Assignment 1: Regression

Due October 4 at 11:59pm

This assignment is to be done individually.

Important Note: The university policy on academic dishonesty (cheating) will be taken very seriously in this course. You may not provide or use any solution, in whole or in part, to or by another student.

You are encouraged to discuss the concepts involved in the questions with other students. If you are in doubt as to what constitutes acceptable discussion, please ask! Further, please take advantage of office hours offered by the TAs if you are having difficulties with this assignment.

DO NOT:

- Give/receive code or proofs to/from other students
- Use Google to find solutions for assignment

DO:

- Meet with other students to discuss assignment (it is best not to take any notes during such meetings, and to re-work assignment on your own)
- Use online resources (e.g. Wikipedia) to understand the concepts needed to solve the assignment

---

1 Probabilistic Modeling

In lecture we went over an example of modeling coin tossing – estimating a parameter μ, the probability the coin comes up heads.

Consider instead the problem of modeling a 6-sided die.

1. What is the parameter that explains the behaviour of the die in this case (in analogy to the μ for the coin)?
2. What is the value of the parameter for a fair die (equal probability of rolling any number)?
3. What is the value of the parameter for a die that always rolls a 2?
4. Specify the domain of the parameter – which settings of the parameter are valid.

2 Weighted Squared Error

The sum-of-squares error function for regression (Eqn. 3.12 in PRML) treats every training data point equally. In some instances, we may wish to place different weights on different training data points. This could arise if we have confidence estimates of the accuracy of each training data point.

Consider the weighted sum-of-squares error function:

$$E_{\hat{D}}(\boldsymbol{w}) = \frac{1}{2}\sum_{n=1}^{N} \alpha_n \{t_n - \boldsymbol{w}^T \boldsymbol{\phi}(\boldsymbol{x}_n)\}^2 \tag{1}$$

with weights $\alpha_n > 0$ on each training data point.

Derive the optimal weights $\boldsymbol{w}$ given this weighted sum-of-squares error function.

3 Training vs. Test Error

For the questions below, assume that error means RMS (root mean squared error).

1. Suppose we perform unregularized regression on a dataset. Is the **validation error** always higher than the **training error**? Explain.
2. Suppose we perform **unregularized** regression on a dataset. Is the **training error** with a degree 10 polynomial always lower than or equal to that using a degree 9 polynomial? Explain.
3. Suppose we perform both **regularized** and **unregularized** regression on a dataset. Is the **testing error** with a degree 20 polynomial always lower using **regularized** regression compared to **unregularized** regression? Explain.

4 Basis Function Dependent Regularization

In lecture we saw a regularization technique applied to linear regression where all weights in the regression model are regularized in the same fashion (like L_1, or L_2), and with a common value for λ. Consider the case where for each weight w_n, we have a different tradeoff parameter λ_n, and a choice from among one of L_1 or L_2 regularizer. Derive the formula of the gradient for the regularized squared error loss function in this scenario.

$$\nabla E(\boldsymbol{w}) = \ ?$$

(Hint for notation: Let $\mathcal{J}_1$ be the set of indices of basis functions whose weights have L_1 regularization, and $\mathcal{J}_2$ be the set of indices of basis functions whose weights have L_2 regularization. Alternately, you may define and use other suitable notation.)

5 Regression

In this question you will train models for regression and analyze a dataset. Start by downloading the code and dataset from the website.

The dataset is created from data provided by UNICEF's State of the World's Children 2013 report: `http://www.unicef.org/sowc2013/statistics.html`

Child mortality rates (number of children who die before age 5, per 1000 live births) for 195 countries, and a set of other indicators are included.

5.1 Getting started

Run the provided script `polynomial_regression.py` to load the dataset and names of countries / features.

Answer the following questions about the data. Include these answers in your report.

1. Which country had the highest child mortality rate in 1990? What was the rate?
2. Which country had the highest child mortality rate in 2011? What was the rate?
3. Some countries are missing some features (see original .xlsx/.csv spreadsheet). How is this handled in the function `assignment1.load_unicef_data()`?

For the rest of this question use the following data and splits for train/test and cross-validation.

- **Target value**: column 2 (Under-5 mortality rate (U5MR) 2011)[1].
- **Input features**: columns 8-40.
- **Training data**: countries 1-100 (Afghanistan to Luxembourg).

[1]Zero-indexing, hence `values[:,1]`.

- **Testing data**: countries 101-195 (Madagascar to Zimbabwe).
- **Cross-validation**: subdivide training data into folds with countries 1-10 (Afghanistan to Austria), 11-20 (Azerbaijan to Bhutan), ... I.e. train on countries 11-100, validate on 1-10; train on 1-10 and 21-100, validate on 11-20, ...

5.2 Polynomial Regression

Implement linear basis function regression with polynomial basis functions. Use only monomials of a single variable (x_1, x_1^2, x_2^2) and no cross-terms ($x_1 \cdot x_2$).

Perform the following experiments:

1. Create a python script `polynomial_regression.py` for the following.

   Fit a polynomial basis function regression (unregularized) for degree 1 to degree 6 polynomials. Include bias term. Plot training error and test error (in RMS error) versus polynomial degree.

   Put this plot in your report, along with a brief comment about what is "wrong" in your report.

   Normalize the input features before using them (not the targets, just the inputs $\boldsymbol{x}$). Use `assignment1.normalize_data()`.

   Run the code again, and put this new plot in your report.

2. Create a python script `polynomial_regression_1d.py` for the following.

   Perform regression using just a single input feature.

   Try features 8-15 (Total population - Low birthweight). For each (un-normalized) feature fit a degree 3 polynomial (unregularized). Try with and without a bias term.

   Plot training error and test error (in RMS error) for each of the 8 features. This should be as bar charts (e.g. use `matplotlib.pyplot.bar()`) — one for models with bias term, and another for models without bias term.

   Put the two bar charts in your report.

   The testing error for feature 11 (GNI per capita) is very high. To see what happened, produce plots of the training data points, learned polynomial, and test data points. The code `visualize_1d.py` may be useful.

   In your report, include plots of the fits for degree 3 polynomials for features 11 (GNI), 12 (Life expectancy), 13 (literacy).

5.3 Sigmoid Basis Functions

1. Create a python script `sigmoid_regression.py` for the following.

   Implement regression using sigmoid basis functions for a single input feature. Use two sigmoid basis functions, with $\mu = 100, 10000$ and $s = 2000.0$. Include a bias term. Use un-normalized features.

Fit this regression model using feature 11 (GNI per capita).

In your report, include a plot of the fit for feature 11 (GNI).

In your report, include the training and testing error for this regression model.

5.4 Regularized Polynomial Regression

1. Create a python script `polynomial_regression_reg.py` for the following.

   Implement L_2-regularized regression. Fit a degree 2 polynomial using $\lambda = \{0, .01, .1, 1, 10, 10^2, 10^3, 10^4\}$. Use normalized features as input. Include a bias term. Use 10-fold cross-validation to decide on the best value for λ. Produce a plot of average validation set error versus λ. Use a `matplotlib.pyplot.semilogx` plot, putting λ on a log scale[2].

   Put this plot in your report, and note which λ value you would choose from the cross-validation.

[2]The unregularized result will not appear on this scale. You can either add it as a separate horizontal line as a baseline, or report this number separately.

Submitting Your Assignment

The assignment must be submitted online at `https://courses.cs.sfu.ca`. In order to simplify grading, you must adhere to the following structure.

You must submit two files:

1. You must create an assignment report in **PDF format**, called `report.pdf`. This report must contain the solutions to questions 1-4 as well as the figures / explanations requested for 5.

2. You must submit a .zip file of all your code, called `code.zip`. This must contain a single directory called `code` (no sub-directories, no leading path names), in which all of your files must appear[3]. There must be the 4 scripts with the specific names referred to in Question 4, as well as a common codebase you create and name.

   As a check, if one runs

   ```
   unzip code.zip
   cd code
   ./polynomial_regression_1d.py
   ```

   the script produces the plots in your report from the relevant question.

[3]This includes the data files and others which are provided as part of the assignment.