

# Homework 1

## 1 Directions:

- **Due: Sunday February 16, 2020 at 10:00pm** Late submissions will be accepted for 12 hours after that time, with a 15% penalty. (the enforcement is strict, beginning at 10:01pm, except for extreme situations; having a poor wifi connection or minor computer problems is not sufficient for the penalty to be waived.)
- Upload the homework to Canvas as a pdf file. Written responses must be typed. Text should be Times New Roman font size 12 or similar size in other fonts.

Unless specified otherwise, plots should be computer generated (make sure axis ranges are appropriate, tick marks and labels are legible, etc.). You can use Microsoft Word (or similar) or latex, then convert to pdf.

- Any non-administrative questions must be asked in office hours or (if a brief response is sufficient) Piazza.
- We recommend using Python with the machine learning library scikit-learn. Anaconda packages that library and other useful ones (like Matplotlib, Numpy, Statsmodels, and Pandas) together with both Spyder (a nice IDE similar to Matlab or RStudio) and Jupyter notebook. We recommend getting Python 3.7:

<https://www.anaconda.com/distribution/>

## 2 Problems

**Problem 1.** [60 points] For this problem, you will model feature  $y$  using polynomials of  $x$  up to order 30,

$$y = \sum_{i=0}^{30} a_i x^i.$$

In addition to looking at issues like training error, test error, overfitting, and so on, we will also explore how the amount of data (# of samples  $n$ ) matter.

- (a). First, let's generate a *ground-truth* model that the data comes from for reference (in practice we usually won't have this, but this will help us gain intuition). Recall that if we have  $p + 1$  data points, we can fit that perfectly with a  $p$ th order polynomial. Fit a third order polynomial to the following four points

$$(0, 10) \quad (5, 5) \quad (8, 12) \quad (12, 0).$$

You can fit it using linear regression (such as with scikit-learn

[https://scikit-learn.org/stable/modules/generated/sklearn.linear\\_model.LinearRegression.html](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html)

and treating  $x^0$ ,  $x^1$ ,  $x^2$ , and  $x^3$  as separate covariates.

- (b). Make a plot of those points (big black circles) and the fit polynomial (red curve) with an  $x$  range of  $[-5, 20]$  and a  $y$  range of  $[-5, 20]$  to confirm the fit.
- (c). The scikit-learn built-in linear regression function (and most built-in functions for other languages) use mean square error (MSE) as the only, or at least default, total-loss function. In 1-3 sentences, explain why the loss-function matters or not *for that fit specifically*. In other words, if you used a different total-loss function in part (a)., like mean absolute error or a weighted average of an asymmetric loss function, would you have gotten a different polynomial?
- (d). Now let's generate some data using that polynomial.

- First, there might be some distribution to the  $x$  values in the data. For this problem, we'll use the uniform distribution over the interval  $[0, 15]$ . Numpy has a function for generating uniform random variables:

<https://docs.scipy.org/doc/numpy-1.15.0/reference/generated/numpy.random.uniform.html>

Generate  $n = 30$  random variables. Each of these is an  $x$  value.

- Now for the  $y$  values. Generate  $y$  using the polynomial you fit in part (a). *plus* independent and identically distributed noise. Use the normal distribution,  $\mathcal{N}(0, 10)$ ,

<https://numpy.org/devdocs/reference/random/generated/numpy.random.Generator.normal.html#numpy.random.Generator.normal>

- Now make a plot of the fit polynomial (red) and the  $n = 30$  data points (black circles) you generated. The data should follow the curves of the polynomial but be scattered about it.
- (e). We are now ready to start fitting models. First, let's look at how well constant models  $y = a_0$  fit the data under different loss functions. Make a plot with  $a_0$ -values as the horizontal axis, ranging from  $[-10, 20]$  with 100 values linearly spaced, and the total-loss along the vertical axis, for each of the total-loss functions
- mean squared error (MSE)  $\frac{1}{n} \sum_{i=1}^n |res(i)|^2$
  - mean absolute error (MAE)  $\frac{1}{n} \sum_{i=1}^n |res(i)|^1$
  - a special total-loss function  $\sum_{i=1}^n \frac{1}{|x(i)-5|+0.01} l(res(i))$  where  $x(i)$  is the  $x$ -feature value of sample  $i$  and the loss function  $l(\cdot)$  is

$$l(res) = \begin{cases} -\frac{1}{5}res & \text{if } res < 0 \\ 10res & \text{if } res \geq 0 \end{cases}.$$

where  $res(i) = y(i) - \hat{y}(i)$  denotes the residual of the  $i$ th sample.

- (f). In 2-4 sentences, explain
- i. does the loss function  $l(\cdot)$  prefer over-estimating or under-estimating  $y$  values?
  - ii. does the loss function put more emphasis or less emphasis on points close to  $x = 5$ ?
- (g). Calculate the mean and the median of the  $y$ -values for the data. Do you notice anything special about MAE and MSE minimizers? Explain in 2-5 sentences why that makes sense (or not) based on how residual magnitudes get penalized. Alternatively, you can use calculus to prove that they make sense (or not); i.e. you can solve for the minimizers.
- (h). Now let's look at higher-order polynomials. Treating each  $x^i$  as a different covariate, we will fit the data with polynomials from order 0 to order 30. **Caution: Some regression functions return wrong answers when the number of coefficients approaches the number of samples. If your results make sense for polynomials up to 10 or so, but do not make sense for higher orders, that's fine, submit what you have.** Since MSE is the built-in total-loss function, we will use that (though keep in mind we could use others if someone had already coded the fitting procedure).
- i. First, to evaluate over-fitting, we will use validation. Use the first 20 samples for fitting, reserving the rest as a validation set.
  - ii. Make a plot with title "Training Loss" plotting the MSE for the 20 samples used to fit for each polynomial for  $p = 0$  to  $p = 30$ . In this plot, include a green horizontal line for the ground-truth model's MSE on the training data.
  - iii. In 2-5 sentences, comment on how the training loss curve behaves, such as where it has a steep slope, where it has a shallow slope, where it goes to 0. Do not simply describe the plot, but explain why it is that way, using knowledge of the ground-truth model.
  - iv. Make another plot with the same axes, titled "Validation Testing Loss," plotting the MSE for the 10 samples held out. In this plot, include a green horizontal line for the ground-truth model's MSE on the validation data.
  - v. In 3-6 sentences, comment on how the validation testing loss curve behaves, explaining why it is that way, using knowledge of the ground-truth model. Your comment should also explain the similarities and differences between the training loss curve and the validation loss curve.
  - vi. Now generate a new data-set with  $n = 1000$ . Make another plot, titled "Ground-Truth Testing Loss  $n = 1000$ " with the MSE of the polynomials fitted to the  $n = 30$  data set evaluated on this new, larger data-set. **(This is a similar exercise to (h).iv except you are using a much more data)** Use the same axes as the other plots. (keep in mind that in practice, we cannot do this since we won't know the ground truth.) In this plot, include a green horizontal line for the ground-truth model's MSE on this  $n = 1000$  sample data-set.
  - vii. In 2-4 sentences, comment on how well (or not) the validation set testing loss approximates the ground-truth testing loss.

- viii. Next, let's look at model complexity instead of using validation data. Fit polynomials using all  $n = 30$  data points and keep track of the corresponding MSE. Then calculate

$$\begin{aligned} \text{Total Complexity} &= \text{Total Training Loss} + \lambda \cdot \text{Model Complexity} \\ &= \text{MSE} + \lambda p \end{aligned}$$

where  $n$  refers to the number of samples used to fit (so  $n = 30$ ),  $p$  is the model order, and  $\lambda$  is a threshold we choose. In class, for examples, we've used  $\lambda = 1$  for illustration, but in practice we need to be a bit careful. We need to convert scales, like if we want to add

$$3.2 \text{ miles} + 17 \text{ inches},$$

the summed number is not 20.2. Instead we need to make sure they are on the same *scale*. (Chapter 6.1.3 of the textbook describes a few  $\lambda$  values that have been derived by statisticians, such as Mallows's  $C_p$ , AIC, and BIC. ) For us, let's pick a  $\lambda$  such that the model with  $p = 30$  has the same total-complexity as the model with  $p = 0$ .

- Write what that  $\lambda$  should be in terms of  $p$  and the MSE of certain models. Use the notation  $MSE(i)$  for the MSE for the fitted polynomial of order  $i$ .
  - Make a plot of the total-complexity as a function of the model order  $p$ . Use the title "Total Complexity  $p$ "
  - In 2-5 sentences, explain the behavior of the total complexity curve with reference to the validation testing curve and the ground truth testing curves.
- ix. We can also use other penalties. Reusing the fitting from part (h).viii, select a new  $\lambda$  and make new plots for penalties other than  $\lambda p$
- $\lambda_1 \sum_{i=0}^p |a_i|^1$ , with plot titled "Total Complexity L1"
  - $\lambda_2 \sum_{i=0}^p |a_i|^2$  with plot titled "Total Complexity L2."

In 2-5 sentences, discuss why the curves behave the way that they do in comparison/contrast to the curves using validation testing data and ground-truth testing data.

- (i). Lastly, let's see how much getting more data helps to find the correct model.
- i. Generate  $n = 5000$  samples which will be used for fitting.
  - ii. For  $i = 1, \dots, 100$ , fit the best  $p = 30$  polynomial using  $n = 50i$  samples from the 5000 sample data set.
  - iii. Plot each coefficient  $\{a_0, a_2, a_4, a_6, a_8\}$  as a function of  $50i$  in separate plots. (so one plot for  $a_0$ , etc). In each plot, draw a green line for the corresponding coefficient in the ground-truth polynomial.
  - iv. In 2-5 sentences, discuss what is happening to the coefficients and comment on how the amount of over-fitting depends on the number of samples used.

**Problem 2.** [40 points] For this problem, you will find a model to predict housing prices using the data set `housingdata.csv` on canvas. Background information is described in <http://lib.stat.cmu.edu/datasets/boston>

**Description:** There are 14 attributes for about 500 samples. The attributes are:

- CRIM - per capita crime rate by town
- ZN - proportion of residential land zoned for lots over 25,000 sq.ft.
- INDUS - proportion of non-retail business acres per town.
- CHAS - Charles River dummy variable (1 if tract bounds river; 0 otherwise)
- NOX - nitric oxides concentration (parts per 10 million)
- RM - average number of rooms per dwelling
- AGE - proportion of owner-occupied units built prior to 1940
- DIS - weighted distances to five Boston employment centres
- RAD - index of accessibility to radial highways
- TAX - full-value property-tax rate per \$10,000
- PTRATIO - pupil-teacher ratio by town
- B -  $1000(Bk - 0.63)^2$  where  $Bk$  is the proportion of black residents by town
- LSTAT - % lower status of the population
- MEDV - Median value of owner-occupied homes in \$1000's

Our goal will be to use this dataset to predict median values (MEDV) for new towns or towns whose socio-economics have changed, based on the other factors. We will use MSE as the total-loss function throughout this problem.

- (a). First, make a scatterplot for each of the attributes with MEDV (with MEDV along the vertical axis). You should have 13 plots total.
  - The vertical axis should be MEDV, and the range should be the same for all 13 plots. Label the axis "MEDV."
  - the title and x-axis label of each plot should be the attribute name.
  - In 3-6 sentences describe which features might be most and least relevant to include and explain why. (Though keep in mind these figures are just low-dimensional projections and sometimes there might be complex interactions between features.)
- (b). Find the best fitting linear model for every subset of AGE, INDUS, NOX, RM, TAX using the first  $n = 400$  samples. Reserve the remaining samples for validation.
- (c). For  $i = 0$  to  $i = 5$ , select the model with  $i$  attributes from part (b). that has the lowest validation set MSE.
  - i. Write down what the best subset was for each  $i$ . Briefly comment on to what extent they are nested.

- ii. Make a plot of the training data total loss of the best fitting model with  $i$  variables
  - The horizontal axis is the number of variables  $i$  for  $i = 0$  to  $i = 5$ .
  - Title the plot “Training loss - best subsets”
  - Using the validation data set (samples not used in fitting), make a similar plot of the validation set’s MSE, titled “Validation loss - best subsets”
  - In 2-4 sentences, comment on how the the two plots are similar/differ.
- iii. Now find the best fitting linear model for every subset of AGE, INDUS, NOX, RM, TAX using all the samples. Instead of using a validation set, we will use a complexity penalty. We will measure total complexity using Mallows’s  $C_p$ ,

$$C_p = \text{MSE} + \frac{2\hat{\sigma}^2}{n}i$$

where  $i$  is the number of attributes used and  $\hat{\sigma}^2$  is the MSE using all 5 variables. Thus, you can view  $\frac{2\hat{\sigma}^2}{n}$  as a carefully chosen  $\lambda$ .

- Make a plot of the total complexity  $C_p$  as a function of the number of variables  $i$ , where for each  $i$  you are using the best fitting model that has  $i$  variables.
  - Discuss in 2-5 sentences how the results from using total complexity are similar/different to using a validation set.
- iv. Next, let’s consider alternative model complexities. Documentation for scikit-learn’s implementation of ridge regression (with L2 norm on coefficients) is at [https://scikit-learn.org/stable/modules/generated/sklearn.linear\\_model.Ridge.html](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Ridge.html) and LASSO (with L1 norm on coefficients) is at [https://scikit-learn.org/stable/modules/generated/sklearn.linear\\_model.Lasso.html](https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.Lasso.html)
    - i. Split up the data again, with the first  $n = 400$  samples for fitting. Reserve the remaining samples for validation.
    - ii. For ridge regression, make a plot of the validation set’s total complexity (MSE +  $\lambda$ \*L2 norm) as a function of  $\lambda$ .  $\lambda$  should range from 0 (no penalty on model complexity) to a value large enough that the total complexity (MSE +  $\lambda$ \*L2 norm) is close to **that of the  $\lambda = 0$  model**.
    - iii. Report what model minimizes that curve and discuss how it compares with the model selected using Mallows  $C_p$  and using validation error without model complexity penalties.
    - iv. Repeat the previous two steps for LASSO.
    - v. In the steps above, we did not normalize the attributes to each have values in  $[0, 1]$ . In 3-5 sentences, discuss whether that matters for the the feature being predicted and/or the features we are using in the prediction; would the results be different?
- (d). The above model selections were for a subset of attributes. Now let’s find a good model among all possible subsets of features. But we need to be smart about how we search, because there are  $2^{13}$  subsets of features.

- i. Perform a forward search (using validation MSE to compare) to select which models to branch off of. Use  $n = 400$  samples for fitting and the rest for validation. Go from the 0th order model to the full model (using all features).
  - i. Make a plot of training set MSE as a function of the number of features
  - ii. Make a plot of validation set MSE as a function of the number of features
  - iii. Report which subset of features is best (in terms of validation MSE) across all possible subsets of features.
- ii. Repeat using a backward search. Use  $n = 400$  samples for fitting and the rest for validation. Go from the the full model (using all features) to the 0th order model. Plot and report similar to the forward search.
- iii. In 3-5 sentences, comment on similarities/differences in the nested sequence of models considered and the corresponding best models found using each method. Note that in practice, we would stop the forward/backward early unless the results kept improving.
- iv. Repeat the steps from part (c).iv for LASSO and Ridge regression, except use all features and in the discussion compare to the results using validation instead of Mallows's  $C_p$ .