8C_LR_SVM

January 26, 2020

0.1 Task-C: Regression outlier effect.

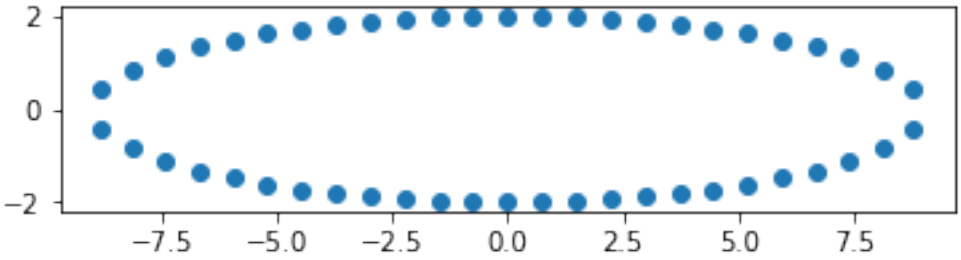
Objective:Visualization best fit linear regression line for different scenarios

In [1]:
```
# you should not import any other packages
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings("ignore")
import numpy as np
from sklearn.linear_model import SGDRegressor
```

In [2]:
```
import numpy as np
import scipy as sp
import scipy.optimize

def angles_in_ellipse(num,a,b):
    assert(num > 0)
    assert(a < b)
    angles = 2 * np.pi * np.arange(num) / num
    if a != b:
        e = (1.0 - a ** 2.0 / b ** 2.0) ** 0.5
        tot_size = sp.special.ellipeinc(2.0 * np.pi, e)
        arc_size = tot_size / num
        arcs = np.arange(num) * arc_size
        res = sp.optimize.root(
            lambda x: (sp.special.ellipeinc(x, e) - arcs), angles)
        angles = res.x
    return angles
```

In [3]:
```
a = 2
b = 9
n = 50

phi = angles_in_ellipse(n, a, b)
e = (1.0 - a ** 2.0 / b ** 2.0) ** 0.5
arcs = sp.special.ellipeinc(phi, e)

fig = plt.figure()
```

```
ax = fig.gca()
ax.axes.set_aspect('equal')
ax.scatter(b * np.sin(phi), a * np.cos(phi))
plt.show()
```



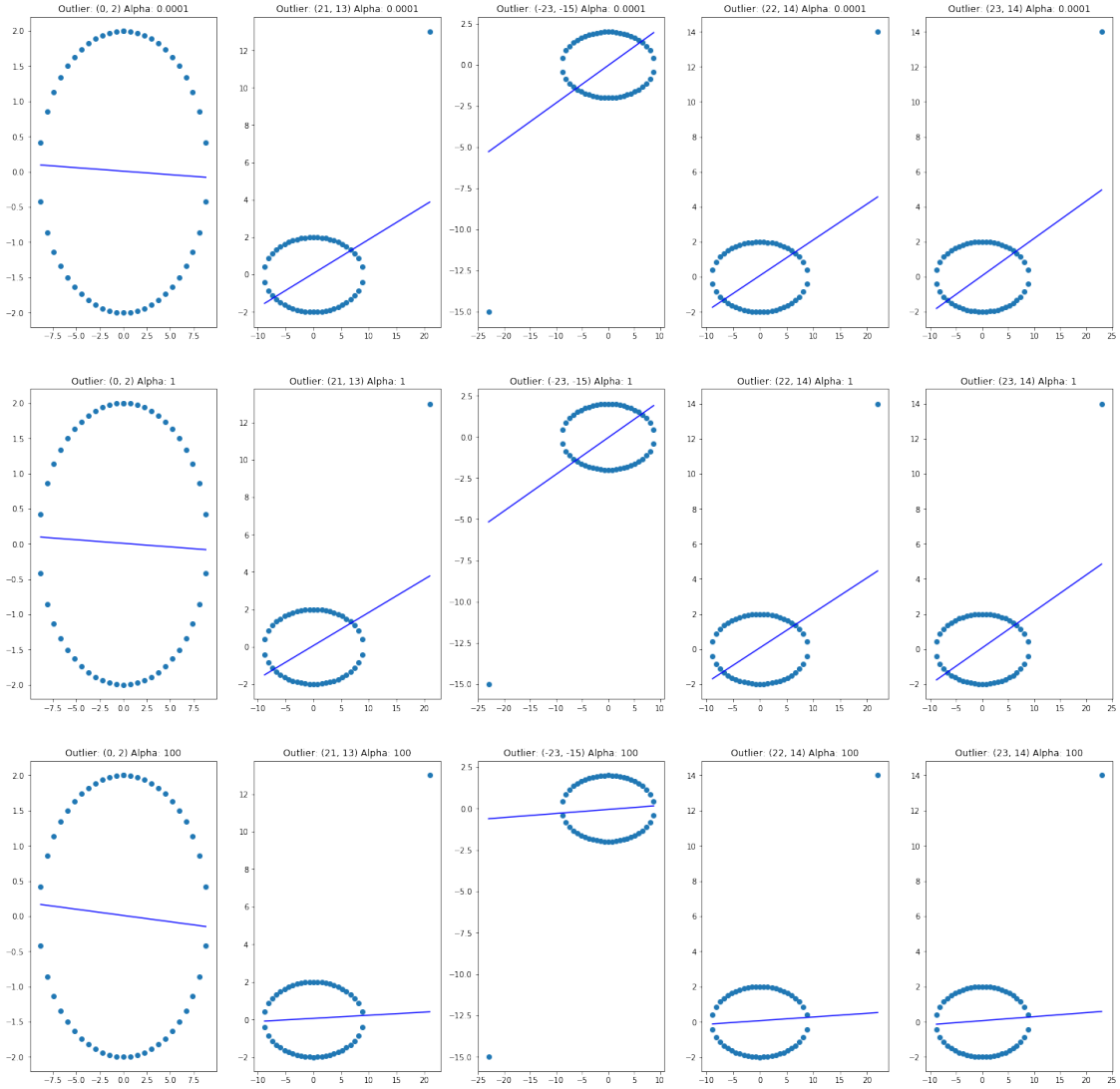
In [4]:
```
X= b * np.sin(phi)
Y= a * np.cos(phi)
```

In [5]:
```
from sklearn import linear_model
```

In [6]:
```
alphas = [0.0001, 1, 100]
outlier = [(0,2),(21, 13), (-23, -15), (22,14), (23, 14)]
```

In [7]:
```
def draw_line(coef,intercept, mi, ma):
    # for the separating hyper plane ax+by+c=0, the weights are [a, b] and the interce
    # to draw the hyper plane we are creating two points
    # 1. ((b*min-c)/a, min) i.e ax+by+c=0 ==> ax = (-by-c) ==> x = (-by-c)/a here in p
    # 2. ((b*max-c)/a, max) i.e ax+by+c=0 ==> ax = (-by-c) ==> x = (-by-c)/a here in p
    points=np.array([[mi, coef*mi+intercept],[ma, coef*ma+intercept]])
    plt.plot(points[:,0], points[:,1],'b')
```

In [8]:
```
plt.figure(figsize=(25,25))
k = 1
for i in range(len(alphas)):
    for j in range(len(outlier)):
        plt.subplot(3, 5,k+j)
        plt.title('Outlier: {} Alpha: {}'.format(outlier[j],alphas[i]))
        X_new = np.append(X,outlier[j][0]).reshape(-1,1)
        Y_new = np.append(Y,outlier[j][1])
#         print(len(X_new))
        cfg = linear_model.SGDRegressor(alpha=alphas[i], eta0=0.001, learning_rate='co
        cfg.fit(X_new, Y_new)
        draw_line(cfg.coef_.ravel(),cfg.intercept_, min(X_new), max(X_new))
        plt.scatter(X_new,Y_new)
    k+=5
plt.show()
```


Outlier: (0, 2) Alpha: 0.0001
Outlier: (21, 13) Alpha: 0.0001
Outlier: (-23, -15) Alpha: 0.0001
Outlier: (22, 14) Alpha: 0.0001
Outlier: (23, 14) Alpha: 0.0001
Outlier: (0, 2) Alpha: 1
Outlier: (21, 13) Alpha: 1
Outlier: (-23, -15) Alpha: 1
Outlier: (22, 14) Alpha: 1
Outlier: (23, 14) Alpha: 1
Outlier: (0, 2) Alpha: 100
Outlier: (21, 13) Alpha: 100
Outlier: (-23, -15) Alpha: 100
Outlier: (22, 14) Alpha: 100
Outlier: (23, 14) Alpha: 100