CS 181 Spring 2019 Section 3 Notes (Model Selection)

1 Model Selection

1.1 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.

1.2 Exercise: Bias-Variance Decomposition

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 $D \sim F^n$.
- $\mathbf{x}$: A new input.
- y: The true result of input $\mathbf{x}$. Conditioned on $\mathbf{x}$, 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:

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

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

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

   where the *s denote a third term. What is this 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:

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

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.

Solution:

1. Follow the hint:

$$\begin{aligned}
&\mathbb{E}_{D,\,y|\mathbf{x}}[(y - h_D(\mathbf{x}))^2] \\
&= \mathbb{E}_{D,\,y|\mathbf{x}}[(y - \overline{y} + \overline{y} - h_D(\mathbf{x}))^2] \\
&= \underbrace{\mathbb{E}_{y|\mathbf{x}}[(y - \overline{y})^2]}_{\text{noise}} + \underbrace{\mathbb{E}_D[(\overline{y} - h_D(\mathbf{x}))^2]}_{\text{bias+var}} + \underbrace{2\mathbb{E}_{D,\,y|\mathbf{x}}[(y - \overline{y})(\overline{y} - h_D(\mathbf{x}))]}_{0}
\end{aligned}$$

2. Using the hint:

$$2\mathbb{E}_D[\overline{y} - h_D(\mathbf{x}) \cdot \mathbb{E}_{y|\mathbf{x}}[y - \overline{y}]] = 2\mathbb{E}_D[\overline{y} - h_D(\mathbf{x}) \cdot 0] = 0.$$

3. Following a similar procedure as in part 1:

$$\begin{aligned}
&\mathbb{E}_D[(\overline{y} - h_D(\mathbf{x}))^2] \\
&= \mathbb{E}_D[(\overline{y} - \overline{h}(\mathbf{x}) + \overline{h}(\mathbf{x}) - h_D(\mathbf{x}))^2] \\
&= \underbrace{(\overline{y} - \overline{h}(\mathbf{x}))^2}_{\text{bias squared}} + \underbrace{\mathbb{E}_D[(\overline{h}(\mathbf{x}) - h_D(\mathbf{x}))^2]}_{\text{variance}} + \underbrace{2\mathbb{E}_D[(\overline{y} - \overline{h}(\mathbf{x}))(\overline{h}(\mathbf{x}) - h_D(\mathbf{x}))]}_{0}
\end{aligned}$$

where the third term is 0 by:

$$2(\overline{y} - \overline{h}(\mathbf{x}))\mathbb{E}_D[\overline{h}(\mathbf{x}) - h_D(\mathbf{x})] = 2(\overline{y} - \overline{h}(\mathbf{x}))(0) = 0.$$

4. Substituting (2) back into (1), we have:

$$\begin{aligned}
&\mathbb{E}_{D,\,y|\mathbf{x}}[(y - h_D(\mathbf{x}))^2] \\
&= \mathbb{E}_{y|\mathbf{x}}[(y - \overline{y})^2] + (\overline{y} - \overline{h}(\mathbf{x}))^2 + \mathbb{E}_D[(\overline{h}(\mathbf{x}) - h_D(\mathbf{x}))^2] \\
&= \text{noise}(\mathbf{x}) + \text{bias}^2(h(\mathbf{x})) + \text{Var}_D(h_D(\mathbf{x})).
\end{aligned}$$

Considering the expectation over $\mathbf{x}$ (you are not asked to do this in the exercises), the generalization error is:

$$\mathbb{E}_{\mathbf{x}}\left[\text{noise}(\mathbf{x}) + \text{bias}^2(h(\mathbf{x})) + \text{Var}_D(h_D(\mathbf{x}))\right]$$

1.3 Exercise: More 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:

1. Estimate 1: Use the same mean of data D.
2. Estimate 2: Use the constant hypothesis, 0.

1. The bias is $\mu - E_D[\overline{x}] = \mu - E_D[(1/n)\sum x_i] = 0$. The variance is $E_D[(\overline{x} - \mu)^2] = \sigma^2/n$, the variance of the sample mean on n examples (as is standard).
2. The bias equals μ, the expected difference between the estimator and the true value. This prediction is constant, so the variance of our prediction is 0.

1.4 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.

1.5 Validation Set

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.

Note that this is a general approach that is highly recommended for any sort of model selection. In the context of regularization, you would use a validation set to fit the regularized linear model using multiple values of λ and choose the one that results in the best performance.

1.6 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.

1.7 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.

1.7.1 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.

1.7.2 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.

1.8 Exercise: Model Selection Using Bias and Variance

You are given a data set containing information about a set of dogs. Your goal is to model the weights of the dogs (the dependent variable) using the ages of the dogs (the independent variable). You fit and evaluate three models using the same train-test split, a linear model, a quadratic model, and a cubic model. Below is a table of the train and test accuracies.

	Linear	Quadratic	Cubic
Train	0.60	0.82	0.93
Test	0.62	0.73	0.54

1. For each of the three models, would you choose to regularize the model? What would be the effect of regularization?
2. For each of the three models, what would be the effect of adding more data and why?
3. How would each of these models perform on a freshly drawn set of dogs? Assume that the draws across both data sets are i.i.d. (i.e. using same breeds, etc. in both data sets).
4. Which model do you think is the most appropriate one for this data (based on the numerical results here, not using your intuition about dogs)?

1. The linear model is already fitting quite well as seen by the fact that the test performance exceeds the train performance, so no regularization is needed. In fact, regularization would probably only serve to reduce its power, making the model even weaker than it is.

   The quadratic and cubic models both appear to do worse on the test than the train set, suggesting that they are both overfit and would therefore benefit from regularization. Due to the nature of regularization in making a more parsimonious model, the training accuracies in these cases will likely decrease, but this is not a problem since the test accuracies should consequently increase with regularization and the test accuracy is much more important in evaluating the performance of a model.

2. The linear model will see no significant difference in performance with more data, since it is already not overfit. In fact, it might be somewhat underfit.

The quadratic model, as mentioned before, is overfit and would likely benefit from more data, since the addition of data tends to reduce overfitting.

Similarly, the cubic model appears even more overfit than the quadratic model, in which case data augmentation will be beneficial in fitting a more representative model.

3. Since both the quadratic and cubic models are overfit, there is likely to be high variance and therefore substantially different performance results if they are applied to another data set. It will be difficult to say which of the quadratic and cubic models will perform better, but it can be fair to presume that the cubic model will have higher variance due to its more extreme overfitting.

   The linear model, on the other hand, might actually be somewhat underfit, and therefore be quite biased. Based on what we know of bias-variance relationships, that suggests that the linear model will have little variance across data sets and therefore perform similarly.

4. The quadratic model almost certainly looks better than the linear model, due to the higher test performance, even if the quadratic model is overfit. However, it is unlikely but still possible that the cubic model will outperform the quadratic model once proper regularization techniques are applied.

2 Regularization

2.1 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?

2.2 Penalized Loss Function

Recall that the standard linear regression problem, known as *ordinary least squares (OLS)*, uses the following loss function (which is actually the mean squared error):

$$\mathcal{L}_{OLS}(D) = MSE = \sum_{i=1}^{n} (y_i - h(x_i; \mathbf{w}))^2$$

Regularization refers to the general practice of modifying the model-fitting process to avoid overfitting and other potential problems like multicollinearity. Linear models are typically regularized by adding a *penalization term* to the loss function. The penalization term is simply any function p of the weights $\mathbf{w}$ scaled by a penalization factor λ. The loss then becomes:

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

There are some common choices for $p(\mathbf{w})$ that will be discussed. They frequently leverage the idea of a vector norm, where $||\mathbf{w}||_n$ represents the L_n-norm of the vector $\mathbf{w}$ for $n \geq 1$:

$$||\mathbf{w}||_n = \left(\sum_{i=1}^{|\mathbf{w}|} |\mathbf{w}_i|^p \right)^{1/p}$$

2.3 LASSO Regression

One common choice for a penalization term is simply $p(\mathbf{w}) = ||\mathbf{w}||$. This is just the L_1-norm of the weights vector, which quite naively means that the penalization term here is

just the sum of the magnitudes of all the weights for the model. This form of regularized regression is known as *LASSO (Least Absolute Shrinkage and Selection Operator)* regression. The full modified loss is then:

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

There are some notable properties of LASSO regression. One main disadvantage is that it does not have a closed-form solution, meaning that it cannot be analytically solved.

> **Concept Question**: Why do you think LASSO has no closed-form solution? Try to solve for it using the same process as for the OLS solution; what goes wrong?

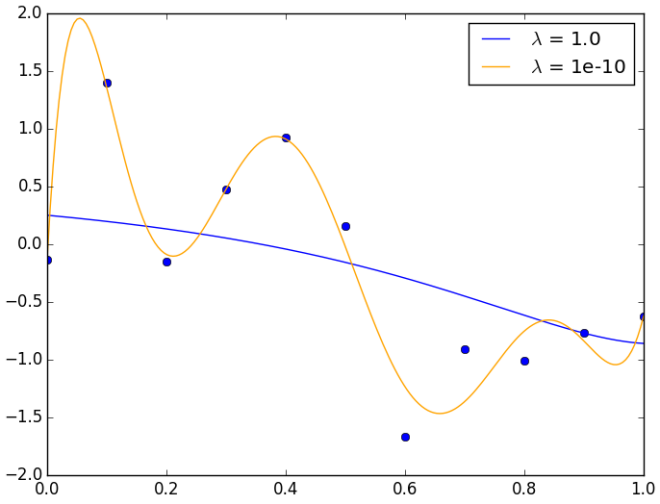
Rather, it needs to be numerically solved through an iterative process, which can be much slower. However, it does have the benefit, as the name suggests, that it is good for *variable selection*, meaning that coefficients might be "shrunk" directly to zero, which has intuitive interpretation that the variables are not meaningful in the model after regularization.

2.4 Ridge Regression

Another 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:

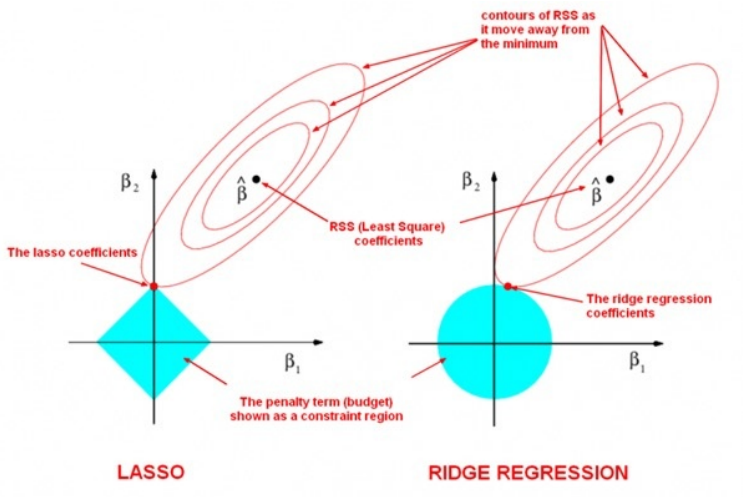


Unlike LASSO, ridge regression has a closed form solution, which makes the solution much more computationally efficient. While it does not shrink coefficients to zero, it has other intuitive properties, such as connection to a Normal prior. The analytical solution is:

$$\mathbf{w}_{ridge} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$$

The above expression can be compared to the OLS solution. The only additional term is $\lambda\mathbf{I}$, which looks like a "ridge" of λ values (hence the name ridge regression). This also helps avoid problems with singular data.

Concept Question: Solve for the closed form solution to ridge regression.



Source: https://www.quora.com/Why-does-Lasso-regression-lead-to-sparse-solutions

The above diagram is a visual comparison between the LASSO and ridge regularizations and may help you understand how they each behave when shrinking coefficients.

2.5 Exercise: 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

Recall that the linear regression solution is

$$\widehat{\mathbf{w}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$$

and recall from homework that the ridge regression solution is

$$\widehat{\mathbf{w}}_{ridge} = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I})^{-1} \mathbf{X}^\top \mathbf{y}$$

If $\mathbf{X}^\top \mathbf{X} = \mathbf{I}$, we see that

$$\widehat{\mathbf{w}} = \mathbf{X}^\top \mathbf{y}$$
$$\widehat{\mathbf{w}}_{ridge} = \frac{1}{1+\lambda} \mathbf{X}^\top \mathbf{y}$$

hence giving us the result.

2.6 Exercise: Lasso Regularization from 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

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

Rewrite the constraint as

$$\sum_{j=1}^{M} |w_j| - \eta \leq 0$$

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.

We get the Lagrangain function

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

where we introduce the factor of $1/2$ in front of the second term for convenience. We see immediately that the above function is equal to the regularized error function plus the terms of η which do not depend on $\mathbf{w}$. Therefore, minimizing the Lagrangian with respect to $\mathbf{w}$ will give the same $\mathbf{w}^*$ as minimizing the regularized error function.