.0	2.325635	39.43	-
	121.2							
20638	1.8672	18.0	5.329513	1.171920	741.0	2.123209	39.43	-
	121.3							
20639	2.3886	16.0	5.254717	1.162264	1387.0	2.616981	39.37	-
	121.2							

20640 rows × 8 columns

In [12]:
```
target = housing.target_names
target
```

Out[12]:
```
['MedHouseVal']
```

In [13]:
```
data1 = pd.DataFrame(data = np.c_[housing ['data'], housing['target']],
columns = housing ['feature_names'] + ['target'])
```

In [14]:
```
from sklearn.datasets import fetch_openml housing =
fetch_openml(name = 'house_prices', as_frame = True)
```

```
C:\Users\SSOS19\anaconda3\Lib\site-packages\sklearn\datasets\_openml.py:96
8: FutureWarning: The default value of `parser` will change from `'liac-ar
ff'` to `'auto'` in 1.4. You can set `parser='auto'` to silence this warni
```

In

```
ng. Therefore, an `ImportError` will be raised from 1.4 if the dataset is
dense and pandas is not installed. Note that the pandas parser may return
different data types. See the Notes Section in fetch_openml's API doc for
```

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitud
0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.2

```
details.  warn(
```

In [15]:
```
df
```

Out[15]:

1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.2
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.2
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.2
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.2
...	...	...	...	...	...	...	...	...
20635	1.5603	25.0	5.045455	1.133333	845.0	2.560606	39.48	-121.0
20636	2.5568	18.0	6.114035	1.315789	356.0	3.122807	39.49	-121.2
20637	1.7000	17.0	5.205543	1.120092	1007.0	2.325635	39.43	-121.2
20638	1.8672	18.0	5.329513	1.171920	741.0	2.123209	39.43	-121.3
20639	2.3886	16.0	5.254717	1.162264	1387.0	2.616981	39.37	-121.2

20640 rows × 8 columns

In [16]:
```
data1
```

	MedInc	HouseAge	AveRooms	AveBedrms	Population	AveOccup	Latitude	Longitud
0	8.3252	41.0	6.984127	1.023810	322.0	2.555556	37.88	-122.2

Out[16]:

In

1	8.3014	21.0	6.238137	0.971880	2401.0	2.109842	37.86	-122.2
2	7.2574	52.0	8.288136	1.073446	496.0	2.802260	37.85	-122.2
3	5.6431	52.0	5.817352	1.073059	558.0	2.547945	37.85	-122.2
4	3.8462	52.0	6.281853	1.081081	565.0	2.181467	37.85	-122.2
...	...	...	...	...	...	...	...	...
20635	1.5603	25.0	5.045455	1.133333	845.0	2.560606	39.48	-121.0
20636	2.5568	18.0	6.114035	1.315789	356.0	3.122807	39.49	-121.2
20637	1.7000	17.0	5.205543	1.120092	1007.0	2.325635	39.43	-121.2
20638	1.8672	18.0	5.329513	1.171920	741.0	2.123209	39.43	-121.3
20639	2.3886	16.0	5.254717	1.162264	1387.0	2.616981	39.37	-121.2

20640 rows × 9 columns

[17]:
```
df.isnull().sum()
```

Out[17]:
```
MedInc        0
HouseAge      0
AveRooms      0
AveBedrms     0
Population    0
AveOccup      0
Latitude      0
Longitude     0
```

In

```
dtype: int64
```

In [18]:
```
x = data1.drop(['target'], axis = 1)
y = data1['target']
```

In [19]:
```
from sklearn.model_selection import train_test_split
xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size = 0.2, rando
```

In [20]:
```
import sklearn
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
model=lm.fit(xtrain, ytrain)
```

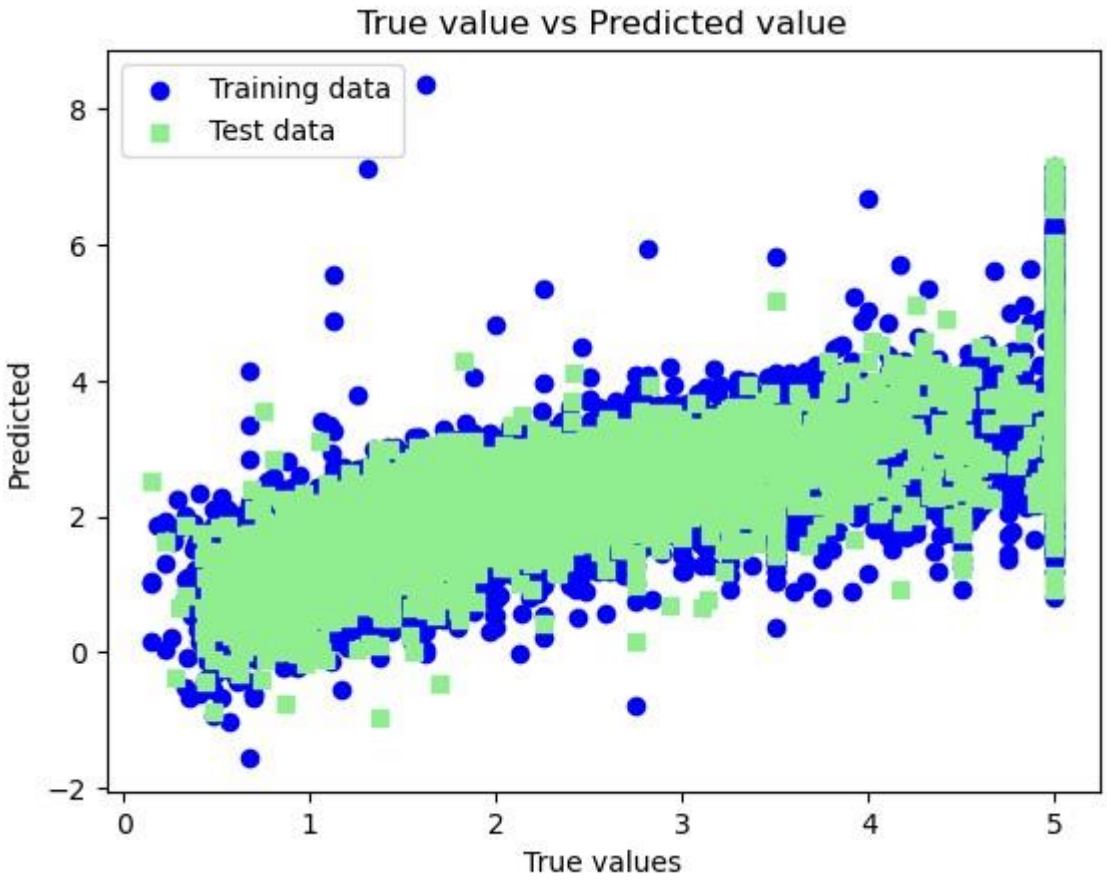
In [21]:
```
ytrain_pred = lm.predict(xtrain)
ytest_pred = lm.predict(xtest)
```

In [22]:
```
df=pd.DataFrame(ytrain_pred,ytrain)
df=pd.DataFrame(ytest_pred,ytest)
```

In [23]:
```
from sklearn.metrics import mean_squared_error, r2_score
mse = mean_squared_error(ytest, ytest_pred)
print(mse)
mse = mean_squared_error(ytrain_pred,ytrain)
print(mse)
```

```
0.5289841670367192
0.5234413607125448
```

In [24]:
```
plt.scatter(ytrain ,ytrain_pred,c='blue',marker='o',label='Training data')
plt.scatter(ytest,ytest_pred ,c='lightgreen',marker='s',label='Test data')
plt.xlabel('True values')
plt.ylabel('Predicted')
plt.title("True value vs Predicted value")
plt.legend(loc= 'upper left')
#plt.hlines(y=0,xmin=0,xmax=50)
plt.plot()
plt.show()
```



#Tanmay_Dixit_TE_13143