Benjamin Ye

CS/CNS/EE 156a: Learning Systems (Fall 2023)

November 20, 2023

**Homework 8**

|  |  |
| --- | --- |
| **Problem** | **Answer** |
| 1 | [d] |
| 2 | [a] |
| 3 | [a] |
| 4 | [c] |
| 5 | [d] |
| 6 | [b] |
| 7 | [b] |
| 8 | [c] |
| 9 | [e] |
| 10 | [c] |

**Primal Versus Dual Problem**

1. Recall that is the size of the data set and is the dimensionality of the input space. The original formulation of the hard-margin SVM problem (minimize subject to the inequality constants), without going through the Lagrangian dual problem, is

**Answer: [d] a quadratic programming problem with variables**

In the hard-margin SVM problem, we minimize using the constraint . Since the only unknowns in the constraint are and , which have dimensions of and 1, respectively, the hard-margin SVM problem is a quadratic programming problem with variables.

**SVM with Soft Margins**

*Notice: The following problems deal with a real-life data set. In addition, the computational packages you use may employ different heuristics and require different tweaks. This is a typical situation that a machine learning (ML) practitioner faces. There are uncertainties, and the answers may or may not match our expectations. Although this situation is not as “sanitized” as other homework problems, it is important to go through it as part of the learning experience.*

In the rest of the problems of this homework set, we apply soft-margin SVM to the handwritten digits from the processed U.S. Postal Service Zip Code data set. Download the data (extracted features of intensity and symmetry) for training and testing:

<http://www.amlbook.com/data/zip/features.train>

<http://www.amlbook.com/data/zip/features.test>

(The format of each row is: **digit, intensity, symmetry**.) We will train two types of binary classifiers; one-versus-one (one digit is class +1 and another digit is class , with the rest of the digits disregarded), and one-versus-all (one digit is class +1 and the rest of the digits are class ).

The data set has thousands of points, and some quadratic programming packages cannot handle this size. You may need stronger SVM packages such as **fitcsvm** in MATLAB or the free **libsvm**.

Implement SVM with soft margins on the above zip code data set by solving

When evaluating and of the resulting classifier, use binary classification error. is estimated using the test set.

Practical remarks:

1. For this homework, do not scale the data when you use libsvm or other packages or you may inadvertently change the (effective) kernel and get different results.
2. In some packages, you need to specify double precision.
3. In 10-fold cross validation, if the data size is not a multiple of 10, the sizes of the 10 subsets may be off by 1 data point.
4. Some packages have software parameters whose values affect the outcome. ML practitioners must deal with this kind of added uncertainty.

**Polynomial Kernels**

Consider the polynomial kernel , where is the degree of polynomial.

1. With and , which of the following classifiers has the highest ?

**Answer: [a] 0 versus all**

1. With and , which of the following classifiers has the lowest ?

**Answer: [a] 1 versus all**

1. Comparing the two selected classifiers from Problems 2 and 3, which of the following values is the closest to the difference between the number of support vectors of these two classifiers?

**Answer: [c] 1,800**

1. Consider the 1 versus 5 classifier with and . Which of the following statements is correct? Going up or down means strictly so.

**Answer: [d] Maximum achieves the lowest .**

1. In the 1 versus 5 classifier, comparing with , which of the following statements is correct?

**Answer: [b] When , the number of support vectors is lower at .**

(The sample program output is on the next page, and the

Python 3 source code is at the end of this document. )

**Sample program output**

[Homework 8 Problems 2–4]

SVM with soft margins (C=0.01) and polynomial kernel (Q=2):

classifier number of support vectors in-sample error out-of-sample error

0 vs. all **2179** **0.105884** 0.111609

1 vs. all **386** **0.014401** 0.021923

2 vs. all 1970 0.100261 0.098655

3 vs. all 1950 0.090248 0.082711

4 vs. all 1856 0.089425 0.099651

5 vs. all 1585 0.076258 0.079721

6 vs. all 1893 0.091071 0.084704

7 vs. all 1704 0.088465 0.073244

8 vs. all 1776 0.074338 0.082711

9 vs. all 1978 0.088328 0.088191

[Homework 8 Problems 5–6]

SVM with soft margins (C=1) and polynomial kernel (Q=5) for 1 vs. 5 classifier:

C Q number of support vectors in-sample error out-of-sample error

0.0001 2.0 236.0 0.008969 0.016509

0.0010 2.0 **76.0** 0.004484 0.016509

0.0100 2.0 34.0 0.004484 0.018868

0.1000 2.0 24.0 0.004484 0.018868

1.0000 2.0 24.0 **0.003203** 0.018868

0.0001 5.0 26.0 0.004484 0.018868

0.0010 5.0 **25.0** 0.004484 0.021226

0.0100 5.0 23.0 0.003844 0.021226

0.1000 5.0 25.0 0.003203 0.018868

1.0000 5.0 21.0 0.003203 0.021226

**Cross Validation**

In the next two problems, we will experiment with 10-fold cross validation for the polynomial kernel. Because is a random variable that depends on the random partition of the data, we will try 100 runs with different partitions and base our answer on how many runs lead to a particular choice.

1. Consider the 1 versus 5 classifier with . We use to select . If there is a tie in , select the smaller . Within the 100 random runs, which of the following statements is correct?

**Answer: [b] is selected most often.**

1. Again, consider the 1 versus 5 classifier with . For the overall winning selection in the previous problem, the average value of over the 100 runs is closest to

**Answer: [c] 0.005**

**Sample program output**

[Homework 8 Problems 7–8]

Cross-validation error for soft margin (C=1) SVM with polynomial kernel (Q=2) for 1 vs. 5 classifier:

C cross-validation error selection rate

0.0001 0.009711 0.00

**0.0010** **0.004734 0.52**

0.0100 0.004695 0.21

0.1000 0.004772 0.11

1.0000 0.004785 0.16

**RBF Kernel**

Consider the radial basis function (RBF) kernel in the soft-margin SVM approach. Focus on the 1 versus 5 classifier.

1. Which of the following values of results in the lowest ?

**Answer: [e]**

1. Which of the following values of results in the lowest ?

**Answer: [c]**

**Sample program output**

[Homework 8 Problems 9–10]

Soft margin SVM with RBF kernel for 1 vs. 5 classifier:

C in-sample error out-of-sample error

0.01 0.003844 0.023585

1.00 0.004484 0.021226

100.00 0.003203 **0.018868**

10000.00 0.002562 0.023585

1000000.00 **0.000641** 0.023585

(The Python 3 source code is on the following page. )

**Python 3 source code**

from pathlib import Path

import numpy as np

import pandas as pd

import requests

from sklearn import svm

from sklearn.model\_selection import cross\_val\_score

from sklearn.utils import shuffle

DATA\_DIR = Path(\_\_file\_\_).parents[2] / "data"

if \_\_name\_\_ == "\_\_main\_\_":

    rng = np.random.default\_rng()

    DATA\_DIR.mkdir(exist\_ok=True)

    data = {}

    for dataset in ["train", "test"]:

        file = f"features.{dataset}"

        if not (DATA\_DIR / file).exists():

            r = requests.get(f"http://www.amlbook.com/data/zip/{file}")

            with open(DATA\_DIR / file, "wb") as f:

                f.write(r.content)

        data[dataset] = np.loadtxt(DATA\_DIR / file)

    C = 0.01

    Q = 2

    clf = svm.SVC(C=C, kernel="poly", degree=Q, gamma=1, coef0=1)

    df = pd.DataFrame(columns=["classifier", "number of support vectors",

                               "in-sample error", "out-of-sample error"])

    for digit in range(10):

        x\_train = data["train"][:, 1:]

        y\_train = 2 \* (data["train"][:, 0] == digit) - 1

        clf.fit(x\_train, y\_train)

        df.loc[digit] = (

            f"{digit} vs. all",

            clf.n\_support\_.sum(),

            1 - clf.score(x\_train, y\_train),

            1 - clf.score(data["test"][:, 1:],

                          2 \* (data["test"][:, 0] == digit) - 1)

        )

    print("\n[Homework 8 Problems 2–4]\n"

          f"Soft margin ({C=}) SVM with polynomial kernel ({Q=}):\n",

          df.to\_string(index=False), sep="")

    x\_train = data["train"][np.isin(data["train"][:, 0], (1, 5))]

    y\_train = 2 \* (x\_train[:, 0] == 1) - 1

    x\_test = data["test"][np.isin(data["test"][:, 0], (1, 5))]

    y\_test = 2 \* (x\_test[:, 0] == 1) - 1

    df = pd.DataFrame(columns=["C", "Q", "number of support vectors",

                               "in-sample error", "out-of-sample error"])

    for Q in (2, 5):

        for C in (Cs := (0.0001, 0.001, 0.01, 0.1, 1)):

            clf = svm.SVC(C=C, kernel="poly", degree=Q, gamma=1, coef0=1)

            clf.fit(x\_train[:, 1:], y\_train)

            df.loc[len(df)] = (

                C, Q, clf.n\_support\_.sum(),

                1 - clf.score(x\_train[:, 1:], y\_train),

                1 - clf.score(x\_test[:, 1:], y\_test)

            )

    print("\n[Homework 8 Problems 5–6]\n"

          f"Soft margin ({C=}) SVM with polynomial kernel ({Q=}) for "

          "1 vs. 5 classifier:\n",

          df.to\_string(index=False), sep="")

    Q = 2

    N\_runs = 100

    N\_folds = 10

    clfs = [svm.SVC(C=C, kernel="poly", degree=Q, gamma=1, coef0=1)

            for C in Cs]

    counters = np.zeros((2, len(Cs)), dtype=float)

    for \_ in range(N\_runs):

        Es\_cv = tuple(1 - cross\_val\_score(clf, x\_train[:, 1:], y\_train,

                                          cv=N\_folds).mean()

                      for clf in clfs)

        counters[0] += Es\_cv

        counters[1, np.argmin(Es\_cv)] += 1

        x\_train, y\_train = shuffle(x\_train, y\_train)

    counters /= N\_runs

    df = pd.DataFrame({"C": Cs, "cross-validation error": counters[0],

                       "selection rate": counters[1]})

    print("\n[Homework 8 Problems 7–8]\n"

          f"Cross-validation error for soft margin ({C=}) SVM with "

          f"polynomial kernel ({Q=}) for 1 vs. 5 classifier:\n",

          df.to\_string(index=False), sep="")

    df = pd.DataFrame(columns=["C", "in-sample error", "out-of-sample error"])

    for C in (0.01, 1, 100, 1e4, 1e6):

        clf = svm.SVC(C=C, gamma=1)

        clf.fit(x\_train[:, 1:], y\_train)

        df.loc[len(df)] = (

            clf.C,

            1 - clf.score(x\_train[:, 1:], y\_train),

            1 - clf.score(x\_test[:, 1:], y\_test)

        )

    print("\n[Homework 8 Problems 9–10]\n"

          "Soft margin SVM with RBF kernel for 1 vs. 5 classifier:\n",

          df.to\_string(index=False), sep="")