Instituto Superior Técnico

Departamento de Engenharia Electrotécnica e de Computadores

Machine Learning

1st Lab Assignment

Shift: 4ª 14h Group Number: 1

Number 78486 Name Luís Bernardo de Brito Mendes Rei

Number 78761 Name João Miguel Limpo de Lacerda Pestana Girão

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}.

$\hat{y} = \beta_0 + \beta_1 x_1 + \cdots + \beta_p x_p$, using vector notation we get $\hat{y} = \begin{bmatrix} 1 & x^T \end{bmatrix} \beta$

Considering a training set of N points, the linear model $f(x) = \begin{bmatrix} 1 & x^T \end{bmatrix} \beta$

is trained minimizing the least square cost function $SSE = \sum_{i=1}^{n} (y_i - f(x_i))^2$

Adopting matrix notation,

$$X = \begin{bmatrix} 1 & x_{11} & \ldots & x_{1p} \\ 1 & x_{21} & \ldots & x_{2p} \\ \vdots & \vdots & \ldots & \vdots \\ 1 & x_{n1} & \ldots & x_{np} \end{bmatrix} \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}$$

$$\begin{aligned} SSE &= \|y - X\beta\|^2 = (y - X\beta)^T (y - X\beta) \\ &= y^T y - 2y^T X\beta + \beta^T X^T X\beta \end{aligned}$$

computing the gradient and making it zero

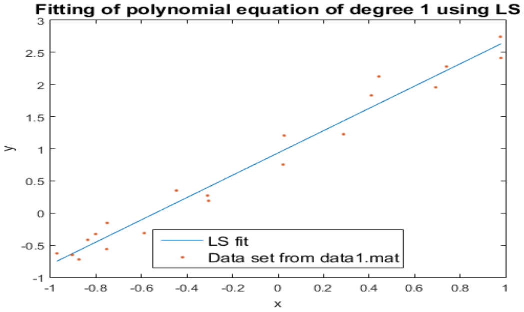
$$\nabla_\beta SSE = -2X^T y + 2X^T X\beta = 0$$

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

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.



Data set can be interpreted with a linear regression, correlating x and y.

Fit is intuitive, and well related to the training data.

   b. Indicate the coefficients and the error you obtained.

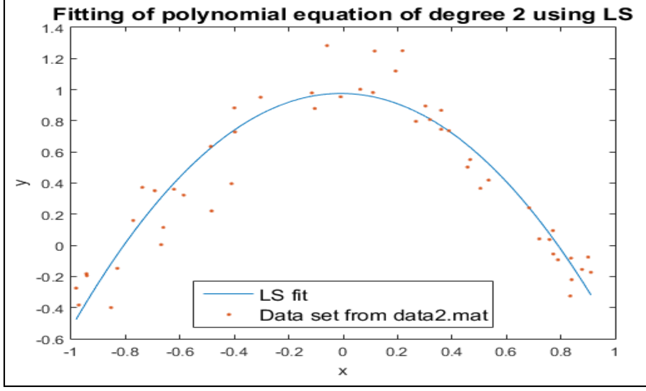
```
beta =          e =

    0.9351          0.7433
    1.7332
```

Linear regression encompasses two beta parameters.
The first one, beta_0, represents the ordinate of the regression line when the abcissa is null (y = 0.9351).

Second parameter, beta_1, is the slope of the line (m = 1.8332).

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.



Second-degree polynomial regression. Fit seems to adjust relatively well to the data set.

Registered noise difficults a probable more precise fit.

As the training set increases its size, the error of the fit tends to increase as well.

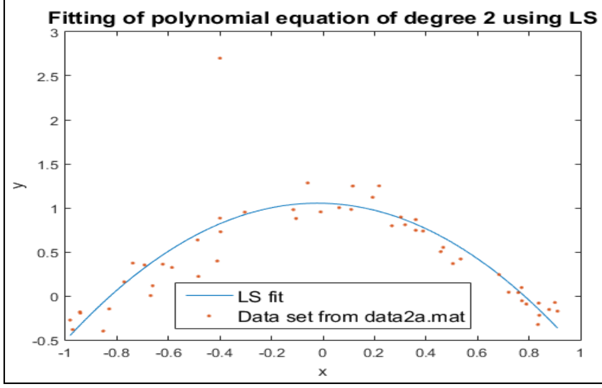
   b. Indicate the coefficients and the error you obtained. Comment.

```
beta =          e =

    0.9757          1.3416
   -0.0257
   -1.5322
```

Second degree polynomial admits three parameters beta (beta_0, beta_1, beta_2). Last parameter is negative, indicating that the concavity of the curve is growing downwards.
The coefficients beta_1 and beta_2 together control the location of the axis of symmetry of the parabola.
Error increased compared to the previous question as expected due to the Gaussian noise added to the observations and the increased training set size.

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.



The fit remains similar to the previous plot.

The single outlier, when attempting to fit the data to a second degree polynomial, doesn't drastically alter the fit we saw in the previous question. The outlier weight will increase if we attempt to fit a bigger degree polynomial (the model will start tending upwards to that outlier point).

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

```
beta =

    1.0523
   -0.0716
   -1.6313

e =

    5.0249
```

The outlier highjacks the values of the error, bringing them up as expected. The parameters are slightly changed also due to the effect of the outlier that corrupts the data.

2. Regularization

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

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

Computing the gradient vector and making it equal to zero we get

$$\hat{\beta}_{\text{ridge}} = (X^T X + \lambda I)^{-1} X^T y$$

Similar to LS but with an extra term lambda.

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

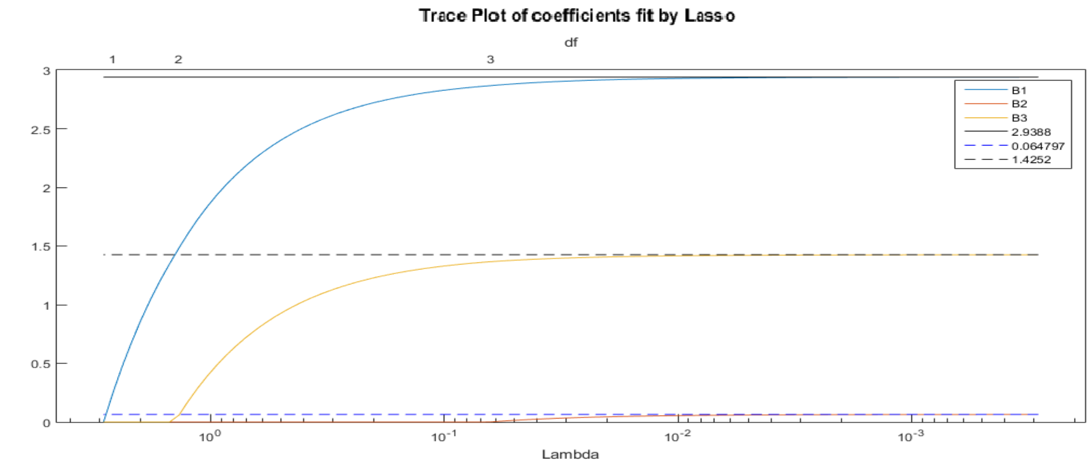
This optimization problem cannot be solved by a linear system of equation, and needs to be solved numerically. This is due to the fact that the regularization term in lasso is a l1 norm instead of the euclidean norm of beta seen in Ridge.

Lasso can, contrary to Ridge regression, attribute zero weight to a feature, rendering it useless and dispendable. This can be interpreted as a feature selection operation. Since unimportant features are removed other features that are considered more important are better estimated.

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`). Use function `lassoPlot` to plot the coefficients against λ. For comparison plot the LS coefficients in the same figure ($\lambda = 0$).

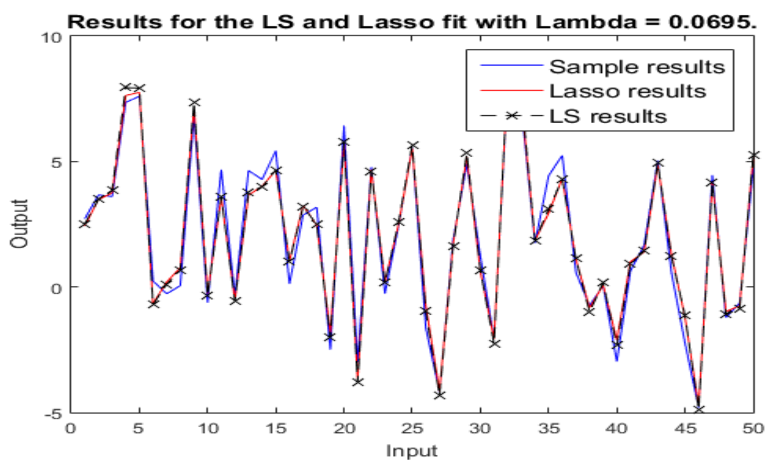
```
[B,FitInfo] = lasso(x,y);
lassoPlot(B,FitInfo,'PlotType','Lambda','XScale','log');
```

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



Features 1, 2 and 3 are legended as B1, B2 and B3 respectively. LS coefficients are the horizontal lines (last 3 legends). The plot allows us to gather information about the weight of the features in the given data set. Through observation we see that B2 is weightless throughout most values of lamba. It is considered irrelevant because it carries no significance in the fit. We notice that with lambda growth, beta values decrease to zero.

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 error in both cases. Comment.



lassoError =

15.7192

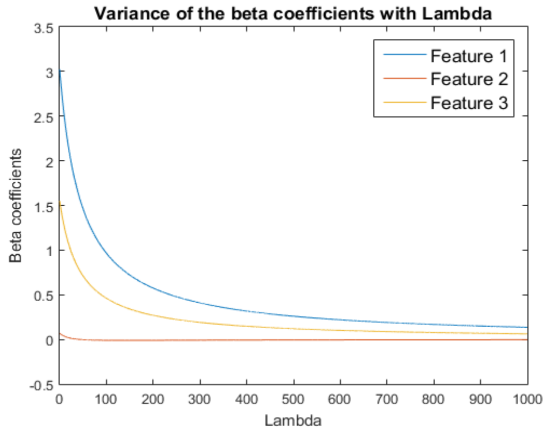
LSError =

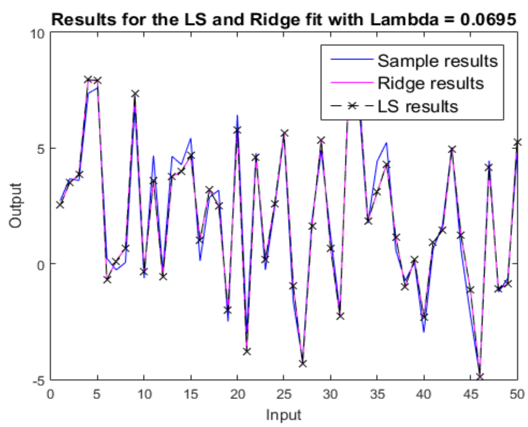
14.9823

Parameter beta_0 is fundamental to achieve an overall better fit to the data available to us.

We chose a lambda from just before feature 2 gains any weight - the results we obtain for both fits is very similar because the coefficients are also similar (see 2.3). Increasing the lambda will obviously disturb the results obtained for the lasso fit. The coefficients will be significantly different, the points will distance themselves from the LS fit results and the error will increase accordingly.

5. Repeat the previous items but using ridge regression (function `ridge)` instead of Lasso. Use the same λ values as in Lasso.





Coefficient values tend to, but never reach, zero.
We also observe that the evolution of beta remains the same in comparison to 2.3 - where the coefficients decreased with the lambda scalar increase.

Of the 3, feature 2 is the one that carries less significance to the fit, just like we concluded before.

Beta_0 is similar in both cases (ridge and lasso) although we aren't showing it.

```
ridgeError =

   14.9832

LSError =

   14.9820
```

Ridge results overlap LS results.
The coefficients are very similar, and that is translated in the almost identical values of the errors.