Task-C: Regression outlier effect.

Objective:Visualization best fit linear regression line for different scenarios

```
# 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
```

```
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


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()
```

```
X= b * np.sin(phi)
Y= a * np.cos(phi)
```

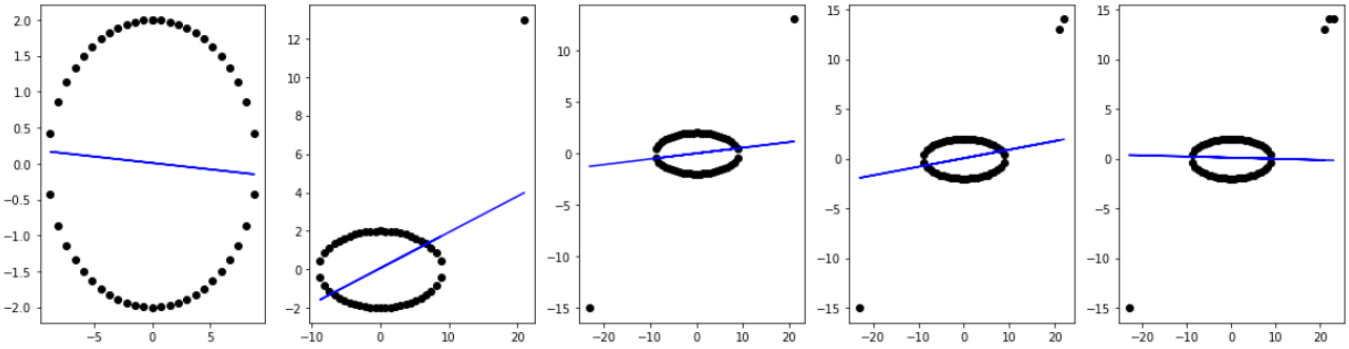
1. As a part of this assignment you will be working the regression problem and how regularization helps to get r

2. Use the above created X, Y for this experiment.

3. to do this task you can either implement your own SGDRegression(prefered) excatly similar to "SGD assignm you can use the SGDRegression of sklearn, for example "SGDRegressor(alpha=0.001, eta0=0.001, learning_rat note that you have to use the constant learning rate and learning rate ***eta0*** *initialized.*

4. as a part of this experiment you will train your linear regression on the data (X, Y) with different regularizat observe how prediction hyper plan moves with respect to the outliers

5. This the results of one of the experiment we did (title of the plot was not metioned intentionally)



in each iteration we were adding single outlier and observed the movement of the hyper plane.

6. please consider this list of outliers: [(0,2),(21, 13), (-23, -15), (22,14), (23, 14)] in each of tuple the first elemet is the input feature(X) and the second element is the output(Y)

7. for each regularizer, you need to add these outliers one at time to data and then train your model again on the updated data.

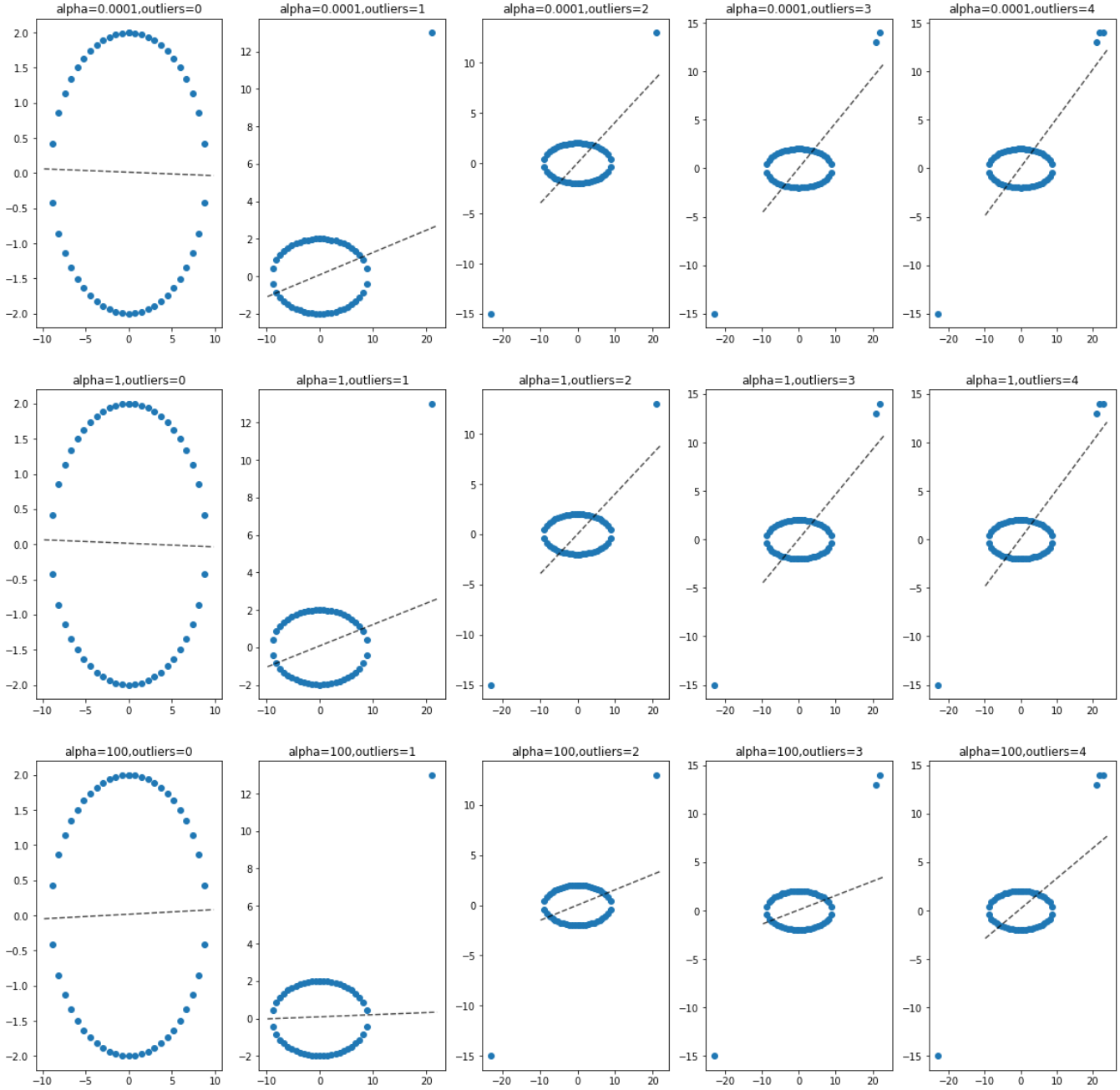
*8. you should plot a 3*5 grid of subplots,*
where each row corresponds to results of model with a single regularizer.

9. Algorithm:

for each regularizer:
for each outlier:
#add the outlier to the data
#fit the linear regression to the updated data
#get the hyper plane
#plot the hyperplane along with the data points

10. MAKE SURE YOU WRITE THE DETAILED OBSERVATIONS, PLEASE CHECK THE LOSS FUNCTION IN T
(please do search for it).

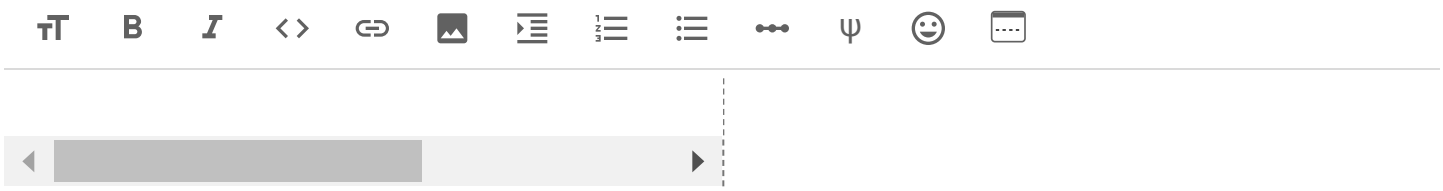
```
from sklearn.linear_model import SGDRegressor

index=1
plt.figure(figsize=(20,20))

outliers=[(0,2),(21,13),(-23,-15),(22,14),(23,14)]
reg_12=[0.0001,1,100]


for alpha in reg_12:
  X_,Y_=X,Y

  for i,outlier in enumerate(outliers):
    plt.subplot(len(reg_12),len(outliers),index)
    index+=1

    X_,Y_=np.hstack((X_,outlier[0])),np.hstack((Y_,outlier[1]))

    plt.scatter(X_,Y_)
    plt.title(f'alpha={alpha},outliers={i}')

    sgd=SGDRegressor(alpha=alpha,eta0=0.001,learning_rate='constant',random_state=0)
    sgd.fit(X_.reshape(-1,1),Y_)
    coef,intercept=sgd.coef_,sgd.intercept_
    xs=np.linspace(X.min()-1,X_.max()+1)
    ys=xs*coef+intercept

    plt.plot(xs,ys,'black',linestyle='--',alpha=0.7)

plt.show()
```


alpha=0.0001,outliers=0
alpha=0.0001,outliers=1
alpha=0.0001,outliers=2
alpha=0.0001,outliers=3
alpha=0.0001,outliers=4
alpha=1,outliers=0
alpha=1,outliers=1
alpha=1,outliers=2
alpha=1,outliers=3
alpha=1,outliers=4
alpha=100,outliers=0
alpha=100,outliers=1
alpha=100,outliers=2
alpha=100,outliers=3
alpha=100,outliers=4

When there are no outliers there is no need of regularization as seen it doesn't show mucj effect

when a single outlier is added the model performance get started decreasing so we need large Regularization it seems

with more than one outlier the model becomes bad



✓ 2s completed at 12:48 PM