M 124 Coursework 1

Y. Panagakis and **K. Panousis**

Due Date: March 26, 2021

1 Introduction

This coursework is divided in two different sections, each focusing on one of the first two lectures in the course.

In the first lecture, we covered the basic computational tools that we'll be using throughout this course. We focused on Linear Algebra and derivatives. Thus, in the first section, you'll need to perform some basic computations to get the hang of them. At the same time, you'll be asked to use simple python commands, to generate some variables and perform various computations.

In the second lecture, we focused on Linear Regression, one of the most important approaches in Machine Learning. We talked about the accuracy of the estimator and the potential drawbacks of Linear Regression. Hence, in the second part of the coursework, you'll delve deeper into the derivations and the implementation of Linear Regression in Python.

2 Vector and Matrices (10 Points)

It is of utmost importance to familiarize yourselves with the basic computations used in Machine Learning. Even though most computations are performed via statistical packages (e.g., numpy, scipy, pytorch, e.t.c.), you need to get the basics right.

Tasks

Use the NumPy python package to generate:

1. Two random integer matrices $X \in \mathbb{Z}^{3\times 4}$, $Y \in \mathbb{Z}^{4\times 3}$.
2. Two random integer vectors $\boldsymbol{g} \in \mathbb{Z}^4$ and $\boldsymbol{z} \in \mathbb{Z}^4$.

For the generation of the matrices you can use the *random* package of NumPy.

1. Compute the inner product of $\boldsymbol{g}$ and $\boldsymbol{z}$, $\boldsymbol{g}^T\boldsymbol{z} \in \mathbb{Z}$.
2. Compute the matrix-vector product $Xg \in \mathbb{Z}^3$.
3. Compute the dot product $XY \in \mathbb{Z}^{3\times 3}$.
4. Compute the L_2/Frobenius norms of the vectors and matrices.

In all cases perform the computations by hand, showing the intermediate steps and confirm the results with a NumPy implementation. Set the seed to be the last digit of your student number to get consistent random results. You'll need to hand in both the handwritten notes (it will be better to typeset them using LaTeX, as mentioned in the general notes) and the python notebooks.

2.1 Gradient Computation (10 Points)

Recall the gradient of a function $f : \mathbb{R}^n \to \mathbb{R}$, $\nabla f(x)$ with respect to a vector $\boldsymbol{x} \in \mathbb{R}^n$:

$$\nabla f(x) = \begin{bmatrix} \frac{\partial f(x)}{\partial x_1} \\ \vdots \\ \frac{\partial f(x)}{\partial x_n} \end{bmatrix} \tag{1}$$

Tasks

1. Let $f(\boldsymbol{x}) = \boldsymbol{x}^T A \boldsymbol{x} + \boldsymbol{b}^T \boldsymbol{x}$, where A is a $n \times n$ symmetric matrix, such that $A^T = A$ and $\boldsymbol{b} \in \mathbb{R}^n$. Compute the gradient of $f(\boldsymbol{x})$, $\nabla f(\boldsymbol{x})$ with respect to x by hand showing the intermediate steps.
2. Chain Rule: Assume that we have three functions f, g and h, such that $f(x) = sin(x)$, $g(x) = x^3$ and $h(x) = 5x$. Compute the derivative of $c(x) = (f \circ g \circ h)(x) = f(g(h(x)))$ by hand showing the intermediate steps.

The Hessian, denoted $\nabla^2 f(\boldsymbol{x})$ of a function $f : \mathbb{R}^n \to \mathbb{R}$ is the $n \times n$ symmetric matrix, when we take partial derivatives twice:

$$\nabla_x^2 f(\boldsymbol{x}) = \begin{bmatrix} \frac{\partial^2 f(x)}{\partial x_1^2} & \frac{\partial^2 f(x)}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f(x)}{\partial x_2 \partial x_1} & \frac{\partial^2 f(x)}{\partial x_2^2} & \cdots & \frac{\partial^2 f(x)}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f(x)}{\partial x_n \partial x_1} & \frac{\partial^2 f(x)}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f(x)}{\partial x_n^2} \end{bmatrix} \tag{2}$$

Task

1. Compute the Hessian $\nabla_x^2 f(\boldsymbol{x})$ of $f(\boldsymbol{x}) = \boldsymbol{x}^T A \boldsymbol{x} + \boldsymbol{b}^T \boldsymbol{x}$, where A is a $n \times n$ symmetric matrix, such that $A^T = A$ and $\boldsymbol{b} \in \mathbb{R}^n$ by hand.

3 Linear Regression (80 points)

In the first lecture, we showed the process of obtaining the least squares coefficient estimates in the univariate case. This was done by taking the derivative of the loss function and setting to zero.

3.1 Derivation of the Ordinary Least Squares estimator for multiple regressors (5 points)

In Linear Regression, we assume a dataset containing n pairs of observations $\{\boldsymbol{x}_i, y_i\}_{i=1}^N$, where $\boldsymbol{x}_i$ is a p-dimensional vector of regressors. We seek to derive an estimator/linear predictor $f(\cdot)$ to predict y given

X, such that:

$$f(X) = \boldsymbol{\beta}^T X \tag{3}$$

where we have absorbed the bias factor in the vector notation.

Task:

- Derive the Least Squares estimates for the multivariate Linear Regression case.**(5 points)**

Show the intermediate steps of the computations.

3.2 Implementation

In this task, you'll be using python and specifically NumPy, Scikit-learn, Matplotlib and Pandas to explore Linear Regression in a common introductory dataset: The Boston Housing Dataset.

Pandas is a python package for Data manipulation and analysis. It provides some very useful functions and makes data access easier and more intuitive. There are many tutorials online about the usage, but the main object you'll be using is a pandas Dataframe.

Dataset Description

The dataset consists of 506 samples and 13 features/regressors. We want to use this data in order to construct an estimator to be able to predict the price of a house given the features. You can use the sklearn datasets package to load the dataset and create a *single* panda dataframe for both the dataset *data* (.data) and the *target* variable (.target).

The feature of the Boston House Dataset are:

- CRIM: Per capita crime rate by town.
- ZN: Proportion of residential land zoned for lots over 25,000 sq. ft.
- INDUS: Proportion of non-retail business acres per town.
- CHAS: Charles River dummy variable (= 1 if tract bounds river; 0 otherwise).
- NOX: Nitric oxide concentration (parts per 10 million).
- RM: Average number of rooms per dwelling.
- AGE: Proportion of owner-occupied units built prior to 1940.
- DIS: Weighted distances to five Boston employment centers.
- RAD: Index of accessibility to radial highways.
- TAX: Full-value property tax rate per $10,000$.
- PTRATIO: Pupil-teacher ratio by town.
- B: 1000(Bk — 0.63)^2, where Bk is the proportion of [people of African American descent] by town.
- LSTAT: Percentage of lower status of the population.

Target Variable: MEDV: Median value of owner-occupied homes in 1000.

Some preliminaries :
In ML, it is bad practice to use the training data to evaluate the model. We'll talk about that in the following lectures. For now, let's say that we need to split the available data into two different subsets: the *training* and the *test* data. There exist implementations that perform this task, and in this exercise you can use the train_test_split function of the sklearn package to do that. Essentially you train the model using the train data and evaluate it on both the training and test data.

Tasks

- **Train the model (30 points)**:
  1. Use the train_test_split sklearn function to split the data into train and test sets using a ratio of your choice, e.g. 90% training, 10% test, e.t.c.
  2. Train a linear regression model using the train data. Do not use the built in implementation of the sklearn package, but instead employ the coefficient estimates that you derived in the previous section.
- **Evaluate the results (15 points)**:
  1. Evaluate the model, using the statistics introduced in Lecture 2 for both training and test data (RSE, R^2).
  2. Is there a difference between the produced RSE and R^2 statistics of the train and test sets? If yes, why do you think this happens?
  3. Choose three additional different ratios for train_test_split and repeat the process. Show the corresponding results for each split and explain the potential differences between the obtained estimators.
- **Further Investigation (30 points)**:
  1. Try some different combinations of the features. To this end, use various subsets (e.g., try 3 different combinations of 2-3 out of the 13 features) and report the results.
  2. How does the performance change according to different features? Can you guess why is this happening?
  3. Use the corr function of the Pandas library and examine the resulting correlation matrix. You can plot the correlation matrix using a heatmap to more easily decipher the results (you can use the sns package or any implementation you find online).
  4. Using the resulting matrix, choose the features that you think will produce the best results and explain your selection. Re-train the model. How do the results in this case fare against your previous experiments?
  5. One would think that using more features would produce better results since we have more information. However, does this happen? Expand a bit on the results obtained from your analysis of the correlation matrix.

General Notes

- This coursework is an introduction to the basics of Machine Learning. I highly recommend to take your time and understand the fundamental calculus and rationale behind the introduced approaches and avoid searching ready-made solutions online.
- Individual Coursework.

- Must be submitted via e-class. E-mailed submissions will not be accepted.
- Programming:
  - For the programming part, you can turn to StackOverflow if you bump into some kind of bug or problem, but again I strongly advise against copying code for these simple tasks.
  - As mentioned in the first tutorial, it is best practice to use Virtual Environments for your projects to avoid breaking the system. You can use either Virtual Environments of native python venv or install an Anaconda/Miniconda platform. Or simply use Google Colab.
  - Implement your approach using a Jupyter Notebook, with sufficient but not redundant comments.
- Typesetting:
  - I would suggest using LaTeXfor reporting the results. LaTeXis a very powerful typesetting system, most commonly employed in research papers and reports. It provides all the necessary tools for easily typesetting equations, arrays and more, while avoiding the hassles of other editors. This CS was written in LaTeX. You can use Overleaf for online editing.
  - A template will be provided.