Homework #4

Due on December 26, 2019 at 11:55pm

Professor Hoda Mohammadzade

Amirhossein Yousefi

97206984

Problem 7

Section a and b:

In order to do LDA locally we should make data matrix for each test sample respective to *kernal* we choose to use that here is *RBF*. So for each test sample, each row of data matrix is multiplied by corresponding *RBF kernel* between that test sample and mentioned row. Overally each row of data matrix is multiplied by a coefficient obtained by kernel and then new data matrix is created.So we have data matrices as much as the cardinality of test set. Then we train our model by new data matrix and then make prediction according to all models corresponding to individual test sample and then the accuracy is computed for each model.We should note that here we train models as much as test samples we have and then we evaluate all models in one loop.This problem is done by *PCA* and without *PCA* and then the corresponding accuracy reported. After that decision boundary is drawn.We also used*DML* package which calculate *local LDA* and then the corresponding accuracy is reported.

Figure 1: accuracy for different parameters of γ

```
10 gammaa=[.00001,.00004,.00008,.0001,.0005,.0008,.001,.005,.01,.1,.5,1,5,10,20,40,80,100]
```

Figure 2: different γ values for *RBf* kernel

```
{'num_dims': 64, 'acum_eig': 0.9975602429526055}
/usr/local/lib/python3.6/dist-packages/ipykernel_launc
```

Figure 3: local *LDA* using DML package

```
accuracy for LDA is 0.9444444444444444
/usr/local/lib/python3.6/dist-packages/sklearn/discriminan
  warnings.warn("Variables are collinear.")
```

Figure 4: Local *LDA* from scratch using all 64 features

```
0.9351851851851852
```

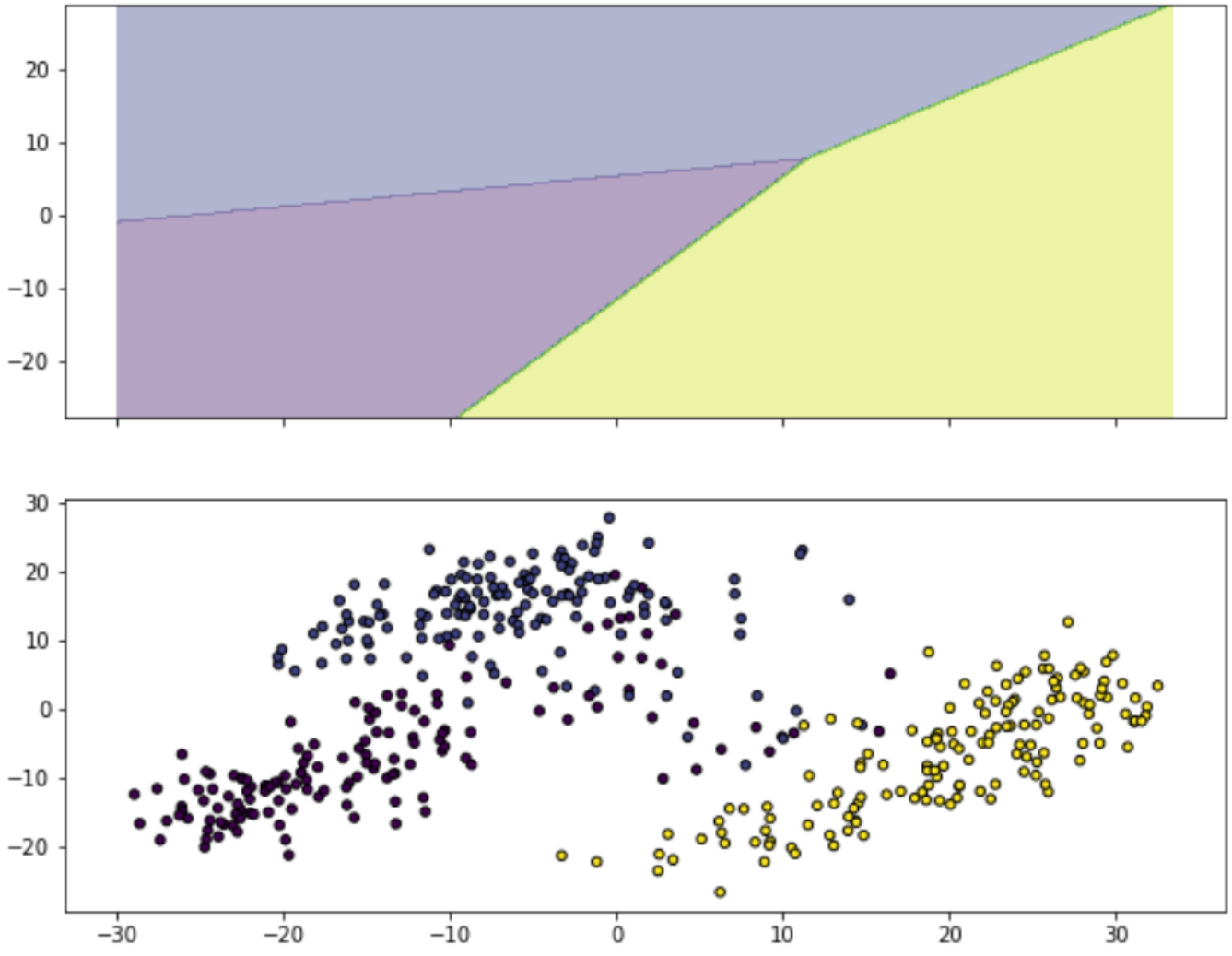

Figure 5: Local *LDA* from scratch using just two features by using *PCA* and decision boundary for that

Problem 8

Section a and b:
Soft margin svm is done by *scikit* and then by cross validation, optimal value of C is obtained which controls the degree of being hard margin or soft margin or in other word its specify the amount of passing the margin. Result are shown in bellow.Note that optimal hyper parameters is found by *GridSearchCV*. At both part a dictionary for parameters fed into the learning procedure.

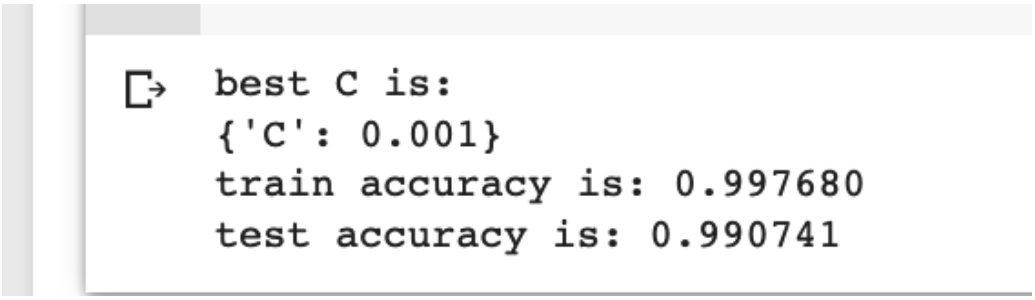

Figure 6: Results of section a

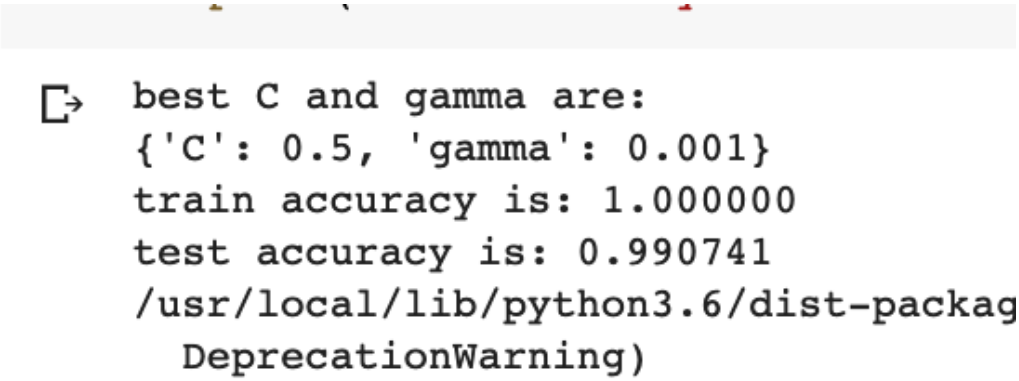

Figure 7: Results of section b

```
parameters = {'C': [0.0001, 0.001, 0.01, 0.1,.5, 1,2,3, 10, 100],'gamma':[0.000001,.001,.005,.01,.05,.1,.5,1,5,10,20]}
```

Figure 8: Hyper parameters for part b

```
parameters = {'C': [0.0001, 0.001, 0.01, 0.1, 1, 10, 100]}
```

Figure 9: Hyper parameters for part a