Assignmnet 1

Model Selection (Part 1)

In [30]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
matr = []
Y = []
```

In [72]:

```
# Read the data
data = pd.read_csv("Dataset_1_train.csv", header = -1,usecols=range(2))
cols = ["x","y"]
data.columns = cols
data.head()
```

Out[72]:

	x	y
0	0.516220	1.609671
1	0.850085	14.814006
2	-0.840629	-8.738649
3	0.227433	-0.274344
4	-0.649508	-8.683412

In [73]:

```
x = data['x']
y = data['y']
```

In [74]:

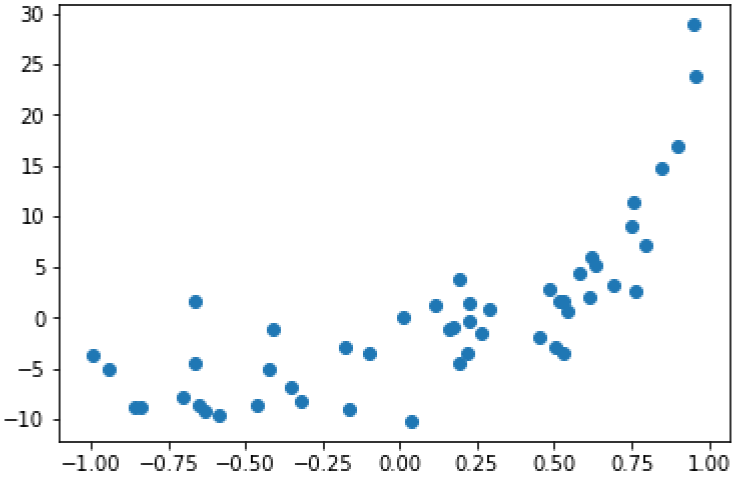
```
plt.scatter(x,y)
```

Out[74]:

```
<matplotlib.collections.PathCollection at 0x7f524e220a20>
```

In [75]:

```
plt.show()
```



In [77]:

```
matr = []
for a in x:
    row =[]
    for p in range(20,-1,-1):
        row = row + [a**(p)]
    matr = matr + [row]
```

In [79]:

```
#Initialization for X, X^T and Y.
Y=[]
for i in y:
    Y = Y +[[i]]

X = np.array(matr)
XT = X.transpose()
```

$$W^* = (X^T X)^{-1} X^T Y$$

Using the formula we derived from class, we get the following parameters for the polynomial

In [81]:

```
# Get the parameters
A = np.dot(XT,X)
A1 = np.matrix(A)
A_I = A1.I
XTX = np.dot(A_I,XT)
W = np.dot(XTX,Y)
# Get the Transpose because its easier to use as a row vector
WT = W.transpose()
# flatten the list.
p = WT.tolist()[0]
# Define the polynomial
Poly = np.poly1d(p)
```

Get the new Y values

Plot the curve and compare it with training data.

In [82]:

```
x1 = x.tolist()
x1.sort()
new_y = [Poly(i) for i in x ]
plt_y = [Poly(i) for i in x1 ]
```

Calculate The training MSE

In [83]:

```
sigma = 0
for i in range(len(new_y)):
    sigma = sigma + (new_y[i] - y[i])**2
print("The Training mean square error is:",sigma/len(y))
```

```
The Training mean square error is: 6.474747793
```

Read the Validation data

In [84]:

```
valid = pd.read_csv("Dataset_1_valid.csv",header=-1,usecols=range(2))
cols = ["x","y"]
valid.columns = cols
#data.head()
u = valid['x']
v = valid['y']
```

calculate the validation MSE

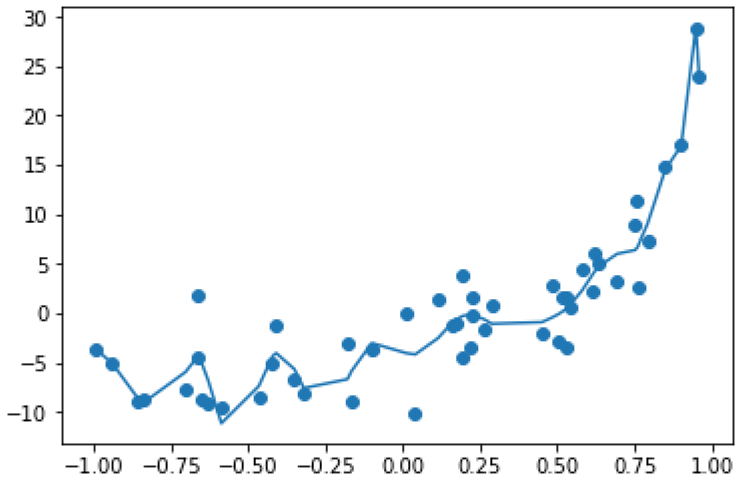
In [87]:
```
new_v = [Poly(i) for i in u ]
sigma = 0
for i in range(len(new_v)):
    sigma = sigma + (new_v[i] - v[i])**2
print("The validation mean square error is:",sigma/len(v))
```

```
The validation mean square error is: 1418.46219822
```

Curve Fit with Training data

In [88]:
```
plt.scatter(x,y)
plt.plot(x1,plt_y)

plt.show()
```



TEST CURVE FIT

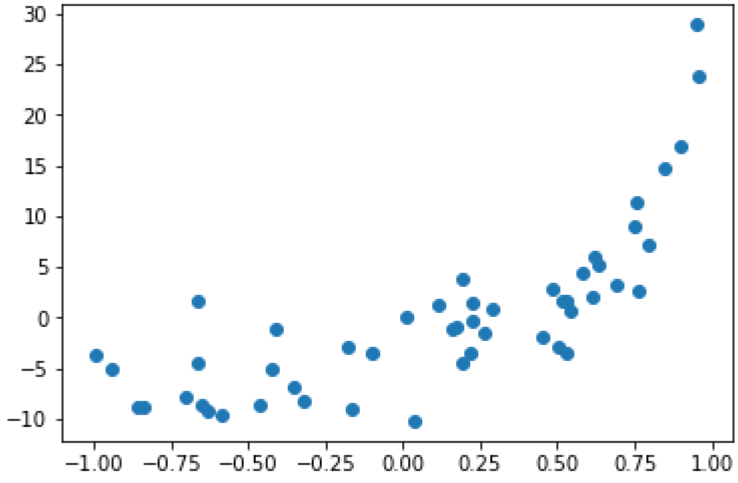
In [90]:
```
test = pd.read_csv("Dataset_1_train.csv", header = -1,usecols=range(2))
cols = ["x","y"]
test.columns = cols
test.head()
```

Out[90]:

	x	y
0	0.516220	1.609671
1	0.850085	14.814006
2	-0.840629	-8.738649
3	0.227433	-0.274344
4	-0.649508	-8.683412

In [91]:

```
plt.scatter(test['x'],test['y'])
```

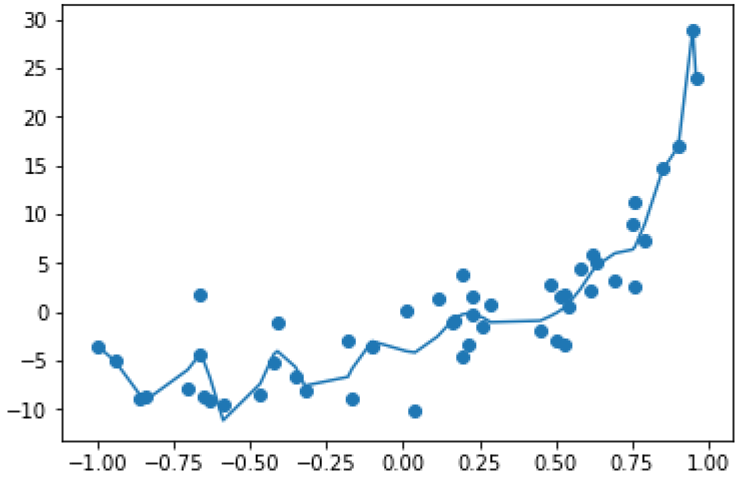


In [95]:

```
X = test['x'].tolist()
X.sort()
Y = [Poly(i) for i in X ]
```

In [97]:

```
plt.plot(X,Y)
plt.scatter(test['x'],test['y'])
plt.show()
```



In [ ]:

```
```