Demo : Learn how to do math calculations

Author: Jim Xie

Date: 2020-08-07

In [1]:
```
import sys
#!{sys.executable} -m pip install seaborn==0.9.0
import seaborn
print(seaborn.__version__)
import random
from mpl_toolkits.mplot3d import Axes3D
%matplotlib inline
import matplotlib.pyplot as plt
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn import preprocessing
#plt.figure(figsize=(15,10))
```

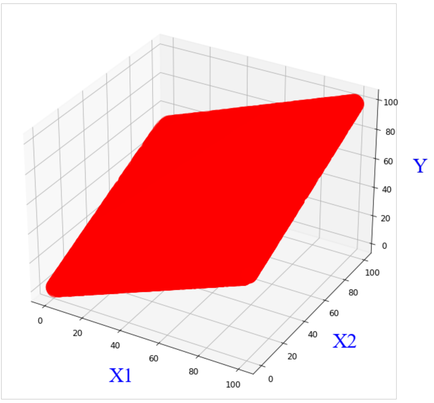
```
0.10.1
```

In [2]:
```
def random_data():
    x = random.randint(0,100)
    #x = random.random()
    return x

def generate_data(x1,x2):
    y = (x1 * x2)/2
    y = (x1 + x2) /2
    return y
```

In [3]:
```
train_x = []
train_y = []
for i in range(20000):
    x1 = random_data()
    x2 = random_data()
    x = [x1,x2]
    train_x.append(x)
    y = generate_data(x1,x2)
    train_y.append(y)
```

In [4]:
```
plt.style.use({'figure.figsize':(24, 8)})
fig = plt.figure()
ax = Axes3D(fig)
x1_axis = []
x2_axis = []
y_axis = []
for x,y in zip(train_x,train_y):
    x1_axis.append(x[0])
    x2_axis.append(x[1])
    y_axis.append(y)
ax.scatter(x1_axis, x2_axis, y_axis,depthshade=True,s=600,c='r')
ax.ticklabel_format(style='plain',axis='both')
font2 = {'family' : 'Times New Roman','weight' : 'normal','size'   : 30,"color":"blue"}
ax.set_xlabel("X1",font2,labelpad=20)
ax.set_ylabel("X2",font2,labelpad=20)
ax.set_zlabel("Y",font2,labelpad=20)
ax.grid(False)
plt.grid(linestyle=':')
plt.tick_params(labelsize=12)
plt.show()
```



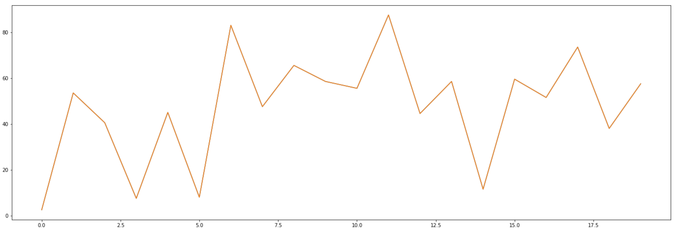
In [5]:
```
model = LinearRegression()
model.fit(train_x,train_y)
```

Out[5]:
```
LinearRegression()
```

In [6]:
```
test_x = []
test_y = []
for i in range(20):
    x1 = random_data()
    x2 = random_data()
    test_x.append([x1,x2])
    y = generate_data(x1,x2)
    test_y.append(y)
pred_y = model.predict(test_x)
#pred_y = pred_y - 10
print("x1    x2     y")
for x,y in zip(test_x,test_y):
    print(x[0],"   ",x[1],"     ",y)
```

```
x1    x2     y
1     4      2.5
57    50      53.5
43    38      40.5
14    1      7.5
43    47      45.0
0     16     8.0
86    80      83.0
87    8      47.5
63    68      65.5
88    29      58.5
53    58      55.5
76    99      87.5
57    32      44.5
65    52      58.5
2     21     11.5
37    82      59.5
58    45      51.5
82    65      73.5
56    20      38.0
40    75      57.5
```

In [7]:
```
plt.plot(test_y)
plt.plot(pred_y)
plt.show()
```



In [8]:
```
input_x1 = 10 #random_data()
input_x2 = 50 #random_data()
result = model.predict([[input_x1,input_x2]])
msg = "F(%.02f,%.02f)=%.02f"%(input_x1,input_x2,result[0])
print(msg)
```

```
F(10.00,50.00)=30.00
```

In [9]:
```
from sklearn import metrics
from sklearn.metrics import auc
from sklearn.metrics import accuracy_score
```

In [10]:
```
print(metrics.r2_score(test_y,pred_y,multioutput="uniform_average"))
print(metrics.r2_score(test_y,pred_y,multioutput="raw_values"))
```

```
1.0
[1.]
```

In [11]:
```
print(model.score(test_x,test_y))
print(metrics.r2_score(test_y, pred_y))
```

```
1.0
1.0
```

In [12]:
```
metrics.explained_variance_score(test_y,pred_y)
```

Out[12]:
```
1.0
```

In [13]:
```
e = metrics.mean_absolute_error(test_y,pred_y)
print("%0.8f"%e)
```

```
0.00000000
```

In [14]:
```
e = metrics.mean_squared_error(test_y,pred_y)
print("%0.8f"%e)
```

```
0.00000000
```

In [15]:
```
e = metrics.median_absolute_error(test_y,pred_y)
print("%0.8f"%e)
```

```
0.00000000
```

In [ ]:
```

```

In [ ]:
```

```