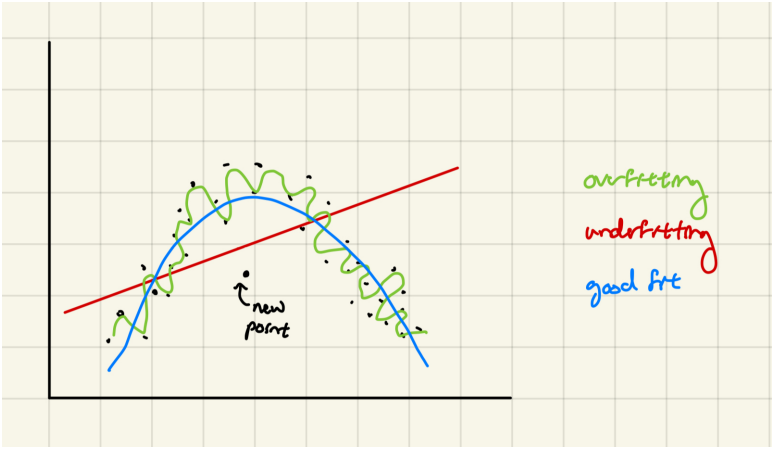
CS 181 LECTURE 2/8

Scribe Notes

Contents

1 Model Selection

How do we decide if a model is good?



Red line: something very simple, but not capturing the nonlinearity

Green line: captures the training data very well, fits every single point

Intuitively, we are looking for something between the two.

Why would we prefer blue?

There is the tension between the complexity of the model and capturing what we want to capture. We want the least complex model that captures what we want.

1.1 Fukushima Nuclear Plant Accident

This real world example demonstrates how overfitting can be detrimental. By overfitting a model that predicts the magnitudes of earthquakes, scientists believed that an earthquake of magnitude 9.0 was highly unlikely so the disaster came unexpectedly. The tsunami ensuing the earthquake disabled the power supply and cooling of three Fukushima Daiichi reactors, causing a nuclear accident.

The overfitting occurred when the model fit to a few outliers rather than following the general trend of the data.

1.2 What is fit?

We care about how well the model performs on the test data. So, the idea of "fit" is seeing how good the model predicts on data that it has not seen yet. A common way to do this evaluation is using: validation.

1.2.1 Validation

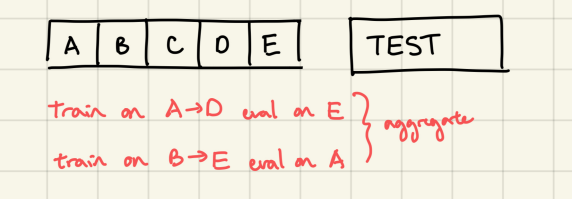
We split the data into three subsets: 1) train set to train the model, 2) validation set for comparison and selection, 3) a test set for final evaluation.

The validation set is used for choosing a model, so we need the third dataset that has been completely untouched for us to have a final evaluation.

What if we don't have a lot of data? If we have the validation and test sets, then we don't have a lot of data for model training. We can simulate the validation process when we have smaller datasets using: cross-validation.

1.2.2 Cross-Validation

You are taking random re-shuffles of your data and training on that.



You also save a part of your data for testing in the end.

Main takeaway: Cross-validation is better when you have less data. When you don't have enough data to do validation splitting the dataset into three datasets, you can implement cross-validation. However, cross-validation is more computationally expensive.

2 Bias-Variance Trade Off

Decompose the generalization error (or mean squared error) into the sum of squared bias (systematic error), variance (sensitivity of prediction), and noise (irreducible error) by following the steps below. You will find the following notation useful:

- f_D: The trained model, $f_D : \mathcal{X} \mapsto \mathbb{R}$.
- D: The data, a random variable sampled from some distribution $D \sim F^n$.
- $\mathbf{x}$: A new input.
- y: The true result of input $\mathbf{x}$. Conditioned on $\mathbf{x}$, y is a r.v. (there may be noise involved.)
- $\overline{y}$: The true conditional mean, $\overline{y} = \mathbb{E}_{y|\mathbf{x}}[y|\mathbf{x}]$.
- $\overline{f}(\mathbf{x})$: The predicted mean from the model, $\overline{f}(\mathbf{x}) = \mathbb{E}_D[f_D(\mathbf{x})]$.
- $E_D(\cdot)$ is the expectation with respect to the data (for a quantity that depends on the data, and over the data's distribution). $E_{y|\mathbf{x}}(\cdot)$ is the expectation of a quantity over the conditional distribution of y given $\mathbf{x}$.

We start with $\mathbb{E}_{D,y|\mathbf{x}}[(y - f_D(\mathbf{x}))^2]$ and use the trick of adding and subtracting the same thing to maintain equality. (We add and subtract $\overline{y}$.)

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

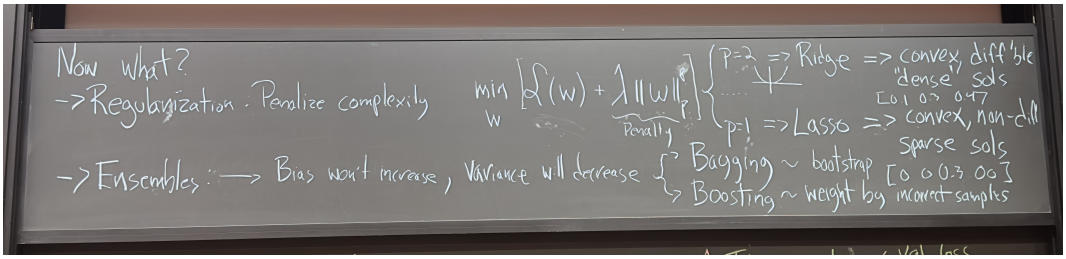
We can then further decompose the term $\mathbb{E}_D[(\overline{y} - f_D(\mathbf{x}))^2]$, by adding and subtracting $\overline{f}(\mathbf{x})$.

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

We substitute our decomposition of the second term back into the first equation and consider the expectation over $\mathbf{x}$ to find that the generalization error is:

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

3 Regularization



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

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

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

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

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

$$||\mathbf{w}||_p = \left(\sum_d |\mathbf{w}_d|^p\right)^{1/p}$$

3.1 Ridge Regularization

Set $p = 2$.

The term that we add to our loss function is the following:

$$\frac{\lambda}{2}\|w\|^2$$

Ridge regression prevents any individual weight from growing too large, providing us with solutions that are generally moderate.

3.2 LASSO Regularization

Set $p = 1$.

The term that we add to our loss function is the following:

$$\frac{\lambda}{2}\|w\|$$

Unlike ridge regression, lasso regression will drive some parameters w_i to zero if they aren't informative for our final solution. Thus, lasso regression is good if you wish to recover a sparse solution that will allow you to throw out some of your basis functions.

4 Ensembles

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

4.1 Bootstrap aggregating (Bagging)

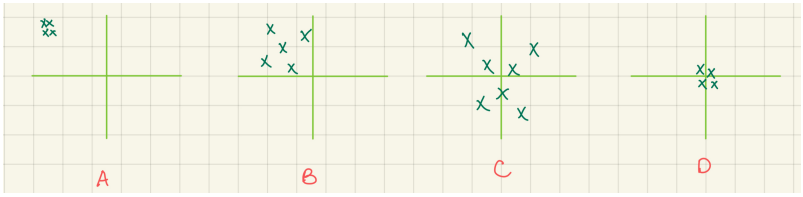
- In bagging, we fit each individual model on a random sample of the training set.
- To predict data in the test set, we either use an average of the predictions from the individual models (for regression) or take the majority vote (for classification).
- This tends to lower variance without changing bias, since it's an average of models!

4.2 Boosting

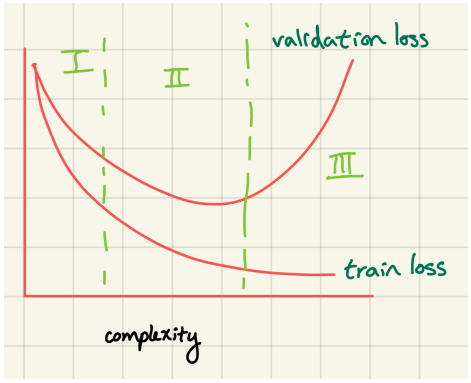
- In boosting, we train the individual models sequentially. After training the i^{th} model on a sample of the training set, we train the $(i+1)^{th}$ model on a new sample based on the performance of the i^{th} model.
- Thus, examples classified incorrectly in the previous step receive higher weights 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).

5 Concept Check

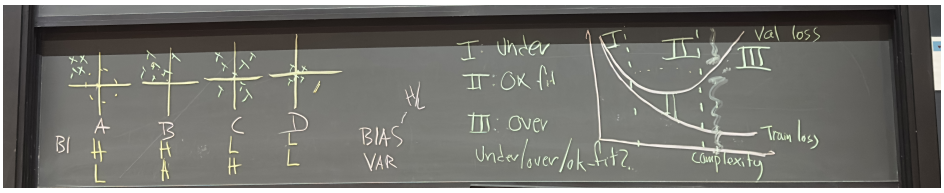
For each of the four graphs, our goal is the center of the graph and the green X's represent our prediction. Describe whether each graph has 1) high or low bias and 2) high or low variance.



Label each out of I, II, III as overfitting, underfitting, or good fit.



Solution on the next page.



Graph A is high bias, because the X's are far from the center. Graph A is low variance because the X's are all close to each other.

Graph B is high bias and high variance.

Graph C is low bias and high variance.

Graph D is low bias and low variance.

For the second question: Graphs that lie in section I. are underfitting. When the loss from the training data and the loss from the validation set are both high, then the graph is underfitting.

Graphs in III. are overfitting. A good sign that your model is overfitting is when the train loss is low but the validation set loss is high. This means that the model is overfitting to the training dataset but performs poorly on the validation set.

Graphs in II. are therefore a good fit; we want the training and validation losses to be relatively low.