Instituto Superior Técnico

Departamento de Engenharia Electrotécnica e de Computadores

Machine Learning

1st Lab Assignment

Shift: Sexta 14h Group Number: 1

Number 84053 Name Francisco Raposo de Melo

Number 89213 Name Rodrigo Tavares Rego

Linear Regression

Linear Regression is a simple technique for predicting a real output *y* given an input $\mathbf{x}=(x_1, x_2,\ldots, x_P,)$ via the linear model

$$f(\mathbf{x}) = \beta_0 + \sum_{k=1}^{P} \beta_k x_k$$

Typically there is a set of training data $T=\{(\mathbf{x}^i, y^i), i=1,.., N\}$ from which to estimate the coefficients $\boldsymbol{\beta}=[\beta_0, \beta_1, \ldots,\beta_P]^T$. The Least Squares (LS) approach finds these coefficients by minimizing the sum of squares error

$$SSE = \sum_{i=1}^{N} (y^i - f(\mathbf{x}^i))^2$$

The linear model is limited because the output is a linear function of the input variables x_k. However, it can easily be extended to more complex models by considering linear combinations of nonlinear functions, $\phi_k(\mathbf{x})$, of the input variables

$$f(\mathbf{x}) = \beta_0 + \sum_{k=1}^{K} \beta_k \phi_k(\mathbf{x})$$

In this case the model is still linear in the parameters although it is nonlinear in **x**. Examples of nonlinear function include polynomial functions and Radial basis functions.

This assignment aims at illustrating Linear Regression. In the first part, we'll experiment linear and polynomial models. In the second part, we'll illustrate regularized Least Squares Regression. The second part of this assignment requires MatLab's Statistics Toolbox.

1. Least Squares Fitting

1. Write the matrix expressions for the LS estimate of the coefficients of a polynomial fit of degree P and of the corresponding sum of squares error, from training data $T=\{(x^i, y^i),\ i= 1,.., N\}$.

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

Assuming $X^T X$ is invertible. $X = \begin{bmatrix} 1 & x_1 & {x_1}^2 & \ldots & {x_1}^P \\ 1 & x_2 & {x_2}^2 & \ldots & {x_2}^P \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_N & {x_N}^2 & \ldots & {x_N}^P \end{bmatrix}$

$$\hat{y} = X\beta$$

$$SSE = ||y - X\hat{\beta}||^2$$

$$dim(\hat{\beta}) = (P+1) \times 1 \qquad dim(X) = N \times (P+1)$$

$$dim(\hat{y}) = N \times 1$$

Result Images in html folder

2. Write Matlab code to fit a polynomial of degree P to 1D data variables *x* and *y*. Write your own code, do not use any Matlab ready made function for LS estimation or for polynomial fitting. You should submit your code along with your report.

3. Load the data in file 'data1.mat' and use your code to fit a straight line to variables *y* and *x*.

   a. Plot the fit on the same graph as the training data. Comment.

Using P=1 to fit a straight line to the variables y and x, we notice that the LS estimation successfully finds the linear function that best minimizes the sum of squared errors, as a result we conclude that P=1 is adequate to fit the given data.

   b. Indicate the coefficients and the error you obtained.

$\hat{\beta}_0 = 0,6351$ $\hat{\beta}_1 = 1,7332$ $\hat{y} = 1,7332x + 0,6351$

$SSE = 0,7433$

4. Load the data in file 'data2.mat', which contains noisy observations of a cosine function $y = cos(2x) + \varepsilon$, with $x \in [-1,1]$, in which ε is Gaussian noise with a standard deviation of 0.15. Use your code to fit a second-degree polynomial to these data.

   a. Plot the training data and the fit. Comment.

Using P=2 to fit a parabola to the variables y and x, since the y values behave as a parabola within the given domain, we notice that the LS estimation successfully finds the second degree polynomial function that best minimizes the sum of squared errors, as a result we conclude that P=2 is adequate to fit the given data, without overfitting.

   b. Indicate the coefficients and the error you obtained. Comment.

$\hat{\beta}_0 = 0,9757$ $\hat{\beta}_1 = -0,0257$ $\hat{\beta}_2 = -1,5322$

$\hat{y} = -1,5322x^2 - 0,0257x + 0,9757$

$SSE = 1,3416$

The SSE is relatively higher in this case, since the signal has added Gaussian noise and we're fitting a 2nd degree polynomial to training data of a cossine signal, however considering the small deviation of 0,15 the data points are still reasonably close to each other (small errors) and thus the approximation is acceptable, without overfitting.

5. Repeat item 4 using as input the data from file 'data2a.mat'. This file contains the same data used in the previous exercise except for the presence of an outlier point.

   a. Plot the training data and the fit. Comment.

Using P=2 to fit a parabola to the variables y and x, since the y values behave as a parabola within the given domain, even with the added outlier, we found that the 2nd degree polynomial estimation with the outlier is visibly drawn by the outlier position and, as a result, the maximum of the parabola is higher than in the estimation without the outlier. This significant difference occured for only one outlier, which means that the outlier weights in relatively more than a non-outlier point.

   b. Indicate the coefficients and the error you obtained. Comment on the sensitivity of LS to outliers.

$\hat{\beta}_0 = 1,0523 \qquad \hat{\beta}_1 = -0,0716 \qquad \hat{\beta}_2 = -1,6313$

$\hat{y} = -1,6313x^2 - 0,0716x + 1,0523$

$SSE = 5,0249$

The LS estimation aims to minimize the sum of the squared distances from the regression line to the data points. Since the distance in relation to an outlier is relatively much higher than in relation to the other points and it is also squared, the contribution of an outlier weights more and thus the LS estimation is considerably sensitive to outliers. That explains the higher SSE we obtained. In order to improve this situation, we could eliminate the outlier or use other robust elimination methods. The SSE with the outlier is 3,7 times higher than the SSE with the previous training set.

2. Regularization

The goal of this second part is to illustrate linear regression with regularization, we'll experiment with Lasso.

1. (T) Write the expression for the cost function used in Ridge Regression and Lasso and explain how Lasso can be used for feature selection.

$\hat{\beta}_{ridge} = arg\min_{\beta} ||y - X\beta||^2 + \lambda||\beta||^2$

Ridge Cost Function: $||y - X\beta||^2 + \lambda||\beta||^2$

$\beta_{lasso} = arg\min_{\beta} ||y - X\beta||_2^2 + \lambda||\beta||_1$

Lasso Cost Function: $||y - X\beta||_2^2 + \lambda||\beta||_1$

$||.||_1 = L1 - norm$

$||.||_2 = L2 - norm$

The new term $||\beta||^2$ penalizes the use of large coefficients (large errors) and it is denoted a regularization term. Regarding Lasso Regression, if a feature does not contribute to predict the outcome y, it will probably make the corresponding coefficient equal to zero. As a result, the Lasso estimate β_{lasso} is thus a sparse vector of coefficients since unimportant features, to predict a certain outcome y, receive a zero coefficient. This can be interpreted as a feature selection operation, since unimportant features are removed, the other ones are better estimated.

Ridge method penalizes large coefficients by lowering them, however it does not force them to be actually 0 (if they're unimportant). Lasso method on the other hand, since it uses the L1-norm in the regularization term, its absolute value not only penalizes the use of large coefficients, but also actually sets the irrelevant ones to 0.

2. Load the data in file 'data3.mat' which contains 3-dimensional features in variable **x** and a single output *y*. One of the features in **x** is irrelevant. Use function `lasso` with default parameters (type help for more information on this function) and obtain regression parameters for different values of the regularization parameter λ (the values for lambda are returned in `FitInfo.Lambda`). Note that although these data are not zero centered, that is not a problem because the lasso function does not require centered data. Use function `lassoPlot` to plot the coefficients against λ. For comparison plot the LS coefficients in the same figure (λ = 0).

```
[B,FitInfo] = lasso(X,Y);
lassoPlot(B,FitInfo,'PlotType','Lambda','XScale','log');
```

3. Comment on what you observe in the plot. Identify the irrelevant feature.

Lasso Regression applies a shrinking regularization process where it penalizes the coefficients of the regression variables gradually shrinking some of them to zero - as λ increases eventually all of them shrink to zero. When λ = 0 we have the LS estimation, with no feature selection. What we can observe is the coefficients gradually shrinking to zero as λ increases, however one of the features is visibly irrelevant and the last to enter the model. ***Feature 1*** *of X is the first one to be considered in the model since it reaches 0 only for the highest values of λ (positively affects the response variable). The second one to enter the model is* ***feature 3*** *of X by the same logic as feature 1 of X. Finally,* ***feature 2*** *of X is the first one to shrink to zero and it also has a weak positive affect in the response variable to begin with, comparing to the other two. Therefore, the* ***irrelevant feature*** *is feature 2 of X.*

4. Choose an adequate value for λ. Plot *y* and the fit obtained for that value of λ. Compare with the LS fit. Compute the SSE in both cases. Comment.

We decided to analyse the vector with the degrees of freedom given by FitInfo to establish since what index we stop having 3 degrees of freedom and instead we get 2, in other others, when feature 2 of X is 0.

We chose $\boldsymbol{\lambda = 0{,}0695}$.

Since feature 2 of X is irrelevant, the plot of the predicted y, with (λ = 0,0695) and without feature selection (LS fit), is very similar, which proves that the chosen λ indeed eliminated an unimportant feature.

SSE (λ = 0,0695) = 15,7192

SSE (λ = 0 - LS) = 14,9820

As was previously said, the elimination of the irrelevant feature barely affects the fit of the given data, as we can see by the obtained SSE values for each case.