DDA3020 Homework 1

Due date: Oct 14, 2024

Instructions

- The **deadline** is 23:59, Oct 14, 2024.
- The weight of this assignment in the final grade is 20%.
- **Electronic submission**: Turn in solutions electronically via Blackboard. Be sure to submit your homework as one pdf file plus two python scripts. Please name your solution files as "DDA3020HW1_studentID_name.pdf", "HW1_yourID_Q1.ipynb" and "HW1_yourID_Q2.ipynb". (.py files also acceptable)
- Note that **late submissions** will result in discounted scores: 0-24 hours → 80%, 24-120 hours → 50%, 120 or more hours → 0%.
- Answer the questions in English. Otherwise, you'll lose half of the points.
- Collaboration policy: You need to solve all questions independently and collaboration between students is **NOT** allowed.

1 Written Problems (50 points)

1.1. (Learning of Linear Regression, 25 points) Suppose we have training data:

$$\{(\boldsymbol{x}_1, \boldsymbol{y}_1), (\boldsymbol{x}_2, \boldsymbol{y}_2), \ldots, (\boldsymbol{x}_N, \boldsymbol{y}_N)\},$$

where $\boldsymbol{x}_i \in \mathbb{R}^d$ and $\boldsymbol{y}_i \in \mathbb{R}^k$, $i = 1, 2, \ldots, N$.

i) (9 pts) Find the closed-form solution of the following problem.

$$\min_{\boldsymbol{W},\boldsymbol{b}} \sum_{i=1}^{N} \|\boldsymbol{y}_i - \boldsymbol{W}\boldsymbol{x}_i - \boldsymbol{b}\|_2^2,$$

ii) (8 pts) Show how to use gradient descent to solve the problem. (Please state at least one possible Stopping Criterion)

iii) (8 pts) We further suppose that $x_1, x_2, \dots, x_N$ are drawn from $\mathcal{N}(\mu, \sigma^2)$. Show that the maximum likelihood estimation (MLE) of σ^2 is $\hat{\sigma}^2_{MLE} = \frac{1}{N}\sum_{n=1}^{N}(x_n - \mu_{MLE})^2$.

1.2. (Support Vector Machine, 25 points) Given two positive samples $x_1 = (3,3)^T$, $x_2 = (4,3)^T$, and one negative sample $x_3 = (1,1)^T$, find the maximum-margin separating hyperplane and support vectors.
Solution steps:

i) Formulating the Optimization Problem (5 pts)

ii) Constructing the Lagrangian (5 pts)

iii) Using KKT Conditions (5 pts)

iv) Solving the Equations (5 pts)

v) Determining the Hyperplane Equation and Support Vectors (5 pts)

2 Programming (50 points)

2.1. (Linear regression, 25 points) We have a labeled dataset $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_n, y_n)\}$, with $\mathbf{x}_i \in \mathbb{R}^d$ being the d-dimensional feature vector of the i-th sample, and $y_i \in \mathbb{R}$ being real valued target (label).

A linear regression model is give by

$$f_{w_0,\dots,w_d}(\mathbf{x}) = w_0 + w_1x_1 + w_2x_2 + \cdots + w_dx_d, \tag{1}$$

where w_0 is often called bias and $w_1, w_2, \dots, w_d$ are often called coefficients.

Now, we want to utilize the dataset $\mathcal{D}$ to build a linear model based on linear regression. We provide a training set $\mathcal{D}_{\text{train}}$ that includes 2024 labeled samples with 11 features (See linear_regression_train.txt) to fit model, and a test set $\mathcal{D}_{\text{test}}$ that includes 10 unlabeled samples with 11 features (see linear_regression_test.txt) to estimate model.

1. Using the LinearRegression class from Sklearn package to get the bias w_0 and the coefficients $w_1, w_2, \dots, w_{11}$, then computing the $\hat{y} = f(\mathbf{x})$ of test set $\mathcal{D}_{\text{test}}$ by the model trained well. (Put the estimation of $w_0, w_1, \dots, w_{11}$ and these $\hat{y}$ in your answers.)

2. Implementing the linear regression by yourself to obtain the bias w_0 and the coefficients $w_1, w_2, \dots, w_{11}$, then computing the $\hat{y} = f(\mathbf{x})$ of test set $\mathcal{D}_{\text{test}}$. (Put the estimation of $w_0, w_1, \dots, w_{11}$ and these $\hat{y}$ in your answers. It is allowed to compute the inverse of a matrix using the existing python package.)

(Hint: Note that for linear_regression_train.txt, there are 2024 rows with 12 columns where the first 11 columns are features $\mathbf{x}$ and the last column is target y and linear_regression_test.txt only contains 10 rows with 11 columns (features). Both of two tasks require the submission of code and results. Put all the code in a "HW1_yourID_Q1.ipynb" Jupyter notebook. file.(".py" file is also acceptable))

2.2. (SVM, 25 points)

Task Description You are asked to write a program that constructs support vector machine models with different kernel functions and slack variables.

Datasets You are provided with the iris dataset. The data set contains 3 classes of 50 instances each, where each class refers to a type of iris plant. There are four features: 1. sepal length in cm; 2. sepal width in cm; 3. petal length in cm; 4. petal width in cm. You need to use these features to classify each iris plant as one of the three possible types.

What you should do You should use the SVM function from python sklearn package, which provides various forms of SVM functions. For multiclass SVM you should use the one vs rest strategy. You are recommended to use sklearn.svm.svc() function. You can use numpy for vector manipulation. For technical report, you should report the results required as mentioned below (e.g. training error, testing error, and so on).

1. **(2 points) Split training set and test set**. Split the data into a training set and a test set. The training set should contain 70% of the samples, while the test set should include 30%. The number of samples from each category in both the training and test sets should reflect this 70-30 split; for each category, the first 70% of the samples will form the training set, and the remaining 30% will form the test set. Ensure that the split maintains the original order of the data. You should report instance ids in the split training set and test set. The output format is as follows:

   ```
   Q2.2.1 Split training set and test set:
   Training set:  xx

   Test set:  xx
   ```

   You should fill up xx in the template. You should write ids for each set in the same line with comma separated, e.g. `Training set:[1, 4, 19]`.

2. **(10 points) Calculation using Standard SVM Model (Linear Kernel)**. Employ the standard SVM model with a linear kernel. Train your SVM on the split training dataset and

validate it on the testing dataset. Calculate the classification error for both the training and testing datasets, output the weight vector $\mathbf{w}$, the bias b, and the indices of support vectors (start with 0). Note that the scikit-learn package does not offer a function with hard margin, so we will simulate this using $C = 1e5$. You should first print out the total training error and testing error, where the error is $\frac{\text{wrong prediction}}{\text{number of data}}$. Then, print out the results for each class separately (note that you should calculate errors for each class separately in this part). You should also mention in your report which classes are linear separable with SVM without slack. The output format is as follows:

```
Q2.2.2 Calculation using Standard SVM Model:
total training error:  xx, total testing error:  xx,

class setosa:
training error:  xx, testing error:  xx,
w:  xx, b:  xx,
support vector indices:  xx,

class versicolor:
training error:  xx, testing error:  xx,
w:  xx, b:  xx,
support vector indices:  xx,

class virginica:
training error:  xx, testing error:  xx,
w:  xx, b:  xx,
support vector indices:  xx,

Linear separable classes:  xx
```

If we view the one vs all strategy as combining the multiple different SVM, each one being a separating hyperplane for one class and the rest of the points, then the w, b and support vector indices for that class is the corresponding parameters for the SVM separating this class and the rest of the points. If a variable is of vector form, say $a = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$, then you should write each entry in the same line with comma separated e.g. `[1,2,3]`.

3. **(6 points) Calculation using SVM with Slack Variables (Linear Kernel)**. For each $C = 0.25 \times t$, where $t = 1, 2, \ldots, 4$, train your SVM on the training dataset, and subsequently validate it on the testing dataset. Calculate the classification error for both the training and testing datasets, the weight vector $\mathbf{w}$, the bias b, and the indices of support vectors, and the slack variable ζ of support vectors (you may compute it as $max(0, 1 - y \cdot f(X))$. The output format is as follows:

```
Q2.2.3 Calculation using SVM with Slack Variables (C = 0.25 × t, where t = 1,...,4):
```

```
-------------------------------------------
C=0.25,
total training error: xx, total testing error: xx,

class setosa:
training error: xx, testing error: xx,
w: xx, b: xx,
support vector indices: xx,
slack variable: xx,

class versicolor:
training error: xx, testing error: xx,
w: xx, b: xx,
support vector indices: xx,
slack variable: xx,


class virginica:
training error: xx, testing error: xx,
w: xx, b: xx,
support vector indices: xx,
slack variable: xx,
-------------------------------------------
C=0.5,
<... results for (C=0.5) ...>
-------------------------------------------
C=0.75,
<... results for (C=0.75) ...>
-------------------------------------------
C=1,
<... results for (C=1) ...>
```

4. **(7 points) Calculation using SVM with Kernel Functions**. Conduct experiments with different kernel functions for SVM without slack variable. Calculate the classification error for both the training and testing datasets, and the indices of support vectors for each kernel type:

   (a) 2nd-order Polynomial Kernel

   (b) 3nd-order Polynomial Kernel

   (c) Radial Basis Function Kernel with $\sigma = 1$

   (d) Sigmoidal Kernel with $\sigma = 1$

   The output format is as follows:

```
Q2.2.4 Calculation using SVM with Kernel Functions:
-------------------------------------------
(a) 2nd-order Polynomial Kernel,
total training error: xx, total testing error: xx,

class setosa:
training error: xx, testing error: xx,
w: xx, b: xx,
support vector indices: xx,

class versicolor:
training error: xx, testing error: xx,
w: xx, b: xx,
support vector indices: xx,


class virginica:

training error: xx, testing error: xx,

w: xx, b: xx,

support vector indices: xx,

-------------------------------------------

(b) 3nd-order Polynomial Kernel,

<... results for (b) ...>

-------------------------------------------

(c) Radial Basis Function Kernel with σ = 1,

<... results for (c) ...>

-------------------------------------------

(d) Sigmoidal Kernel with σ = 1,

<... results for (d) ...>
```

Submission Submit your executable code in a "HW1_yourID_Q2.ipynb" Jupyter notebook(".py" file is also acceptable). Indicate the corresponding question number in the comment for each cell, and ensure that your code can logically produce the required results for each question in the required format. Please note that you need to write clear comments and use appropriate function/variable names. Excessively unreadable code may result in point deductions.