22417.202210 Homework 4 - Support Vector Machines

Pankaj Kumar Jatav

TOTAL POINTS

22 / 25

QUESTION 1

1 Plot normal 1 / 1

✓ **- 0 pts** *Correct*

- 1 pts Blank

QUESTION 2

2 Plot banana 1 / 1

✓ **- 0 pts** *Correct*

QUESTION 3

3 1.b 1 / 1

✓ **- 0 pts** *Correct*

QUESTION 4

4 1.c 0 / 1

- 0 pts Correct

✓ **- 1 pts** *Incorrect*

QUESTION 5

5 1.d 0 / 1

- 0 pts Correct

✓ **- 1 pts** *Incorrect*

QUESTION 6

6 1.e 1 / 1

✓ **- 0 pts** *Correct*

QUESTION 7

7 Discussion 3 / 3

✓ **- 0 pts** *Correct*

- 1 pts Did not mention things related to smaller margin 'and' higher accuracy

QUESTION 8

8 C Values 2 / 2

✓ **- 0 pts** *Correct*

- 1 pts Different from one correct answer (correct: C=50 and 100)

- 2 pts Different from two or more answers (correct: C=50 and 100)

QUESTION 9

9 Meshgrid C=20 1 / 1

✓ **- 0 pts** *Correct*

QUESTION 10

10 1.i 1 / 1

✓ **- 0 pts** *Correct*

- 1 pts Incorrect

QUESTION 11

11 1.j 1 / 2

- 0 pts Correct

✓ **- 1 pts** *The question asks for maximum examples*

- 2 pts Incorrect (correct: 197)

QUESTION 12

12 1.k 2 / 2

✓ **- 0 pts** *Correct*

- 2 pts Incorrect

QUESTION 13

13 Default rbf on banana **2 / 2**

✓ **- 0 pts** *Correct*

- 1 pts Imprecise non-linear decision boundaries

- 2 pts Incorrect

QUESTION 14

14 average CV score **1 / 1**

✓ **- 0 pts** *Correct*

- 1 pts Incorrect (correct: 0.830)

QUESTION 15

15 params1 **1 / 1**

✓ **- 0 pts** *Correct*

- 0.5 pts One of parameters is incorrect

- 1 pts Incorrect (ans: C = 2^-3, sigma = 2^1)

QUESTION 16

16 params2 **1 / 1**

✓ **- 0 pts** *Correct*

- 0.5 pts One of parameters is incorrect

- 1 pts Incorrect (ans: C = 8, sigma = 8)

QUESTION 17

17 accuracy 1 **1 / 1**

✓ **- 0 pts** *Correct*

- 0.5 pts Close to correct answer 0.985

- 1 pts Incorrect (ans: 0.985)

QUESTION 18

18 accuracy 2 **1 / 1**

✓ **- 0 pts** *Correct*

- 0.5 pts Almost correct

- 1 pts Incorrect

QUESTION 19

19 1.p **1 / 1**

✓ **- 0 pts** *Correct*

- 1 pts Incorrect (Yes/More or less)

QUESTION 20

20 Collaboration Questions **0 / 0**

✓ **- 0 pts** *Correct*

HOMEWORK 4
SUPPORT VECTOR MACHINES AND KERNELS[1]

CS 688 MACHINE LEARNING (SPRING 2022)
https://nlp.cs.gmu.edu/course/cs688-spring22/

OUT: March 31, 2022
DUE: April 7, 2022

Your name: Pankaj Kumar Jatav

Your GID: 01338769

[1]Compiled on Sunday 17th April, 2022 at 03:32

1 Written Questions [25 pts]

1.1 Support Vector Machines

1. In this programming exercise we will implement SVMs and play a bit with their hyperparameters, in order to better understand them. We will use the Python `scikit-learn` library, so make sure you have your Python working and install the package.

   We will use two datasets: `mynormaldistdataset.mat` and `mybananadataset.mat`, released with the homework package.

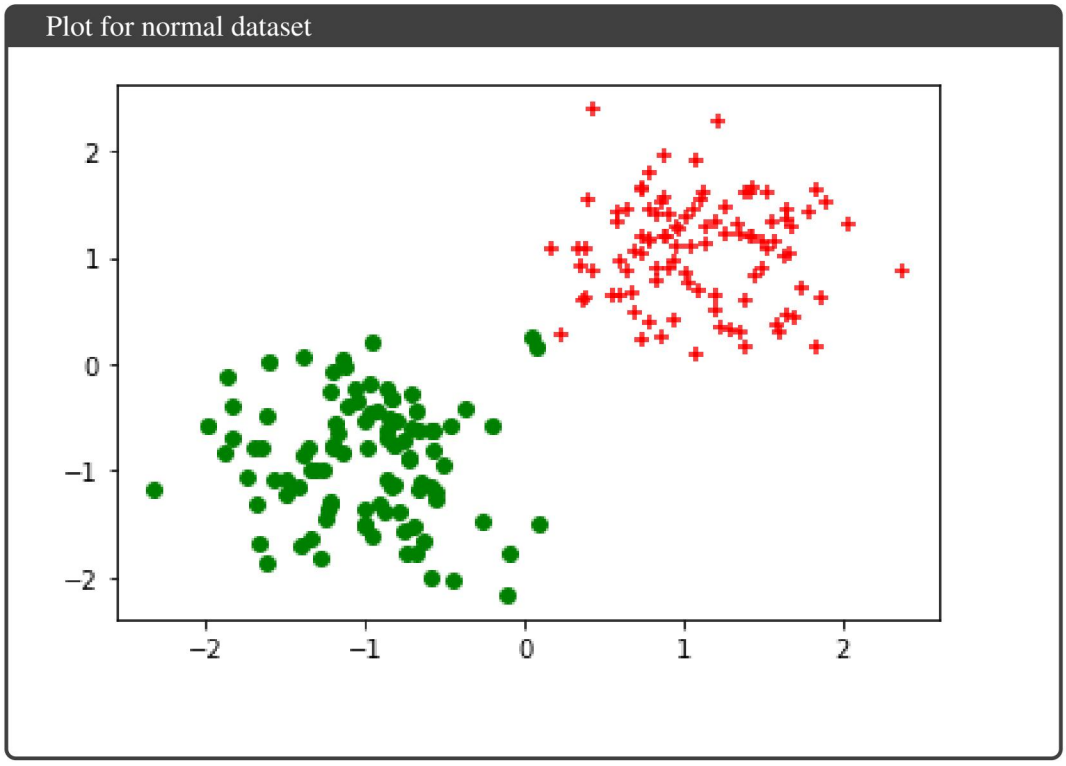
   Let's load the first dataset:

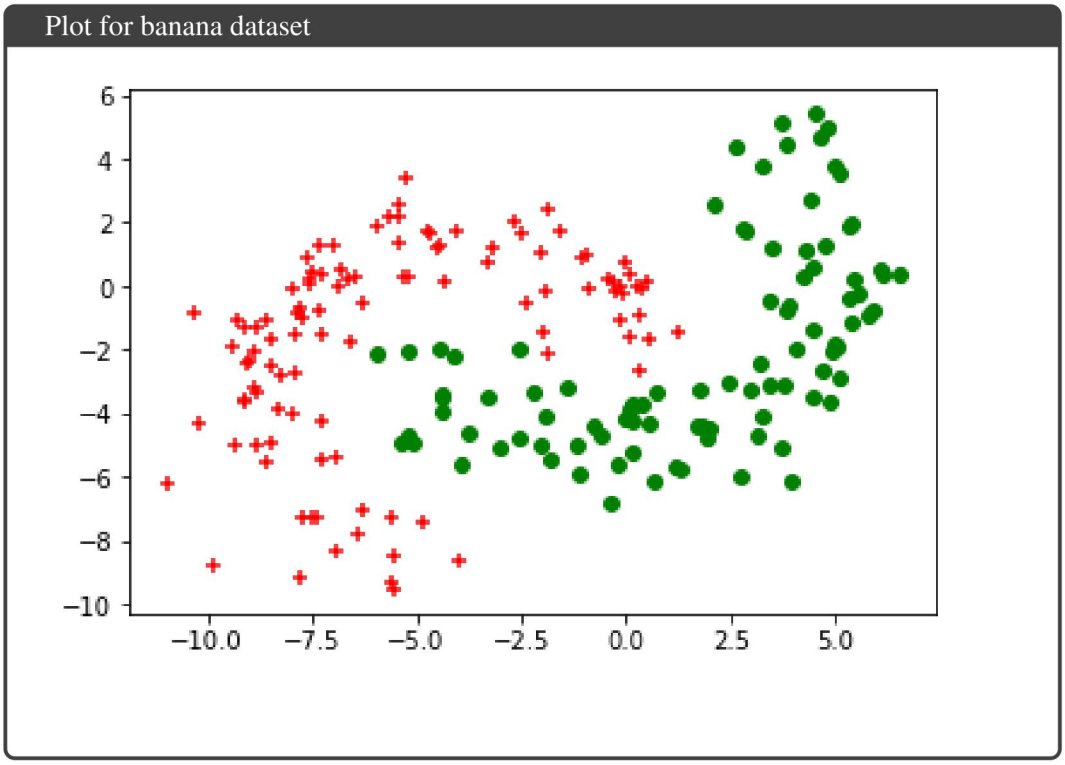
```
import scipy.io
import numpy as np
normal = scipy.io.loadmat('mynormaldistdataset.mat')
normal_train_X = normal['A']
normal_train_Y = np.ravel(normal['a'])
```

   For the second dataset, you'll need to load both train and test data:

```
banana = scipy.io.loadmat('mybananadataset.mat')
banana_train_X = banana['A']
banana_train_Y = np.ravel(banana['a'])
banana_test_X = banana['B']
banana_test_Y = np.ravel(banana['b'])
```

   (a) (2 points) Let's first visualize the datasets. You can use matplotlib or seaborn or any other package you prefer. Create 2-D scatter plots for each dataset, with each class having either a different color or a different marker (or both, for color-blind-friendliness!)





(b) (1 point) Would a perceptron reach zero training error in the normal dataset?

- [x] Yes
- [ ] No

(c) (1 point) Would a perceptron reach zero training error in the banana dataset?

- [x] Yes
- [ ] No

(d) (1 point) Would a linear SVM reach zero training error in the normal dataset?

- [ ] Yes
- [x] No

(e) (1 point) Would a linear SVM reach zero training error in the banana dataset?

- [ ] Yes
- [x] No

Now, we will train our SVM. With scikit-learn, we can do something like the following to train and obtain a prediction for a new example. **Note: in all experiments below, make sure to set the `random_state` to 42, as shown below.**:

```
from sklearn import svm

# In this case we assume 2d features.
# X: numpy array with shape (n, 2), all n data points.
# Y: numpy array of shape (n), labels in {0, 1, 2, 3, ...} corresponding to X

classifier = svm.SVC(C=10, kernel='linear', random_state=42)
classifier.fit(X, Y)

print("The feature [1, 2] is classified as:", classifier.predict([[1, 2]]))
```

First, train on the normal dataset. We'll stick with the linear SVM for now, but we will vary the C parameter. Rerun the experiments a couple of times, and visualize the data using something like the following:

```
import numpy as np
import matplotlib.pyplot as plt

def make_meshgrid(X, h=.02):
    """Make a meshgrid covering the range of X. This will be used to draw classification regions

    Args:
        X: numpy array with shape [n, 2] containing 2d feature vectors.
        h: parameter controlling the resolution of the meshgrid
    """
    x = X[:, 0]
    y = X[: ,1]
    x_min, x_max = x.min() - 1, x.max() + 1
    y_min, y_max = y.min() - 1, y.max() + 1
    xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))
    return xx, yy

def scatter(X, Y, xx, yy, Z):
    """
    Scatter plot with classification regions

    Args:
        X: numpy array of shape [n, 2] where n is the total number of datapoints
        Y: numpy array of shape [n] containing the labels {1, 2, 3, ...} of X
        xx: meshgrid x
        yy: meshgrid y
        Z: The result of applying some prediction function on all points in xx and yy
    """
    x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
    y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1

    plt.figure()
    # Color class regions
    plt.gca().contourf(xx, yy, Z, alpha=0.7)
    # Data points
```

```
    plt.scatter(X[:, 0], X[:, 1], c=Y, cmap=plt.cm.viridis, marker='o',
    edgecolors='k')
    plt.xlim(x_min, x_max)
    plt.ylim(y_min, y_max)
    plt.axes().set_aspect('equal')
    plt.grid()
    plt.tight_layout()

    plt.show()

# Given X and Y as explained above, we can display the data and classification
    boundaries as
classifier = svm.SVC(C=100.0, kernel='linear')
classifier.fit(X, Y)

xx, yy = make_meshgrid(X)
Z = classifier.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)
scatter(X, Y, xx, yy, Z)
```

(f) (3 points) How do the support vectors and the boundary change with the parameter? Write one-two sentences.

Linear SVM varying C

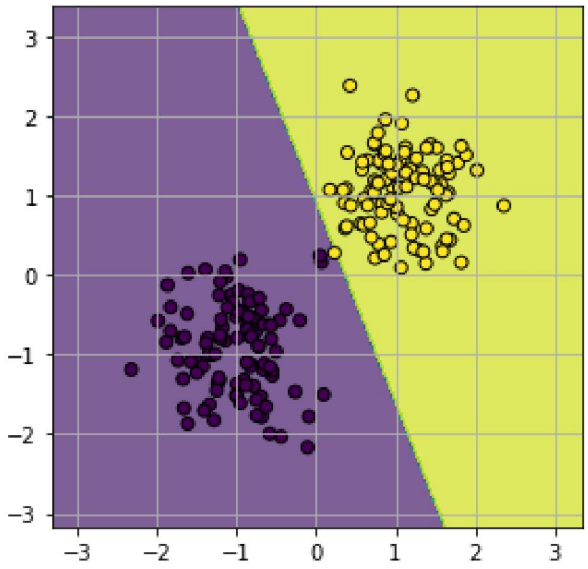
The C parameter instructs the SVM optimizer how much you wish to avoid misclassifying each training example. For large values of C, the optimization will select a smaller-margin hyperplane if it performs a better job of accurately classifying all of the training points. A very small value of C, on the other hand, will encourage the optimizer to seek a larger-margin separating hyperplane, even if that hyperplane misclassifies more points. Misclassified cases should be expected for very small values of C, even if your training data is linearly separable.

(g) (2 points) Which of the following values for C help the SVM reach 0 training error on the normal dataset?

- ☐ $C = 0.01$
- ☐ $C = 1$
- ☐ $C = 20$
- ☒ $C = 50$
- ☒ $C = 100$

(h) (1 point) Show the `meshgrid` you obtain on the normal dataset with $C = 20$:

Meshgrid for normal dataset



(i) (1 point) Try to remove some of the non-support-vectors from the dataset and rerun (e.g. with $C = 100$). Does the solution change?

- ○ Yes
- ● No

(j) (2 points) For $C = 100$, how many training examples could you remove before the solution changes?

Your answer

1

(k) (2 points) Now let's switch to the banana dataset. Try various values of the C parameter with a linear SVM. Can the linear SVM classifier make a good separation of the feature space?

- ○ Yes
- ● No

Now let's change kernel to a RBF (Radial Basis Function), and rerun. Try changing the σ-parameter. Notice that in the lecture we defined the rbf kernel as

$$K(x, y) = \exp\{-\frac{1}{\sigma^2}||x - y||^2\},$$

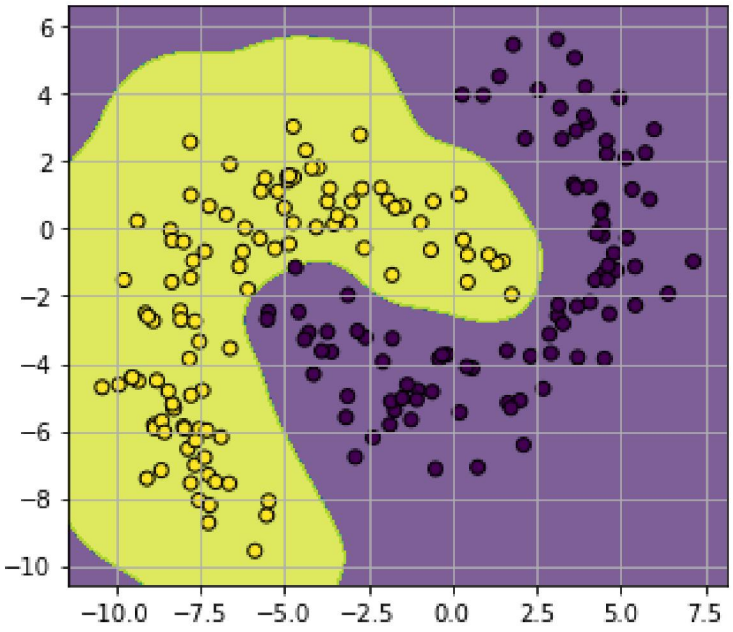
but in sklearn, σ is expressed through a γ parameter, and the rbf kernel is given as

$$K(x, y) = \exp\{-\gamma||x - y||^2\}.$$

```
# svm.SVC() with its default parameter values:
num_features=2
classifier = svm.SVC(C=1.0, kernel='rbf', gamma=1/num_features)
```

(l) (2 points) Plot the meshgrid on the banana dataset with the default settings. Make sure you know why we now get non-linear decision boundaries.

Default RBF kernel on banana dataset



Now, implement a grid search of the C and σ parameters based on 10-fold cross-validation of the banana training data (the A-dataset).

You can either implement cross-validation yourself (a good exercise!), or you can use the scikit-learn helper functions. See here: https://scikit-learn.org/stable/modules/cross_validation.html. You should be able to use something like this:

```
# This following example demonstrates how to run 5-fold cross-validation,
# estimating the accuracy of a linear kernel support vector machine on a
# dataset by splitting the data, fitting a model and computing the score 5
# consecutive times (with different splits each time):

from sklearn.model_selection import cross_val_score

# We have a dataset (X,y)
# Define a classifier, e.g.:
clf = svm.SVC(kernel='linear', C=1, random_state=42)
scores = cross_val_score(clf, X, y, cv=5)

```

```
# Scores is a list of the accuracy in each fold.
# print(scores) would something like print: array([0.96..., 1. , 0.96...,
    0.96..., 1. ])
```

To check that you have correctly implemented cross-validation, try performing 10-fold CV with the default rbf kernel we used above. With a fixed random_state, you should get scores:

```
>>> scores
[0.95 1.   1.   1.   1.   0.95 1.   1.   1.   1.  ]

```

With cross-validation implemented, now run a grid-search over C and σ (set $\gamma = frac1\sigma$ for simplicity), using the following ranges:

$$C \in \{2^{-5}, 2^{-3}, 2^{-1}, \ldots, 2^{13}, 2^{15}\}$$
$$\sigma \in \{2^{-15}, 2^{-13}, 2^{-11}, \ldots, 2^{1}, 2^{3}\}$$

(m) (1 point) What is the average CV score for $C = 7$ and $\sigma = -3$? Write your answer with up to 3 decimal digits.

Average CV score

0.83

(n) (2 points) Find the values of C and σ that yield the best CV scores. There should be two pairs of values. What are they?

Optimal parameters 1

$C = 0.125 \quad , \sigma = 2$

Optimal parameters 2

$C = 8 \quad , \sigma = 8$

Now, use these two sets of parameters to retrain two on the entire A–dataset, and then test on the B-dataset (banana_test). As a reminder, after you fit (train) your classifier, you can easily obtain predictions $\hat{Y}$ with:

```
Y_hat = clf.predict(banana_test_X)
```

(o) (2 points) What accuracy do you obtain on the test dataset when using the optimal parameters with the smaller and the larger C values?

Accuracy with small C	Accuracy with large C
accuracy = 0.985	99 accuracy =

(p) (1 point) Does the average 10-fold cross-validation estimate of the overall classification error match the result we get when testing on the independent test dataset?

- ● Yes
- ○ No
- ○ More or less

2 Collaboration Questions

After you have completed all other components of this assignment, report your answers to these questions regarding the collaboration policy. Details of the policy can be found in the syllabus.

1. Did you receive any help whatsoever from anyone in solving this assignment? If so, include full details.
2. Did you give any help whatsoever to anyone in solving this assignment? If so, include full details.
3. Did you find or come across code that implements any part of this assignment? If so, include full details.

Your Answer

No No No