Lab Course Machine Learning
Exercise 6

Prof. Dr. Dr. Lars Schmidt-Thieme,
Mofassir ul Islam Arif
Information Systems and Machine Learning Lab
University of Hildesheim
Submission: 06.12.2019 LearnWeb 3115

November 29, 2019

Instructions

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

1. You should submit two things a) python scripts(zipped) / jupyter notebook and b) a pdf document.
2. In the pdf document you will explain your approach (i.e. how you solved a given problem), and present your results in form of graphs and tables.
3. The submission should be made before the deadline, only through learnweb.
4. Unless explicitly mentioned, you are not allowed to use scikit, sklearn or any other library for solve any part. All implementations must be done yourself.

1 Exercise Sheet 6

Datasets

- 1. Regression Datasets

  (a) Generate a Sample dataset called D1 :
  i. Initialize matrix $x \in R^{100\times1}$ using Uniform distribution with $\mu = 1$ and $\sigma = 0.05$
  ii. Generate target $y \in R^{100\times1}$ using $y = 1.3x^2 + 4.8x + 8 + \psi$, where $\psi \in R^{100\times1}$ randomly initialized.
  (b) Wine Quality called D2: (use winequality-red.csv)http://archive.ics.uci.edu/ml/datasets/Wine+Quality

You are required to pre-process given datasets.

2 GLMs

Exercise 1: Generalized Linear Models with Scikit Learn (5 Points)
In previous labs you have implemented various optimization algorithms to solve linear or logistic regression problem. In this task you are required to use Scikit Learn to experiment with following linear models and Stochastic Gradient Descent (SGD) [Hint: use *SGDRegressor*]. You may use scikit learn for this question.

1. Ordinary Least Squares
2. Ridge Regression
3. LASSO

Following are required in this task

1. Split your data into Train and Test Splits. Use dataset D2
2. For each model, pick three sets of hyperparameters and learn each model (without cross validation). Measure Train and Test RMSE and plot it on one plot. Explain the plots and relate it to the theory studied in lectures i.e. influence of regularized vs non-regularized models. You have to compare the following models and argument should explain underfitting and overfitting.
3. Now tune the hyperparameters using scikit learn GridSearchCV and plot the results of cross validation for each model. [Hint: use *cv_results* to see different options]
4. Using the optimal hyperparameter you have to evaluate each model using cross_val_score. Plot each model using boxplot and explain how significant are your results.

3 Polynomial Regression

Exercise 2: Higher Order Polynomial Regression (5 Points) In this task you are required to use dataset D1. So far we have only looked at 1st degree polynomial, i.e. linear polynomial and your D1 is also generated using linear polynomial. In this task you have to use higher degrees of polynomial feature for your data i.e. degrees 1, 2, 7, 10, 16 and 100. [Hint: use sklearn.preprocessing to generate polynomial features]. You may use scikit learn for this question. Your tasks are:

1. **Task A**: Prediction with high degree of polynomials
   - a For each newly created dataset learn LinearRegression.
   - b Plot the predicted curves for each dataset. Explain the phenomena you observed for different prediction curves.
2. **Task B**: Effect of Regularization

a Fixed the degree of polynomial to 10

b Pick Four values of λ (regularization constant) and learn Ridge Regression [Hint: use Ridge and your λ values should be far a part i.e. $0,\ 10^{-6}, 10^{-2},\ 1$].

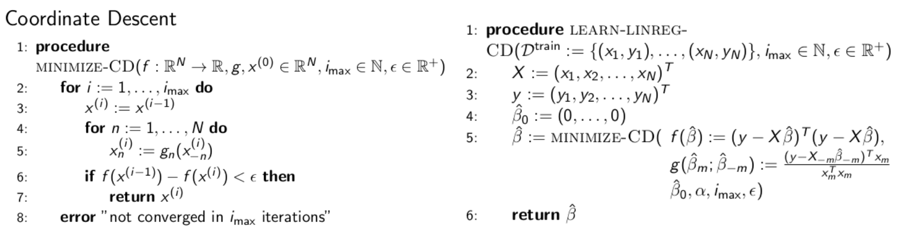
c Plot the predicted curves for each dataset. Explain the phenomena you observed for different prediction curves.

4 Coordinate Descent

Exercise 3: Implementing Coordinate Descent (10 Points)

So far we have looked at Gradient Descent, Stochastic Gradient Descent(Ascent). This week the main task is to implement Coordinate Descent that has been covered in the lecture. To make things a bit more interesting, we will be implementing Lasso Regression along with the Coordinate Descent. You may NOT use scikit-learn for this question . We will use the Wine Dataset for this question.

Fig. 1 Shows the implementation for the Coordinate Descent along with its minimization.



Coordinate Descent

```
procedure MINIMIZE-CD($f : \mathbb{R}^N \to \mathbb{R}, g, x^{(0)} \in \mathbb{R}^N, i_{\max} \in \mathbb{N}, \epsilon \in \mathbb{R}^+$)
    for $i := 1, \dots, i_{\max}$ do
        $x^{(i)} := x^{(i-1)}$
        for $n := 1, \dots, N$ do
            $x_n^{(i)} := g_n(x_{-n}^{(i)})$
        if $f(x^{(i-1)}) - f(x^{(i)}) < \epsilon$ then
            return $x^{(i)}$
    error "not converged in $i_{\max}$ iterations"
```

```
procedure LEARN-LINREG-CD($\mathcal{D}^{\text{train}} := \{(x_1, y_1), \dots, (x_N, y_N)\}, i_{\max} \in \mathbb{N}, \epsilon \in \mathbb{R}^+$)
    $X := (x_1, x_2, \dots, x_N)^T$
    $y := (y_1, y_2, \dots, y_N)^T$
    $\hat\beta_0 := (0, \dots, 0)$
    $\hat\beta :=$ MINIMIZE-CD( $f(\hat\beta) := (y - X\hat\beta)^T(y - X\hat\beta)$,
                             $g(\hat\beta_m; \hat\beta_{-m}) := \frac{(y - X_{-m}\hat\beta_{-m})^T x_m}{x_m^T x_m}$
                             $\hat\beta_0, \alpha, i_{\max}, \epsilon$)
    return $\hat\beta$
```

Figure 1: Coordinate Descent Algorithm

1. **Task A**: Coordinate Descent.

   a Implement Coordinate Descent.

   b Maintain a history of your β values. After training plot them against iterations [hint: If you have 10 features, you should have 11 βs (one bias, 10 features)]. Plot them all in a singel plot. This should show you the progression of your feature values as your train the model.

   c

2. **Task B**: Coordinate Descent with L1 Regularization

   a Implement CD with L1 regularization (Fig. 2). Note that the update step is now including the L1 term.

   b Maintain a history of your β values. After training plot them against iterations.

3. **Task C**: Comparison

```
1: procedure LEARN-LINREG-L1REG-CD($\mathcal{D}^{\text{train}} := \{(x_1, y_1), \ldots, (x_N, y_N)\}, \lambda \in \mathbb{R}^+, i_{\max} \in \mathbb{N}, \epsilon \in \mathbb{R}^+$)
2:     $X := (x_1, x_2, \ldots, x_N)^T$
3:     $y := (y_1, y_2, \ldots, y_N)^T$
4:     $\hat{\beta}_0 := (0, \ldots, 0)$
5:     $\hat{\beta}$ := MINIMIZE-CD( $f(\hat{\beta}) := (y - X\hat{\beta})^T(y - X\hat{\beta}) + \lambda||\beta||_1$,
                        $g(\hat{\beta}_m; \hat{\beta}_{-m}) := \text{soft}(\frac{(y - X_{-m}\hat{\beta}_{-m})^T x_m}{x_m^T x_m}, \frac{\frac{1}{2}\lambda}{x_m^T x_m})$,
                        $\hat{\beta}_0, \alpha, i_{\max}, \epsilon$)
6:     return $\hat{\beta}$
```

$$\text{soft}(x, \epsilon) := \begin{cases} x - \epsilon, & \text{if } x > \epsilon \\ 0, & \text{if } |x| \le \epsilon \\ x + \epsilon, & \text{if } x < -\epsilon \end{cases}$$

Figure 2: Coordinate Descent with Regularization

a Compare the plots of the unregularized and regularized CD

b Highlight the difference. What information can be inferred from these values.

4.1 ANNEX

- Following lecture is relevant this exercise https://www.ismll.uni-hildesheim.de/lehre/ml-16w/script/ml-04-A3-regularization.pdf
- sklearn.model_selection, sklearn.metrics, sklearn.linear_model, sklearn.preprocessing
- Scikit Learn User Guide http://scikit-learn.org/stable/user_guide.html
- You can use matplotlib for plotting.
- sklearn.metrics http://scikit-learn.org/stable/modules/classes.html#module-sklearn.metrics