hwo_1a

August 18, 2018

In [69]:
```
import numpy as np
import matplotlib.pyplot as plt
```

1 Generating training samples

In [70]:
```
poly_order = 4
# Number of training samples
N = 10
# Generate equispaced floats in the interval [0, 2*pi]
x_train = np.linspace(0, 2*np.pi, N)
# Generate noise
mean = 0
std = 0.05
# Generate some numbers from the sine function
y = np.sin(x_train)
# Add noise
y += np.random.normal(mean, std, N)
#defining it as a matrix
y_train = np.asmatrix(y.reshape(N,1))
```

2 adding the bias and higher order terms to x

In [71]:
```
x = np.asmatrix(np.append(np.ones((N,1)),x_train.reshape((N,1)),axis = 1))
```

3 finding the optimum weights

In [72]:
```
w = (x.T*x).I*x.T*y_train
print(w)
```

```
[[ 0.67639965]
 [-0.20934188]]
```

4 generating test samples

```
In [73]: M = 100
         x_t = np.linspace(0, 2*np.pi, M)
         x_test = np.asmatrix(np.append(np.ones((M,1)),x_t.reshape(M,1),axis = 1))
```

5 predicting the outputs for the test sample

```
In [74]: y_test = x_test*w
```

6 Error (cost)

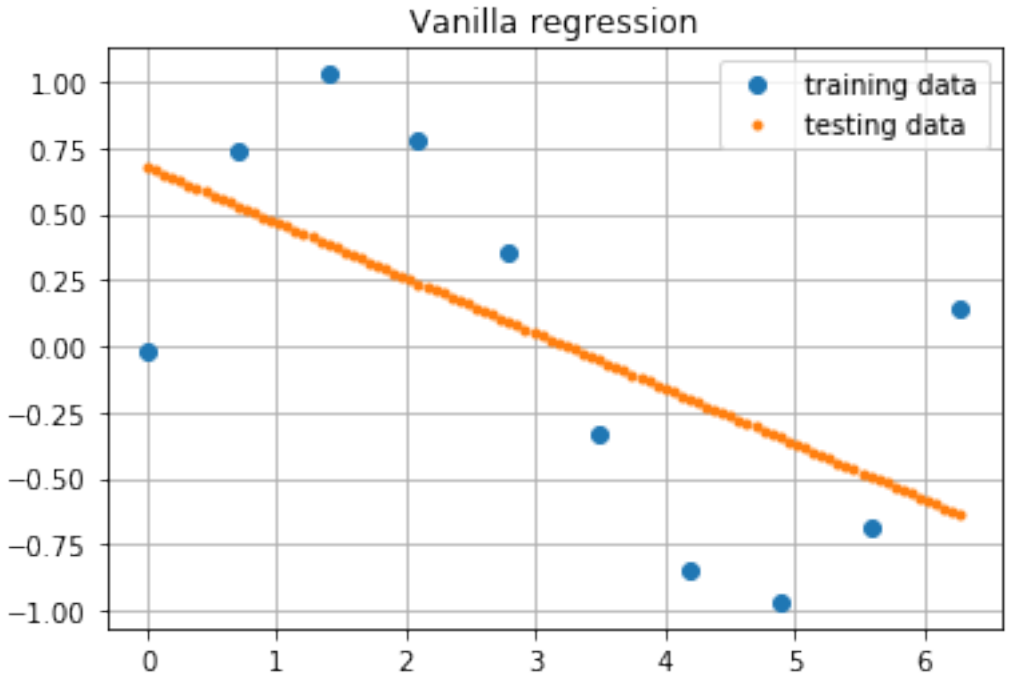
```
In [75]: y_fin = x * w

         print("error:- ",np.asmatrix(y_train-y_fin).T*np.asmatrix(y_train-y_fin))
```

```
error:-  [[2.81674634]]
```

7 ploting the results

```
In [76]: plt.plot(x_train,y_train,'o',label = 'training data')
         plt.plot(x_t,y_test,'.',label = 'testing data')
         plt.legend()
         plt.grid()
         plt.title("Vanilla regression")
         plt.show()
```



8 Observations

As the number of parameters is only 2 , - the model is estimated by a straight line - The error is pretty high

By increasing the variance of the noise - There error has increased - But there is not much shift in the plots