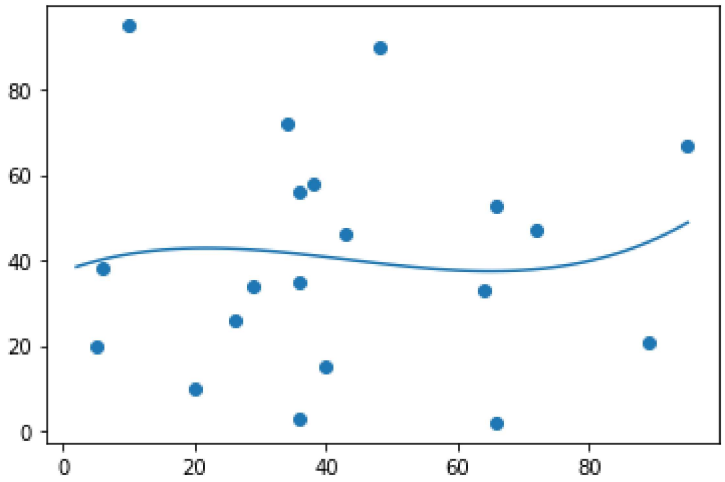
In [ ]:
```
#bad fit
import numpy as np
import matplotlib.pyplot as plt
x= [89,43,36,36,95,10,66,34,38,20,26,29,48,64,6,5,36,66,72,40]
y= [21,46,3,35,67,95,53,72,58,10,26,34,90,33,38,20,56,2,47,15]
mymodel= np.poly1d(np.polyfit(x, y, 3))

myline= np.linspace(2, 95, 100)
plt.scatter(x, y)
plt.plot(myline, mymodel(myline))
plt.show()
```



R-squared for bad fit

In [ ]:
```
import numpy as np
from sklearn.metrics import r2_score

x= [89,43,36,36,95,10,66,34,38,20,26,29,48,64,6,5,36,66,72,40]
y= [21,46,3,35,67,95,53,72,58,10,26,34,90,33,38,20,56,2,47,15]
model= np.poly1d(np.polyfit(x,y, 3))
print(r2_score(y, model(x)))
```

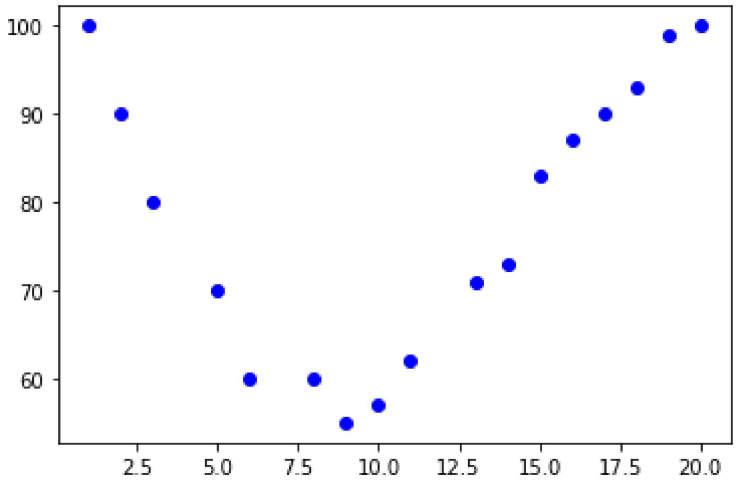
```
0.009952707566680652
```

Note >> r2 value is too low, means x and y aren't in accurate relationship.

Scatter plot\

1-- Data input

In [ ]:
```
import matplotlib.pyplot as plt
x= [1,2,3,5,6,8,9,10,11,13,14,15,16,17,18,19,20]
y= [100,90,80,70,60,60,55,57,62,71,73,83,87,90,93,99,100]
plt.scatter(x, y, color= "blue")
plt.show()
```



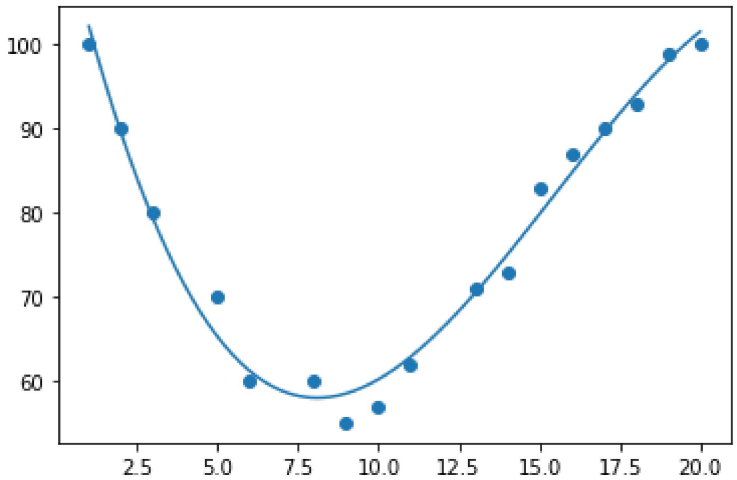
step 2-- drawing line

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

x= [1,2,3,5,6,8,9,10,11,13,14,15,16,17,18,19,20]
y= [100,90,80,70,60,60,55,57,62,71,73,83,87,90,93,99,100]

model= np.poly1d(np.polyfit(x, y, 3))

line= np.linspace(1, 20, 100)
plt.scatter(x, y)
plt.plot(line, model(line))
plt.show()
```



step--3 plotting r squared

In [ ]:
```
import numpy as np
from sklearn.metrics import r2_score

x= [1,2,3,5,6,8,9,10,11,13,14,15,16,17,18,19,20]
y= [100,90,80,70,60,60,55,57,62,71,73,83,87,90,93,99,100]

model= np.poly1d(np.polyfit(x,y, 3))
print(r2_score(y, model(x)))
```

```
0.9799174038012791
```

here, r2 indicates 97% relationship between x and y.

step 4-- speed prediction

In [ ]:
```
import numpy as np
from sklearn.metrics import r2_score

x= [1,2,3,5,6,8,9,10,11,13,14,15,16,17,18,19,20]
y= [100,90,80,70,60,60,55,57,62,71,73,83,87,90,93,99,100]

model= np.poly1d(np.polyfit(x,y, 3))
speed= model(18)
print(speed)
```

```
94.0569762856651
```

Note >> numpy can also be used as resource for ML algorithms.

Example

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

#dataset
dataset= pd.read_csv("polynomial-regression.csv")
```

In [ ]:
```
dataset.head()
```

Out[ ]:

	araba_fiyat	araba_max_hiz
0	60	180
1	70	180
2	80	200
3	100	200
4	120	200

In [ ]:
```
X= dataset["araba_fiyat"]
y= dataset["araba_max_hiz"]
```

In [ ]:
```
X= dataset.iloc[:, :-1] #features
y= dataset.iloc[ : , -1:] #labels
```

In [ ]:
```
X.head()
```

Out[ ]:

	araba_fiyat
0	60

	araba_fiyat
1	70
2	80
3	100
4	120

In [ ]:
```
y.head()
```

Out[ ]:

	araba_max_hiz
0	180
1	180
2	200
3	200
4	200

Train-Test splitting

In [ ]:
```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test= train_test_split(X, y, test_size= 0.3, random_state=
```

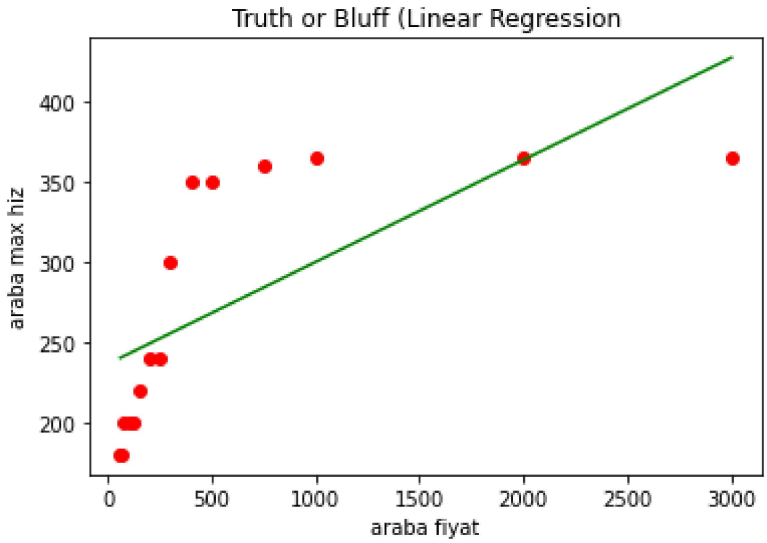
Fit linear regression

In [ ]:
```
from sklearn.linear_model import LinearRegression
lin_reg= LinearRegression()
lin_reg.fit(X, y)
```

Out[ ]:
```
LinearRegression()
```

Visualizing linear regression results

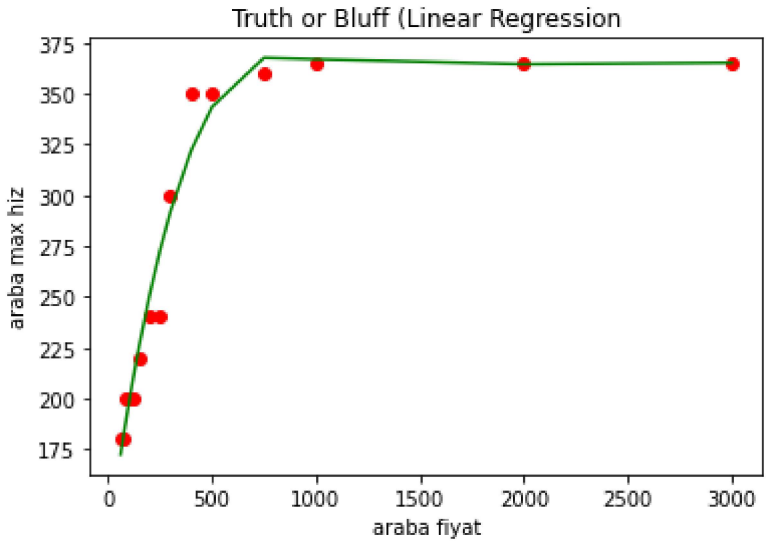
In [ ]:
```
def viz_linear():
    plt.scatter(X, y, color="red")
    plt.plot(X, lin_reg.predict(X), color="green")
    plt.title("Truth or Bluff (Linear Regression")
    plt.xlabel("araba fiyat")
    plt.ylabel("araba max hiz")
    plt.show()
    return
viz_linear()
```



Fitting polynomial regression to the dataset

In [ ]:
```
from sklearn.preprocessing import PolynomialFeatures
poly_reg= PolynomialFeatures(degree=4)
X_poly = poly_reg.fit_transform(X)
pol_reg = LinearRegression()
pol_reg.fit(X_poly, y)

def viz_polynomial():
    plt.scatter(X, y, color="red")
    plt.plot(X, pol_reg.predict(poly_reg.fit_transform(X)), color="green")
    plt.title("Truth or Bluff (Linear Regression")
    plt.xlabel("araba fiyat")
    plt.ylabel("araba max hiz")
    plt.show()
    return
viz_polynomial()
```



predict a new result with linear regression

In [ ]:
```
pred_linear= lin_reg.predict(([[1000]])
```

```
C:\Users\Javeria\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\base.
py:450: UserWarning: X does not have valid feature names, but LinearRegression was fitte
d with feature names
  warnings.warn(
```

In [ ]:
```
pred_linear= lin_reg.predict(([[1000]])
```

```
C:\Users\Javeria\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\base.
py:450: UserWarning: X does not have valid feature names, but LinearRegression was fitte
d with feature names
  warnings.warn(
```

In [ ]:
```
pred_polynomial= pol_reg.predict(poly_reg.fit_transform([[1000]]))
```

In [ ]:
```
print("Linear Regression results    =", pred_linear)
print("polynomial Regression results    =", pred_polynomial)

print("The difference is    =", pred_linear - pred_polynomial)
```

```
Linear Regression results    = [[299.83277117]]
polynomial Regression results    = [[366.80291008]]
The difference is    = [[-66.97013891]]
```