CS 534: Implementation Assignment 1

October 26, 2015

1 Linear regression with L_2 regularization

1.1 Part 1

For the first part of the assignment, we implement linear regression with L_2 (quadratic) regularization, which learns from a set of N training examples $\{x_i, y_i\}_{i=1}^N$ an weight vector w that optimize the following regularized Sum of Squared Error (SSE) objective:

$$F(w) = \sum_{i=1}^{N} (y_i - \mathbf{w}^T x_i)^2 + \lambda \|w\|_2^2. \tag{1}$$

To optimize this objective, we implement the gradient descent algorithm. The update rule for iteration $k+1$ is given as follows

$$w^{(k+1)} = w^{(k)} - \alpha \nabla F(w^{(k)}),$$

where

$$\nabla F(w^{(k)}) = 2\lambda w^{(k)} - 2\sum_{i=1}^{N} \left(y_i - \left(w^{(k)} \right)^T x_i \right) x_i$$

and α is the learning rate.

Note, because some features have very large values, before applying gradient descent algorithm, we normalize the features to have zero mean and unit variance.

In order to determine learning rate we define a range of possible values for the learning rate α and test our algorithm for those values. For the experiment we chose $\alpha \in [1e{-}5, 1e{-}1]$. We fix Lagrangian parameter $\lambda = 0.1$ and as a stopping criterion we chose the maximum number of iterations to be $iterNo = 1e4$.

In Figures 1-4 we depict objective function for several different values of α. As one can notice on Figure 1 $\alpha = 0.001$ is too big and objective function does not converge. When $\alpha = 0.0001$ we observe convergence of the objective function, see Figure 2. However, when we set $\alpha = 1e-5$ we detect much slower convergence rate, as depicted on Figure 4. We conclude that the best learning rate that minimizes the convergence time is $\alpha = 0.0006$, see Figure 3.

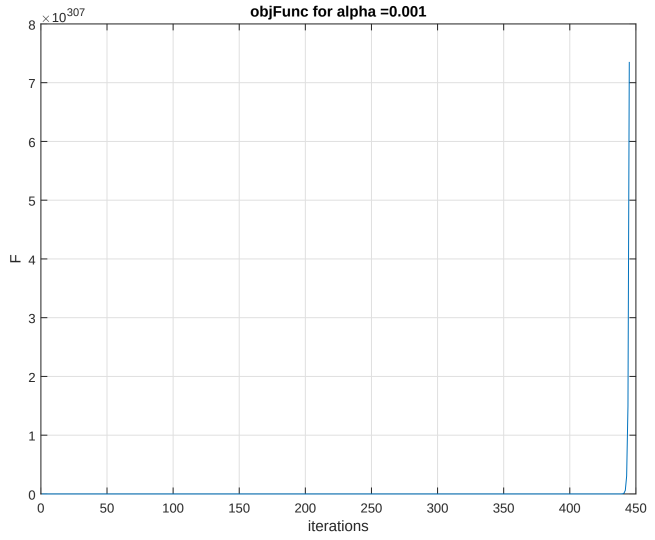

Figure 1: Objective function $F(w)$ versus number of iterations for $\alpha = 0.001$.

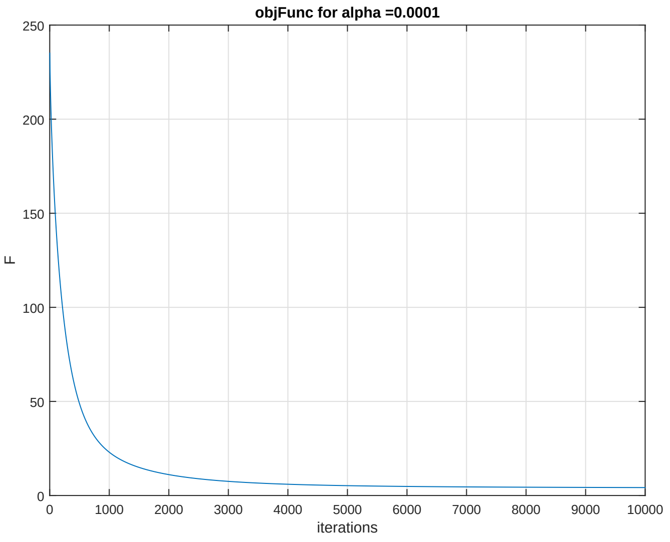

Figure 2: Objective function $F(w)$ versus number of iterations for $\alpha = 0.0001$.

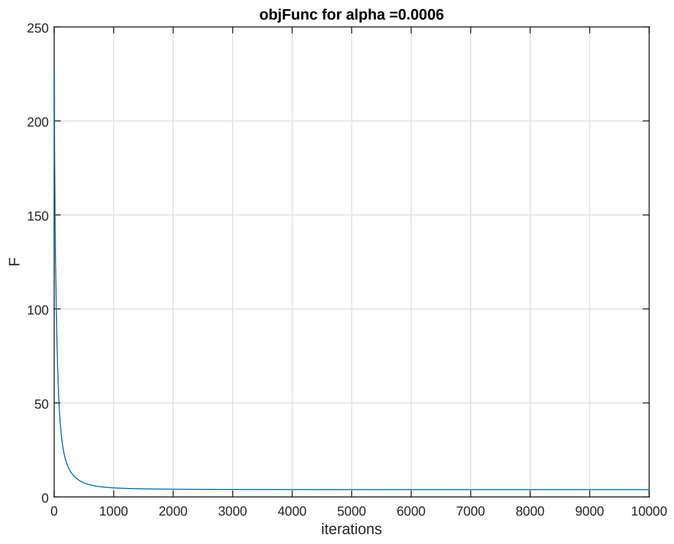

Figure 3: Objective function $F(w)$ versus number of iterations for $\alpha = 0.0006$.

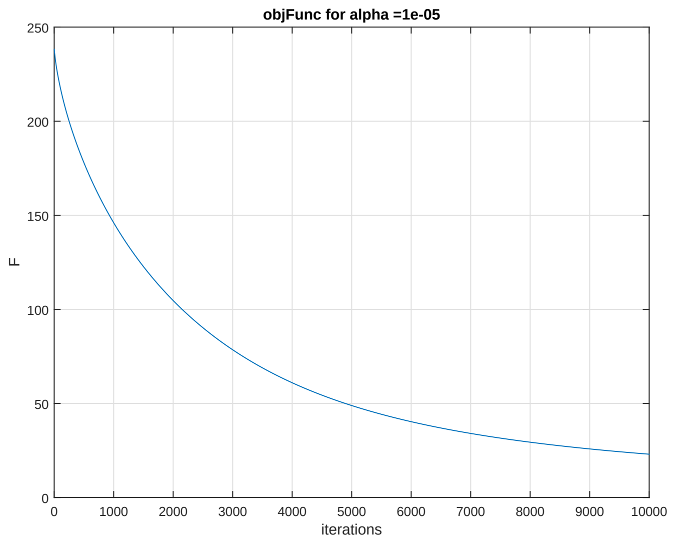

Figure 4: Objective function $F(w)$ versus number of iterations for $\alpha = 1e - 05$.

The results confirm our intuition. Increasing the convergence rate can lead to divergence of the algorithm or to oscillation, when algorithm makes too large steps and bounces over the optimum value. Decreasing the learning rate leads to slower convergence, since algorithm makes smaller steps towards the optimal value. The optimum learning rate allows to achieve the fastest convergence.

1.2 Part 2

Note, that for part 2 we do not need to use gradient descent algorithm. We also observe that there is closed form solution for the optimization problem. Note that objective function can be rewritten in the following matrix form:

$$F(w) = (y - Xw)^T (y - Xw) + \lambda w^T w.$$

Differentiating the objective function with respect to w and setting derivative to zero we obtain a closed-form solution given by

$$w = \left(\lambda I + X^T X\right)^{-1} X^T y.$$

Now we can test the effect of the regularizer on linear regressor by experimenting with different λ values. We consider λ on the interval $[0, 100]$ and for each value of λ we calculate SSE on both training and test data. In Figure 5 we depict SSE on the training data. In Figure 6 we depict SSE on test data. We observe the following trend: training SSE increases as λ increases and test SSE decreases as λ increases. To explain this phenomenon, first, note that the objective function 1 consist of two parts: the SSE part

$$\sum_{i=1}^{N} (y_i - \mathbf{w}^T x_i)^2$$

and the constraint part

$$\|w\|_2^2.$$

Parameter λ can be considered as a weight of the constraint part: the larger the λ the more weight we put on the constraint. Since the objective is to minimize function $F(w)$, larger values of λ will enforce smaller values of $\|w\|_2^2$. This implies that, by increasing λ we enforce smaller value of vector w and thus, worsen the data fit, which makes the training SSE to increase. On the other hand, smaller values of vector w prevent over-fitting, thus making test SSE to decrease.

1.3 Part 3

For selecting the best λ we again use closed form solution derived in part 2. For 10-fold cross validation, we split the training data into 10 parts. We vary λ on interval $[0, 100]$ and for each value of λ, we train on 9 parts and test on the remaining one part and measure its SSE. After 10 times we sum up the 10

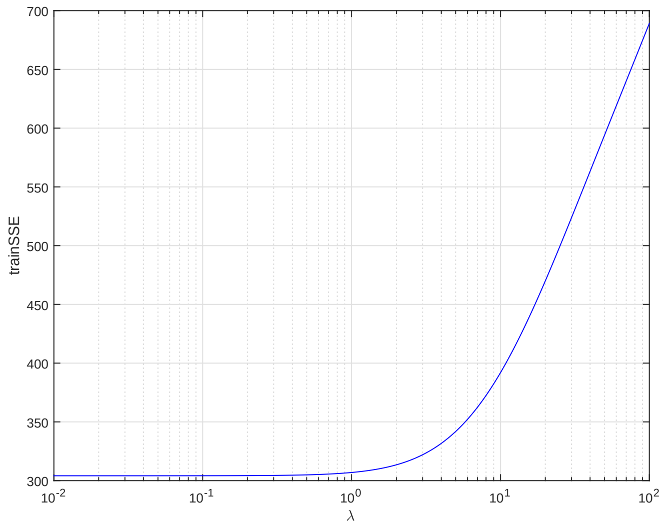

Figure 5: SSE on the training data versus values of λ.

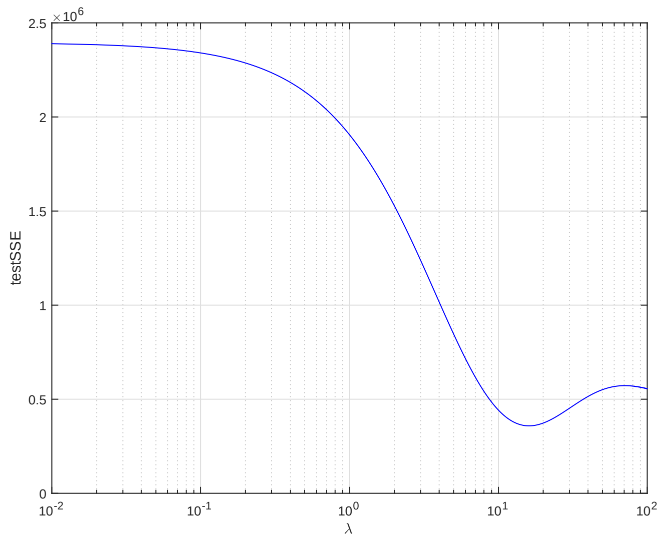

Figure 6: SSE on the test data versus values of λ.

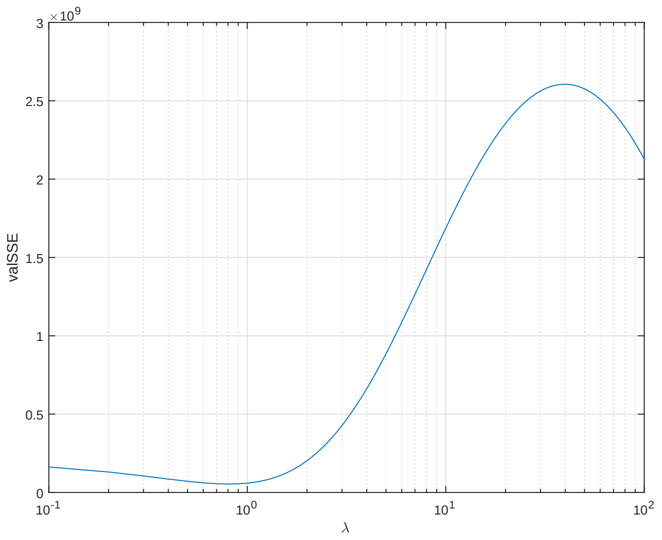

Figure 7: Cross-validated SSE versus values of λ.

SSEs as the cross-validated SSE for the specific value of λ.In Figure 7 we depict cross-validated SSE versus values of λ. Among all candidate λ values, we select the one that has the lowest cross-validated SSE. In our case $\lambda = 0.8$.

We also observe that the cross-validates SSE is larger than the training SSE. This is due to the fact that training SSE is obtained using the same data points (all of them) that were used for learning the model. However, for cross-validation SSE we use only a part of training dataset (i.e. smaller sample to learn the model). Moreover, validation SSE is obtained using the data points (validation sets) that were not used for learning the model. Thus, the cross-validates SSE is larger than the training SSE.

We observe the following behavior of the cross-validates SSE: first, it decreases slightly, reaching its minimum value, then as we increase λ the cross-validates SSE increases. This trend is closer to the trend of test SSE. This is due to the fact that in both cases the model was learned and evaluated on different data sets.