CITS5508 Machine Learning
Semester 1, 2024

Assignment 1

Assessed, worth 15%. Due: 8pm, Friday 12th April 2024

Discussion is encouraged, but all work must be done and submitted individually. This assignment has several assessed tasks, which are total 92 marks.

1 Outline

In this assignment, you will develop Python code for classification tasks using logistic regression and k-NN classifiers. You will use Grid Search and cross-validation to find the optimal hyperparameters of the model (e.g., the regularisation hyperparameter) and discuss and interpret the different decisions and their impact on the model's performance and interpretability.

2 Submission

Your submission consists of a **two files**. The first file is a report describing your analysis/approach. Your analysis should provide the requested plots and tables and your explanations and reflections about the results. Each deliverable task is indicated as **D**, a number and associated marks. For instance, **D1 [3 marks]** is the first deliverable task, and it is worth three marks.

The second file is your Python notebook with the code supporting the part one analysis. It will follow the rules provided in LMS.

Your report should be submitted as ".PDF" file. Name your file as `assig1_<student_id>.pdf` (where you should replace `<student_id>` with your student ID) and submit it to LMS before the due date and time. You can submit your file multiple times. Only the latest version will be marked. Accordingly, your code should be submitted as `assig1_<student_id>.ipynb`, the Jupyter notebook extension.

Important:

- You must submit the first part of your assignment as an electronic file and use a PDF format (do not send DOCX or any other file format). Only PDF format is accepted. Any other file format will get a zero mark.
- You must deliver part one and part two to have your assignment assessed. That is, your submission should contain your analysis and your Jupyter notebook with all coding, both with appropriate formatting.
- By submitting your assignment, you acknowledge you have read all instructions provided in this document and in LMS.
- There is a section in your LMS, Assignments - Assignment 1 - Updates, where you will find updates or clarifications about the tasks when necessary. It is your responsibility to check this page regularly.

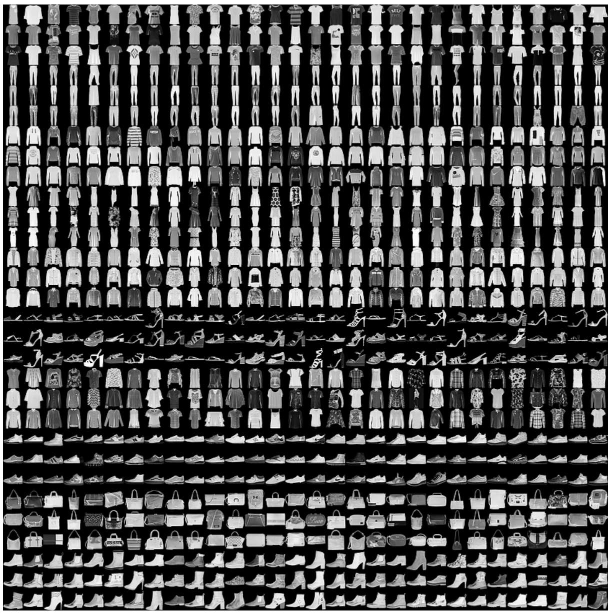
- You will be assessed on your thinking and process, not only on your results. A perfect performance without demonstrating understanding what you have done won't provide you marks.
- Your answer must be concise. Two to three sentences should be enough to answer most of the open questions. If you are writing long answers, rethink what you are doing. Probably, it is the wrong path.
- You can ask in the lab or during consultation if you need any clarification about the assignment questions.
- You should be aware that some algorithms can take a while to run. A good approach to improving their speed in Python is to use the vectorised forms discussed in class. In this case, it is strongly recommended that you start your assignment soon to accommodate the computational time.
- For the functions and tasks that require a random procedure (e.g. splitting the data into 80% training and 20% validation set), you should set the seed of the random generator to the value "5508".

3 Dataset

Fashion-MNIST is a dataset motivated by the popular MNIST dataset used to classify handwritten digits. It contains examples of ten classes (e.g. sneakers, coats, sandals, trousers, pullovers, etc.). Fashion-MNIST has a training set with 60,000 examples and a test set with 10,000 examples. Each example is a 28 x 28 low-resolution and grayscale image (similar to the examples in MNIST) associated with a label from one of the ten classes.

References and information about the dataset can be found here.

An example of how the data looks like and the classes associated (each class takes three rows):



Label	Description
0	T-shirt/top
1	Trouser
2	Pullover
3	Dress
4	Coat
5	Sandal
6	Shirt
7	Sneaker
8	Bag
9	Ankle boot

In LMS, you will find four predefined datasets to support your tasks in this assignment:

- FMNIST_training_set.csv: Training set with 60,000 examples of the cloth items.

- FMNIST_training_set_labels.csv: Labels associated to each example in FMNIST_training_set.csv.
- FMNIST_test_set.csv: Testing set with 10,000 examples of cloth items.
- FMNIST_test_set_labels.csv: Labels associated to each example in FMNIST_test_set.csv.

Each input image in the files FMNIST_training_set.csv and FMNIST_test_set_classes.csv: follows the same representation as in the MNIST dataset; that is, each example is a 28 x 28 gray-scale image reshaped into a 784-dimensional vector. Each input image has a label associated with it, given by the files FMNIST_test_set_labels.csv (training set labels) and FMNIST_test_set_labels.csv (test set labels). Each pixel's feature value can assume values in the range of 0.0 (black) and 1.0 (white). Feature values in between this range represent the possible variation in grey. Therefore, each example i has features $\mathbf{x_i} \in \mathbb{R}^{784}$ and target feature (label) $y_i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. The parametric space has *bias* and *weights*, which is in dimension 785.

4 Problem: Classifying `Sneakers` versus `Sandals`

We will start with a binary classification using the `Fashion-MNIST` dataset. In this problem, your challenge is to build a logistic regression classifier that generalises well to distinguish `Sneakers` from `Sandals`, and you will need to demonstrate you know how to load the data, train a logistic regression classifier and interpret the results.

4.1 Summarising the datasets

From the files provided for the large `Fashion-MNIST` dataset, write a Python code to separate images from "sneakers" and "sandals". You should create a training set containing "sneakers" and "sandals" only, from FMNIST_training_set.csv, and a respectively training set labels where each y_i should be 0 (zero) for "sneakers" and 1 for "sandals" (binary problem). Accordingly, you should create a test set containing "sneakers" and "sandals", from FMNIST_test_set.csv, and a respectively test set labels where each $y_i \in \{0, 1\}$.

You should not reorder, sample, or change your data. You will need to build new data, but the original characteristics and data order must be preserved.

Below are the first five instances/examples in the training data (after selecting "sandals" and "sneakers" and relabelling). You will find the respective label and the first 25 features.

label																								
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	53	102	144	169	149
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

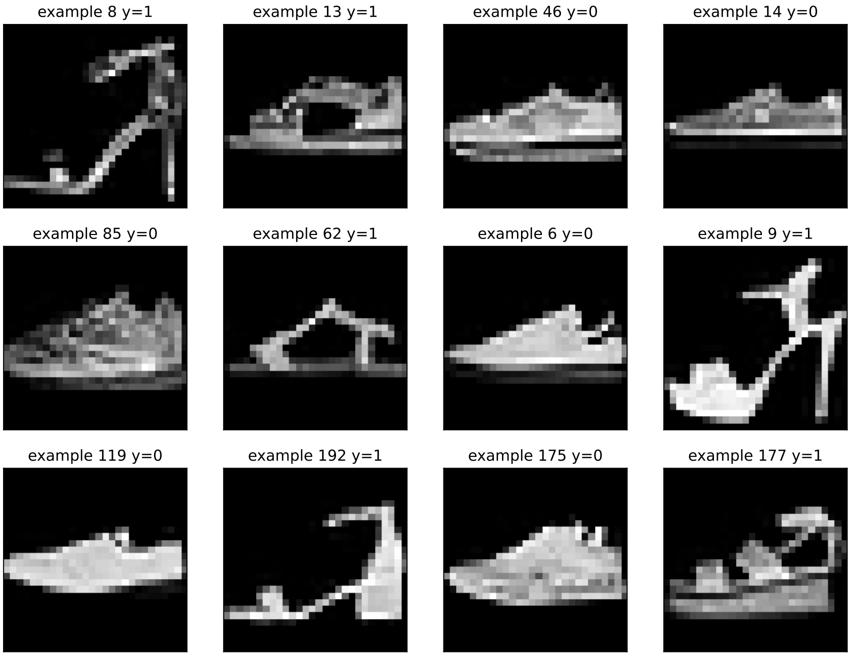
These are rows 12, 30, 36, 41, and 43 from the original training data set (remember that Python starts to count on zero). After you have built the training data, you will also need to build the test data accordingly.

D1 [3 marks]: In a table, describe:

- The number of instances in the training set;
- The number of instances in the test set;
- The total number of instances.

D2 [2 marks]: Provide a bar plot showing the number of instances for each class label. Do you have an imbalanced training set?

D3 [3 marks]: Plot the first six images/examples from each class with the corresponding *example id* and associated label on the top of the plot. Use Python function `imshow()` (you may need to adjust some values for the arguments in the function `imshow()` to better visualise the resulting image). Hint: use a 3 x 4 grid. In the example below, we attributed class '1' to the `sandals` and class '0' to the `sneakers`.



4.2 Fitting your logistic regression classifier

In this part, you will check how your classifier behaves using an iterative optimization algorithm to fit the training data. You will implement the Logistic Regression Classifier with Batch Gradient Descent.

Split your training data (that you constructed on "summarising the datasets") into two sets: training and validation. Select randomly 80% for training and 20% for validation. You must set the random generator seed to "5508" before the splitting. Ensure your validation set is fixed and does not change each time you run your algorithm.

Your implementation should initialise η (the learning rate) and θ (the parameter vector). Then, in each iteration, the main steps will look like:

```
gradients = ... # calculate the gradient vector using all instances
theta = ... # update the parameter vector
est_p = ... # compute the estimated probability for each instance
lr_cost_function = ... # calculate the logistic regression cost function
```

The probability $\hat{p}$ estimated by the logistic regression model is given by:

$$\hat{p} = h_{\boldsymbol{\theta}}(\mathbf{x}) = \sigma(\boldsymbol{\theta}^\top \mathbf{x}) = \frac{1}{1 + \exp(-\boldsymbol{\theta}^\top \mathbf{x})}$$

We calculate the logistic regression cost function as follows:

$$J(\boldsymbol{\theta}) = -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \log(\hat{p}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{p}^{(i)}) \right]$$

Run your algorithm for 10000 iterations and provide the following results and plots.

D4 [3 marks]: Experiment with some values for η, the learning rate of the gradient descent. Provide plots to support your decision for the final value and justify your choice.

D5 [10 marks]: Set $\eta = 10^{-5}$. Provide two (by-side) plots:

- In the left plot, show the values of the cost function (y-axis) for each iteration (x-axis) for the training and validation sets. That is, your plot should contain two results.
- In the right plot, show the fraction of misclassifications (y-axis) for each iteration (x-axis) using the logistic regression model prediction for a threshold of 0.5. Similarly, you should provide two results (one for the training and one for the validation set).

D6 [3 marks]: Give a short interpretation of your results. What are your conclusions?

4.3 Using cross-validation and Scikit-learn logistic regression classifier

You will incorporate a L2 regularisation in your model. Your first task is to select an optimal value for the regularisation hyperparameter C (the hyperparameter α in the textbook) in the training set using the fixed validation set. You must use the same number of iterations as before and explore penalty strength values in the range given by `np.logspace(-10, 5, 30)` (this function will generate 30 numbers spaced evenly on a log scale). Your code will look something as follows:

```
C_range = np.logspace(-10, 5, 30)

for C in C_grid:
# Train your model for this C value
# Compute the cost function on the training set
# Compute the cost function on the validation set
```

D7 [6 marks]: Provide two (by-side) plots:

- In the left plot, show the values of the cost function (y-axis) for each C value (x-axis) for the training and validation sets. That is, your plot should contain two results.
- In the right plot, show the fraction of misclassifications (y-axis) for each C value (x-axis) using the logistic regression model prediction for a threshold of 0.5. Similarly, you should provide two results (one for the training and one for the validation set).

For the next task, you will use `sklearn.linear_model.LogisticRegressionCV`, using 10-fold cross-validation and specifying the same range for the regularisation hyperparameter as before. Similarly, you will use this function to explore penalty strengths, but now using cross-validation.

D8 [4 marks]: Provide two (by-side) plots:

- In the left plot, show the values of the cost function (y-axis) for each C value (x-axis). Note that you will have ten values for each C value (10 folds). Therefore, you should report the average value.
- In the right plot, show the fraction of misclassifications (y-axis) for each C value (x-axis) using the logistic regression model prediction for a threshold of 0.5.

D9 [3 marks]: Give a short interpretation of your results. Which regularisation hyperparameter value would you select and why? What was the impacting of using 10-fold cross-validation instead of a fixed validation set?

Now you will use `sklearn.linear_model.SDGClassifier` together with `sklearn.model_selection.GridSearchCV` to find the optimal regularisation parameter according to Grid Search. `sklearn.linear_model.SDGClassifier` implements logistic regression "log loss". Remember to choose the correct strategy to evaluate the performance of the cross-validated model in the `scoring` argument of the function. (Hint: be careful how `scikit-learn` defines the Grid Search).

D10 [5 marks]: In a table, report:

- The optimal value of the regularisation hyperparameter C according to Grid Search;
- The value of the cost function in the training set for this optimal value;
- The value of the cost function in the validation set;
- The fraction of misclassifications in the training set;
- The fraction of the misclassifications in the validation set.

D11 [3 marks]: Give a short interpretation of your results. How do they compare with what you obtained in task **D8**?

Select the model you just created using `sklearn.linear_model.SDGClassifier` together with `sklearn.model_selection.GridSearchCV`; that is, using the optimal value of the regularisation hyperparameter C according to Grid Search.

D12 [5 marks]: With the obtained C, train your model using the training set. Predict the examples in the validation set, plot precision versus recall for different threshold values and comment on the results. How does the performance measure behave? What threshold would you choose and why?

D13 [4 marks]: Use a grid search (you may need to implement your own) to fine-tune the optimal threshold value using the validation set, fixing the regularisation hyperparameter C as the optimal value obtained in **D10**. What value did you get? Comment comparing with your reflection in D12.

4.4 Analysing the performance closer

So far, we have four logistic regression models:

- LR1: The model in 4.2, which is your implementation of the logistic regression without regularisation and with a fixed validation set;

- LR2: The model in 4.3, which is your implementation of the logistic regression with regularisation and with a fixed validation set;
- LR3: The the model in 4.3 using 10-fold cross-validation and the optimal value of the regularisation hyperparameter C according to Grid Search but keeping the threshold value for the logistic regression prediction model at 0.5;
- LR4: The model in 4.3 using 10-fold cross-validation and the optimal value of the regularisation hyperparameter C according to Grid Search **and** threshold values according to your grid search in **D13**.

We will give a closer investigation into the classification mistakes.

D14 [8 marks]: Provide the model's performance on the test set. For each LR1, LR2, LR3 and LR4, provide:

- The confusion matrix;
- Precision, recall, false positive rate.

D15 [3 marks]: Briefly comment on the results you see. Discuss the generalisation capacity of the four models.

D16 [4 marks]: Consider the model LR4 and show five images that are false positives on the test set and five images that are false negatives.

D17 [2 marks]: Briefly comment on the results. Can you describe what kinds of mistakes the model is making?

D18 [2 marks]: Let's inspect the mistakes further and try to interpret the LR4's estimated weights (parameters). Again, choose model LR4 and provide the model parameter values used for each image by reshaping them into a 28 x 28 matrix to associate the weights with the images. Plot the result using `imshow()`. You may need to adjust some values for the arguments in the function `imshow()` to better visualise the resulting image.

D19 [3 marks]: Inspect the values and magnitude of the weights. Comment on what you think LR4 learned when distinguishing `sandals` and `sneakers`.

4.5 Comparing models

Use the k-nearest neighbours (k-NN) algorithm with the Euclidean distance for the same binary classification task. Try k values in the range $[1, 30]$.

D20 [3 marks]: Provide a plot showing the fraction of misclassifications (y-axis) for each k value (x-axis) for the training and validation set. Use the same validation set you created in section 4.2. That is, your plot should contain two results.

D21 [4 marks]: Comment on the results. What did you observe as the value of k increases? Which value of k would you choose and why? How does this model compare with the one you created in section 4.2?

D22 [2 marks]: Provide the k-NN performance on the test set for the k value you decided in task **D21**, providing:

- The confusion matrix;
- Precision, recall, false positive rate.

D23 [2 marks]: Compare these performances with the logistic regression models (LR1, LR2, LR3 and LR4) obtained on deliverable **D14**. Briefly comment on the results you see, discussing the generalisation capacity of the different models.

4.6 Exploring the ML pipeline

This last part of the assignment is open-ended and will assess your reflections on the machine learning pipeline. You are not restricted to the images' 784-pixel values (features). Here, the idea is to think about different strategies you can use to improve the model's performance.

We can investigate the relevant pixels for our binary classification, `sandals` and `sneakers`, and think of other information about them we could use in the classification.

D24 [5 marks]: Comment on what you would do to improve the modelling.