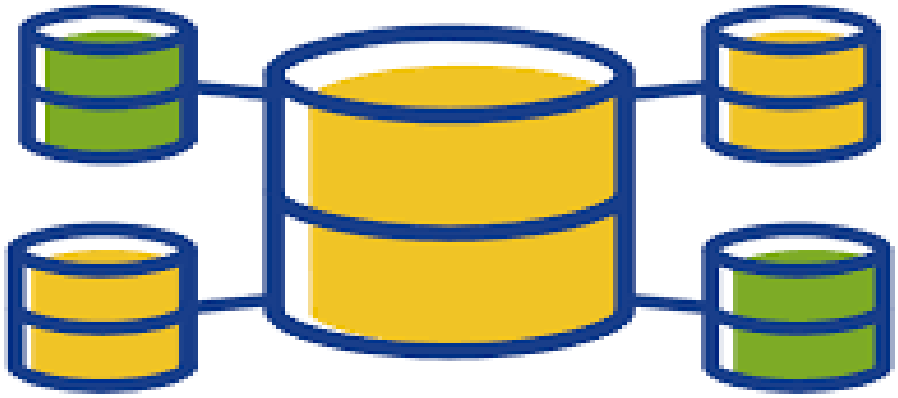
4.1 METHODS FOR DATA AGGREGATION AND ITS RESULTS

Data Aggregation :

Data aggregation is any process in which information is gathered and expressed in a summary form, for purposes such as statistical analysis. A common aggregation purpose is to get more information about particular groups based on specific variables such as age, profession, or income. The information about such groups can then be used for Web site personalization to choose content and advertising likely to appeal to an individual belonging to one or more groups for which data has been collected. For example, a site that sells music CDs might advertise certain CDs based on the age of the user and the data aggregate for their age group. Online analytic processing (OLAP) is a simple type of data aggregation in which the marketer uses an online reporting mechanism to process the information.

Data aggregation can be user-based: personal data aggregation services offer the user a single point for collection of their personal information from other Web sites. The customer uses a single master personal identification number (PIN) to give them access to their various accounts (such as those for financial institutions, airlines, book and music clubs, and so on). Performing this type of data aggregation is sometimes referred to as "screen scraping."



Data Aggregation

Data Aggregation Functions:

- **Display the dataset mean, medium and other values**

Command:

```
print(cc.describe())
```

Output:

	YEAR	Below 18 Years	...	Above 60 Years	Total
count	4180.000000	4180.000000	...	4180.000000	4179.000000
mean	2010.000000	0.202392	...	0.047368	8.351041
std	1.414383	2.056906	...	0.404160	58.046745
min	2008.000000	0.000000	...	0.000000	0.000000
25%	2009.000000	0.000000	...	0.000000	0.000000
50%	2010.000000	0.000000	...	0.000000	0.000000
75%	2011.000000	0.000000	...	0.000000	0.000000
max	2012.000000	65.000000	...	8.000000	1522.000000

- **Display the Aggregation**

1) Mean

The **statistical mean** refers to the **mean** or average that is used to derive the central tendency of the data in question. It is determined by adding all the data points in a population and then dividing the total by the number of points. The resulting number is known as the **mean** or the average.

Command:

```
print("MEAN OF CYBER CRIME")
print("--------------------")
print(cc.mean())
```

Output :

```
MEAN OF CYBER CRIME
--------------------
YEAR                   2010.000000
Below 18 Years            0.202392
Between 18-30 Years       4.430383
Between 30-45 Years       2.973445
Between 45-60 Years       0.695455
Above 60 Years            0.047368
Total                     8.351041
dtype: float64
```

2) Median

The "**median**" **is**the "middle" value in the list of numbers.

Command :

```
print("MEDIAN OF CYBER CRIME DATA SET")
print("----------------------------")
print(cc.median())
```

Output :

```
MEDIAN OF CYBER CRIME DATA SET
-----------------------------
YEAR                   2010.0
Below 18 Years            0.0
Between 18-30 Years       0.0
Between 30-45 Years       0.0
Between 45-60 Years       0.0
Above 60 Years            0.0
Total                     0.0
dtype: float64
```

3) Mode

The **mode is** a **statistical** term that refers to the most frequently occurring number found in a set of numbers. The **mode is** found by collecting and organizing data in order to count the frequency of each result. The result with the highest count of occurrences **is** the **mode** of the set, also referred to as the modal value.

Command :

```
print("MODE OF CYBER CRIME DATA SET")
print("----------------------------")
print(cc.mode())
```

Output :

```
MODE OF CYBER CRIME DATA SET
----------------------------
              STATE/UT  ...  Total
0       A & N ISLANDS   ...    0.0
1      ANDHRA PRADESH   ...    NaN
2   ARUNACHAL PRADESH   ...    NaN
3               ASSAM   ...    NaN
4               BIHAR   ...    NaN
5          CHANDIGARH   ...    NaN
6        CHHATTISGARH   ...    NaN
7        D & N HAVELI   ...    NaN
8         DAMAN & DIU   ...    NaN
9               DELHI   ...    NaN
10                GOA   ...    NaN
```

4) Minimum

In **statistics**, the sample maximum and sample **minimum**, also called the largest observation and smallest observation.

Command :

```
print("MINIMUM OF CYBER CRIME DATA SET")
print("----------------------------")
print(cc.min())
```

Output :

```
MINIMUM OF CYBER CRIME DATA SET
----------------------------
STATE/UT                                                A & N ISLANDS
CRIME HEAD             BREACH OF CONFIDENTIALITY OR PRIVACY (SECTION ...
YEAR                                                               2008
Below 18 Years                                                        0
Between 18-30 Years                                                   0
Between 30-45 Years                                                   0
Between 45-60 Years                                                   0
Above 60 Years                                                        0
Total                                                                 0
dtype: object
```

5) Maximum

In **statistics**, the sample **maximum** and sample minimum, also calle the **largest** observation.

Command :

```
print("MAXIMUM OF CYBER CRIME DATA SET")
print("----------------------------")
print(cc.max())
```

Output :

```
MAXIMUM OF CYBER CRIME DATA SET
----------------------------
STATE/UT                                                  WEST BENGAL
CRIME HEAD             UN-AUTHORISED ACCESS OR ATTEMPT TO ACCESS TO P...
YEAR                                                               2012
Below 18 Years                                                       65
Between 18-30 Years                                                 928
Between 30-45 Years                                                 436
Between 45-60 Years                                                  90
Above 60 Years                                                        8
Total                                                              1522
dtype: object
```

6) Count

In statistics, count data is a statistical data type, a type of data in which the observations can take **only** the non-negative integer values {0, 1, 2, 3, ...}, and where these integers arise from counting rather than ranking.

Command :

```
print("NO.OF ROWS OF CYBER CRIME DATA SET")
print("----------------------------")
print(cc.count())
```

Output :

```
NO.OF ROWS OF CYBER CRIME DATA SET
----------------------------
STATE/UT                4180
CRIME HEAD              4180
YEAR                    4180
Below 18 Years          4180
Between 18-30 Years     4180
Between 30-45 Years     4180
Between 45-60 Years     4180
Above 60 Years          4180
Total                   4179
dtype: int64
```

7) Standard Deviation

In **statistics**, the **standard deviation** (SD, also represented by the lower case Greek letter sigma σ or the Latin letter s) **is** a measure that **is** used to quantify the amount of variation or dispersion of a set of data values. It **is** algebraically simpler, though in practice less robust, than the average absolute **deviation.**

Command :

```
print("SD OF CYBER CRIME DATA SET")
print("----------------------------")
print(cc.std())
```

Output :

```
SD OF CYBER CRIME DATA SET
---------------------------
YEAR                    1.414383
Below 18 Years          2.056906
Between 18-30 Years    33.135255
Between 30-45 Years    19.313636
Between 45-60 Years     4.809044
Above 60 Years          0.404160
Total                  58.046745
dtype: float64
```

8) Quartile(Q1)

A **quartile is** a type of **quantile**. The first **quartile** (Q_1) **is** defined as the middle number between the smallest number and the median of the data set.

Command :

```
print("Q1 OF CYBER CRIME DATA SET")
print("---------------------------")
print(cc.quantile([0.25]))
```

Output :

```
Q1 OF CYBER CRIME DATA SET
---------------------------
        YEAR  Below 18 Years  ...    Above 60 Years  Total
0.25  2009.0             0.0  ...               0.0    0.0

[1 rows x 7 columns]
```

9) Quartile(Q3)

The third **quartile** (Q_3) **is** the middle value between the median and the highest value of the data set.

Command :

```
print("Q3 OF CYBER CRIME DATA SET")
print("----------------------------")
print(cc.quantile([0.75]))
```

Output :

```
Q3 OF CYBER CRIME DATA SET
----------------------------
        YEAR  Below 18 Years  ...    Above 60 Years  Total
0.75  2011.0             0.0  ...               0.0    0.0

[1 rows x 7 columns]
```

- **display count unique top freq(AGGREGATION ON PARTICULER COLUMN)**

1) print(cc['STATE/UT'].describe())

```
count            4180
unique             38
top       LAKSHADWEEP
freq              110
Name: STATE/UT, dtype: object
```

2) print(cc['YEAR'].describe())

```
count    4180.000000
mean     2010.000000
std         1.414383
min      2008.000000
25%      2009.000000
50%      2010.000000
75%      2011.000000
max      2012.000000
Name: YEAR, dtype: float64
```

3) print(cc['Below 18 Years'].describe())

```
count    4180.000000
mean        0.202392
std         2.056906
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max        65.000000
Name: Below 18 Years, dtype:
```

4) print(cc['Between 18-30 Years'].describe())

```
count    4180.000000
mean        4.430383
std        33.135255
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max       928.000000
Name: Between 18-30 Years, dtype: float64
```

5) print(cc['Between 30-45 Years'].describe())

```
count    4180.000000
mean        2.973445
std        19.313636
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max       436.000000
Name: Between 30-45 Years, dtype: float64
```

6) print(cc['Between 45-60 Years'].describe())

```
count    4180.000000
mean        0.695455
std         4.809044
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max        90.000000
Name: Between 45-60 Years, dtype: float64
```

7) print(cc['Above 60 Years'].describe())

```
count    4180.000000
mean        0.047368
std         0.404160
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max         8.000000
Name: Above 60 Years, dtype: float64
```

8) print(cc['Total'].describe())

```
count    4179.000000
mean        8.351041
std        58.046745
min         0.000000
25%         0.000000
50%         0.000000
75%         0.000000
max      1522.000000
Name: Total, dtype: float64
```

➢ Display aggregation step wise

```
print('Mean =',cc['Below 18 Years'].mean())
print('Q1 = ',cc['Below 18 Years'].quantile([0.25]))
print('Median / Q2 =',cc['Below 18 Years'].median())
print('Q3 = ',cc['Below 18 Years'].quantile([0.75]))
print('Mode =',cc['Below 18 Years'].mode())
print('Minimum =',cc['Below 18 Years'].min())
print('Maximum =',cc['Below 18 Years'].max())
print('Standerd Deviation =',cc['Below 18 Years'].std())
print('Total Rows =',cc['Below 18 Years'].count())
```

Output:

```
Mean = 0.20239234449760765
Q1 =  0.25    0.0
Name: Below 18 Years, dtype: float64
Median / Q2 = 0.0
Q3 =  0.75    0.0
Name: Below 18 Years, dtype: float64
Mode = 0    0
dtype: int64
Minimum = 0
Maximum = 65
Standerd Deviation = 2.056906148606878
Total Rows = 4180
```

- **Group by**

In an experiment, a control group is a baseline group that receives no treatment or a neutral treatment. To assess treatment effects, the experiment compares results in the treatment group to result in the control group.

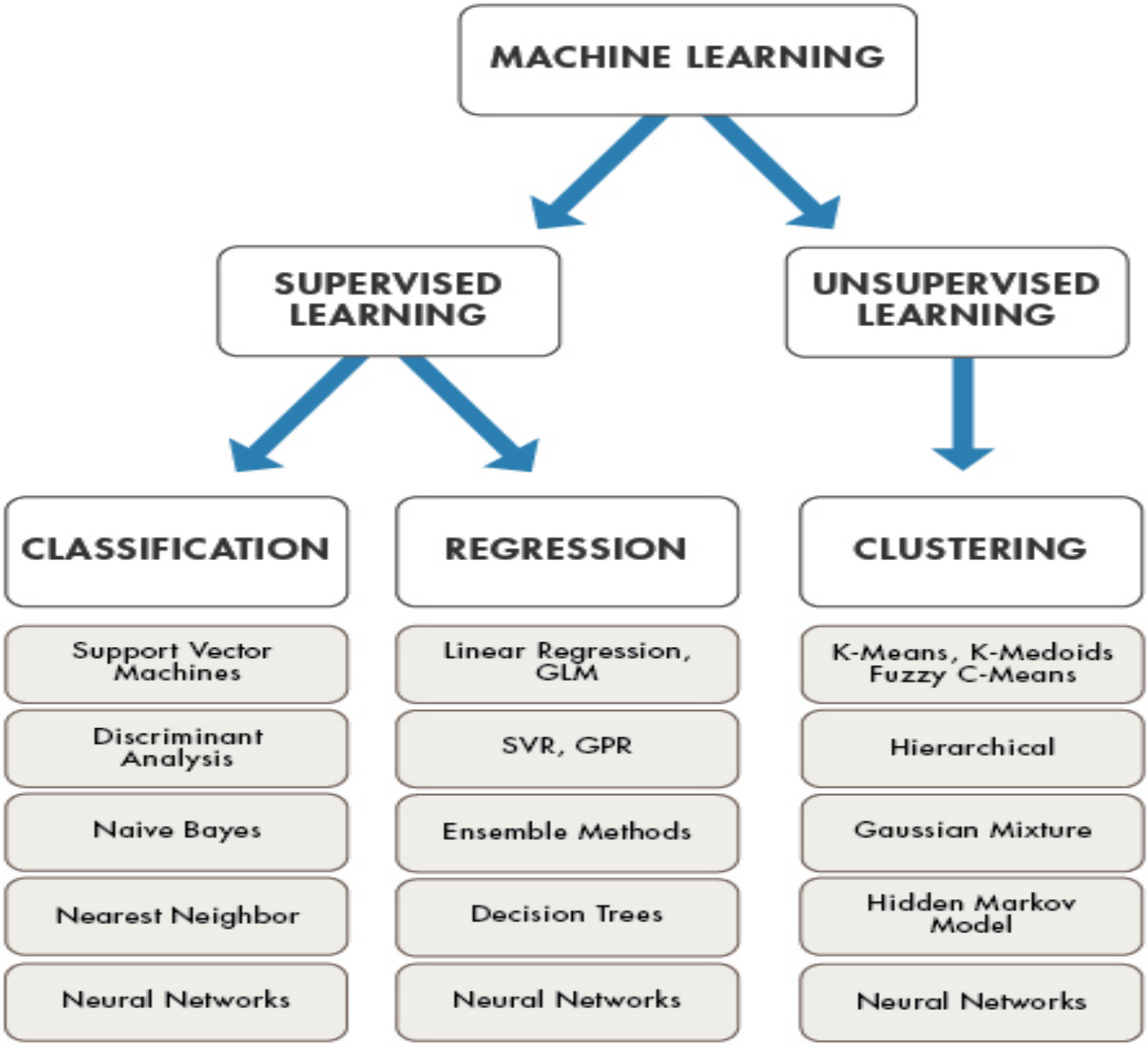
Command :

```
year = cc.groupby('YEAR').sum()

print(year)
```

Output:

```
      Below 18 Years  Between 18-30 Years  ...    Above 60 Years  Total
YEAR                                       ...
2008              36                 1014  ...                 6   2238
2009              60                 1776  ...                12   3306
2010             102                 3375  ...                48   7149
2011             246                 5298  ...                78   9780
2012             402                 7056  ...                54  12426

[5 rows x 6 columns]
```

4.2 MODELS USED FOR TRAINING DATA SET AND TESTING



Cyber Crime Dataset uses Linear Regression model.

Simple linear regression is useful for finding relationship between two continuous variables. One is predictor or independent variable and other is response or dependent variable.

It looks for statistical relationship but not deterministic relationship. Relationship between two variables is said to be deterministic if one variable can be accurately expressed by the other.

For example, using temperature in degree Celsius it is possible to accurately predict Fahrenheit. Statistical relationship is not accurate in determining relationship between two variables. For example, relationship between height and weight.

Randomness and unpredictability are the two main components of a regression model.

Prediction = Deterministic + Statistic

Deterministic part is covered by the predictor variable in the model. Stochastic part reveals the fact that the expected and observed value is unpredictable.

There will always be some information that are missed to cover. This information can be obtained from the residual information.

Residual plot helps in analyzing the model using the values of residues. It is plotted between predicted values and residue. Their values are standardized. The distance of the point from 0 specifies how bad the prediction was for that value. If the value is positive, then the prediction is low. If the value is negative, then the prediction is high. 0 values indicates prefect prediction. Detecting residual pattern can improve the model.

Analytics often involves studying past historical data to research potential trends. Weather condition is the state of crime in a given year in terms of crime head variables like TAMPERING COMPUTER SOURCE DOCUMENTS, HACKING, FRAUD DIGITAL SIGNATURE CERTIFICATE, etc., the existing models use data mining techniques to predict the crime ratio.

The main disadvantage of these systems is that it doesn't provide an estimate of the predicted crime ratio.

The system calculates average of values and understand the state of total crime, which doesn't yield estimate results.

This paper represents a mathematical method called Linear Regression to predict the crime ratio in various districts in India.

The Linear Regression method is modified in order to obtain the most optimum error percentage by iterating and adding some percentage of error to the input values.

This method provides an estimate of crime ratio using different age group.

What is the logic behind simple linear regression model?

As the name suggests, linear regression follows the linear mathematical model for determining the value of one dependent variable from value of one given independent variable.
Remember the linear equation from school?
y=mx+c

where y is the dependent variable, m is slope, x is the independent variable and c is the intercept for a given line.

We also have multiple regression model where multiple independent variables are used to calculate one dependent variable.

I have used Spyder for implementation. Any Python IDE can be used of your choice.[2]

Step 1: Importing libraries

```
import numpy as np #for using data as array
import pandas as pd #for loading csv file data to numpy array
import matplotlib.pyplot as plt #for plotting graph of x,y
from sklearn import linear_model # for model we want to predict by
from sklearn.metrics import mean_squared_error, r2_score #for mean error and variance calculation
from sklearn.model_selection import train_test_split #splitting traing and testing sets
```

- There are already developed libraries in Python for implementation of Machine Learning models.

  - First library called matplotlib is used to plot the graph in last step. ”plt” is used as variable name for using this library in code ahead.

  - sklearn is official machine learning library in python for various model implementation.

  - numpy is used to convert data into arrays for actual use by sklearn library.

  - pandas is used to access .csv file of our dataset.

Step 2: Loading dataset

```
#dataset
dataset = pd.read_csv('C://Users/MALVIKA/Desktop/18mcl2/Project_datascience/cyber_Crime2.csv')
```

Our dataset is in a .csv file type. Pandas variable pd is used to access the dataset with read_csv() function.

Step 3: Split to independent and dependent variables

```
X = dataset.iloc[:,1].values #states
print(X)

Y = dataset.iloc[:,3].values #below 18
print(Y)
```

We define x as the independent variable in dataset by iloc(index location) value. [] is used to define array elements. ":" inside [] indicates consider all rows in dataset and separating by using "," we specify the number of column which we want to use as independent or dependent variable values starting the count from zero in dataset.

Step 4: Splitting data into training and testing data

```
X_train, X_test, Y_train,Y_test=train_test_split(X, Y, test_size=1/3, random_state = 0)
```

Now, entire dataset is divided into training and testing set so that prediction does not overfit or underfit and correct values are obtained. train_test_split() is inbuilt function from scikit learn for splitting x and y variables data. "test_size" parameter is used to divide (1/3)rd of entire dataset(30%) into test data and remaining as training data.Setting random_state as null would not allow random values to be taken from dataset.

Step 5: Choosing the Model

```
#reshaping array to convert from 1D to 2D array
X_test=X_test.reshape(-1,1)
X_train=X_train.reshape(-1,1)


#lin_reg is our model calling model "LinearRegression()"

lin_reg=linear_model.LinearRegression()
```

We reshape our independent variable as sklearn expects a 2D array as input.

Linear Regression is our model here with variable name of our model as “lin_reg”. We can try the same dataset with many other models as well. This part varies for any model otherwise all other steps are similar as described here.

Step 6: Fit our model

```
#fittring our data in linear regression model
lin_reg.fit(X_train,Y_train)
```

We now fit our model to the linear regression model by training the model with our independent variable and dependent variables.

Step 7: Predict the output

```
#making predictions
lin_reg_pred=lin_reg.predict(X_test)

#cof_and intercept_ are cofficients and intercepts resp. for our model
print("Coefficients : \n",lin_reg.coef_)
print("Intercept : \n",lin_reg.intercept_)

#the mean squares error
print("Mean squares error : %.2f" % mean_squared_error(Y_test,lin_reg_pred))


#explained variance scorwe :1 is perfect prediction
print('Variance score : %.2f' % r2_score(Y_test,lin_reg_pred))
```

Finally our model predicts the dependent variable “lin_reg_pred” using the test values of independent variable.

We can see the coefficient,intercept values for our outlier and also the mean squared error and variance for the predicted values(lin_reg_pred) and actual test value of dependent variable(y_test). Inbuilt methods does the math with the predefined formulae for each value.

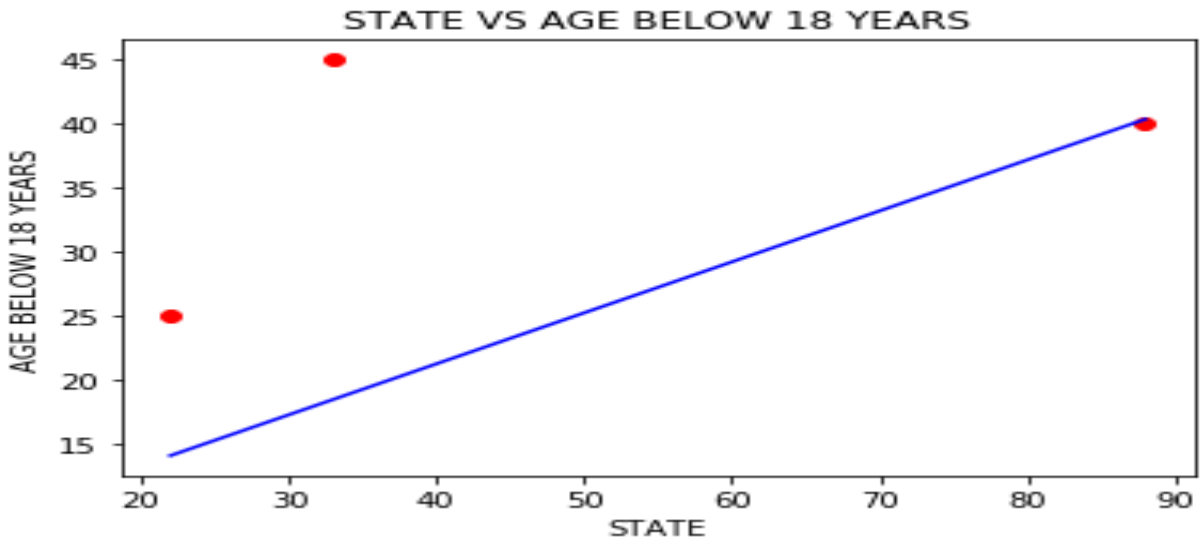
Step 8: Plot the graph

```
#plotting graph
plt.scatter(X_test, Y_test, color= 'red')
plt.plot(X_test,lin_reg_pred,color='blue')
plt.title('STATE VS AGE BELOW 18 YEARS')
plt.xlabel('STATE')
plt.ylabel('AGE BELOW 18 YEARS')

plt.show()
```

We ultimately want to visualize the actual data values and predicted data values in a graphical format. “plt”, matplotlib variable, is used to plot points using “scatter()” and outlier using “plot()” functions.

Output might vary depending on various system features. Output I got is as follows:

```
[11 22 33 44 55 66 77 88 99]
[10 25 45 30 15 17 63 40 37]
Coefficients :
 [0.39853896]
Intercept :
 5.285714285714288
Mean squares error : 275.17
Variance score : -2.81
```



- Here we assume the states as a number because linear model does not support the string data type