Lab Course Machine Learning
Exercise Sheet 3

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

November 3rd, 2016
Submission on November 9th, 2016 at 11:55pm, (on moodle, course code 3112)

Instructions

Please read the lab related instructions, i.e. submission, report format and policies, at `https://www.ismll.uni-hildesheim.de/lehre/prakAIML-16w/exercises/ml_lab_instructions.pdf`

Datasets

1) Airfare and demand: `http://www.stat.ufl.edu/~winner/data/airq402.dat` description: `http://www.stat.ufl.edu/~winner/data/airq402.txt`.
2) Wine Quality: `http://archive.ics.uci.edu/ml/datasets/Wine+Quality`

Exercise 1: Data preprocessing (5 Points)

You are required to pre-process given datasets.

1. convert any non-numeric values to numeric values. For example you can replace a country name with an integer value or more appropriately use hot-one encoding. [Hint: use hashmap (dict) or $pandas.get\_dummies$]. Please explain your solution.
2. If required drop out the rows with missing values or NA. In next lectures we will handle sparse data, which will allow us to use records with missing values.
3. Split the data into a train(80%) and test(20%) .

Exercise 2: Linear Regression with Gradient Descent (15 Points)

Part A: (8 Points): Implement Linear Regression with Gradient Descent In this part you are required to implement linear regression algorithm with gradient descent algorithm. Reference lecture `https://www.ismll.uni-hildesheim.de/lehre/ml-16w/script/ml-02-A1-linear-regression.pdf`
For each dataset given above

1. A set of training data $D_{train} = \{(\mathbf{x}^{(1)}, y^{(1)}), (\mathbf{x}^{(2)}, y^{(2)}), \ldots, (\mathbf{x}^{(N)}, y^{(N)})\}$, where $\mathbf{x} \in \mathcal{R}^M$, $y \in \mathcal{R}$, N is number of training examples and M is number of features
2. Linear Regression model is given as $\hat{y}^n = \sum_{m=1}^{M} \beta_m x_m^n$

3. Least square loss function is given as $l(x, y) = \sum_{n=1}^{N} (y^n - \hat{y}^n)^2$

4. minimize the loss function $l(x, y)$ using Gradient Descent algorithm. **Implement (*learn-linregGD* and *minimize-GD* algorithms given in the lecture slides**). Choose i_{max} between 100 to 1000.

5. You can choose three suitable values of step length $\alpha > 0$. For each value of step length perform the learning and record

   (a) In each iteration of the *minimize-GD* algorithm calculate $|f(x_i{-}1) - f(x_i)|$ and at the end of learning, plot it against iteration number i. Explain the graph.

   (b) In each iteration step also calculate RMSE on test set $RMSE = \sqrt{\frac{\sum_{q=1}^{T}(y_{test}^{q} - \hat{y}^{q})^2}{T}}$ and at the end of learning, plot it against iteration number i. Explain the graph.

Part B: (7 Points): Step Length for Gradient Descent This task is based on Part A. You have to implement two algorithms *steplength-armijo* and *steplengthbolddriver* given in the lecture slides.
For each step length Algorithm

1. In each iteration of the *minimize-GD* algorithm calculate $|f(x_i{-}1) - f(x_i)|$ and at the end of learning, plot it against iteration number i. Explain the graph.

2. In each iteration step also calculate RMSE on test set $RMSE = \sqrt{\frac{\sum_{q=1}^{T}(y_{test}^{q} - \hat{y}^{q})^2}{T}}$ and at the end of learning, plot it against iteration number i. Explain the graph.

Compare different step length algorithms
Compare the RMSE graphs of *steplength-armijo* and *steplengthbolddriver* and the three fixed step length. Explain your graph.

Annex

1. You can use numpy or scipy in build methods for doing linear algebra operations.

2. You can use pandas to read and processing data

3. You can use matplotlib for plotting.

4. You should not use any machine learning library for solving the problem i.e. scikit-learn etc. If you use them you will not get any points for the task.