Bilkent University Department Of Computer Engineering

CS 464

Introduction to Machine Learning

Spring 2024

Homework 2

Due date : April 14, 2024 11:59 PM

1 Instructions

- Submit a soft copy of your homework of all questions to Moodle in PDF format. Submitting a hard copy or scanned files is NOT allowed. You have to prepare your homework digitally. (using Word, Excel, Latex etc.)
- Follow the honor code while doing this homework. This is an individual assignment for each student.That is, you are NOT allowed to share your work with your classmates.
- For this homework, you may code in any programming language you would prefer. In submitting the homework file, please package your file as a gzipped TAR file or a ZIP file with the name *CS464_HW2_Firstname_Lastname*. If your name is David Bowie for instance, then you should submit a file with name *CS464_HW2_David_Bowie*.
- Please do not use Turkish letters in your package name. The file you will upload must contain the following parts:
  - the soft copy of your report in PDF format
  - your source code file(s) in a format easy to run, i.e. with a main script to call other functions.
  - the README file that tells us how we can execute/call your program.
- Unless it is stated otherwise in the questions, you are NOT allowed to use ANY machine learning packages, libraries or toolboxes for this assignment (such as scikit-learn, MATLAB Statistics and Machine Learning Toolbox functions, etc.)
- You can use the following python packages : *numpy, pandas, matplotlib, os*
- If you do not follow the submission routes, deadlines and specifications (codes, report, etc), it will lead to significant grade deduction.
- **You may ask your questions regarding this homework to your TA, Mehmet Alper Yilmaz (mehmet.yilmaz@bilkent.edu.tr)**

2 PCA [30 pts]

Principal component analysis is a linear dimensionality reduction technique which is commonly used in data analysis. It basically transforms the data into a new coordinate system, where the axes are aligned with the directions of maximum variance in the data, allowing for a more compact representation of the original data while preserving most of its information. So, you can use PCA to compress your data. In this question, you are expected to compress the image data and decompress it. You will use the dog image dataset [1] which is provided to you within homework zip file as `q1_dog` folder. For this question, you are not allowed to use PCA from libraries. Appendix 5.1 describes the details of PCA analysis.

You have 3000 grayscale dog images in which each image is represented with 60 x 60 matrices. For PCA task, first you need to flatten each image to get a vector with a length of 3600. So you will end up with 2-D array, X, of size 3000 x 3600 matrix where columns correspond to the feature vector. You can consider each image sample as a 3600 dimensional vector.

Question 2.1 [10 pts] Apply PCA on X's to obtain first 10 principal components. Report proportion of variance explained (PVE) for each of the principal components. What does PVE of principal components show us? Why are they important?

Question 2.2 [10 pts] Using the first 10 principal components, reshape each principal component to a 60 x 60 matrix. Display all images. What do these principal components represent?

Question 2.3 [10 pts] Describe how you can reconstruct an original dog image using the principal components you obtained in question 1.1. Use first k principal components to analyze and reconstruct the first image in the dataset where $k \in \{1, 50, 250, 500\}$. Analyze how different k values affect the compression and why.

3 Logistic Regression [40 pts]

For both logistic regression, and SVM part, you will use Spambase dataset [2]. The dataset contains 57 features of 4601 e-mail and each mail was labeled depending on whether it is spam or not. So, it is basically a binary classification task and your aim is to determine whether a given email is spam or not. You will work on only 680 samples. You can access the train (480), validation (100), and test dataset (100) in `Spambase_train.csv`, `Spambase_val.csv`, and `Spambase_test.csv` files, respectively. Labels are in the *Class* column in each dataset.

Question 3.1 [20 points] You need to perform maximum likelihood estimation to maximize the log-likelihood function. For this purpose, you will implement full batch gradient ascent algorithm to train your model. Initialize all weights to 0. Try different learning rates from the given logarithmic scale [10^{-5}, 10^{-4} , 10^{-3} ,10^{-2}, 10^{-1}]. Use at most 1000 iterations to train your model. Compute F1 scores of the model on validation set and plot a line graph showing F1 scores of models trained with different learning rates. X axis should correspond to the learning rates (in log scale) and Y axis should show the F1 score. Choose the one which works best on validation set. Test your model on test set and report confusion matrix and following performance scores: accuracy, micro and macro averages of precision, recall, negative predictive value (NPV), false positive rate (FPR), false discovery rate (FDR), F1, and F2 scores.

Question 3.2 [**20 points**] You will implement mini-batch gradient ascent algorithm with *batch size = 32* and stochastic gradient ascent algorithm to train your logistic regression model. Initialize all weights to 0. Use the learning rate you have chosen in Question 3.1 and perform at most 1000 iterations to train your model. Test your model on test set and report confusion matrix and following performance scores: accuracy, micro and macro averages of precision, recall, negative predictive value (NPV), false positive rate (FPR), false discovery rate (FDR), F1, and F2 scores

4 Support Vector Machines (SVMs) [30 pts]

In this question, you will train soft margin SVM classifiers on the same dataset you used in Question 3. You must perform Monte Carlo cross-validation WITHOUT using any libraries but you CAN use libraries or software packages to train your SVM. Since you will perform cross validation, you do not need `Spambase_val.csv` file in this part.

Question 4.1 [**15 points**] In this part, you will train a linear SVM model with soft margin without using any kernels. Your model's hyper-parameter is C. Using Monte Carlo cross-validation on your *training set*, find the optimum C value of your model. Look for the best C value in the following range $[10^{-2}, 10^{-1}, 1, 10, 10^2, 10^3]$. Then, run your model on the *test set* with this C value and report confusion matrix and following performance scores: accuracy, micro and macro averages of precision, recall, negative predictive value (NPV), false positive rate (FPR), false discovery rate (FDR), F1, and F2 scores.

Question 4.2 [**15 pts**] This time, use polynomial kernel to train your soft margin SVM model dataset from Question 3. The polynomial kernel is defined as

$$K(x, y) = (x^T y)^d \tag{1}$$

Similar to linear SVM part, train a SVM classifier with polynomial kernel using same training and test sets you have used in linear SVM model above. In addition to the penalty parameter C, d is your new hyper-parameter that needs be optimized. It is the degree of the polynomial. Using Monte Carlo cross-validation and calculating mean cross validation accuracy as described in MCCV, find and report the best d within the interval from $\{2, 3, 4, 5\}$. After tuning d on your *training set*, run your model on the *test set* and report confusion matrix and following performance scores: accuracy, micro and macro averages of precision, recall, negative predictive value (NPV), false positive rate (FPR), false discovery rate (FDR), F1, and F2 scores.

Monte Carlo Cross-Validation Monte Carlo Cross-Validation (MCCV) is a technique used to assess the performance of a machine learning algorithm by repeatedly sampling subsets of the dataset for training and validation. It combines the concepts of Monte Carlo sampling and cross-validation to provide a more reliable estimate of the model's generalization performance. The procedure is as following:

- Randomly split given training dataset into training set and validation set, ensuring that the validation set size falls within the range of 15% to 25% of the total dataset size (training set + validation set). The train-validation split percentage varies for each iteration.
- Train the model using the training data for each iteration and calculate the validation error by applying the trained model to the validation dataset.
- Repeat this process for 10 iterations.
- Calculate the average validation error across all iterations to evaluate average performance of the model with the given hyper-parameter value.

Note: YOU WILL SPLIT GIVEN TRAINING SET TO GET VALIDATION SET. Test set is used only after you find the best parameter via cross-validation. So, do not use test set in cross validation.

5 Appendix

5.1 PCA Compression

Before applying PCA, first you should make your data mean centered. Your input matrix looks like this

$$X = \begin{bmatrix} x_{11} & x_{12} & x_{13} & \dots & x_{1n} \\ x_{21} & x_{22} & x_{23} & \dots & x_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & x_{m3} & \dots & x_{mn} \end{bmatrix} \tag{2}$$

Where m is the number of image samples and n is the the feature size. So, $X \in \mathbb{R}^{3000\times3600}$ in our case.

First, you need to find principal components of the dataset via applying eigendecomposition of $COV(X) \in \mathbb{R}^{3600\times3600}$ which is the covariance matrix of X. It will give you 3600 eigenvectors with size 3600 ($A \in \mathbb{R}^{3600}$) and corresponding eigenvalues (λ). Next, you need to sort eigenvalues and find the largest k eigen-values and corresponding eigenvectors. It will be the new basis vectors of your new dimensionally reduced space. These eigenvectors (or principal components) will be the row matrix of our new matrix $W \in \mathbb{R}^{k\times3600}$.

$$W = \begin{bmatrix} PC_1 \\ PC_2 \\ \vdots \\ PC_k \end{bmatrix} \tag{3}$$

Where $PC_i \in \mathbb{R}^{1\times3600}$. Now you can compress all images with following equation.

$$X_{comp} = XW^T \tag{4}$$

Where $X_{comp} \in \mathbb{R}^{3000\times k}$.

References

[1] Y. Choi, Y. Uh, J. Yoo, and J.-W. Ha, "Stargan v2: Diverse image synthesis for multiple domains," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2020.

[2] R. E. F. G. Hopkins, Mark and J. Suermondt, "Spambase." UCI Machine Learning Repository, 1999. DOI: https://doi.org/10.24432/C53G6X.