Overfitting

Data Mining
Prof. Dawn Woodard
School of ORIE
Cornell University

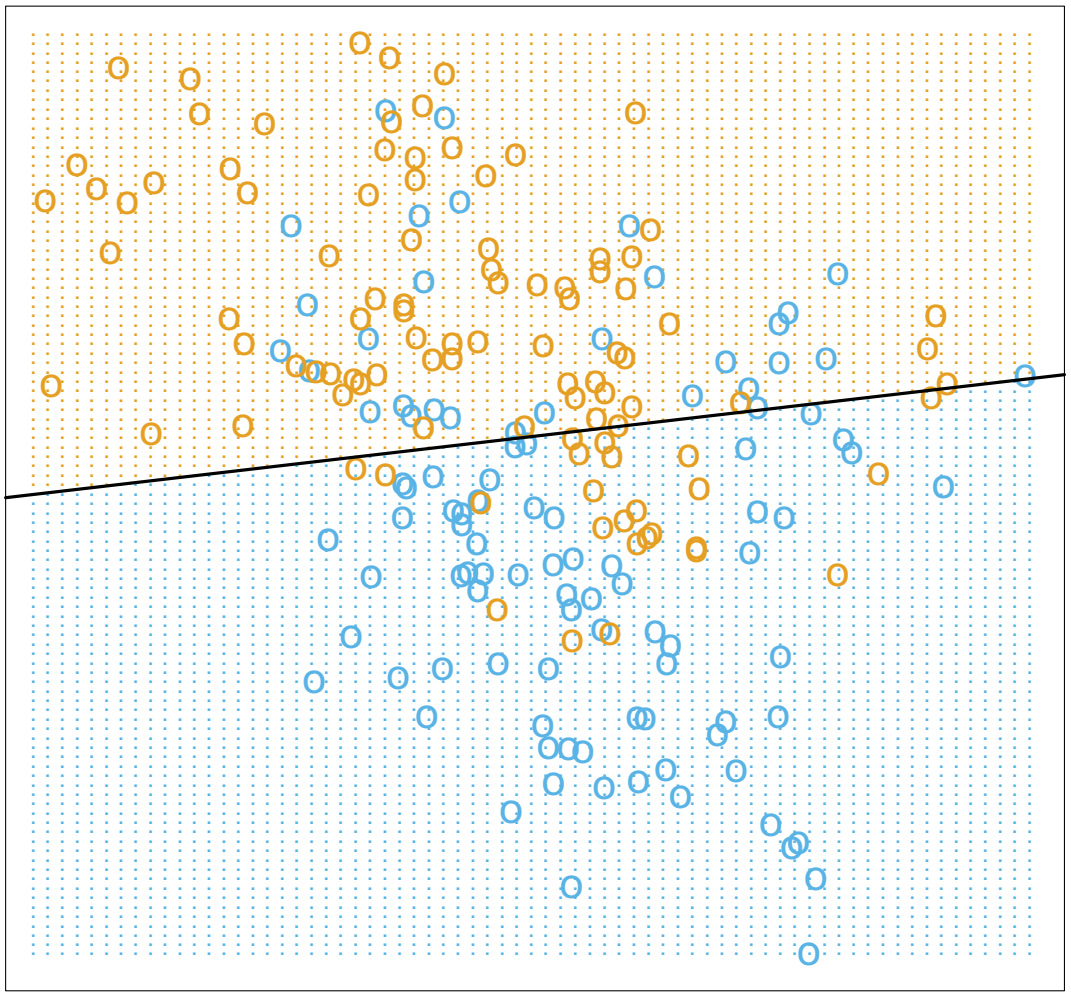
Outline

1 Overfitting

2 Cross-Validation

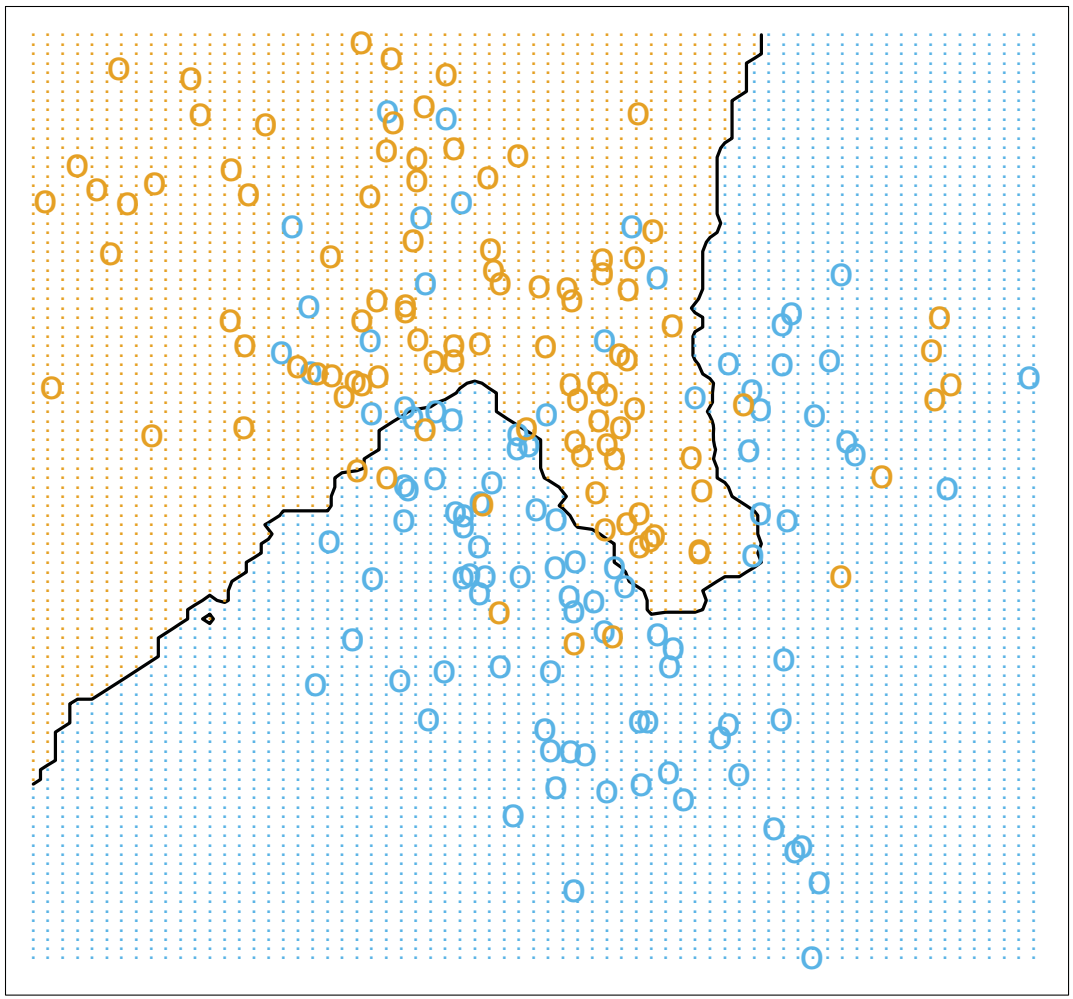
- Consider a classification problem with 2 continuous predictors and a binary outcome.
- We can create a scatterplot of the predictor values (X_1, X_2) of the training data, showing points with $Y = 1$ as blue and $Y = 0$ as orange
- If we can find a good separating boundary for blue vs. orange, this may provide a good classifier
- The following example and plots are from Chap. 2 of Hastie, Tibshirani, and Freedman (2009).

Plot the training data and draw a good linear boundary:

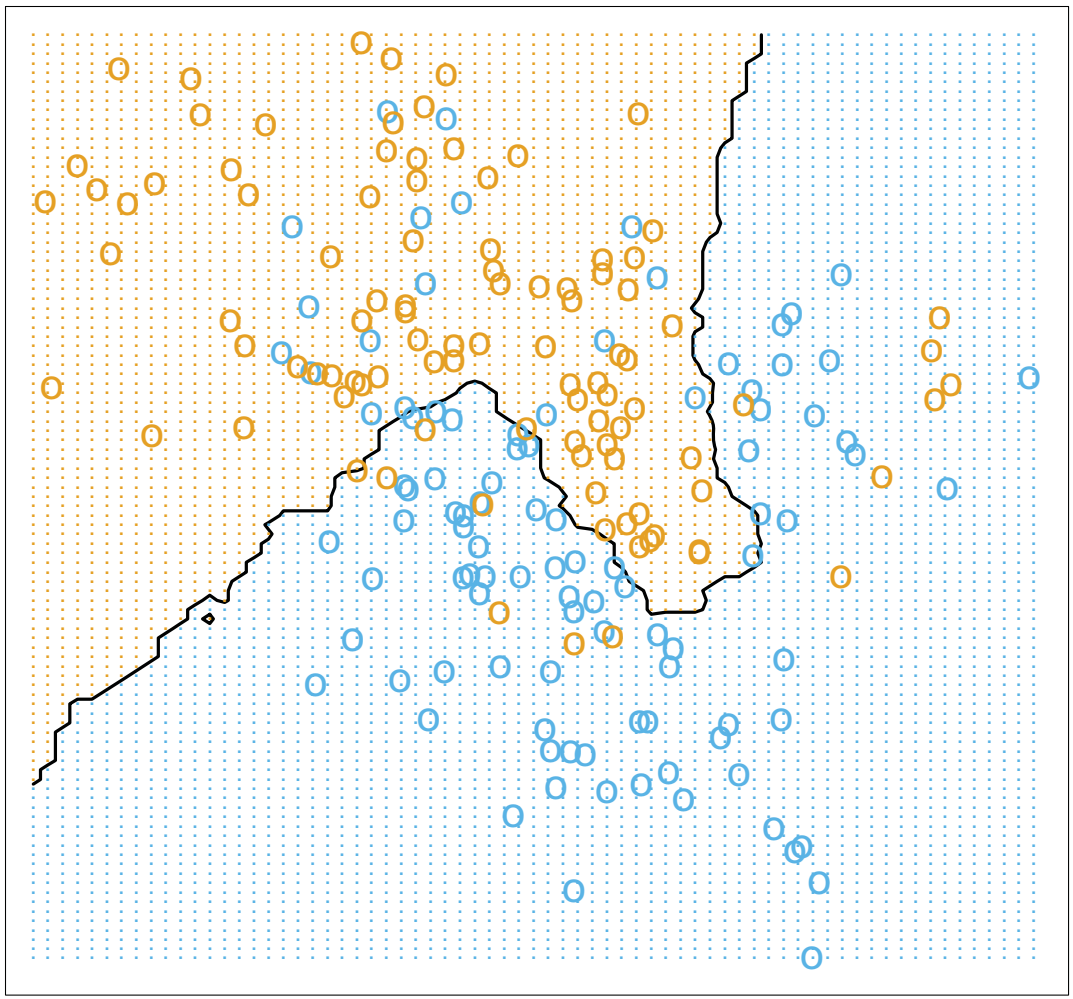


- This boundary misclassifies quite a few points
- To get a better boundary, let's try the simple *k*-Nearest Neighbors method:
  - To predict for a new (test) observation based on the training data:
  - Take the *k* observations in the training data that are closest to the new observation
  - Predict *Y* for the new observation to be the "majority vote" of these points
- This classification rule implies a classification boundary on our plot

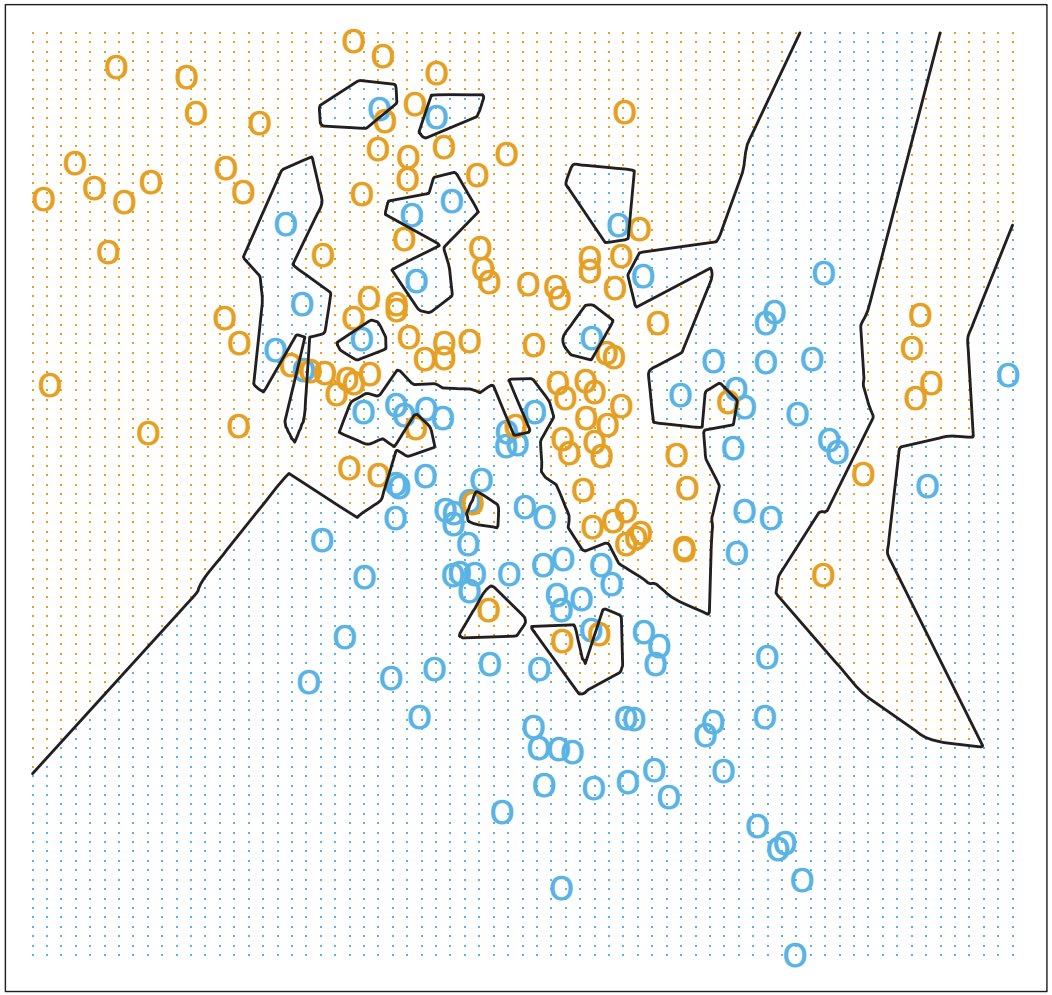
Here's the 15-Nearest Neighbor classification boundary:



Does this boundary appear to be better than the linear boundary?



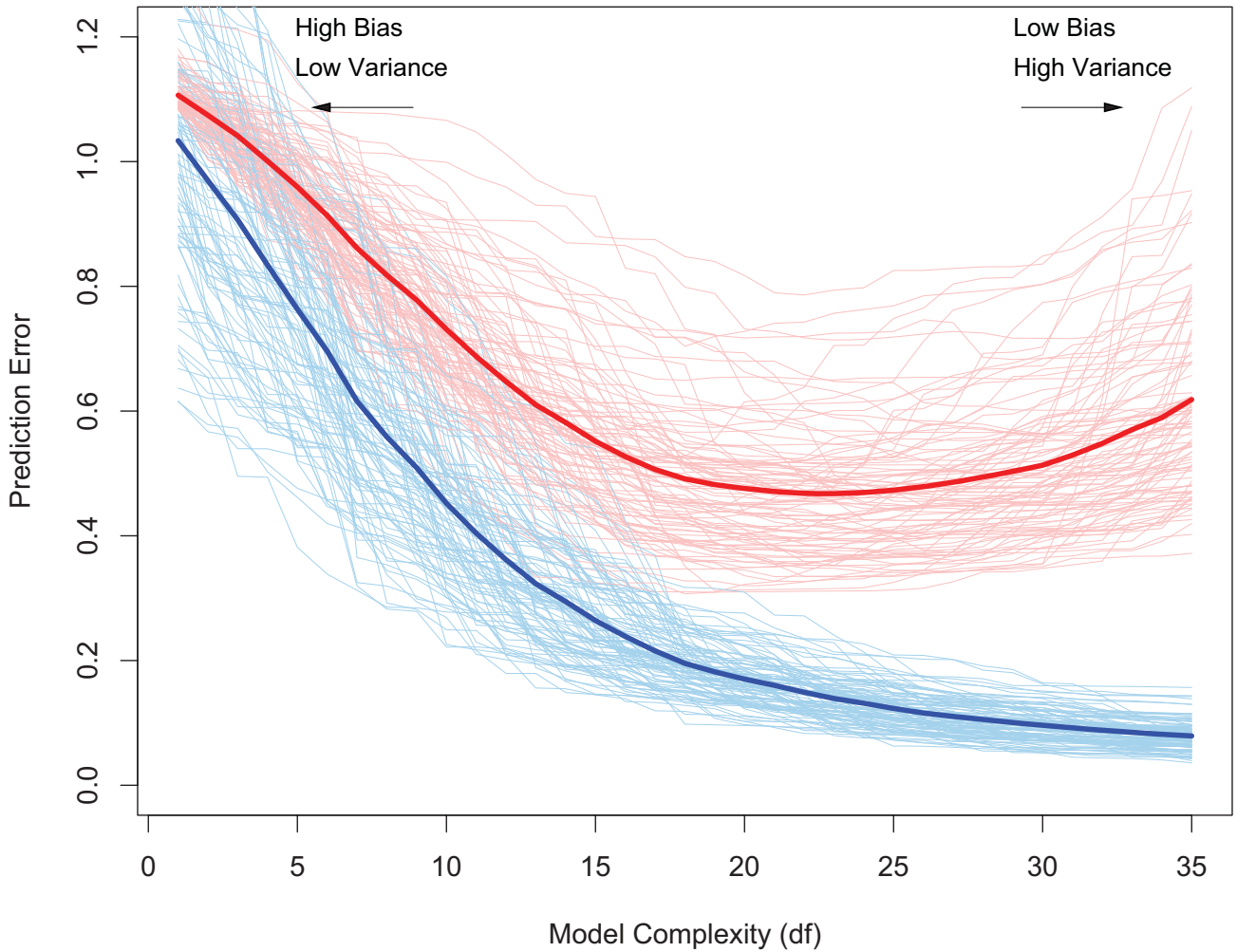
Here's the 1-Nearest Neighbor classification boundary:



- Is this an improvement over the 15-nearest neighbor boundary?
- This boundary is great for the training data, but probably has terrible accuracy on new data (such as the test data)!
- We appear to be fitting our boundary to the noise in the data as well as to the pattern in the data
- This problem is called OVERFITTING

Overfitting

Overfitting



Accuracy on training data shown in blue, on test data in red (Hastie et al.)

Model Selection

But **how do we avoid using a model that will overfit the data?**

- Consider a range of possible models with differing "complexity", e.g. for k-nearest-neighbors consider several possible values of k
- Fit all the models to the same training set, and evaluate the predictive accuracy of each model on another dataset
- Choose the model with the highest predictive accuracy. Models that overfit will have low predictive accuracy, as will models that are too simple.
- In order to do this, we will need to cut our dataset into 3 datasets. **Instead of just train and test, randomly divide data into "train," "validate," and "test" sets**

Model Selection

We can cut our data into train, validate, and test sets:

- **Training data**: for fitting each model under consideration (parameter estimation, “learning”)
  -
- **Validation data**: used to select among the models, on the basis of prediction accuracy on the validation data
  -
- **Test data**: For measuring the accuracy of the chosen model
  -

Model Selection

Procedure:

1. Train each model on the training set
2. Choose the model with the lowest error on the validation set
3. (optional) Re-train that model on the training and validation data
4. Test on the test data

Model Selection

What % of observations should be put into train, validate, and test sets?

Cross-Validation

K-Fold Cross-Validation (See Section 5.1 in JWHT text)

-
-
-
-

Cross-Validation

Step 1. Randomly select a test dataset (e.g., 25% of observations)

Step 2. Randomly divide the **remaining** data into $K > 1$ sets of roughly equal sizes
<Draw>

Cross-Validation

Step 3. For $k = 1, \ldots, K$

For each model m

- Train model m on the dataset obtained by pooling subsets $1, \ldots, k-1, k+1, \ldots, K$
- Evaluate predictive accuracy (e.g. % classified correctly) on subset k

Step 4. For each model m

- Average the accuracy over the K validation sets

Step 5. Choose the model with the highest average accuracy

Step 6. Retrain on all non-test data and report predictive accuracy on the test data

Cross-Validation

Why is this procedure OK?

Cross-Validation

How to choose K?