Lab Course Machine Learning
Exercise Sheet 1

Prof. Dr. Dr. Lars Schmidt-Thieme, Shayan Jawed
Information Systems and Machine Learning Lab
University of Hildesheim

October 28st, 2020
Submission on November 4th, 2020 at 12 noon, (on learnweb, course code 3116)

Instructions

Please following these instructions for solving and submitting the exercise sheet.

1. You should submit a jupyter notebook detailing your solution.
2. Please explain your approach i.e. how you solved a given problem and present your results in form of graphs and tables.
3. Please submit your jupyter notebook to learnweb before the deadline. Please refrain from emailing the solutions except in case of emergencies.
4. Unless explicitly noted, you are not allowed to use scikit, sklearn or any other library for solving any part.
5. Please refrain from plagiarism.

Exercise 1: Python Warmup (10 Points)

Install Anaconda and Jupyter notebooks (latest version; python latest version); `https://www.anaconda.com/products/individual`.

Part A: (2 Points): IPython In this task you are required to use IPython (a web version provided by Jupyter nootbook). You have to write a word count program. Your program should read the provided text document on learnweb named "random_text.txt" and then output the following stats:

- the number of unique words.
- the top 5 most frequent words.

Part B: (8 Points): numpy In this exercise, we shall implement an image blurring program using numpy matrices. Numpy is a popular numerical operations library that is widely used in the python machine learning ecosystem. Please familiarize yourself with this library by following the documentation online `numpy.org`. The exact task details include:

1. Install matplotlib library and use the given utilities to read and display the image 1.

2. Initialize an averaging filter as follows:

$$F = \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix}$$

3. Pad the image with $1s$ on all sides.

4. Convolve the filter over the given image. More concretely:

   - Select a 3×3 region of the image. Since images are stored as matrices, you end up with a 3×3 matrix S. The center element in this matrix S_{ij} needs to be updated with another entry which is calculated as follows:
   - Multiply the filter above with this matrix S element-wise and sum the output (Please refrain from utilizing numpy utilities):

$$S_{ij} = \sum \frac{1}{9}\begin{bmatrix}1 & 1 & 1\\ 1 & 1 & 1\\ 1 & 1 & 1\end{bmatrix} \times \begin{bmatrix}224 & 215 & 47\\ 214 & 90 & 89\\ 225 & 250 & 247\end{bmatrix} = 177.88$$

   - In the above equation, the second matrix is a subregion from the image. We proceed to replace the original entry in the image $S_{ij} = 90$ with the newly computed value of 177.88.
   - Proceed to convolve this filter over the entire image keeping the stride, $s = 1$. In summary, every entry(pixel) in the image is now replaced with an average of its 3×3 neighbors. Padding is necessary to not lose information about the first and last rows/columns.
   - Please refer to the website *PyImageSearch* (`https://bit.ly/3dWll9h`) for a more detailed description on image convolutions.

5. Plot the original and blurred images side by side.

6. Examine the result of applying the filter multiple times.

Listing 1: Utility functions for images from matplotlib library

```
import matplotlib.image as mpimg
import matplotlib.pyplot as plt
img = mpimg.imread('lena_gray.jpg')
plt.imshow(img, cmap='gray', vmin=0, vmax=255)
```

Exercise 2: Linear Regression through exact form (10 Points)

In this exercise you will implement linear regression that was introduced in the introduction lecture of Machine learning. Please refer to the slides on the course webpage.

1. Generate a simple data i.e. a matrix X with dimensions 100×2. Initialize it with normal distribution $\mu = 2$ and $\sigma = 0.01$

2. Generate a simple target vector i.e. a matrix Y with dimensions 100×1. Initialize it with random uniform distribution.

3. Implement LEARN-SIMPLE-LINREG algorithm and train it using matrix X to learn values of β_0 and β_1

4. An important aspect of this algorithm is solve the system of linear equations. Instead of solving the system of linear equations, here we ask you to invert the matrix $A = X^T X$. The task hence involves implementing a subroutine for matrix inversion. Although it must be noted that inverting a matrix should always be avoided but as programming practice here we implement matrix inversion nevertheless.

5. Implement PREDICT-SIMPLE-LINREG and calculate the points for each training example in matrix A.

6. Plot the training points from matrix Y and predicted values in the form of scatter graph.

7. In the end use numpy.linalg lstsq to replace step 2 for learning values of β_0 and β_1.