MIDTERM PROJECT [ROBT407]

In this project, students are required to implement the theoretical knowledge learned about a class of learning machines such as the Linear and Logistic Regression and participate in Kaggle competition. This project aims to give you an important hands on experience in implementation of a specific algorithm and using available tools in ScikitLearn. Please note that the assignment should be implemented in Python - Jupyter Notebook enviroment.

- DUE DATE: 03/04/2020

METHOD OF DELIVERY

Assignment deliverables should be submitted via Moodle to the course instructor before the due date.

LEVEL OF COLLABORATION ALLOWED

Two students can form a team and work on this assignment. Discussions on course materials and implementation of the project are encouraged. Though, each team should write his/her final solutions separately and understand.

ASSIGNMENT DELIVERABLES

- A well documented Jupyter Notebook report describing in sufficient detail the work that include – the source codes, the approach for solving the problem, implementation, results, difficulties, limitations, etc.
- **Statement** in your report clearly stating the contribution of each member in a team.
- **In addition, convert your Jupyter notebook to a PDF file and submit.**

GRADING CRITERIA

- 35% - Implementation (a well documented Jupyter Notebook)
- 10% - The accuracy of the best model that has been selected after cross-validation
- 20% - overall work and report quality
- 20% - discussion (for example of success/failure; limitations, etc.)
- 15% - a video presentation of the entire workflow using a free screen capture software. The presentation should explain each code, and summarize the work including your Kaggle submission, in detail. Note your face does not have to appear in the presentation, instead you can record your screen. The duration of your presentation shoud be 15 - 20 minutes.

1 Linear Regression -20%

Implement the linear regression algorithm in Section 3.2 of LFD to compute the optimal $(d+1)$-dimensional w that solves

$$\underset{w}{\text{minimize}} \sum_{n=1}^{N} (y_n - (w^\intercal x_n))^2 \tag{1}$$

- Generate a training data set of size 100 as directed by Exercise 3.2 of LFD. Generate a test set of size 1000 of the same nature.
- Run the pocket algorithm (Homework 1) on the training set for $T = 1000$ to get w_{pocket}.
- Run the linear regression algorithm to get w_{lin}. Estimate the performance of the two weight vectors with the test set to get $E_{test}(w_{pocket})$ and $E_{test}(w_{lin})$, in terms of the 0/1 loss (classification).
- Repeat the experiment (with fresh data sets) 100 times and plot $E_{test}(w_{pocket})$ versus $E_{test}(w_{lin})$ as a scatter plot.
- Based on your findings in the previous problem, which algorithm would you recommend to your boss for this data set? Why?

2 Gradient Descent for Logistic Regression -20%

Consider the formulation,

$$\begin{aligned} &\underset{w}{\text{minimize}} \quad E(w) \\ &\text{where,} \quad E(w) = \frac{1}{N}\sum_{n=1}^{N} E^{(n)}(w), \\ &E^{(n)}(w) = \ln\Big(1 + \exp(-y_n\big(2w^{\intercal}x_n\big))\Big) \end{aligned} \tag{2}$$

Implement the fixed learning rate stochastic gradient descent algorithm for Eq.2.

a) initialize a $(d+1)$ -dimensional vector $w^{(0)}$, say, $w^{(0)} \leftarrow (0, 0, ..., 0)$

b) for $t = 1, 2, ..., T$

- randomly pick one n from $\{1, 2, ..., N\}$.
- update

$$w^{(t)} \rightarrow w^{(t-1)} - \eta \nabla E^{(n)}(w^{(t-1)}) \tag{3}$$

- Assume that

$$g_1^t(x) = \text{sign}\Big((w^{(t)})^T x\Big) \tag{4}$$

- where $w(t)$ are generated from gradient descent algorithm above.

  Run the algorithm with $\eta = 0.001$ and $T = 2000$ on the IRIS data set after splitting it into D_{train} (80%) and on the D_{test} (20%) You can use get the IRIS data as follows or from the Scikitlearn

```
import seaborn as sns
iris = sns.load_dataset('iris')
iris.head()
```

  .

- Plot $E_{in}(g_1^{(t)})$ and $E_{test}(g_1^{(t)})$ as a function of t, and briefly state your findings.

You can use One-Versus-All (OVA) decomposition to solve the three class problem as below.

- for $k \in \mathcal{Y}$ obtain $\mathbf{w}_{[k]}$ by running logistic regression on

$$\mathcal{D}_{[k]} = \{(\mathbf{x}_n, y_n' = 2\,[y_n = k] - 1)\}_{n=1}^{N}$$

- return $g(\mathbf{x}) = \text{argmax}_{k\in\mathcal{Y}}\left(\mathbf{w}_{[k]}^T\mathbf{x}\right)$

Or you can select to classes from the data and work on binary classification task. However, the OVA approach is preferred.

3 Implement SVM using a convex optimization package 20%

In this task, you will implement the primal form of Support-Vector Machines using a convex optimization toolbox. Tasks as follows:

Code up a linear SVM from scratch to classify Iris setosa and Iris versicolor. You can use the following a quadratic programming solver, CVXOPT [1]

Task 1: - implement the primal problem hard-margin SVM and plot its decision boundary and margins (take two features for visualization).

Task 2: - implement the dual version of the hard-margin SVM quadratic problem. Which data are the support vectors?

$$\min_{\alpha} \quad \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}\alpha_n\alpha_m y_n y_m \mathbf{z}_n^T\mathbf{z}_m - \sum_{n=1}^{N}\alpha_n$$

$$\text{subject to} \quad \sum_{n=1}^{N} y_n\alpha_n = 0;$$

$$0 \le \alpha_n \le C, \text{for } n = 1, 2, \ldots, N;$$

$$\text{implicitly} \quad \mathbf{w} = \sum_{n=1}^{N}\alpha_n y_n \mathbf{z}_n;$$

$$\beta_n = C - \alpha_n, \text{for } n = 1, 2, \ldots, N$$

Figure 1: Dual SVM formulation

For a primal SVM formulation refer to the lecture materials.

[1] https://cvxopt.org/

4 Practical design of a learning algorithm - 30%

- Given training data consisting of input-output pairs, **model selection** in machine learning is a process that builds a model to predict the output from the input usually by learning optimal adjustable hyperparameters. Many models exist in literature to perform such tasks, including linear, logistic models, svms etc. Finding and comparing methods to optimally select models, which will perform best on new test data, is the objective of this assignment.

The following steps summarize the important steps in model selection:

1. Consider a dataset D (from any domain e.g. credit/ medical / digit recognition)
2. Apply an algorithm of your choice (e.g. Lin Reg, Logistic Reg. etc) on D
3. Estimate its generalization error (E_{test})
4. **If**: generalization error smaller than what exists in the literature for the same dataset:
    - End of the process: Outcome
5. **Else**:
    - Go back to step 2 with another algorithm or change the learning strategy

As we have also studies in the model validation lecture (see Texbook section 4.3) the V-fold cross validation is one commonly used method to estimate generalization error (or to perform model selection), especially when there is little training data.

Specific tasks:

- Review the lectures on cross validation

Task 0: Load Optical Recognition of Handwritten Digits Data Set

```
from sklearn import datasets
digits = datasets.load_digits()
# read the description
print(digits.DESCR)
```

Task 1: Using the Linear & Logistic Regression implemented in Section2 and 4 implement a routine that uses tenfold cross validation for model selection on digits dataset.

Task 2: Perform GridSearchCV based tenfold cross validation using the Scikit-learn module and compare the performance of Linear and Logistic regression on digits dataset.

* For each model plot the validation curves using Scikit-Learn to justfy your selection of a final model. Explain shortly how you addressed bias-variance tradeoff, and that your model has not overfitted the digits dataset.

For Task 1 implement the following model selection steps:

1. Write a function that divides the on digits dataset (training) set (of size m) into n disjoint sets $S_1, ..., S_n$ of equal size n/m
2. For each S_i:
   - Train a classifier (e.g. Lin Reg. Log Reg) on $S \backslash S_i$
   - Test it on $S_i \leftarrow error(i)$
3. Output the average error

This gives an estimate of the generalization error of the classifier when trained on $n - n/m$ data. Report the comparative analysis of performance of Linear and Logistic regression and when you change the number of folds in validation (5 fold vs 10 fold vs 20 fold vs loocv). Refer to chapter 4 of the textbook for more details.

5 Kaggle Competition (https://www.kaggle.com/competitions) - 20%

In this task, you need to chose a problem of interest from kaggle website and participate in the competition. You should use scikit-learn: machine learning toolbox along with jupyter notes as your main tool. You have to submit your solutions to the website and share the link to your submission, as well as a well documented jupyter notebook solutions.

For instance, you can chose the following problems to work on (or many others based on your choice on Kaggle platform):

1) Netflix Prize Dataset

The Netflix Prize data set gives 100 million records of the form "user X rated movie Y a 4.0 on 2/12/05". The data is available here: **Netflix Prize**

Project Task:

- Can you predict the rating a user will give on a movie from the movies that user has rated in the past, as well as the ratings similar users have given similar movies?
- Can you discover clusters of similar movies or users?

Can you predict which users rated which movies in 2006? In other words, your task is to predict the probability that each pair was rated in 2006. Note that the actual rating is irrelevant, and we just want whether the movie was rated by that user sometime in 2006. The date in 2006 when the rating was given is also irrelevant. The test data can be found at this website.

- Reference

  https://www.kaggle.com/netflix-inc/netflix-prize-data

2) Fashon-MNIST

Context

Fashion-MNIST is a dataset of Zalando's article images—consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes. Zalando intends Fashion-MNIST to serve as a direct drop-in replacement for the original MNIST dataset for benchmarking machine learning algorithms. It shares the same image size and structure of training and testing splits.

The original MNIST dataset contains a lot of handwritten digits. Members of the AI/ML/Data Science community love this dataset and use it as a benchmark to validate their algorithms. In fact, MNIST is often the first dataset researchers try. "If it doesn't work on MNIST, it won't work at all", they said. "Well, if it does work on MNIST, it may still fail on others."

Content

Each image is 28 pixels in height and 28 pixels in width, for a total of 784 pixels in total. Each pixel has a single pixel-value associated with it, indicating the lightness or darkness of that pixel, with higher numbers meaning darker. This pixel-value is an integer between 0 and 255. The training and test data sets have 785 columns. The first column consists of the class labels (see above), and represents the article of clothing. The rest of the columns contain the pixel-values of the associated image.

To locate a pixel on the image, suppose that we have decomposed x as x = i * 28 + j, where i and j are integers between 0 and 27. The pixel is located on row i and column j of a 28 x 28 matrix. For example, pixel31 indicates the pixel that is in the fourth column from the left, and the second row from the top, as in the ascii-diagram below.