CS 181 Spring 2019 Section 2 Notes (Model Selection)

1 Validation

1.1 Linear Regression

Suppose we have data $\{(x_i, y_i)\}_{i=1}^n$, with $x_i, y_i \in \mathbb{R}$, and we want to fit polynomial basis functions:

$$\boldsymbol{\phi}(x)^\top = [\phi_1(x) = 1, \phi_2(x) = x, \ldots, \phi_{d+1}(x) = x^d]$$

$$h(\mathbf{x}; \mathbf{w}) = \mathbf{w}^\top \boldsymbol{\phi}(x)$$

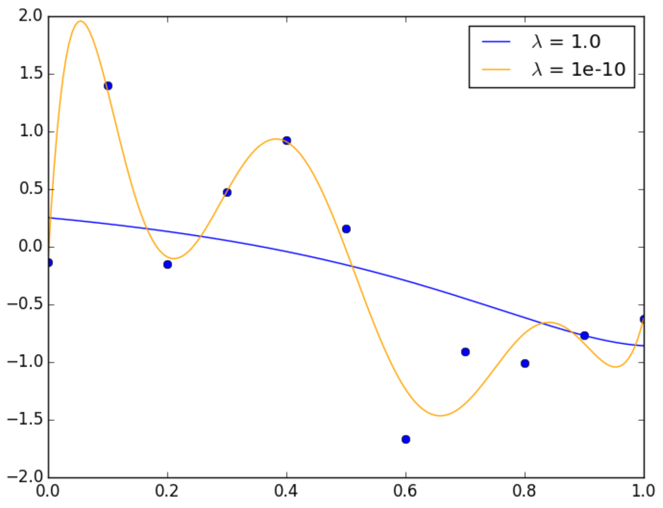
That is, we fit a degree d polynomial. With a small dataset and too high of a d, we get overfitting. Obviously, this will generalize poorly to new data points. How can we solve this problem?

1.2 Ridge Regression

One solution to overfitting linear regression is through ridge regression, which minimizes a modified least squares loss function:

$$\mathcal{L}(D) = \sum_{i=1}^{n} (y_i - h(x_i; \mathbf{w}))^2 + \frac{\lambda}{2} ||\mathbf{w}||^2$$

Ridge regression is used to *regularize* a model, making it simpler and allowing it to generalize better to new data. Indeed, the extra term penalizes overly large weights in $\mathbf{w}$, leading to smaller coefficients for a "flatter" polynomial:



1.3 Validation Set: Model Selection

We can do model selection through a *validation set*, data that are separate from our training set used to fit the regression. By separating our full dataset into a training set and validation set (say in a 90/10 split), we can use our validation set to check our model's generalization ability on data it was not trained on. When tuning model parameters, we can train our models with different parameters on the training set and check their performance on the validation set in order to find the optimal value for the parameter.

1.4 Cross Validation

Cross validation is a more sophisticated technique for obtaining validation losses. Instead of splitting our data once into a 90/10 training/validation set, in k-fold cross validation, we split our data into k equal chunks. For each chunk, we set it to be the validation set and use the rest of the data to fit our model. Then, we obtain a validation loss on our current chunk, and averaging over the 10 chunks gives the final validation loss. Cross validation can also be used to find optimal parameter values as described in the previous section - we simply have an improved way of computing validation losses by averaging. This reduces the variance in the resulting validation loss, as each example is used in estimating the validation loss.

See this notebook for an interactive demo of how cross validation could be used.

2 Bias-Variance Decomposition

Bias-variance decomposition is a way of understanding how different sources of error (bias and variance) can affect the final performance of a model. A tradeoff between bias and variance is often made when selecting models to use, and can be informed by the results of the bias-variance decomposition.

Exercise: Decompose the generalization error into the sum of bias squared (systematic error), variance (sensitivity of prediction), and noise (irreducible error) by following the steps below (**try not to peek at your notes!**). You will find the following notation useful:

- h_D: The trained model, $h_D : \mathcal{X} \mapsto \mathbb{R}$.
- D: The data, a random variable sampled $D \sim F^n$.
- x: A new input.
- y: The true result of input x. Conditioned on x, y is a r.v. (may be noise.)
- $\overline{y}$: The true conditional mean, $\overline{y} = \mathbb{E}_{y|\mathbf{x}}[y]$.
- $\overline{h}(\mathbf{x})$: The prediction mean, $\overline{h}(\mathbf{x}) = \mathbb{E}_D[h_D(\mathbf{x})]$.

1. Start with the equation for the generalization error - the expected error, in least squares terms, on an unseen sample:

$$\mathbb{E}_{D,\,y|\mathbf{x}}[(y - h_D(\mathbf{x}))^2]$$

and use the linearity of expectation to derive an equation of the form:

$$\underbrace{\mathbb{E}_{y|\mathbf{x}}[(y-\overline{y})^2]}_{\text{noise}} + \underbrace{\mathbb{E}_D[(\overline{y} - h_D(\mathbf{x}))^2]}_{\text{bias+var}} + *******\qquad(1)$$

where the *s denote a third term. What is this third term? (**Hint**: add and subtract $\overline{y}$).

2. Show that this third term is equal to 0 (**Hint**: take advantage of the fact that $\overline{y}$ and $h_D(\mathbf{x})$ do not depend on $y|x$.

3. The first term in (1) is the noise. We therefore want to decompose the second term into the bias and variance. Again, using the linearity of expectation, re-write the second term in equation (1) in the form:

$$\underbrace{(\overline{y} - \overline{h}(\mathbf{x}))^2}_{\text{bias squared}} + \underbrace{\mathbb{E}_D[(\overline{h}(\mathbf{x}) - h_D(\mathbf{x}))^2]}_{\text{variance}} + 2\mathbb{E}_D[(\overline{y} - \overline{h}(\mathbf{x}))(\overline{h}(\mathbf{x}) - h_D(\mathbf{x}))]\qquad(2)$$

show that the third term is equal to 0.

4. Plug the results of part 3 back into (1) to show that we have decomposed the error into noise, bias, and variance.

2.1 Limitations

Although the bias-variance decomposition provides some interesting insights into model selection from a complexity perspective, it has limited practical value, as it is based on averages of independent data sets drawn from some distribution. In practice, however, we only have a single observed data set. The bias and variance can be estimated through "bootstrap" style approaches where we sample with replacement to form additional data sets, but still— why not more directly compute validation loss and use this to find the best model? The main interest in the bias-variance decomposition is to gain conceptual insight.

3 Ensemble Methods

Ensemble methods take advantage of multiple models to obtain better predictive accuracy than with a single model alone. The two most common types of ensemble methods are bagging and boosting.

3.1 Bootstrap aggregating (Bagging)

In bagging, we fit each individual model on a random sample of the training set. To predict data in the test set, we either use an average of the predictions from the individual models (for regression) or take the majority vote (for classification). As an average of models, bagging tends to decrease the variance of a learning algorithm without changing the bias. An example is a random forest, which trains multiple decision trees and takes the average prediction from the ensemble of learned models.

3.2 Boosting

In boosting, we train the individual models sequentially. Thus, after training the i^{th} model on a sample of the training set, we train the $(i+1)^{th}$ model on a new sample based on the performance of the i^{th} model. Examples classified incorrectly in the previous step receive higher weights in the new sample, encouraging the new model to focus on those examples. During testing, we take a weighted average or weighted majority vote of the models' predictions based on their respective training accuracies on their reweighted training data (i.e. higher models have larger weights). A common example is the Adaboost algorithm.

4 Practice Questions

1. **Ridge Regression**

Suppose we have some data matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$ and targets $\mathbf{y} \in \mathbb{R}^n$. Suppose the data are orthogonal*, i.e. satisfies $\mathbf{X}^\top \mathbf{X} = \mathbf{I}$. Show that if $\widehat{\mathbf{w}}$ is the solution to linear regression, and $\widehat{\mathbf{w}}_{ridge}$ is the solution to ridge regression, then

$$\widehat{\mathbf{w}}_{ridge} = \frac{1}{1+\lambda}\widehat{\mathbf{w}}$$

This explicitly illustrates the phenomenon of weight shrinkage.

* Orthogonal data is a very special case in which the inner product between any two distinct features is zero. Normally we expect features to be correlated. But it is used to gain this clean illustration of the effect of ridge regression. Technically, we have $\mathbf{X} = [\mathbf{v}_1, \ldots, \mathbf{v}_m]$ where $\mathbf{v}_1, \ldots, \mathbf{v}_m$ are n dimensional, orthogonal column vectors.

2. **Bias and Variance**

We consider a very simple example where the data is a univariate Gaussian, with $x_i \sim \mathcal{N}(\mu, 1)$ with known variance but unknown mean. In this case, there are no features, and the hypothesis doesn't depend on x. A very simple hypothesis, for example, is the sample mean

$$h_D = \overline{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

for data $(x_1, \ldots, x_n) \in \mathbb{R}^n$. Calculate the bias and variance for the following two hypotheses:

(a) Estimate 1: Use the same mean of data D.

(b) Estimate 2: Use the constant hypothesis, 0.

3. **Deriving Lasso Regularization with Lagrange Multipliers**

Show that minimization of the unregularized sum-of-squares error function given by

$$E_D(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}(t_n - \mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n))^2,$$

subject to the constraint

$$\sum_{j=1}^{M}|w_j| \leq \eta,$$

is equivalent to minimizing the regularized error function

$$\frac{1}{2}\sum_{n=1}^{N}(t_n - \mathbf{w}^\mathsf{T}\phi(\mathbf{x}_n))^2 + \frac{\lambda}{2}\sum_{j=1}^{M}|w_j|$$

4. **Priors for Model Selection**

Suppose you had three models, M_1, M_2, M_3, each increasing in complexity. For example, you could imagine that the models represented unregularized polynomial regression, with M_1 linear regression, M_2 quadratic regression, and M_3 cubic regression. Within the context of Bayesian model selection, come up with a way to penalize the complexity of a model so you do not always choose M_3. Additionally, explain why, in many cases, Bayesian model selection will recover the simplest model to explain the data without explicit penalization.