September 29, 2017 Lecture 6, Sep. 25, 2017

Model and Feature Selection

Lecturer: Justin Domke *Scribe: Annamalai Natarajan*

1 Summary

In the last lecture we discussed linear regression and shrinkage. In this lecture we will discuss model selection followed by feature selection.

2 Model Selection

Thought Experiment Goal: Choose a coin that in future sequence of flips result in maximum number of heads

1. Flip a bent coin 10 times, observe the sequence of flips and count the number of heads as,

   ```
   1) H T T T T H T H T H: 4/10
   ```

2. Flip three bent coins 10 times each, observe the sequence of flips and count the number of heads as,

   ```
   1) H T T T T T H T T H: 3/10
   2) H T T H H H T H T H: 6/10
   3) T H H T T H T H T H: 5/10
   ```

3. Flip a thousand bent coins 10 times each, observe the sequence of flips and count the number of heads as,

   ```
   1) H T T T T H T H T H: 4/10
   ...
   772) H H H H T H H T H H: 8/10
   ...
   1000) T T H T T H T T H T: 3/10
   ```

In the first experiment we pick the only coin that was flipped and this gives us an estimate of future performance as 40% heads. In the second experiment if we pick (greedily) coin number 2, then this gives us an estimate of future performance as 60% heads, which is biased and unlikely (on average). In the third experiment if we follow the same heuristic and pick coin number 772 then this gives us an estimate of future performance as 80% heads, which is biased and unlikely (on average).

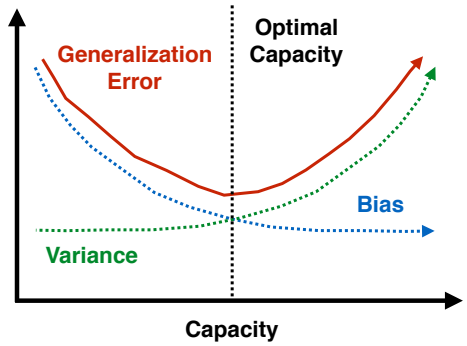


Figure 2.1: Bias and variance as a function of model capacity

The observation is that the more number of coins we flip, the more likely we are in choosing a coin that appeared completely by chance and is less likely to produce the same results (*e.g., number of heads*). This is intuition behind model selection. When we have many models to pick and choose from we need some unbiased estimate to pick and choose a model with good generalization error.

The generalization error is broadly made up of two terms: bias and variance. Hence model selection pertains to picking models that has a good generalization error. The interaction between prediction models and generalization error is represented as a function of model capacity. Prediction models that can represent more has higher capacity *e.g.,* 9^{th} *order polynomial* but tends to overfit to variance in the dataset. On the other hand prediction models that represents less has lower capacity *e.g.,* 1^{st} *order polynomial* but these models exhibit high bias.

The empirical observation between bias, variance as a function of model capacity is shown in Figure 2.1. Higher capacity models have lower bias and higher variance and vice versa for lower capacity models. When we combine both bias and variance we get the red curve which is non-linear. The goal of model selection is to find a model with optimal capacity which strikes a balances between bias and variance.

3 Feature Selection

A popular topic in machine learning is feature selection. The goal is given a dataset with p features find subset of k features that results in better performance. Feature selection is broadly categorized into two groups: one, explicit feature selection which is performed as a separate step in the processing pipeline; two, feature selection baked into prediction models. We first discuss explicit feature selection.

Best subset selection	
Pros	**Cons**
Searches through all feature subsets	Expensive
Sparse	Unstable

Table 1: Pros and cons of best subset selection

3.1 Best subset selection

This approach to feature selection is to pick and choose a subset of features, k, given all features, p. The pseudocode is presented in Algorithm 1 and the pros and cons are presented in Table 1.

Algorithm 1 Best subset selection

```
procedure BEST_SUBSET_SELECTION
    Model_list ← ()
    for k = 0, 1, 2, ..., p do
        for all subsets in {x_1, x_2, ..., x_k} do
            Fit train data for each subset
            Keep subset with best performance on train data
        end for
        Model_list ← Model_list ∪ Model with best performance for each k
    end for
    return Model_list
end procedure
```

3.2 Forward step-wise selection

Like the name indicates this approach to feature selection starts with an empty list and adds features repeatedly using some criterion such as performance The pseudocode is presented in Algorithm 2 and the pros and cons are presented in Table 2.

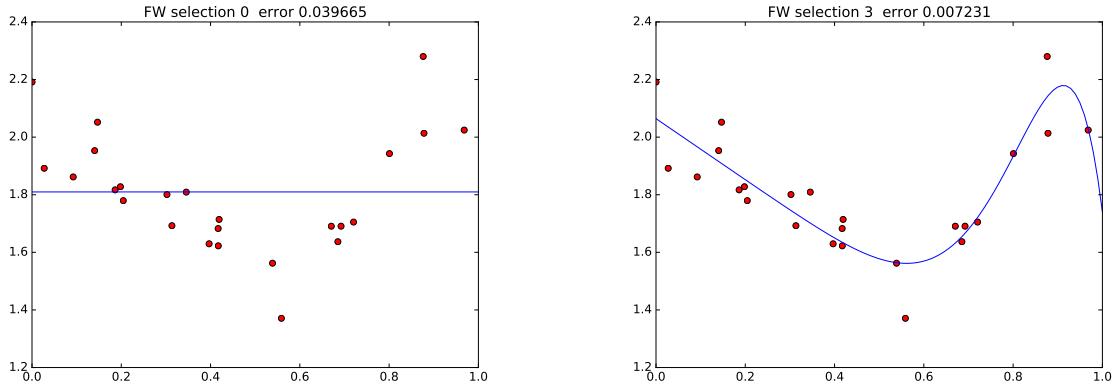
Forward step-wise feature selection	
Pros	**Cons**
Fast	Unstable
Sparse	

Table 2: Pros and cons of forward step-wise feature selection

Algorithm 2 Forward Step-wise selection

```
1: procedure FORWARD_STEP-WISE_SELECTION
2:     Model_list ← ()
3:     for k = 0, 1, 2, ..., p do
4:         Try adding each index of x
5:         Fit training data with subset
6:         Keep the model that reduced the training error the most
7:         Model_list ← Model_list ∪ Model with least training error
8:     end for
9:     return Model_list
10: end procedure
```

An example of forward step-wise feature selection is presented in Figure 3.1. The subplot (a) shows a scatter plot (red points) of the data. On the y-axis is the output values and x-axis is a place holder for input features. In this problem originally there are 11 features. The blue line in subplot (a) shows the prediction model when only selecting one feature. It is basically predicting the mean of the output values with an error of 0.039. When adding three more features for a total of four features the error drops to 0.0073 and the prediction model (blue line) is better representation of the underlying data distribution.



(a) One feature (b) Four features

Figure 3.1: Forward step-wise feature selection

ℓ_2 **regularization**	
Pros	**Cons**
Fast (easy to evaluate for different λ's)	Not sparse
Stable	
Generalizes well	
Most popular regularization technique (works well in practice)	

Table 3: Pros and cons of ridge regression

3.3 Backward Feature Selection

This method is very similar to forward step-wise feature selection with one difference of eliminating one feature at a time versus adding one feature at a time. This method has an added advantage that if two features interact to produce good performance then this method has a better chance of identifying those interactions when compared to forward step-wise feature selection.

3.4 Feature Selection Baked into Models via Regularization

The approach of these methods is too keep all features in the dataset but make their respective coefficients small thereby minimizing their contribution to the predicted outputs. We first discuss ridge regression.

ℓ_2 **Regularization:** Add a term to the loss function that penalizes large coefficients.

$$\hat{\beta}^{ridge} = \arg\min_{\beta} \sum_{i=1}^{N} (y_i - \beta x_i)^2 + \lambda \sum_{i=1}^{p} \beta_i^2 \tag{3.1}$$

where, N is the number of data examples, p is the number of features and λ is the penalization parameter. As λ increases β's decrease which decreases the model capacity. Note that the summation in the second term only goes from 1 and not 0 to avoid penalizing the bias term, β_0. This is the case since we would like the intercept term to represent the mean of the outputs y. Penalizing the intercept term will cause the intercept to shift away from the mean. Additionally only some capacity (typically small) of the model lies in the intercept term hence for model selection we leave the intercept intact and penalize the other coefficients.

The first term in the above equation comes from the ordinary least squares (OLS) approach to regression. This approach has an analytical solution (closed form) as,

$$\hat{\beta}^{OLS} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \tag{3.2}$$

now adding the penalization term to OLS results in ridge regression or ℓ_2 regularization whose closed form solution is given by,

$$\hat{\beta}^{ridge} = (\mathbf{X}^T\mathbf{X} + \lambda I)^{-1}\mathbf{X}^T\mathbf{y} \tag{3.3}$$

where, I is a $p \times p$ identity matrix. Note that as λ increase the inverse grows and $\hat{\beta}^{ridge}$ shrinks hence the name shrinkage. The pros and cons of ridge regression are listed in Table 3.

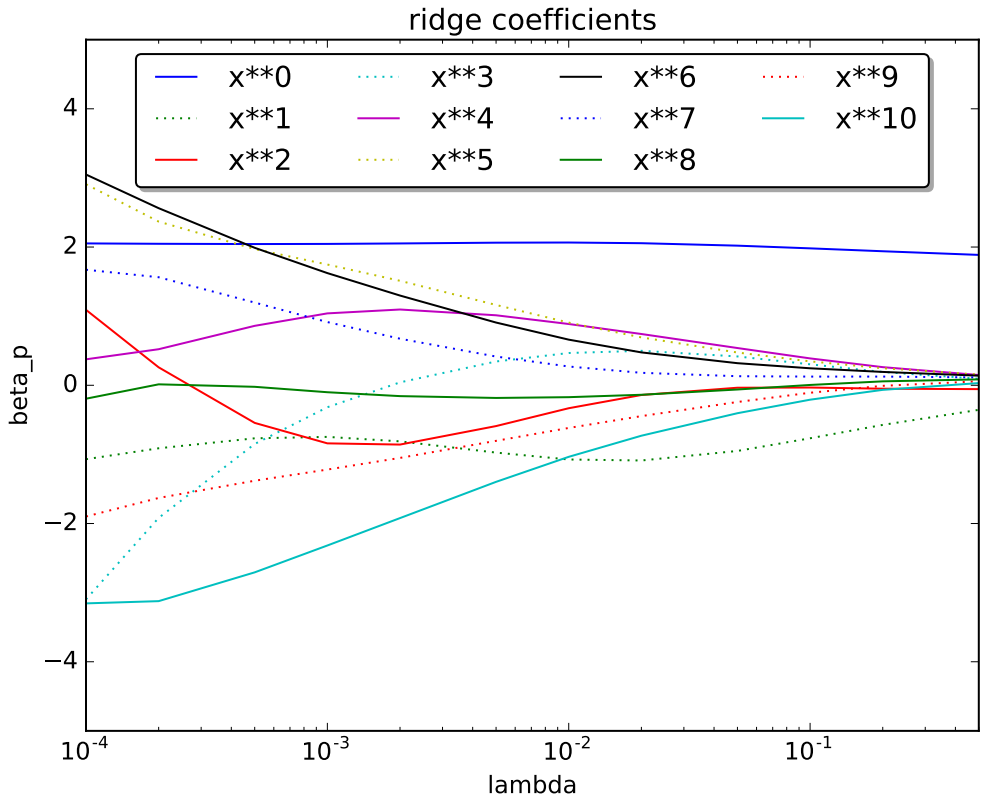


Figure 3.2: Example of ridge regression

An example of ridge regression is shown in Figure 3.2. On the x-axis is the penalty parameter, λ and on the y-axis is the magnitude of the coefficients. The eleven lines correspond to eleven coefficients. Like we observe for smaller magnitudes of λ almost all coefficients are non-zero. As λ increases most coefficients shrink and move towards zero but is not exactly equal to zero.

ℓ_1 **Regularization:** Very similar to ridge regression but with different approach to shrinkage.

$$\hat{\beta}^{lasso} = \arg\min_{\beta} \sum_{i=1}^{N} (y_i - \beta x_i)^2 + \lambda \sum_{i=1}^{p} |\beta_i| \tag{3.4}$$

where, N is the number of data examples, p is the number of features and λ is the penalization parameter.

The pros and cons of lasso regression are listed in Table 4.

An example of lasso regression is shown in Figure 3.3. On the x-axis is the penalty parameter, λ and on the y-axis is the magnitude of the coefficients. Like we observe for smaller magnitudes of λ most coefficients are non-zero. As λ increases most coefficients shrink to zero. Also notice that some coefficients shrink to zero and move back to being non-zero (black line at 10^{-2}). This phenomenon can be explained by the fact that as λ increases some coefficients driven to zero are better candidates when compared to other non-zero coefficients. Hence

ℓ_1 **regularization**	
Pros	**Cons**
Fast (easy to evaluate for different λ's)	Not generalizable
Stable	
Sparse	

Table 4: Pros and cons of lasso regression

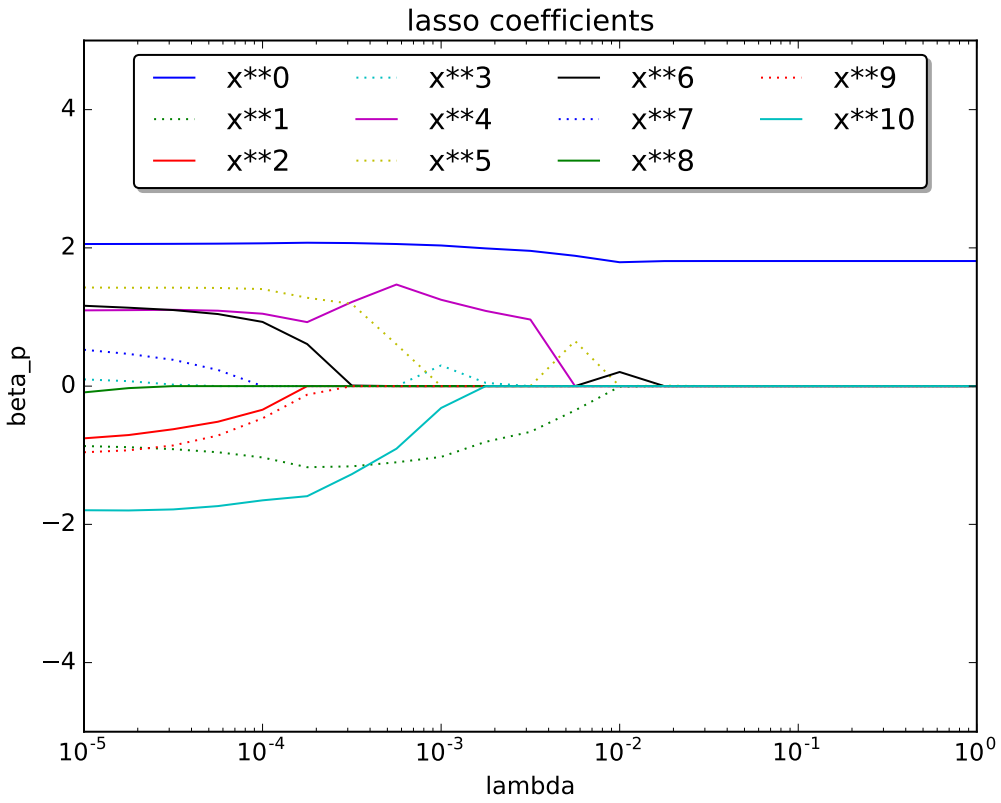


Figure 3.3: Example of lasso regression

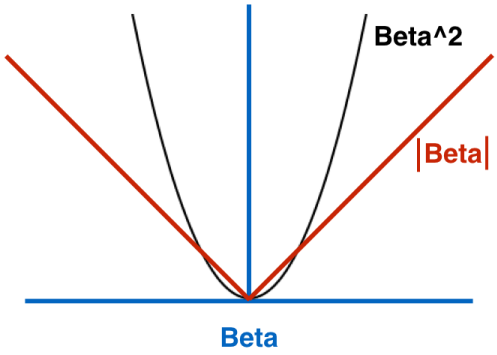


Figure 3.4: ℓ_1 ($|\beta|$) and ℓ_2 (β^2) regularization

these are moved away from zero while others are moved towards zero.

Ridge and lasso regularization can be geometrically viewed as in Figure 3.4. Consider a single coefficient β represented on the x-axis. Ridge regression places a quadratic penalty on *beta* (black line) thus penalizing large values more than small ones. Lasso places a ℓ_1 norm on the coefficient (red line) thus penalizing small values as well. A natural question to ask is can we combine both ℓ_1 and ℓ_2 regularization to get the best of both approaches. The approach to performing both ℓ_1 and ℓ_2 regularization is called elastic net whose objective function looks like,

$$\arg\min_{\beta} \sum_{i=1}^{N}(y_i - \beta x_i)^2 + \lambda \sum_{i=1}^{p} \beta_i^2 + \sigma \sum_{i=1}^{p} |\beta_i| \tag{3.5}$$

Using this objective function will result in the coefficients following the red line for small values and following the black curve for large values in Figure 3.4. There is no closed form solution to this objective function but one typically optimizes using iterative optimization.