HW5

CISC648010 - Fall 2021

Due Date: Oct 22th at 11 PM

1 Kernel Functions 10 pts

a) To what feature map ϕ does the kernel $k(x, z) = x^T z + (x^T z)^2 + 4$ correspond? Assume that x and z have dimension $d = 3$.
b) Assume kernel function $k_1(x, z)$ corresponds to feature vector $\phi_1(x)$, and $k_2(x, z)$ corresponds to $\phi_2(x)$. To what feature map does kernel $k(x, z) = k_1(x, z) + k_2(x, z)$ correspond to (find the feature vector in terms of ϕ_1 and ϕ_2)? Remark: in general if $k_1(x, z)$ and $k_2(x, z)$ are valid kernel functions, then $k_1(x, z) + k_2(x, z)$ is a valid kernel too.

2 Handwritten Image Classification Using Soft Margin Classifier 20 pts

In this problem, we redo the image classification problem in the last homework using the soft margin classifier.

Download the file mnist_49_3000.mat from Canvas. This is a subset of the MNIST handwritten digit database, which is a well-known benchmark database for classification algorithms. This subset contains examples of the digits 4 and 9. The data file contains variables x and y, with the former containing patterns and the latter labels. The images are stored as vectors.

To load the data use the following code:

```
import scipy.io
import numpy as np
data = scipy.io.loadmat('mnist_49_3000.mat')
x = np.array(data['x'])
y = np.array(data['y'][0])
```

To visualize an image, type the followings:

```
from matplotlib import pyplot as plt
index = 0 #change the index to show different images
```

```
image = x[:,index].reshape(28,28)
plt.imshow(image, interpolation='nearest')
plt.show()
```

Implement the gradient descent method to find the optimal soft margin classifier. Try setting $C = 100$. Use the first 2000 examples as training data, and the last 1000 as test data. Please report the following:

(a) (5 points) The test error

(b) (2 points) Your termination criterion (multiple options here)

(c) (3 points) The value of the objective function at the optimum

(d) (5 points) In addition, generate a plot of 5 images. These 5 images should be the 5 misclassified images in the test dataset for which the soft-margin classifier was most confident about its prediction (you will have to define a notion of confidence for the soft-margin classifier). In the title of each subplot, indicate the true label of the image. What you should expect to see is a bunch of 4s that look kind of like 9s and 9s that look like kind of like 4s. upload a printout.

(e) (5 points) To receive credit for this problem, please submit your code via Canvas, in a single file named SMC_lastname.py

Remark: Soft-Margin classifier solves the following optimization problem,

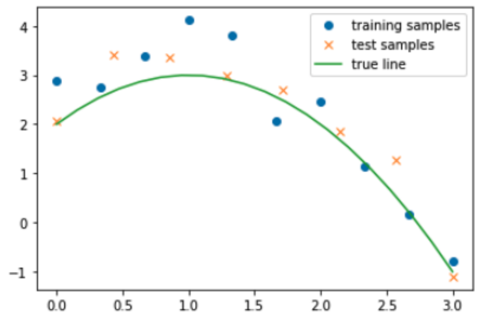
$$\min_{w,b} \frac{1}{2}||w||^2 + \frac{C}{n}\sum_{i=1}^{n} \max\{0, 1 - y_i(w^T x_i + b)\} \tag{1}$$

3 Non-Linear Regression (10pts each part)

Use the following line of code to generate a training and test data set.

```
import numpy as np
from matplotlib import pyplot as plt
np.random.seed(0)
n=10
m=8
x_train = np.linspace(0,3,n)
y_train = -x_train**2 +2*x_train + 2 +0.5*np.random.randn(n)
x_test = np.linspace(0,3,m)
y_test = -x_test**2 +2*x_test + 2 + 0.5*np.random.randn(m)
plt.plot(x_train,y_train,'o')
plt.plot(x_test,y_test,'x')
plt.plot(x,-x**2 +2*x + 2)
```

plt.legend(['training samples','test samples','true line'])



(a) Define a new feature vector $\phi(x) = [x, x^2, x^3, x^4]^T$ and fit function $f(x) = w^T \cdot \phi(x) + b$ to the training dataset. In order to do that use ridge regression with $\lambda = 0.1$. Report the value of w and b. In this section use only the training dataset.

b) Use a **For** loop and repeat part (a) for different value of $\lambda \in [0.001, 0.1]$. Set the initial value of $\lambda = 0.001$ and increase λ by 0.001 in each iteration. In each iteration, calculate the training error and test error. In two separate figures, plot the training error (mean squared error) and test error (mean squared error) as a function of λ. What is the right value for λ? Please include those figures in your report. Upload your code in a single file named NonLinear_lastname.py on canvas.

Remark: when you want to use new feature vector $\phi(x)$, you only need to plug in $\phi(x)$ to the ridge regression formula.

$$\overline{\phi} = \frac{1}{n}\sum_{i=1}^{n} \phi(x_i), \qquad \overline{y} = \sum_{i=1}^{n} y_i,$$

where n is the number of training points.

$$\tilde{\phi}(x_i) = \phi(x_i) - \overline{\phi}, \qquad \tilde{y}_i = y_i - \overline{y}$$

Note that $\tilde{\phi}(x_i)$ is a column vector with dimension 4 in our example. Define matrix $\tilde{\Phi}$ and vector $\tilde{y}$ as follow,

$$\tilde{\Phi}^T = \begin{bmatrix} \tilde{\phi}(x_1) & \tilde{\phi}(x_2) & \cdots & \tilde{\phi}(x_n) \end{bmatrix}, \qquad \tilde{y} = \begin{bmatrix} \tilde{y}_1 \\ \tilde{y}_2 \\ \vdots \\ \tilde{y}_n \end{bmatrix}$$

By Ridge Regression, we have,

$$\hat{w} = (\tilde{\Phi}^T \tilde{\Phi} + n\lambda I)^{-1} \cdot \tilde{\Phi}^T \tilde{y}$$

$$\hat{b} = \overline{y} - \hat{w}^T \overline{\phi}$$

Later when you want to predict the output of a new sample x, use function $f(x) = \hat{w}^T \phi(x) + b$.

If $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ is the training data set, the training error would be,

$$\frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{w}^T \phi(x_i) - b)^2$$

Similarly, if $\{(x_{n+1}, y_{n+1}), (x_{n+2}, y_{n+2}), \ldots, (x_{n+m}, y_{n+m})\}$ is the test dataset, the test error would be,

$$\frac{1}{m} \sum_{i=n+1}^{n+m} (y_i - \hat{w}^T \phi(x_i) - b)^2$$

4 SVM 20pts

In this part, you are allowed to use the sklearn python library. The following code on googlecolab generates a dataset and determine the decision boundary using linear SVM (Linear Kernel):

https://colab.research.google.com/drive/1wQUng7v3i9PxD8VDKkdtkW7Gx4jtNUwv?usp=sharing
You change the code to determine the decision boundaries using the following kernels:

- Polynomial Kernel with degree 3
- Polynomial Kernel with degree 10
- Gaussian Kernel with $\gamma = \frac{1}{2\sigma^2} = 0.1$
- Gaussian Kernel with $\gamma = \frac{1}{2\sigma^2} = 1$

Plot and save 4 figures that shows the decision boundary. Please include them in your report. Upload your code in a single file named SVM_lastname.py on canvas to get the full score.

Remark: Sklearn uses the term "Radial basis function (RBF) kernel" for Gaussian kernel. Using sklearn, you can set the parameter γ not σ.

$$\text{Gaussian Kernel}: \ k(x,z) = \exp\{-\frac{||x-z||^2}{2\sigma^2}\} = \exp\{-\gamma||x-z||^2\}, \gamma := \frac{1}{2\sigma^2}$$