ML Workshop

Methods

- Least squares
- Forward stepwise selection
- Ridge
- Lasso
- Regression (Decision) tree
- Bagging
- Random Forests
- Boosting

Basics

- No one method dominates all others over all possible data sets.
- Selecting the best method is the challenging part of ML.
- How close are the assumptions of a method to the data generating process?
- Mean squared error (MSE) is a common metric for accuracy of a method:

$$\textbf{MSE} = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{f}(x_i))^2$$

- MSE measures closeness of the predicted response to the true response value.
- **Training MSE**: To fit/train the model (to get $\hat{f}$).
- **Test MSE:** To test the accuracy of our model we apply our method to previously unseen data.
- We would like to select a model with the **Smallest Test MSE**.

Bias-Variance Trade-Off

- **Expected Test MSE** for a given point x_0 can be decomposed into,

$$E[y_0 - \hat{f}(x_0)]^2 = Var(\hat{f}(x_0)) + [Bias(\hat{f}(x_0))]^2 + Var(\varepsilon)$$

- We want to select a ML method with a low variance and low bias to minimize the expected Test MSE.
- **Lower bound = Var(ε)** i.e. irreducible error
- **Variance** refers to the amount by which $\hat{f}$ will change if we estimated it using a different training data set.
- A ML method with **high variance** will result in a large change in $\hat{f}$ with a small change in the training data.
- <u>More flexible</u> learning methods generally have a **high variance**.
- Do number of observations in a training data impact variance?

Bias variance Trade-Off

- **Bias** refers to the error introduced by approximating a complicated problem by a much simpler model.
- **High bias:** If the true *f* is highly non-linear, increasing any amount of training obs. will not improve the prediction with a linear model.
- <u>More flexible</u> learning methods have a **lower bias**.
- The **<u>relative rate of change of bias and variance</u>** determines whether Test MSE decreases or increases.
- As we increase flexibility of a method, bias tends to decrease faster than the variance increases.
- However, at some point increasing flexibility has little impact on bias but starts to significantly increase the variance. Hence, the **U-shaped Test MSE curve**.

Forward Stepwise Selection

- **Forward stepwise selection** begins with no predictors and then adds predictors, one-at-a-time, until all of the predictors are in the model.

Algorithm 6.2 *Forward stepwise selection*

---

1. Let $\mathcal{M}_0$ denote the *null* model, which contains no predictors.
2. For $k = 0, \ldots, p-1$:
   (a) Consider all $p-k$ models that augment the predictors in $\mathcal{M}_k$ with one additional predictor.
   (b) Choose the *best* among these $p-k$ models, and call it $\mathcal{M}_{k+1}$. Here *best* is defined as having smallest RSS or highest R^2.
3. Select a single best model from among $\mathcal{M}_0, \ldots, \mathcal{M}_p$ using cross-validated prediction error, C_p (AIC), BIC, or adjusted R^2.

Shrinkage Methods: Ridge and Lasso

- **Least squares** regression estimates $\widehat{\boldsymbol{\beta}}$ by minimizing **RSS**:

$$\mathbf{RSS} = \sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij}\right)^2$$

- **Ridge** regression estimates coefficient $\widehat{\boldsymbol{\beta}}^{R}$ that minimizes:

$$\mathbf{RSS} + \boldsymbol{\lambda} \sum_{j=1}^{p} \beta_j^2$$

- **Lasso** regression estimates coefficient $\widehat{\boldsymbol{\beta}}^{L}$ that minimize

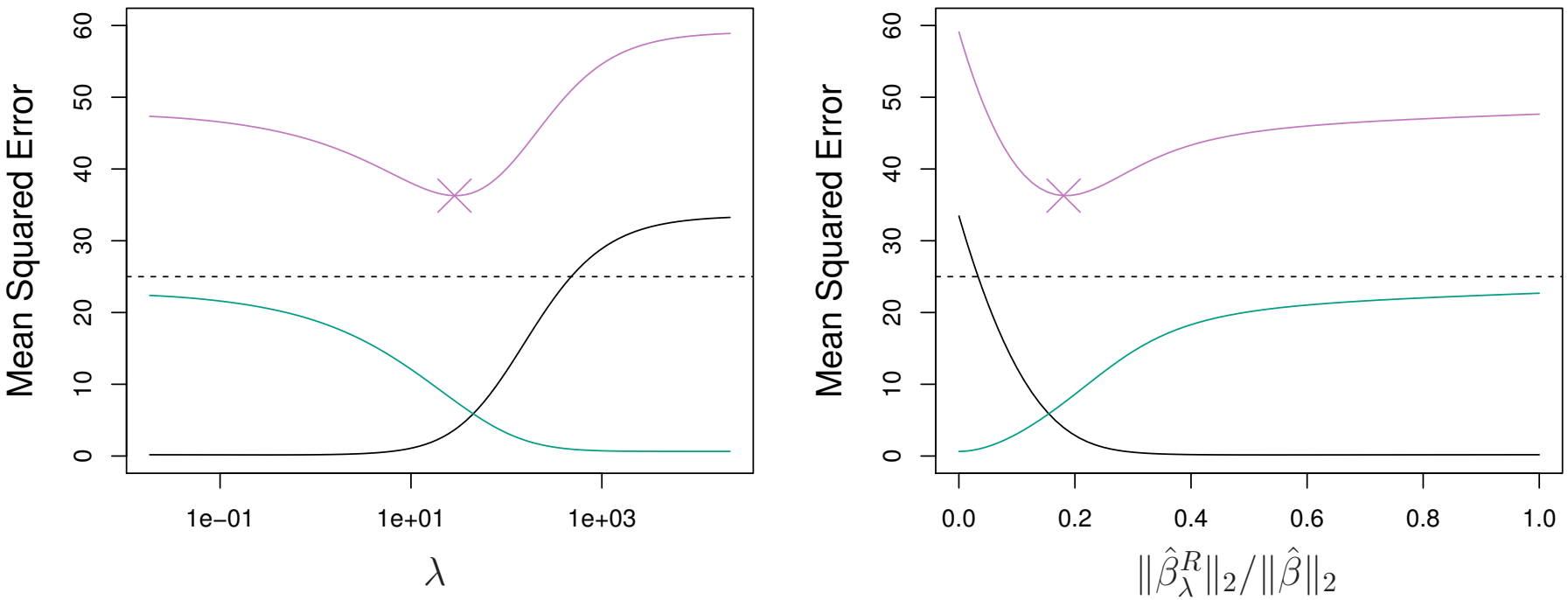
$$\mathbf{RSS} + \boldsymbol{\lambda} \sum_{j=1}^{p} |\beta_j|$$

where $\lambda \geq 0$ is a tuning parameter to be determined separately.

- $\boldsymbol{\lambda}$**=0** gives us the least squares estimates. As $\boldsymbol{\lambda} \rightarrow \infty$, coefficients approach zero.

Why does Ridge/Lasso improve over least squares?

- The answer is rooted in **bias-variance trade-off**.
- As λ increases, flexibility of the regression fit decreases (**↓Var** and **↑Bias**).
- $\boldsymbol{\lambda}$**=0** (LS variance) and as $\lambda \rightarrow \infty$ (Var → 0).



Squared bias (black), variance (green), Test MSE (purple) for ridge regression (p=45, n=50).

- **Lasso/Ridge** regression works best where the <u>LS estimates have high variance</u> i.e. a small change in data can cause large change in parameters.

Alternative formulation for Ridge and Lasso

Lasso :

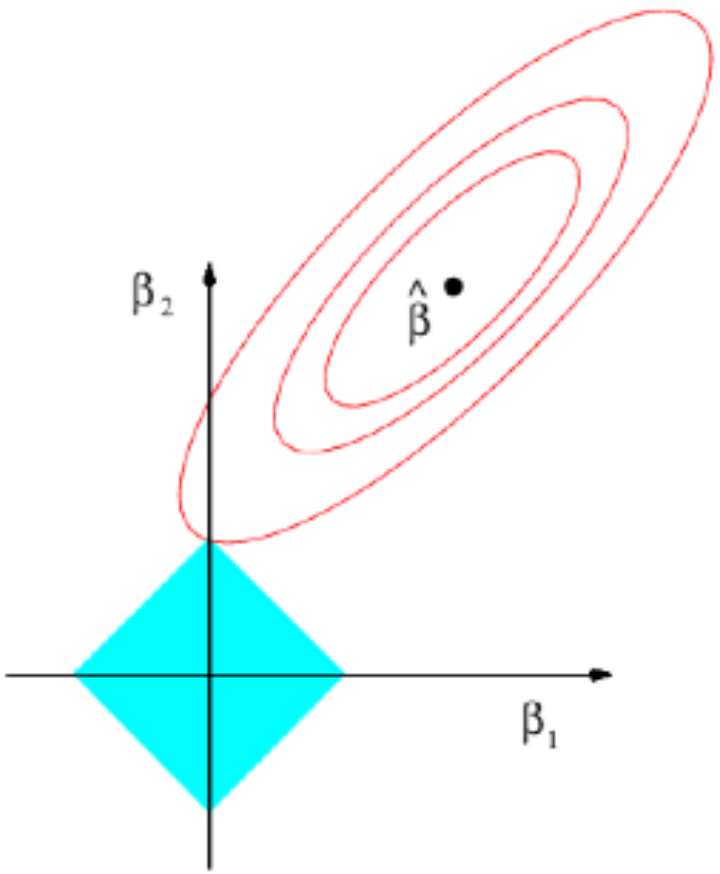
$$\underset{\beta}{\text{minimize}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right\} \quad \text{subject to} \quad \sum_{j=1}^{p} |\beta_j| \leq s$$

Ridge :

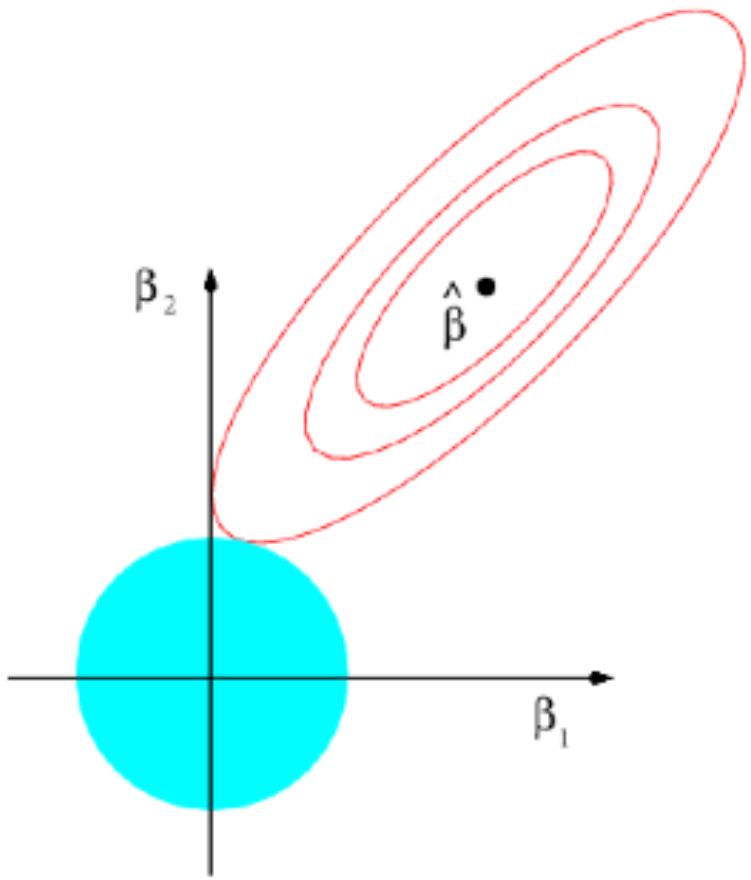
$$\underset{\beta}{\text{minimize}} \left\{ \sum_{i=1}^{n} \left(y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 \right\} \quad \text{subject to} \quad \sum_{j=1}^{p} \beta_j^2 \leq s$$

- In **Lasso** we are trying to find coefficients that minimize RSS, subject to the constraint that there is a **budget** ***s*** for how large $\sum_{j=1}^{p} |\boldsymbol{\beta}_j|$ can be.
- If *s* is large enough, that the **LS solution falls in the budget** we get LS estimates as the Lasso solution.
- If s is small, $\sum_{j=1}^{p} |\boldsymbol{\beta}_j|$ must be small so as not to violate the budget.

Variable selection in Lasso



Lasso constraint : Diamond



Ridge constraint : Circle

- **Ridge:** Since circular constraint has no sharp points, the intersection will typically not occur on axes i.e. no coefficient is set to zero.
- **Lasso** constraint has corners at each axes, so the ellipse will often intersect at one of the axes i.e. some coefficient will be set to zero.

Regression Tree

- **Regression tree (Decision tree) is** simple and useful for interpretation.
  - It **segments the predictor space** into a number of simple regions.
  - Mean of the training observations in a given region is used for prediction.

- The best split is made for each step – **without looking ahead**.

1. Select the predictor $\mathbf{X_j}$ and the cut-point **s** such that splitting the predictor space into regions $\mathbf{R_1 = \{X \mid X_j < s\}}$ and $\mathbf{R_2 = \{X \mid X_j \geq s\}}$ leads to the greatest possible reduction in **RSS**.

2. Repeat the process i.e. looking for the best predictor and its cut-point so as to minimize the **RSS** within each of the resulting regions.

3. This process continues until some **stopping criteria** is reached e.g. $\mathbf{|R_k| < 10}$.

Bagging

- **Regression trees** have high variance.
- **Bootstrap aggregation** (or **Bagging**) is a general procedure for reducing variance of any ML method.
- **Insight:** If **n** independent obs. $\mathbf{Z_1, Z_2, \ldots, Z_n}$ each has a variance σ^2 ⟹ The variance of the mean $\bar{\mathbf{Z}}$ is $\boldsymbol{\sigma^2/n}$.
- **Bagging:** Take many training sets and build separate prediction models using each training set, and average the resulting predictions.
- But we do not have many training sets – **Bootstrap** !
- We train our method on **b = 1, …, B** bootstrapped training sets in order to get $\widehat{\mathbf{f}^{*\mathbf{b}}}$, and finally average all the predictions

$$\widehat{\mathbf{f}_{\mathbf{bag}}}\,(\mathbf{x}) = \frac{1}{\mathbf{B}}\sum_{\mathbf{b}=1}^{\mathbf{B}} \widehat{\mathbf{f}^{*\mathbf{b}}}(\mathbf{x}) \qquad [\textbf{Bagging}]$$

Random Forests

- **Random forests** improve over bagged trees by <u>de-correlating</u> the trees.
- Method for building **random forests**:
  - We build a number of decision trees using bootstrapped training samples.
  - Each time a split in a tree is considered, a random sample of **m** predictors is considered from the full set of **p** predictors.
  - The split is allowed to use one of the **m** chosen predictors.
  - A fresh sample of **m** (typically, **m =** $\sqrt{\mathbf{p}}$) predictors is taken at each split.
- Why **random forests**?
  - Suppose there is one very strong predictor with other moderately strong ones.
  - In a collection of bagged trees, all of them will have this strong predictor at the top split.
  - All the bagged trees will start to look similar - high correlation among trees.
  - Averaging highly correlated quantities doesn't lead to a large decrease in variance.
  - **Random forests** force each split to consider only a subset of the predictors. Hence, de-correlating the trees

Boosting

- **Boosting** grows trees sequentially – each tree uses information from previously grown trees.
- Unlike bagging, the construction of a tree depends strongly on the trees already grown.
- Unlike making a single large decision tree (*fitting the data hard*) – boosting instead *learns slowly*.
- **Boosting** improves $\hat{f}$ in areas where it didn't perform well.
- The **shrinkage parameter** $\boldsymbol{\lambda}$ slows the process further, allowing more trees to work on the residuals. (Slow is good, but slower is better !)

Algorithm 8.2 *Boosting for Regression Trees*

1. Set $\hat{f}(x) = 0$ and $r_i = y_i$ for all i in the training set.

2. For $b = 1, 2, \ldots, B$, repeat:

   (a) Fit a tree $\hat{f}^b$ with d splits ($d+1$ terminal nodes) to the training data (X, r).

   (b) Update $\hat{f}$ by adding in a shrunken version of the new tree:

   $$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^b(x). \tag{8.10}$$

   (c) Update the residuals,

   $$r_i \leftarrow r_i - \lambda \hat{f}^b(x_i). \tag{8.11}$$

3. Output the boosted model,

$$\hat{f}(x) = \sum_{b=1}^{B} \lambda \hat{f}^b(x). \tag{8.12}$$

Boosting

- **Boosting** has three tuning parameters:
  1. **Number of trees**: Unlike bagging or random forests, boosting can overfit if **B** is too large.
  2. **Shrinkage parameter λ**: It controls the rate of learning. Typical values are 0.01 or 0.001. Very small λ can require using a large value of **B**.
  3. **Number of splits d** in each tree: It controls the complexity of the boosted trees. Often d=1 or d=2 works well. The number d is often called *interaction depth*, as it controls the interaction order of the boosted model.

Method	Test MSE	Test R^2
OLS	24.4301	0.7217
Forward Stepwise	24.4301	0.7217
Lasso	24.6004	0.7198
Ridge	24.4502	0.7215
Regression Tree	23.5125	0.7322
Bagging	14.9331	0.8299
Random Forest	12.5293	0.8573
Boosting	14.1929	0.8383

End of Workshop