ML2 Assignment Problems

July 3, 2020

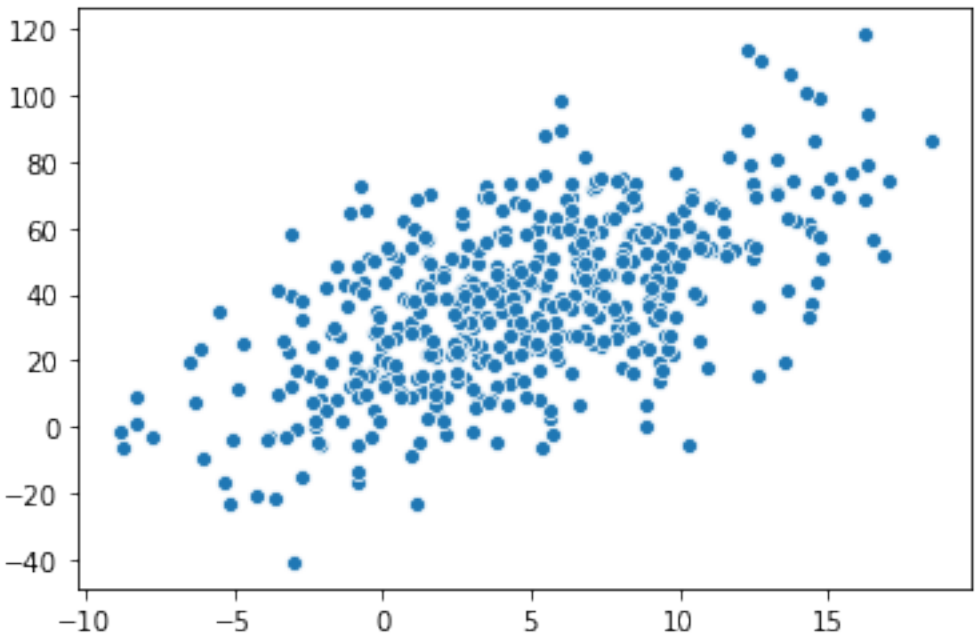
0.1 Explain the reasons in your own words

Look at the codes mentioned below and explain the output

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

np.random.seed(0)
x = np.random.normal(5,5,500)
y = 3*x + 20 + np.random.normal(5,20,500)

sns.scatterplot(x,y)
plt.show()
```



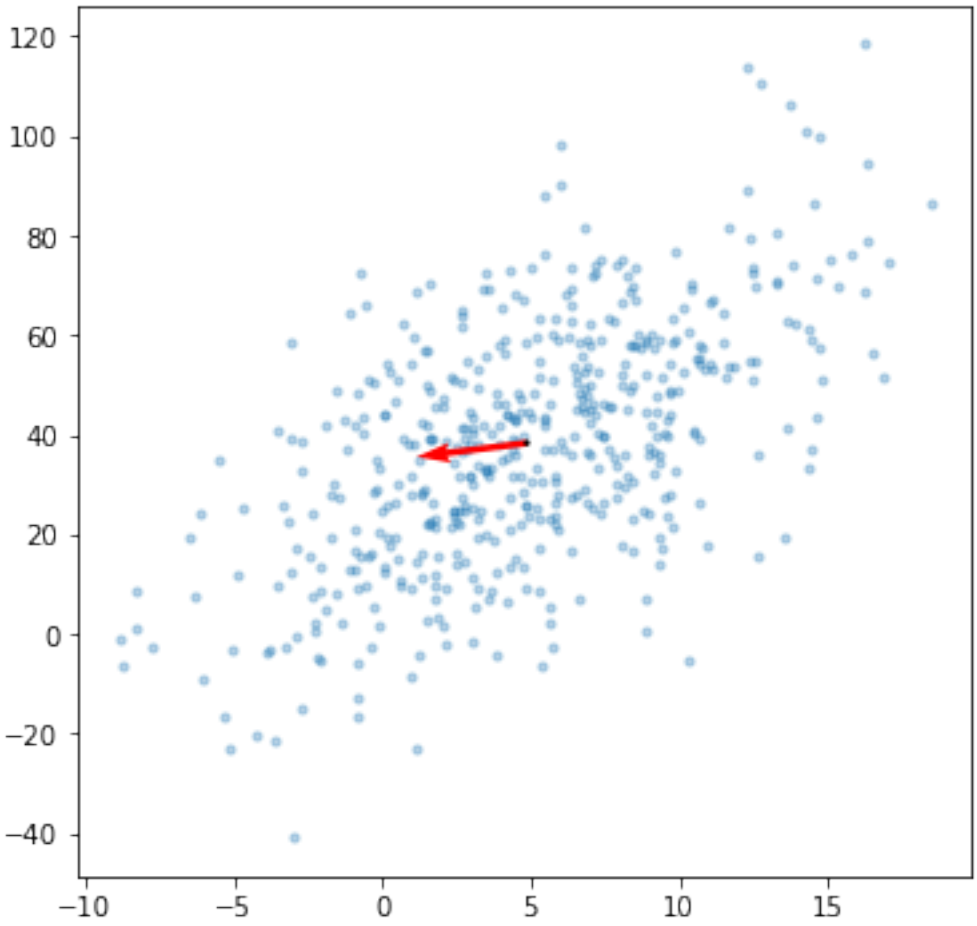
[2]:
```
arr = np.vstack([x,y]).T
```

[3]:
```
covar_mat = pd.DataFrame(arr).cov()
correl_mat = pd.DataFrame(arr).corr()
```

[4]:
```
def plot_pca_2D(arr, symm_mat, cov_mat=True):
    n_comp = 2
    mean_x, mean_y = (np.mean(arr[:,0]), np.mean(arr[:,1]))
    print(mean_x, mean_y)
    plt.figure(figsize=(6,6))
    if cov_mat:
        eigen_val, eigen_vec = np.linalg.eig(symm_mat)
        plt.scatter(arr[:,0],arr[:,1], alpha=0.3, s=10)
        plt.quiver(np.array([mean_x,mean_x]),np.array([mean_y,mean_y]),␣
↪eigen_vec[:,0], eigen_vec[:,1],
                   color=['r','black'], scale=eigen_val*0.5)
        plt.show()
    else:
        cor = symm_mat
        eigen_val, eigen_vec = np.linalg.eig(symm_mat)
        plt.scatter(arr[:,0],arr[:,1], alpha=0.3, s=10)
        plt.quiver(np.array([mean_x,mean_x]),np.array([mean_y,mean_y]),␣
↪eigen_vec[:,0], eigen_vec[:,1],
                   color=['r','black'], scale=eigen_val*5)
        plt.show()
```

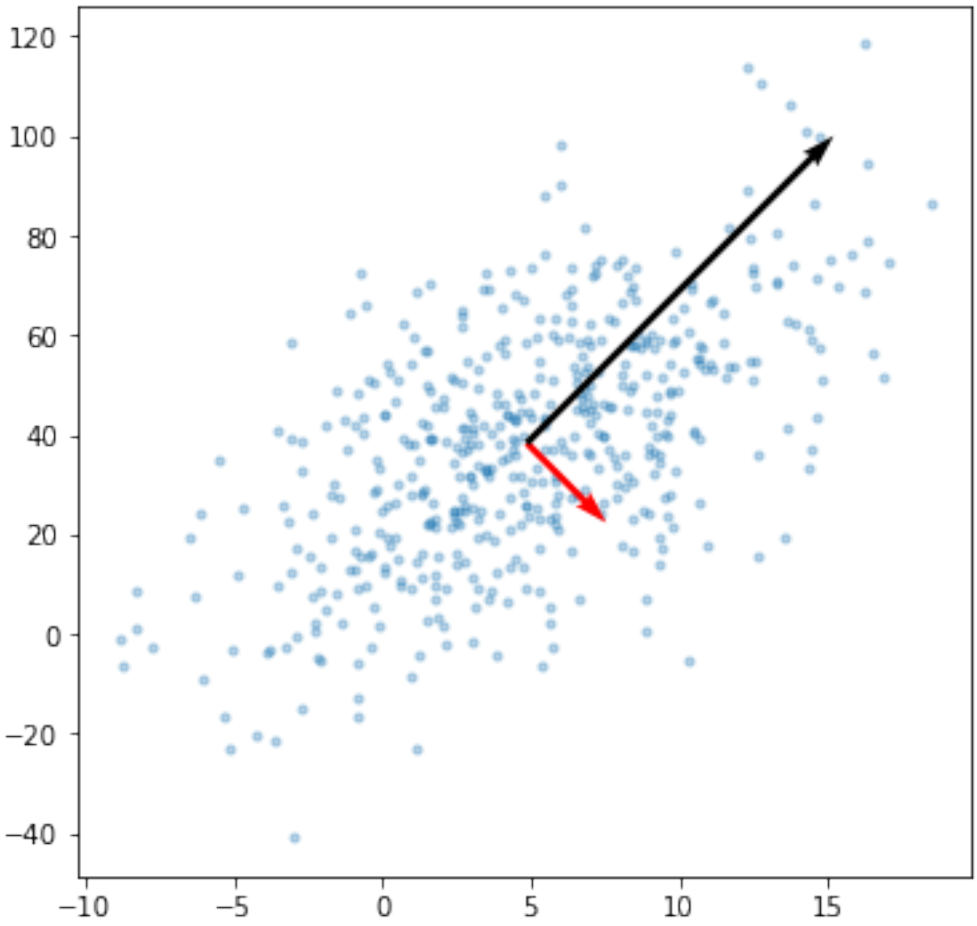
[5]:
```
plot_pca_2D(arr, symm_mat=covar_mat)
```

```
4.873227803337831 38.31650389705435
```



[6]:
```
plot_pca_2D(arr, symm_mat=correl_mat, cov_mat=False)
```

```
4.873227803337831 38.31650389705435
```

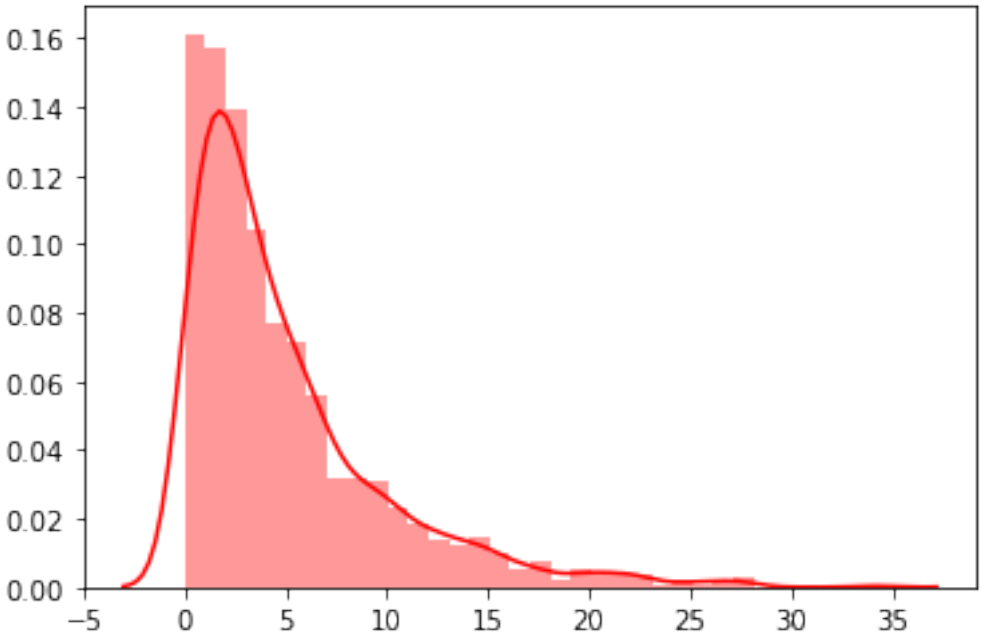


What wrong is happenning with covariance matrix?

1 Solve the problem with MLE

You are given a dataset which was generated using exponential distribution. Find out the model parameters to extract the pattern using MLE. Write a simple python program to solve the problem

[7]:
```
np.random.seed(123)
exp_data = np.random.exponential(5,size=1000)
sns.distplot(exp_data,color='r')
plt.show()
```



2 Solve using EM algorithm

There are two biased coins available. You can choose any one of the coins and toss it for 20 times. The outcomes are to be recorded. The problem is, the coins cannot be differentiated from each other and hence you cannot know which coin was chosen and tossed for 20 times in a particular experiment. Assume that the experiment was run 50 times and you have got the full information about the outcome. Try to estimate the probability of success of each coin (head = success). The outcomes are generated using the codes below. Use your own python codes from scratch.

[8]:
```
np.random.seed(123)
r1 = np.random.binomial(1,0.7,(25,20)) # p for one of the coins is 0.7
r2 = np.random.binomial(1,0.4,(25,20)) # p for the other coins is 0.4
outcome = np.vstack([r1,r2])
np.random.shuffle(outcome)
outcome[:5,:]
```

[8]:
```
array([[1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0],
       [1, 0, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 1],
       [1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1],
       [1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1],
       [1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1, 0, 1, 1, 1, 1, 1, 1]])
```

3 Visualize impact of parameter tuning

Take any dataset of you choice (at least 10k data points) and take any one of the following models:

- SVM
- GBM
- Xgboost
- Random Forest

and vary one hyperparameter at a time to estimate training error and validation error. Vary 5 such hyperparameters to plot 5x2 lineplots for training error and validation error. Based on the line plots, can you infer anythin?

[ ]: