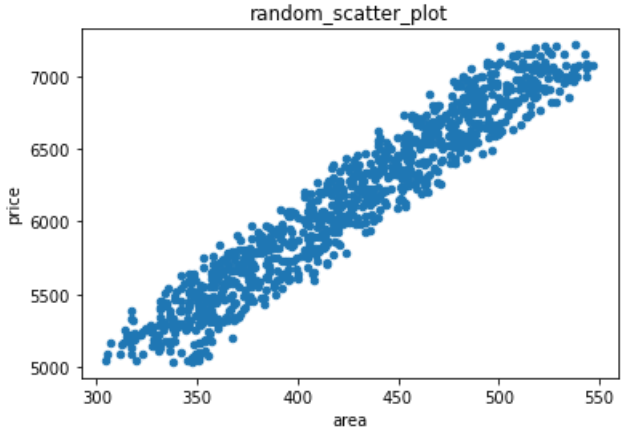
Linear Regression

68_Adnan Shaikh

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
```

In [2]:
```
obj = pd.DataFrame({"x":np.arange(300,500,0.2)+np.random.uniform(low=0,high=50,size=1000), "y":np.
arange(5000,7000,2)+np.random.uniform(low=0,high=250,size=1000)})
```

In [3]:
```
fig,axis = plt.subplots()
obj.plot(x="x",y="y",ax=axis,kind="scatter",title="random_scatter_plot",xlabel="area",ylabel="pric
e",layout=(10,9))
plt.show()
```



In [4]:
```
def minmax_normalize(data,new_max,new_min):
  mini = data.min()
  maxi = data.max()
  normalize_data = []
  for x in data:
    normalize_data.append((x-mini)/(mini-maxi)*(new_max-new_min)+new_min)
  return np.array(normalize_data)
```

In [5]:
```
class LinearModel:
  def fit(self,x,y):
    x_mean,y_mean = x.mean(), y.mean()
    x_norm, y_norm = x.apply(lambda k: k-x_mean), y.apply(lambda k: k-y_mean)
    sum_x = x_norm.sum()
    sum_y = y_norm.sum()
    xy_sum = (x_norm*y_norm).sum()
    sum_x_sq =  (x_norm**2).sum()
    intercept = xy_sum/sum_x_sq
    slope = y_mean-intercept*x_mean
    self.intercept, self.slope = intercept,slope
    return intercept,slope

  def predict(self,x):
    result = []
    for data in x:
      result.append(self.slope+self.intercept*data)
    return np.array(result)

  def average_error(self,x,y):

    predicted_y = self.predict(x)
    rmse = np.sqrt(np.sum(np.square(predicted_y-y))/x.shape[0])
    return rmse
```

In [6]:
```
model = LinearModel()
```

In [7]:
```
model.fit(obj.x,obj.y)
```

Out[7]:
```
(9.379816850089613, 2143.430221913603)
```

In [8]:
```
model.average_error(obj.x,obj.y)
```

Out[8]:
```
157.9563099222744
```

In [9]:
```
obj.head()
```

Out[9]:

	x	y
0	329.923993	5172.183623
1	330.659897	5156.647979
2	338.970577	5248.960807
3	313.505977	5168.992688
4	304.432691	5046.956522

In [10]:
```
model.predict([325])
```

Out[10]:
```
array([5191.87069819])
```