CSE 881: Data Mining (Fall 2016 Homework 4)

Due date: Oct 4, 2016 (before midnight).

A soft copy of your homework must be submitted via handin. All submitted homework must be your own work.

1. In this exercise, you will apply different regression techniques to predict the popularity of a blog article posted on Mashable web site. Download the data set from the UCI machine learning repository located at `http://archive.ics.uci.edu/ml/datasets/Online+News+Popularity`. After unzipping the file, remove the first two columns, `url` and `timedelta`, of the data file `OnlineNewsPopularity.csv`. You can use Excel to do this. Of the remaining 59 columns, use the first 58 columns as predictor variables and the last column (`share`) as the response/target variable. The target variable indicates the popularity of the online article (measured in terms of the number of times it was shared). Since the distribution is skewed, you should first apply natural log to transform the target variable y before applying the regression techniques below, i.e., $y \rightarrow \log_e y$.

   (a) Excluding the first two attributes that you've removed, there are 6 groups of predictor attributes in the data. The first group (attributes 1-11) are content-based attributes. The second group (attributes 12-17) is related to the article category (e.g., lifestyle, entertainment, tech, etc). The third group (attributes 18-29) is related to the number of shares of its keywords and referenced articles. The fourth group (attributes 30-37) are related to the day of the week when the article was published. The fifth group (attributes 38-42) are related to the topics within the article. The sixth group (attributes 43-58) are related to the subjectivity and polarity of the words in the article.

      i. Load the data into Matlab, check to make sure the resulting matrix has 58 columns (excluding the response variable).

      ii. Use the `rank` command in Matlab to calculate the rank of the matrix. You should find the rank of the matrix is only 56, which means there are two non-linearly independent columns in the matrix.

      iii. Explain how you will identify the non-linearly independent columns. Hint: the columns should be in the same attribute group. State which are the two non-linearly independent columns.

      iv. Discard the two non-linearly independent columns. Your remaining data matrix (of predictor variables) should contain only 56 columns.

   (b) Use the first 2000 rows of the data for training and the remaining rows for testing. Standardize each predictor variable by subtracting their means and dividing by their standard deviations. Apply the following regression methods to build your model on the training data you have created.

i. Multiple linear regression (MLR). You can use Matlab's built-in `regress` function to perform linear regression.

ii. Lasso regression. You can use Matlab's built-in `lasso` function to do this. For example:

```
[w,stats] = lasso(X,y,'Alpha',1,'CV',10);
lassoPlot(w,stats,'PlotType','CV');     % use this plot to identify the best regularizer
wbest = w(:,stats.Index1SE);    % this will use the best regularizer based on
                                % 10-fold cross-validation
```

The CV option enables the function to choose the best regularization parameter (λ) based on 10-fold cross-validation. By default, the function will evaluate 100 candidate values for λ. You can use `stats.Index1SE` to identify the best regularization parameter based on cross-validation. Type `help lasso` for more information about the function. Since it is a gradient-based approach, the solution is not stable. You should repeat the lasso method several times and choose the solution that produces the best result.

Make sure you add a column vector of 1s to the predictor data matrix before applying MLR and lasso regression. For each model, you need to provide the following information in your homework solution:

i. Identify the top-10 attributes with the largest absolute parameter values (you must also show their parameter values).

ii. Calculate the correlation between the predicted and actual values on the test set. Which method performs the best on the test set (in terms of its correlation)?

iii. Append the Matlab source code you use with your PDF solution file.

(c) Repeat part (b) using kernel ridge regression, with the following Gaussian radial basis function kernel:

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\bigg(-\gamma \|\mathbf{x}_i - \mathbf{x}_j\|_2^2 \bigg),$$

There are two hyperparameters you need to tune: the kernel parameter γ and the ridge regression regularize λ. Set $\gamma = 10^{-7}$ and $\lambda = 0.001$.

i. Calculate the correlation between the predicted and actual values on the test set. Compare the result against MLR and lasso regression. Which method works the best?

ii. Attach the source code with your PDF solution file.

Hint: Calculating the kernel between training and test sets to make your predictions can be very expensive since there are more than 30,000 test examples. To speed up the computations, you need to partition the test data into smaller blocks and calculate the predicted values for each block. See the example `applyKernel.m` file.

(d) The third group of predictor variables also have a skewed distribution. Apply natural log transform to each of these attributes (i.e., attributes 18-29) before standardizing them. Build MLR, lasso, and kernel ridge regression models on the log-transformed, standardized data (on all 56 attributes). For kernel ridge regression, set $\gamma = 10^{-7}$ and $\lambda = 0.1$. Calculate the correlation between the predicted and actual values on the test set. Which method, MLR, lasso, or kernel ridge regression gives the best result? Do the results improve or become worse after natural log transform?

2. Real-world data sets are often imperfect. The objective of this exercise is to illustrate the effect of outliers on regression methods.

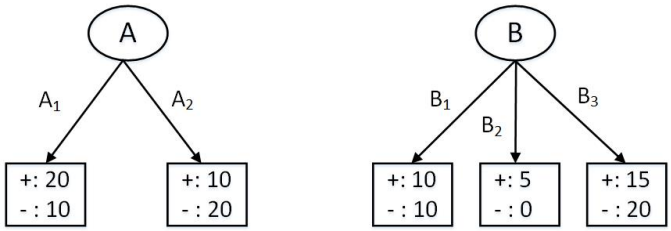
   (a) Explain how the presence of outliers can affect the parameters of a simple linear regression function. If you like, you can also provide a graphical illustration of this using a 1-dimensional input data (x) and draw the regression lines when outliers are included and excluded from the analysis.

   (b) Suppose you apply a preprocessing step to detect outliers before applying the regression method. After preprocessing, each training data point $(\mathbf{x}_i, y_i)$ has a weight α_i to indicate the degree to which it is not an outlier (i.e., the lower the weight, the more likely it is an outlier). You want to modify the ridge regression formulation to account for weights of the data points. So, you decide to use the following objective function:

$$\min_{\mathbf{w}} \sum_i \alpha_i (y_i - \mathbf{w}^T \mathbf{x}_i)^2 + \lambda \|\mathbf{w}\|^2$$

   Intuitively, the objective function will incur a smaller loss if the outlying data points are predicted incorrectly compare to a wrong prediction for the non-outliers. Assuming the set of weights for α are known, derive a formula for computing the weights of the ridge regression parameter $\mathbf{w}$ in terms of $\mathbf{X}$, $\mathbf{y}$, and $\boldsymbol{\alpha}$.

3. Consider a training set that has 60 examples. Half of the examples belong to the positive (+) class while the remaining examples belong to the negative (−) class. Suppose you are given two candidate splitting attributes, $\mathbf{A}$ and $\mathbf{B}$, as shown in the figure below. Calculate the average weighted entropy for each candidate. Based on their entropy values, which attribute, A or B, should be chosen to split the data? Show you calculations clearly.


A
A_1
A_2
+: 20
- : 10
+: 10
- : 20
B
B_1
B_2
B_3
+: 10
- : 10
+: 5
- : 0
+: 15
- : 20