University of Michigan – Dearborn
Department of Computer and Information Science
CIS 581: Computational Learning
(Winter 2023)
Instructor: Luis E. Ortiz

Project 1: (Generalized) Linear Regression, Model Selection (via Cross Validation and Regularization), and Model Evaluation; Application: Polynomial Curve-Fitting Regression for Working-Age Data

Overview

In this project students will apply polynomial curve-fitting for regression learning based on *root-mean-squared error (RMSE)* on a data set of U.S.A Working-Age Population Data. [1] Students will use *cross-validation (CV)* for model selection. Students will use CV to select the optimal value for the degree of a low-degree polynomial to use on all the training data. Students will use a *test set* to evaluate the RMSE of the optimal polynomial models students selected at the end of learning process.

As stated in the previous paragraph, in this project, the *hypothesis function h* would correspond to some *polynomial* from the different hypothesis classes under consideration depending on the polynomial *degree*. Students will need to *compute and record the RMSE* for *each h* found for *each* corresponding hypothesis class on *each* fold of CV. Then students will need to *compute the average RMSE* that each hypotheses h's, for *each* hypothesis class, achieved *over the different folds.* For each case of the low-degree polynomial up to degree 12, students will *select the hypothesis-class model achieving the minimum average RMSE.* Such a hypothesis class corresponds to the *best CV model class* for your final hypothesis. Finally, students will *apply the ML curve-fitting algorithm* on *all* the training data, using the best model class they found using CV.

U.S.A Working-Age Population Data

You are given a dataset of an indicator of working-age population in the U.S.A. through time. The working-age population is defined as those aged 15 to 64. This indicator measures the share of the working-age population in the total population for a number of years between 1970 and 2021, not necessarily consecutive. The *only input attribute* is the *year*. The *output* is the *(numerical) indicator of the working-age population* for the given input year. The files `train.dat` and `test.dat` contain the *training* and *test* datasets, respectively. Each consists of *two columns*. The *first column* corresponds to the *calendar-year values (input)* and the *second column* is the *indicator of the working-age population (output)* (**NOTE**: The examples are not chronologically sorted by the year input.)

Regularized Squared-Error Regression using Polynomial Hypothesis Classes

For the working-age data, we have a single real-valued input feature (the normalized year) and a single real-valued output (the normalized working-age indicator), so that the domain, or feature

[1] `https://data.oecd.org/pop/working-age-population.htm`

space, and the range, or output space, are both one-dimensional. Note that, for numerical reasons, the input-output example pairs must be appropriately normalized, as described above.

In this project, you will use polynomials up to degree d as the hypothesis class. Hence, recall that each hypothesis h in our class has the form $h(x) = \sum_{i=0}^{d} w_i x^i = w_0 + w_1 x + w_2 x^2 + \ldots + w_d x^d$. Also recall that if the training data is of size m, and we denote the lth example as $\left(x^{(l)}, y^{(l)}\right)$, the ($\ell_2$)-regularized squared-error function is

$$L(\mathbf{w}) = \frac{1}{2} \sum_{l=1}^{m} \left(y^{(l)} - \sum_{i=0}^{d} w_i \left(x^{(l)}\right)^i \right)^2 + \frac{\lambda}{2} \|\mathbf{w}\|_2^2$$

where $\|\mathbf{w}\|_2 \equiv \sqrt{w_0^2 + w_1^2 + \cdots + w_d^2} = \sqrt{\sum_{i=0}^{d} w_i^2}$ is the ℓ_2-norm, or magnitude, of the vector of polynomial coefficient-weights.

Recall that the idea of adding the regularization term is to penalize for large-magnitude coefficient-weights which tend to lead to more complex polynomial curves, and thus are more likely to overfit the training data. Note that when $\lambda = 0$, we have the standard squared-error function used for polynomial curve-fitting regression, without regularization. Thus, if we do not use regularization ($\lambda = 0$), minimizing $L(\mathbf{w})$ is equivalent to minimizing RMSE.

In this project, you will consider several values for the degree of the polynomial $d = 0, 1, \ldots, 12$ without regularization ($\lambda = 0$), and for $d = 12$, consider several values for the regularized parameter $\lambda = 0, \exp(-25), \exp(-20), \exp(-14), \exp(-7), \exp(-3), 1, \exp(3), \exp(7)$, where $\exp(z) = e^z$ is the natural-exponential function.

Data Scaling/Normalization for Robust Learning

As presented and discussed during lectures, students must "normalize" the data in order to help the learning algorithm output a hypothesis in a more numerically robust and accurate way. Students must use the same simple normalization algorithm, called "standard scaling," discussed during lectures in the similar context of the climate data; that is, apply a simple linear transformation to the input and output values separately that leads to the average of the values being 0 and the (empirical, unbiased estimator of the) standard deviation being 1. Recall that errors must still be evaluated in the original output space, not the transformed/normalized output space.

The ML Regression Algorithm

In this project students will apply the learning algorithm discussed in class, including the data normalization step stated above to the training data and to appropriate evaluate the learned classifier on test data. Students can either implement the entire algorithm from scratch or write code that uses appropriate existing libraries to perform the learning and evaluation tasks. Note that if students use existing libraries, they must make sure to use the appropriate methods and call them with the appropriate parameters to produce *exactly the same results that they would have obtained if they had coded the entire process themselves*; that is, they must understand exactly how the methods in the libraries they use work and what they are producing when called.

Learning to Predict U.S.A. Working-Age Population Indicator

You will learn the best d-degree polynomial using 6-fold CV on the training data to select the optimal polynomial-degree value d to use from the set $\{0, 1, 2, 3, 4, 5, \ldots, 12\}$ (that is, $d = 0$ means using just a constant, $d = 1$ means standard linear regression, etc., up to $d = 12$ degree polynomial).

You will also learn the regularized coefficient-weights of a 12-degree polynomial using 6-fold CV on the training data to select the optimal regularization parameter λ to use from the set $\{0, \exp(-25), \exp(-20), \exp(-14), \exp(-7), \exp(-3), 1, \exp(3), \exp(7)\}$.

For CV, you must create each fold by splitting the examples in the training dataset using the *same order* as they appear in the data file. Specifically, the *first* fold corresponds to the examples with indexes $1, 2, 3, 4, 5, 6, 7$; the *sixth* fold corresponds to the examples with indexes $36, 37, 38, 39, 40, 41, 42$; and similarly for the indexes to examples in each of the other folds.

During each of the 6 folds of CV, you will obtain a RMSE value for each d or λ value considered. Student must appropriately record those values so that they can evaluate/report the *average* of the RMSE values obtained for each d or λ value during each of the 6 folds of CV.

Suppose d^* or λ^* is the d or λ value with the lowest RMSE after 6-fold CV. Then students must obtain the coefficient-weights of the d^*-degree polynomial using *all* the training data. Students must also run regularized polynomial curve-fitting regression for a 12-degree polynomial using *all* the training data with λ^* as the regularization parameter. Report the *coefficient-weights* for d^* and λ^*, and the corresponding values of the *training and test* RMSE obtained for the resulting polynomials. Student must also plot all the training data along with the resulting polynomial curves for d^* and λ^*,, for the range of years 1968-2023 as input.

Once again, as with the ML regression algorithm described above, students are encouraged to write code for the CV process and the final learning, both including the data transformation/normalization described earlier, as weel as the final evaluation tasks from scratch; alternatively, students can use existing libraries or tools as long as they call those libraries or set up those tools in a way that would produce the same results as they would have obtained if they had coded the entire process themselves from scratch.

What to Turn In

You must submit the following (electronically via Canvas):

1. A **written report** (*in PDF*) that includes (1) the averages of the RMSE values obtained during the 6-fold CV for each case; (2) the optimal degree d^* and regularization parameter λ^* obtained via the 6-fold CV; (3) the coefficient-weights of the d^*-degree polynomial and the λ^*-regularized 12-degree learned on all the training data; (4) the training and test RMSE of that final, learned polynomials; (5) the 2 plots containing all the training data along with the resulting polynomial curves for d^* and λ^*, for the range of years 1968-2023 as input; and (6) a brief discussion of your findings and observations.

2. All your **code and executable** (as a tared-and-gziped compressed file), with instructions on how to run your program. A platform-independent standalone executable is preferred; otherwise, also provide instructions on how to compile your program. In general, the submitted program must be able to be executed as a standalone program from the command-line; and maybe only requiring an additional, previous compilation step, in which case, the compiler must also

be excutable from the command-line. (*Student must use standard tools/compilers/etc., generally available for* **all** *popular platforms. Also, students* **must not** *submit source code that relies on or assumes that the resulting program will be run within a specific software-development IDE such as Microsoft's Visual Studio, Sun Microsystem's Eclipse, or Apple's Xcode.*)

Collaboration Policy: *While discussing general aspects of the project with peers is generally OK, each student must write and turn in their own report, code, and other required materials, based on the student's own work.*