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 N is the size of the data set and d is the dimensionality of the input space. The original formulation of the hard-margin SVM problem (minimize $\mathbf{w}^\top\mathbf{w}/2$ subject to the inequality constants), without going through the Lagrangian dual problem, is

   Answer: [d] a quadratic programming problem with $d + 1$ variables

   In the hard-margin SVM problem, we minimize $\mathbf{w}^\top\mathbf{w}/2$ using the constraint $y_n(\mathbf{w}^\top x_n + b) \geq 1$. Since the only unknowns in the constraint are $\mathbf{w}$ and b, which have dimensions of d and 1, respectively, the hard-margin SVM problem is a quadratic programming problem with $d + 1$ 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 -1, with the rest of the digits disregarded), and one-versus-all (one digit is class $+1$ and the rest of the digits are class -1).

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

$$\min_{\alpha} \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_n\alpha_m y_n y_m K(\mathbf{x}_n, \mathbf{x}_m) - \sum_{n=1}^{N}\alpha_n \quad \text{s.t.} \quad \sum_{n=1}^{N} y_n\alpha_n = 0\,, \;\; 0 \le \alpha_n \le C \;\; \text{for n} = 1, \ldots, N$$

When evaluating E_{in} and E_{out} of the resulting classifier, use binary classification error. E_{out} is estimated using the test set.

Practical remarks:

(i) 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.
(ii) In some packages, you need to specify double precision.
(iii) 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.
(iv) 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 $K(\mathbf{x}_n, \mathbf{x}_m) = (1 + \mathbf{x}_n^\top \mathbf{x}_m)^Q$, where Q is the degree of polynomial.

2. With $C = 0.01$ and $Q = 2$, which of the following classifiers has the highest E_{in}?

   Answer: [a] 0 versus all

3. With $C = 0.01$ and $Q = 2$, which of the following classifiers has the lowest E_{in}?

   Answer: [a] 1 versus all

4. 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

5. Consider the 1 versus 5 classifier with $Q = 2$ and $C \in \{0.001, 0.01, 0.1, 1\}$. Which of the following statements is correct? Going up or down means strictly so.

   Answer: [d] Maximum C achieves the lowest E_{in}.

6. In the 1 versus 5 classifier, comparing $Q = 2$ with $Q = 5$, which of the following statements is correct?

   Answer: [b] When $C = 0.001$, the number of support vectors is lower at $Q = 5$.

(The Python 3 source code is on pages 6–8.) The output used to answer Problems 2–6 is

```
[HW8 P2-4]
SVM with soft margins (C=0.01) and polynomial kernel (Q=2):
  0 vs. all: N_sv=2,179, E_in=0.1059, E_out=0.1116
  1 vs. all: N_sv=386, E_in=0.0144, E_out=0.0219
  2 vs. all: N_sv=1,970, E_in=0.1003, E_out=0.0987
  3 vs. all: N_sv=1,950, E_in=0.0902, E_out=0.0827
  4 vs. all: N_sv=1,856, E_in=0.0894, E_out=0.0997
  5 vs. all: N_sv=1,585, E_in=0.0763, E_out=0.0797
  6 vs. all: N_sv=1,893, E_in=0.0911, E_out=0.0847
  7 vs. all: N_sv=1,704, E_in=0.0885, E_out=0.0732
  8 vs. all: N_sv=1,776, E_in=0.0743, E_out=0.0827
  9 vs. all: N_sv=1,978, E_in=0.0883, E_out=0.0882

[HW8 P5-6]
SVM with soft margins and polynomial kernel (1 vs. 5 classifier):
  C=0.0001, Q=2: N_sv=236, E_in=0.0090, E_out=0.0165
  C=0.001, Q=2: N_sv=76, E_in=0.0045, E_out=0.0165
  C=0.01, Q=2: N_sv=34, E_in=0.0045, E_out=0.0189
  C=0.1, Q=2: N_sv=24, E_in=0.0045, E_out=0.0189
  C=1, Q=2: N_sv=24, E_in=0.0032, E_out=0.0189
  C=0.0001, Q=5: N_sv=26, E_in=0.0045, E_out=0.0189
  C=0.001, Q=5: N_sv=25, E_in=0.0045, E_out=0.0212
  C=0.01, Q=5: N_sv=23, E_in=0.0038, E_out=0.0212
  C=0.1, Q=5: N_sv=25, E_in=0.0032, E_out=0.0189
  C=1, Q=5: N_sv=21, E_in=0.0032, E_out=0.0212
```

Cross Validation

In the next two problems, we will experiment with 10-fold cross validation for the polynomial kernel. Because E_{cv} 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.

7. Consider the 1 versus 5 classifier with $Q = 2$. We use E_{cv} to select $C \in \{0.0001, 0.001, 0.01, 0.1, 1\}$. If there is a tie in E_{cv}, select the smaller C. Within the 100 random runs, which of the following statements is correct?

   Answer: [b] $C = 0.001$ is selected most often.

8. Again, consider the 1 versus 5 classifier with $Q = 2$. For the overall winning selection in the previous problem, the average value of E_{cv} over the 100 runs is closest to

   Answer: [c] 0.005

(The Python 3 source code is on pages 6–8.) The output used to answer Problems 7–8 is

```
[HW8 P7-8]
Cross validation for SVM with soft margins and polynomial kernel (1 vs. 5
classifier):
   C=0.001 is selected most often.
   E_cv=0.005
```

RBF Kernel

Consider the radial basis function (RBF) kernel $K(\mathbf{x}_n, \mathbf{x}_m) = \exp(-\|\mathbf{x}_n - \mathbf{x}_m\|^2)$ in the soft-margin SVM approach. Focus on the 1 versus 5 classifier.

9. Which of the following values of C results in the lowest E_{in}?

   Answer: [e] $C = 10^6$

10. Which of the following values of C results in the lowest E_{out}?

    Answer: [c] $C = 100$

(The Python 3 source code is on pages 6–8.) The output used to answer Problems 9–10 is

```
[HW8 P9-10]
SVM with soft margins and RBF kernel (1 vs. 5 classifier):
  C=0.01, E_in=0.0038, E_out=0.0236
  C=1, E_in=0.0045, E_out=0.0212
  C=100, E_in=0.0032, E_out=0.0189
  C=10,000, E_in=0.0026, E_out=0.0236
  C=1e+06, E_in=0.0006, E_out=0.0236
```

```
import pathlib
import sys

import numpy as np
import requests
from sklearn import svm
from sklearn.model_selection import cross_val_score
from sklearn.utils import shuffle

CWD = pathlib.Path(__file__).resolve().parent
sys.path.insert(0, str(CWD.parent))

DATA_DIR = (CWD / "../data").resolve()

def support_vector_machine(
        N=None, f=None, vf=None, *, x=None, y=None, N_test=1_000, x_test=None,
        y_test=None, clf=None, rng=None, seed=None, hyp=False, **kwargs):
    if clf is None:
        clf = svm.SVC(**kwargs)
    is_linear_kernel = clf.kernel == "linear"

    if rng is None:
        rng = np.random.default_rng(seed)
    if y is None:
        if x is None:
            x, y = generate_data(N, f, bias=is_linear_kernel, rng=rng)
        else:
            y = f(x)

    clf.fit(x[:, is_linear_kernel:], y)
    N_sv = clf.n_support_.sum()

    if x_test is None or y_test is None:
        if x_test is None:
            x_test, y_test = generate_data(N_test, f, bias=is_linear_kernel,
                                           rng=rng)
        else:
            y_test = f(x_test)

    if is_linear_kernel:
        w = np.concatenate((clf.intercept_, clf.coef_[0]))
        return (
            w, N_sv,
            vf(w, x_test, y_test) if vf
            else (1 - clf.score(x_test[:, is_linear_kernel:], y_test))
        )[1 - hyp:]
    return (N_sv, 1 - clf.score(x_test, y_test))
```

```python
if __name__ == "__main__":
    rng = np.random.default_rng()

    # SVM with Soft Margins
    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)

    # Problems 2-4
    C = 0.01
    Q = 2
    print(f"\n[HW8 P2-4]\nSVM with soft margins ({C=}) "
          f"and polynomial kernel ({Q=}):")
    clf = svm.SVC(C=C, kernel="poly", degree=Q, gamma=1, coef0=1)
    for digit in range(0, 10):
        x = data["train"][:, 1:]
        y = 2 * (data["train"][:, 0] == digit) - 1
        N_sv, E_out = support_vector_machine(
            x=x, y=y,
            x_test=data["test"][:, 1:],
            y_test=2 * (data["test"][:, 0] == digit) - 1,
            clf=clf
        )
        E_in = 1 - clf.score(x, y)
        print(f"  {digit} vs. all: {N_sv=:,}, {E_in=:.4f}, {E_out=:.4f}")

    # Problems 5-6
    _x = data["train"][np.isin(data["train"][:, 0], (1, 5))]
    y = 2 * (_x[:, 0] == 1) - 1
    _x_test = data["test"][np.isin(data["test"][:, 0], (1, 5))]
    y_test = 2 * (_x_test[:, 0] == 1) - 1

    Cs = [0.0001, 0.001, 0.01, 0.1, 1]
    Qs = [2, 5]
    print("\n[HW8 P5-6]\nSVM with soft margins and polynomial kernel "
          "for 1 vs. 5 classifier:")
    for Q in Qs:
        for C in Cs:
            clf = svm.SVC(C=C, kernel="poly", degree=Q, gamma=1, coef0=1)
            N_sv, E_out = support_vector_machine(
                x=_x[:, 1:], y=y,
                x_test=_x_test[:, 1:], y_test=y_test,
                clf=clf
            )
            E_in = 1 - clf.score(_x[:, 1:], y)
            print(f"  {C=}, {Q=}: {N_sv=:,}, {E_in=:.4f}, {E_out=:.4f}")
```

```python
# Problems 7-8
Q = 2
n_fold = 10
n_runs = 100

clfs = [svm.SVC(C=C, kernel="poly", degree=Q, gamma=1, coef0=1) for C in Cs]
ns_C = np.zeros_like(Cs, dtype=int)
Es_cv_C = np.zeros_like(Cs, dtype=float)
for _ in range(n_runs):
    Es_cv = np.fromiter((1 - cross_val_score(clf, _x[:, 1:], y,
                                             cv=n_fold).mean()
                         for clf in clfs), dtype=float, count=len(clfs))
    ns_C[np.argmin(Es_cv)] += 1
    Es_cv_C += Es_cv
    _x, y = shuffle(_x, y)
Es_cv_C /= n_runs
best_C = np.argmax(ns_C)
print("\n[HW8 P7-8]\nCross validation for SVM with soft margins "
    "and polynomial kernel (1 vs. 5 classifier):\n",
    f"  C={Cs[best_C]} is selected most often.\n",
    f"  E_cv={Es_cv_C[best_C]:.3f}",)

# Problems 9-10
print("\n[HW8 P9-10]\nSVM with soft margins and RBF kernel "
      "(1 vs. 5 classifier):")
clfs = [svm.SVC(C=C, gamma=1) for C in [0.01, 1, 100, 1e4, 1e6]]
for clf in clfs:
    _, E_out = support_vector_machine(
        x=_x[:, 1:], y=y,
        x_test=_x_test[:, 1:], y_test=y_test,
        clf=clf, rng=rng
    )
    E_in = 1 - clf.score(_x[:, 1:], y)
    print(f"  C={clf.C:,g}, {E_in=:.4f}, {E_out=:.4f}")
```