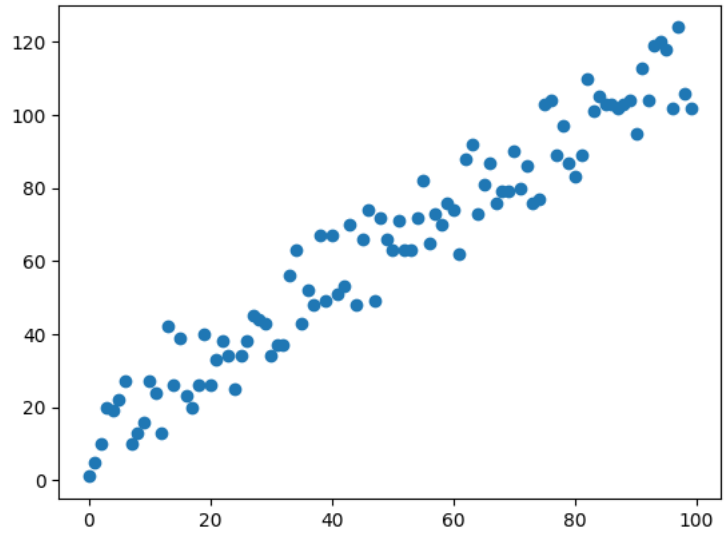
Generating an array of floating numbers

```
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import PolynomialFeatures
import matplotlib.pyplot as plt

x = [x for x in range(100)]
x=((np.array(x)))
y=np.random.randint(1,30,len(x))+x
# print(x)
plt.scatter(x,y)
```

```
<matplotlib.collections.PathCollection at 0x7fb3079836a0>
```




```
model1=LinearRegression()
model1.fit(x.reshape(-1,1),y)
y_predicted=model1.predict(x.reshape(-1,1))
print('Score=>', model1.score(x.reshape(-1,1),y))
```

```
Score=> 0.9358495051747704
```

Applying polynomial regression with degree 20 polynomial

```
model=LinearRegression()
poly = PolynomialFeatures(degree=20)
x_poly = poly.fit_transform(x.reshape(-1, 1))
model.fit(x_poly, y)
y_pred=model.predict(x_poly)
```

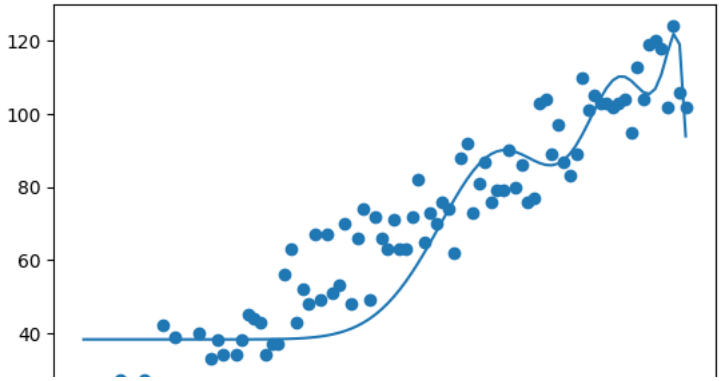
```
score = model.score(x_poly, y)
print('Score=>', score)
```

```
Score=> 0.7894693914724538
```

Plotting the results

```
plt.scatter(x,y)
# fig = plt.figure
plt.plot(x,y_pred)
plt.show()
```

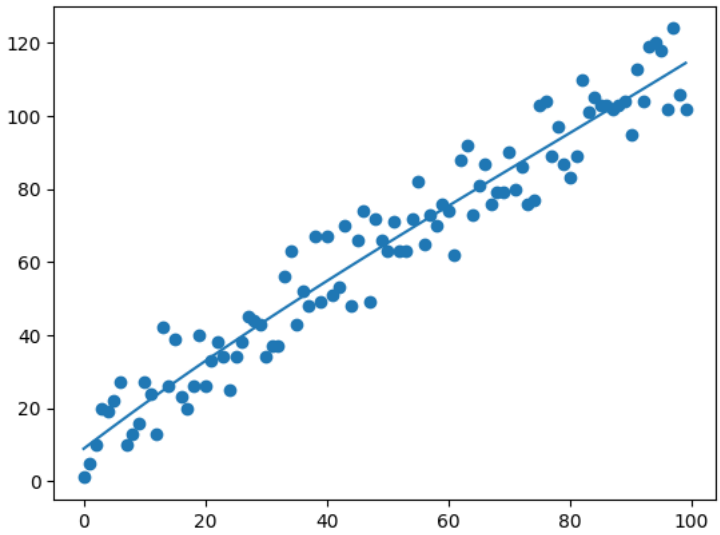
Alt+A



Degree 4 polynomial

Degree 4 polynomial

```
poly = PolynomialFeatures(degree=4)
x_poly = poly.fit_transform(x.reshape(-1,1))
model.fit(x_poly,y);
y_pred=model.predict(x_poly)
plt.scatter(x,y)
plt.plot(x,y_pred)
plt.show()
```




```
score = model.score(x_poly, y)
print('Score=>', score)
```

```
Score=> 0.9367913895941227
```

✓ 0s completed at 11:41 PM