Machine Learning, Spring 2023
Homework 5

Due on 23:59 May 4, 2023

1 Kernel function (40 points)

Suppose we are given the following positively ("+1") labeled data points $\mathbb{R}^2$:

$$\left\{\begin{pmatrix}1\\1\end{pmatrix}, \begin{pmatrix}1\\-1\end{pmatrix}, \begin{pmatrix}-1\\1\end{pmatrix}, \begin{pmatrix}-1\\-1\end{pmatrix}\right\},$$

and the following negatively labeled ("-1") data points in $\mathbb{R}^2$ (see Figure 1):

$$\left\{\begin{pmatrix}2\\2\end{pmatrix}, \begin{pmatrix}2\\-2\end{pmatrix}, \begin{pmatrix}-2\\2\end{pmatrix}, \begin{pmatrix}-2\\-2\end{pmatrix}\right\}.$$

Question:

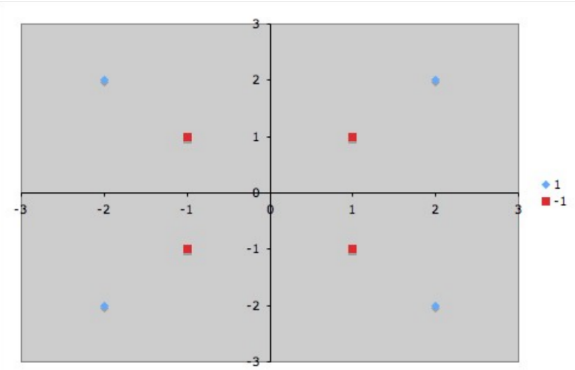

Figure 1: Blue diamonds are positive examples and red squares are negative examples.

1. Find a kernel function to map the data into a $\mathbb{R}^3$ feature space and make the new data linearly separable. (Show your kernel function please!) (8 points)

2. Use SVM classifier to seperate the data. Show the SVM problem in your report. (16 points) Solve this SVM problem, write down the expression of the final separating hyperplane, and plot the data and the separating hyperplane in a figure. (16 points) (You can solve the SVM problem by apllying a convex problem solver.)

Solution:

1. We suppose the data in $\mathbb{R}^2$ can be written (x_1, x_2). Our mapping function $\phi(x) = (x_1, x_2, x_1^2 + x_2^2)$
Then we calculate our kernel function:

$$
\begin{aligned}
K(x, x') &= \phi(x)^T \phi(x') \\
&= (x_1, x_2, x_1^2 + x_2^2)^T (x_1', x_2', x_1'^2 + x_2'^2) \\
&= x_1 x_1' + x_2 x_2' + (x_1^2 + x_2^2)(x_1'^2 + x_2'^2) \\
&= (x_1 x_1' + x_2 x_2') + (x_1^2 x_1'^2 + x_2^2 x_2'^2 + x_1^2 x_2'^2 + x_2^2 x_1'^2)
\end{aligned}
$$

We can see that the kernel function is a polynomial kernel function of degree 2.
Then we can solve the SVM problem, the data becomes:

$$
\left\{ \begin{pmatrix} 1 \\ 1 \\ 2 \end{pmatrix}, \quad \begin{pmatrix} 1 \\ -1 \\ 2 \end{pmatrix}, \quad \begin{pmatrix} -1 \\ 1 \\ 2 \end{pmatrix}, \quad \begin{pmatrix} -1 \\ -1 \\ 2 \end{pmatrix} \right\},
$$

$$
\left\{ \begin{pmatrix} 2 \\ 2 \\ 8 \end{pmatrix}, \quad \begin{pmatrix} 2 \\ -2 \\ 8 \end{pmatrix}, \quad \begin{pmatrix} -2 \\ 2 \\ 8 \end{pmatrix}, \quad \begin{pmatrix} -2 \\ -2 \\ 8 \end{pmatrix} \right\},
$$

These data are in $\mathbb{R}^3$, and we can see that they are linearly separable using the hyperplane z = 5.

2. Using SVM classifier to seperate the data, we get the following result:

```
Support vectors:
 [[ 2.  2.  8.]
 [-2. -2.  8.]
 [-1. -1.  2.]]
Dual coefficients:
 [[-0.01388686 -0.04166059  0.05554746]]
Bias term:
 [1.66637508]
```

Figure 2: svm results

The SVM problem can be defined as:

$$
minimize \frac{1}{2} ||w||_2^2 + C \sum_{i=1}^{8} \epsilon^i
$$

$$
subject\ to\ y_i(w^T \phi(x_i) + b) \geq 1 - \epsilon^i, \epsilon^i \geq 0, i = 1, 2, ..., 8
$$

$$
min \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j (x_i^T x_j)
$$

subject to linear constraint: $y^T \alpha = 0$
The quadratic coefficients are: $y_i y_j K(x_i, x_j)$
Express $g(x) = sign(w^T z + b)$ in terms of Kernel function and
$w = \sum_{z_n isSV} \alpha_n y_n z_n$, so $g(x) = sign(\sum_{\alpha_n > 0} \alpha_n y_n K(x_n, x) + b)$
where $b = y_m - \sum_{\alpha_n > 0} \alpha_n y_n K(x_n, x_m)$ for any support vector $\alpha_m > 0$
The picture and the seperated hyperplane is as Figure 3 showed:

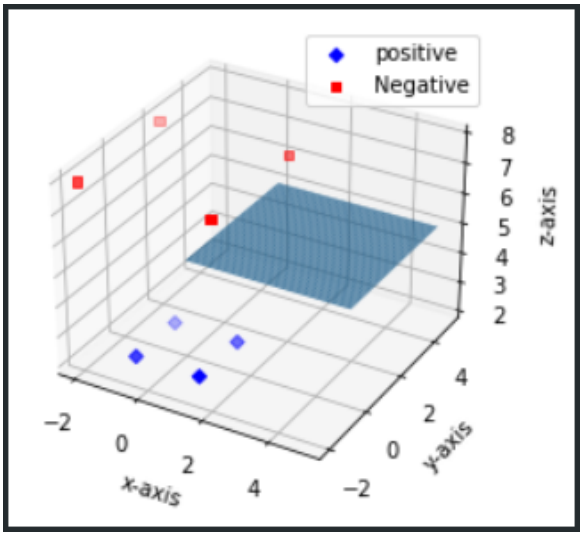

Figure 3: data points and the separating hyperplane

2 Cross Validation And L1 Regularization(60 points)

Given the training data set "crime-train.txt" and the test data set "crime-test.txt", you can read them in the files with in Python:

```
import pandas as pd
df_train = pd.read_table("crime-train.txt")
df_test = pd.read_table("crime-test.txt")
```

This stores the data as Pandas DataFrame objects. DataFrames are similar to Numpy arrays but more flexible; unlike Numpy arrays, they store row and column indices along with the values of the data. Each column of a DataFrame can also, in principle, store data of a different type. For this assignment, however, all data are floats. Here are a few commands that will get you working with Pandas for this assignment:

```
df.head()                    # Print the first few lines of DataFrame df.
df.index                     # Get the row indices for df.
df.columns                   # Get the column indices.
df[``foo'''']                # Return the column named ``foo''''.
df.drop(``foo'', axis = 1)   # Return all columns except ``foo''.
df.values                    # Return the values as a Numpy array.
df[``foo''''].values         # Grab column foo and convert to Numpy array.
df.iloc[:3,:3]               # Use numerical indices (like Numpy) to get 3 rows and cols.
```

The data consist of local crime statistics for 1,994 US communities. The response y is the crime rate. The name of the response variable is `ViolentCrimesPerPop`, and it is held in the first column of `df_train` and `df_test`. There are 95 features x_i. These features include possibly relevant variables such as the size of the police force or the percentage of children that graduate high school. The data have been split for you into a training and test set with 1,595 and 399 entries, respectively[1].

We'd like to use the training set to fit a model which can predict the crime rate in new communities, and evaluate model performance on the test set. As there are a considerable number of input variables, overfitting is a serious issue. In order to avoid this, implement the L1 regularization.

The main goal of this homework is to give you some experience using L1 regularization as a method for variable selection and using 10-folder cross-validation as a technique to get an insight on how the model will generalize to an

[1]The features have been standardized to have mean 0 and variance 1.

independent dataset.Your function should accept a scalar value of λ, a vector-valued response variable ($\boldsymbol{y}$), a matrix of input variables ($\boldsymbol{X}$), and an initial vector of weights ($\boldsymbol{w}_0$). It should output a vector of coefficient values ($\hat{\boldsymbol{w}}$).

Moreover, in order to solve the Lasso problem, we want you to implement the following algorithm.

Consider the following generalized Lasso problem

$$\min_{\boldsymbol{x}} \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2 + \lambda \|\boldsymbol{F}\boldsymbol{x}\|_1 , \tag{1}$$

where $\boldsymbol{A}$ is the under-determined sensing matrix, $\boldsymbol{F}$ is the transformed matrix. In particular, it can be reduced to Lasso problem if $\boldsymbol{F} = \boldsymbol{I}$. The above problem is equivalent to the following formulation

$$\begin{aligned} &\min_{\boldsymbol{x},\boldsymbol{z}} \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2 + \lambda \|\boldsymbol{z}\|_1 \\ &\text{s.t. } \boldsymbol{F}\boldsymbol{x} = \boldsymbol{z}, \end{aligned} \tag{2}$$

and one can employ augmented Lagrangian multiplier method to solve it. Specifically, the augmented Lagrangian is

$$\mathcal{L} = \frac{1}{2} \|\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}\|_2^2 + \lambda \|\boldsymbol{z}\|_1 + < \boldsymbol{y}, \boldsymbol{F}\boldsymbol{x} - \boldsymbol{z} > + \frac{1}{2}\rho \|\boldsymbol{F}\boldsymbol{x} - \boldsymbol{z}\|_2^2 ,$$

which yields the ADMM algorithm, see Algorithm 2. The soft-thresholding operator $\mathbb{S}_{\frac{\lambda}{\rho}}$ is defined as

$$\mathbb{S}_{\frac{\lambda}{\rho}}(x_i) = \begin{cases} x_i - \frac{\lambda}{\rho}, \ x_i \geq \frac{\lambda}{\rho} \\ 0, \ |x_i| < \frac{\lambda}{\rho} \\ x_i + \frac{\lambda}{\rho}, \ x_i \leq -\frac{\lambda}{\rho}. \end{cases} \tag{3}$$

Algorithm 1 ALM for generalized Lasso problem

Input: $\boldsymbol{A}, \boldsymbol{F}, \boldsymbol{b}, \lambda, \mu$ (for augmented Lagrange multiplier)

Initialized $\rho, \boldsymbol{x}_0, \boldsymbol{z}_0$, and $\boldsymbol{y}_0$, $k = 0$

while not converged **do**

  $\boldsymbol{x}^{(k+1)} = update\ \boldsymbol{x}$?

  $\boldsymbol{z}^{(k+1)} = update\ \boldsymbol{z}$? (you may want to use $\mathbb{S}_{\frac{\lambda}{\rho}}$ element-wise)

  $\boldsymbol{y}^{(k+1)} = \boldsymbol{y}^{(k)} + \rho(\boldsymbol{F}\boldsymbol{x}^{(k+1)} - \boldsymbol{z}^{(k+1)})$

  $\rho = \rho\mu$

  $k = k + 1$

end while

Output: $\boldsymbol{x}^* = \boldsymbol{x}^{(k)}$

In your analysis, you need to finish:

1. Derive the steps of update x and z in Algorithm 1. (10 points)
2. A plot of $\log(\lambda)$ against the squared error in the 10-folder splited training data. (15 points)
3. A plot of $\log(\lambda)$ against the squared error in the test data. (10 points)
4. A plot of λ against the number of small coefficients (you can set a threshold), and a brief commentary on the task of selecting λ. (15 points)
5. For the λ that gave the best test set performance, which variable had the largest (most positive) coefficient? What about the most negative? Discuss briefly. (10 points)

Solution:

1. The update of $\boldsymbol{x}$ and $\boldsymbol{z}$ is given by

Algorithm 2 ALM for generalized Lasso problem

Input: $\boldsymbol{A}, \boldsymbol{F}, \boldsymbol{b}, \lambda, \mu$ (for augmented Lagrange multiplier)

Initialize $\rho, \boldsymbol{x}_0, \boldsymbol{z}_0$, and $\boldsymbol{y}_0$, $k = 0$

while not converged **do**

$\boldsymbol{x}^{(k+1)} = (\boldsymbol{A}^T\boldsymbol{A} + \rho\boldsymbol{F}^T\boldsymbol{F})^{-1}(\boldsymbol{A}^T\boldsymbol{b} + \rho\boldsymbol{F}^T(\boldsymbol{z}^{(k)} - \frac{1}{\rho}\boldsymbol{y}^{(k)}))$

$\boldsymbol{z}^{(k+1)} = \mathbb{S}_{\frac{\lambda}{\rho}}(\boldsymbol{F}\boldsymbol{x}^{(k+1)} + \frac{1}{\rho}\boldsymbol{y}^{(k)})$ (element-wise soft-thresholding)

$\boldsymbol{y}^{(k+1)} = \boldsymbol{y}^{(k)} + \rho(\boldsymbol{F}\boldsymbol{x}^{(k+1)} - \boldsymbol{z}^{(k+1)})$

$\rho = \rho\mu$

$k = k + 1$

end while

Output: $\boldsymbol{x}^* = \boldsymbol{x}^{(k)}$

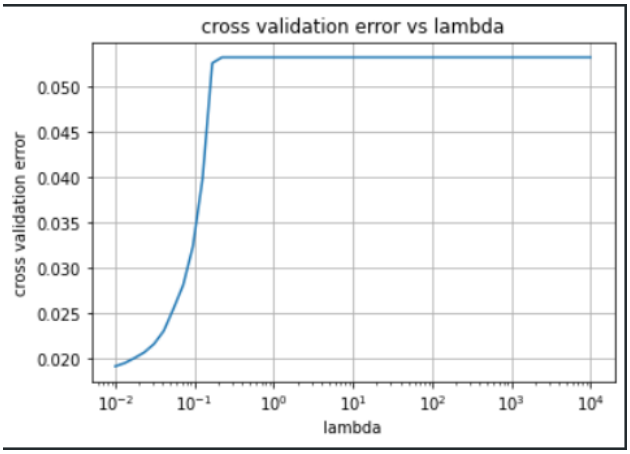

Figure 4: The squared error in the 10-folder splited training data

2. The result picture is as figure2 shows:

3. The result picture is as figure3 shows:

4. The result picture is as figure4 shows:

A brief commentary on the task of selecting λ:
The parameter λ in the generalized Lasso problem is a regularization parameter that controls the trade-off between fitting the data well and imposing sparsity on the solution. In this problem, A larger value of λ will lead to a sparser solution as it imposes a stronger penalty on the magnitude of the coefficients in the solutions. In other words the larger the λ is, the smaller the coefficients are and the smaller λ is, the larger the coefficients are.
5. For the λ that gave the best test set performance, the variable $\boldsymbol{PctIlleg}$ has the largest(most positive) coefficient: 0.0687287360561848, the variable $\boldsymbol{PctKids2Par}$ has the most negative coefficient: -0.06872331883867289

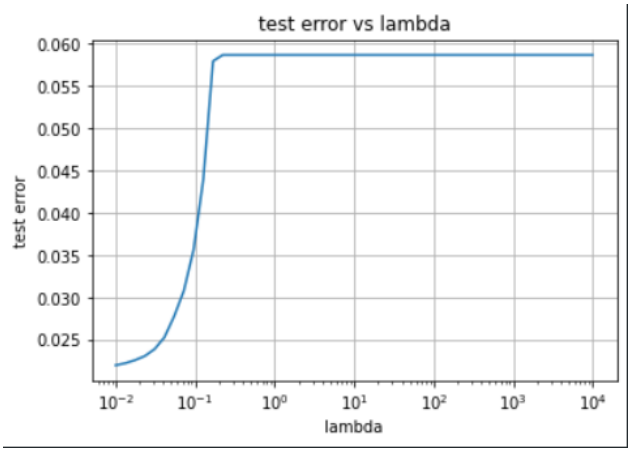

Figure 5: The squared error in the test data

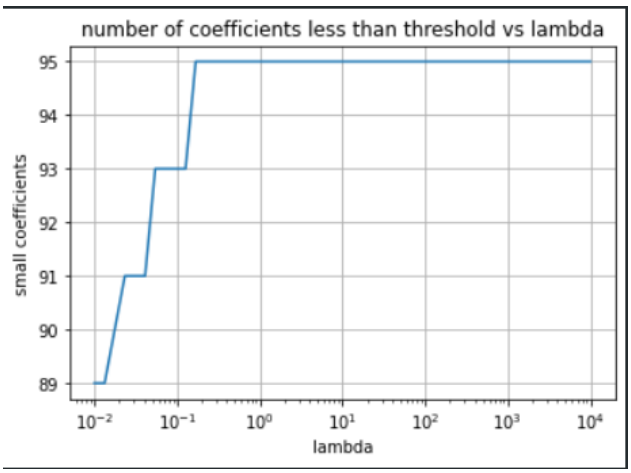

Figure 6: The number of small coefficients