ML LAB REPORT : 1

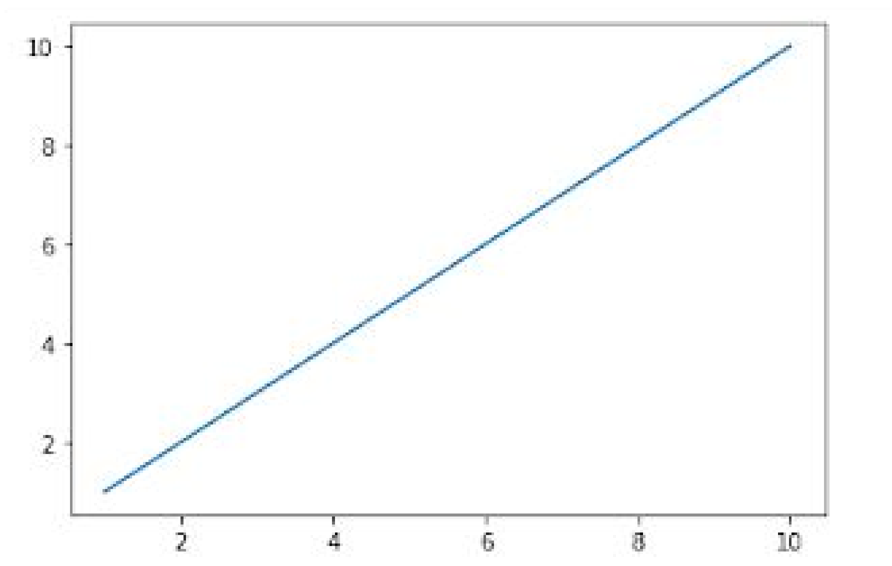
Name : Maurya Sujeet
Roll no. : 17BCS016

Error function for 1 degree polynomial with zero intercept

Equation of the line is y = mx + c

1. Slope of the line is 1 i.e m = 1.
2. Intercept is 0 i.e. c = 0.

```
import numpy as np
import matplotlib.pyplot as plt

m = 1
x = np.linspace(1,10,11)
y = m*x
plt.plot(x,y)
plt.show()
```



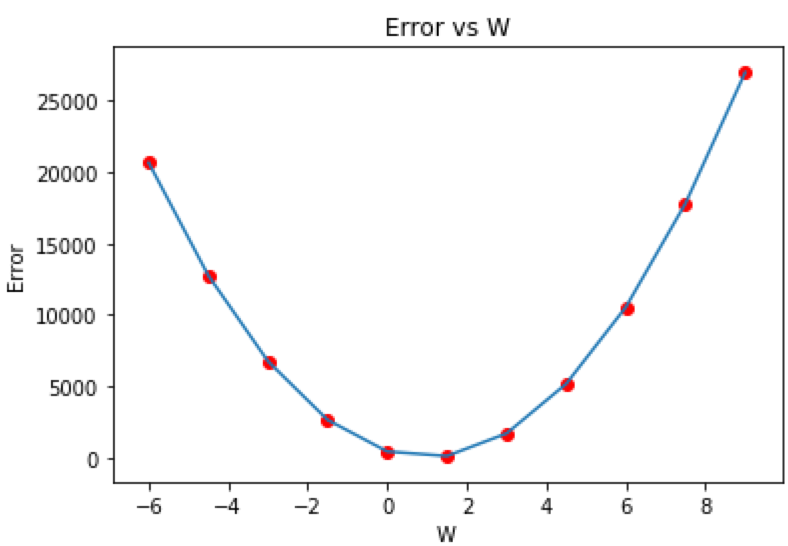
Graph of equation y = x .

Plotting Error function

Equation of error function is $Error = \sum_{i=1}^{n} (w_1 * x_i - y_i)^2$

Above equation has only 1 parameter that is w1.

```
w1 = np.linspace(-6,9,11)
l = 0
it = len(x)
for i in range(it):
    l += (w1*x[i]- y[i])**2


plt.plot(w1,l)
plt.xlabel('W')
plt.ylabel('Error')
plt.title('W vs Error')
plt.scatter(w1,l,c='red')
plt.show()
```



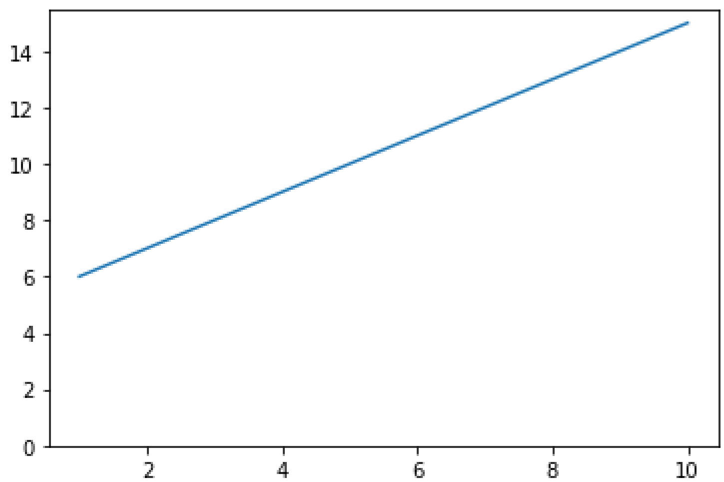
From the above graph we can see that minimum value of Error is 0 when w is equivalent to 1.

Error function for 1 degree polynomial with non-zero intercept

Equation of the line is $y = mx + c$

1. Slope of the line is 1, i.e $m = 1$.
2. Intercept is 5, i.e. $c = 5$.

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

m = 1
x = np.linspace(1,10,10)
y = m*x + 5

f, ax = plt.subplots(1)
ax.plot(x,y)
ax.set_ylim(bottom=0)
plt.show(f)
```



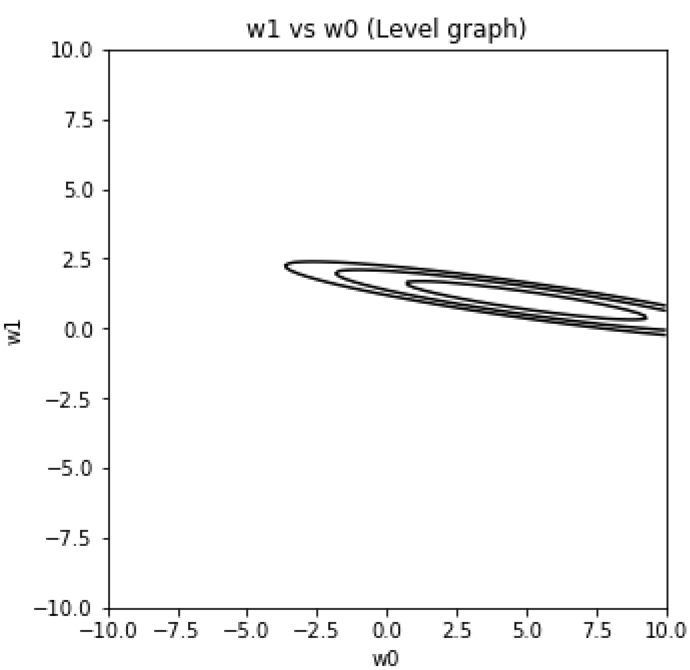
Graph of equation y = x + 5.

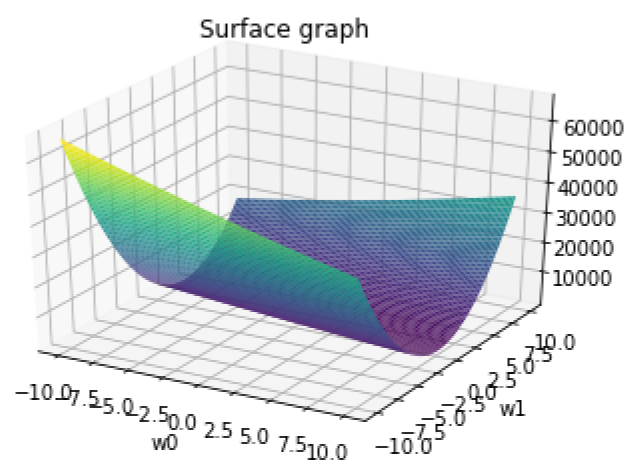
Plotting error function for equation y = x + 5.

Equation of error function is $Error = \sum_{i=1}^{n} ((w_1 * x_i + w_0) - y_i)^2$

Above equation has 2 parameters that is w1 and w0.

```
w1 = np.linspace(-10,10,200)
w0 = np.linspace(-10,10,200)
W0, W1 = np.meshgrid (w0, w1)
l = 0
it = len(x)
for i in range(it):
   l += (W1*x[i] + W0 - y[i])**2

plt.gca().set_aspect('equal',adjustable = 'box')   #to make the graph square
plt.draw()
plt.xlabel('w0')
plt.ylabel('w1')
plt.title('w1 vs w0 (Level graph)')
plt.contour(W0,W1,l,levels = [i for i in np.arange(-200,200,60)],colors = "black")

fig = plt.figure()
ax = plt.axes(projection='3d')

ax.plot_surface(W0, W1, l, cmap='viridis', rstride=1, cstride=1, edgecolor='none')
```



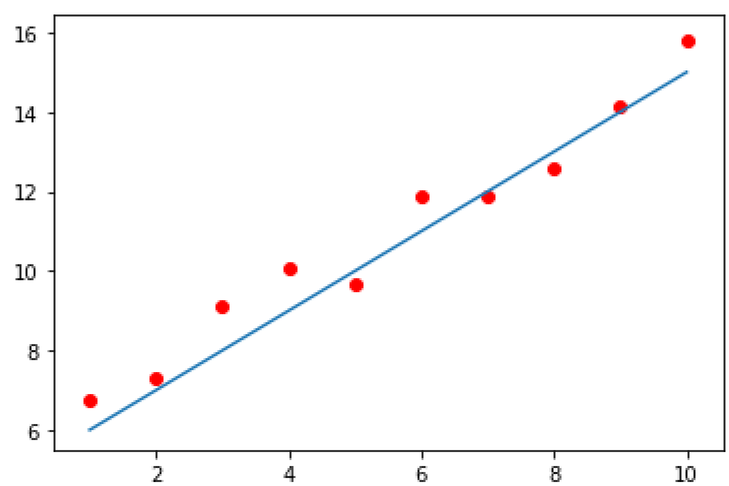


Error function for 1 degree polynomial with non-zero intercept with noise

Equation of the line is e = mx + c + noise

1. Slope of the line is 1 i.e m = 1.
2. Intercept is 0 i.e. c = 0.

```
import numpy as np
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D
import random

random.seed(1)
noise = np.random.randn(10)

m = 1
x = np.linspace(1,10,10)
y = m*x + 5

plt.plot(x,y)

e = m*x + 5 + noise        #represents equation with noise
plt.scatter(x,e,color = "red")
```



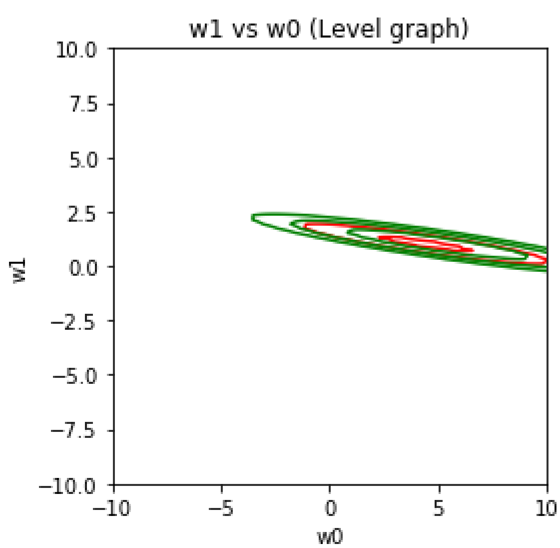
In the above graph , Blue line represents y = x + 5 .
While scattered red dots represents e = x + 5 + noise.

Potting error function

Equation of error function with noise is $Error = \sum_{i=1}^{n} ((w_1 * x_i + w_0) - e_i)^2$

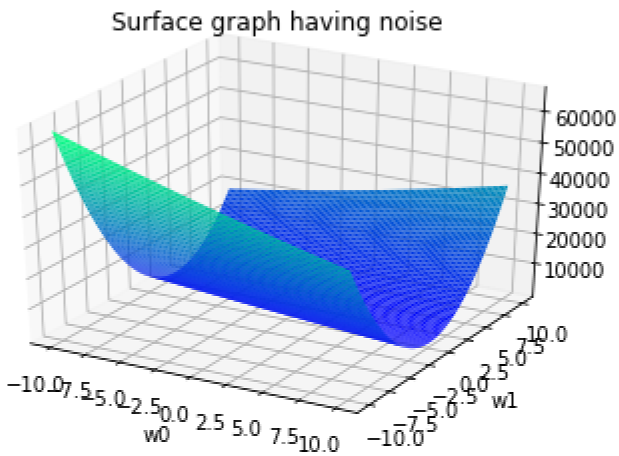
Above equation has 2 parameters that is w1 and w0.

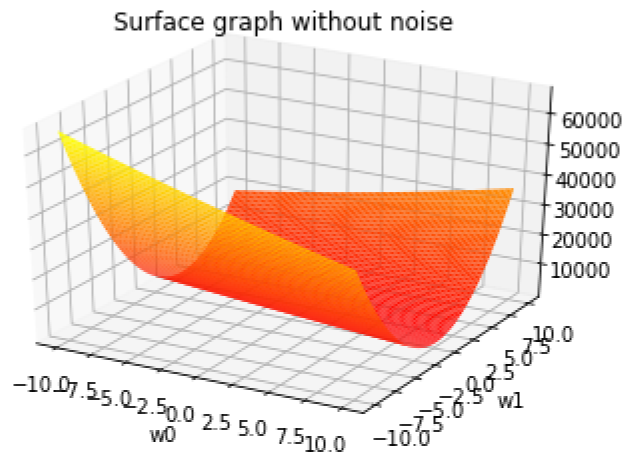
```
w1 = np.linspace(-10,10,100)
w0 = np.linspace(-10,10,100)
W0, W1 = np.meshgrid (w0, w1)

l = 0
it = len(x)
for i in range(it):
   l += (W1*x[i] + W0 - e[i])**2          #calculates error value  for equation having loss

plt.gca().set_aspect('equal',adjustable = 'box')
plt.draw()

plt.contour(W0,W1,l,levels = [i for i in np.arange(-100,100,60)],colors = "red")

l1 = 0
for i in range(it):
   l1 += (W1*x[i] + W0 - y[i])**2        #calculates error value for equation without loss
plt.contour(W0,W1,l1,levels = [i for i in np.arange(-200,200,60)],colors = "green")
```

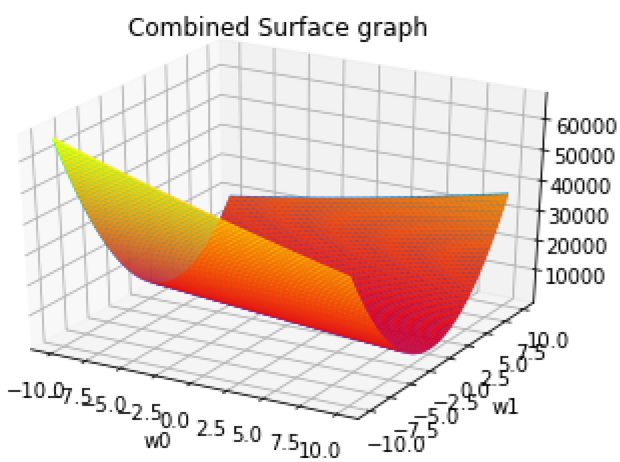


In the above graph , red represents level graph having noise while green represents level graph without noise.

```
fig = plt.figure()
ax = plt.axes(projection='3d')
plt.xlabel('w0')
plt.ylabel('w1')
plt.title('Combined Surface graph')

ax.plot_surface(W0, W1, l, cmap='winter', rstride=1, cstride=1, edgecolor='none')
ax.plot_surface(W0, W1, l1, cmap='autumn', rstride=1, cstride=1, edgecolor='none')
```







After combing surface graph having noise and surface graph without noise , we can see that there is negligible difference.