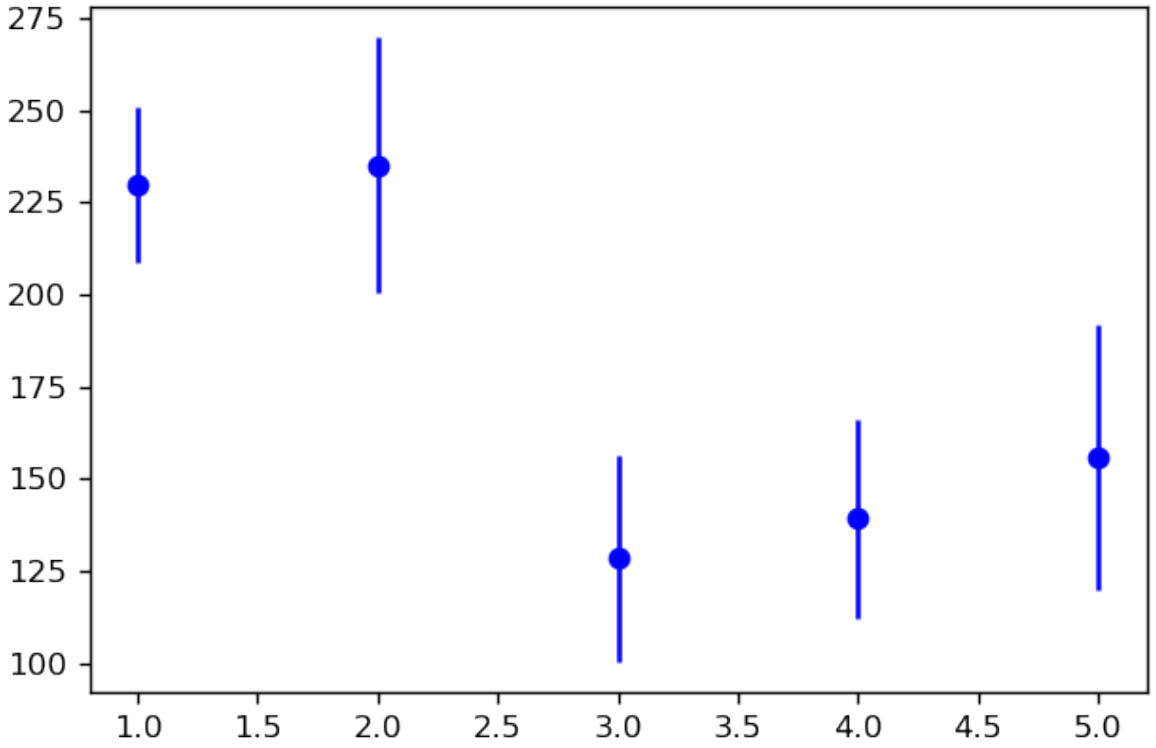
Week 06:
Test Error, Cross-Validation, and Model Selection

Week 5.8
Model selection

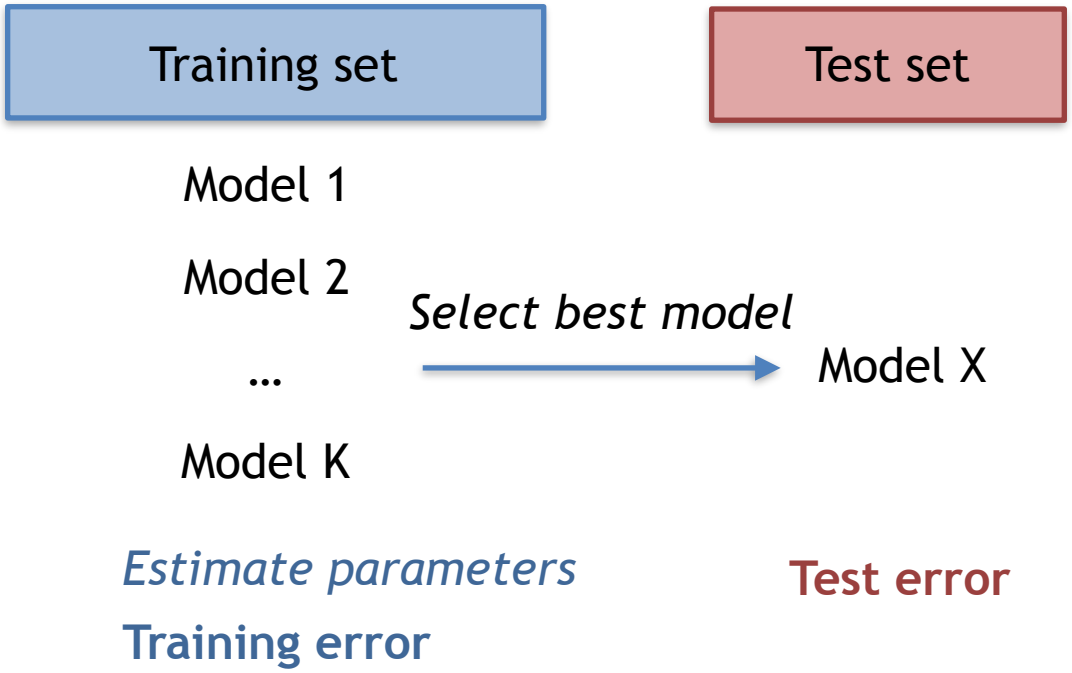
CV for Model Selection

Select “most parsimonious model whose error is no more than one standard error above the error of the best model.” (HTF, p.244)



Model selection strategies

1. Choosing "best fitting" model



This strategy is bad, as it will select the most complex model -> overfitting

Model Selection Based on Penalized Training Error

- Training error is biased downward.
- For simple models, including linear ones, we can get a less-biased estimate of generalization error by adjusting the training error upwards. These adjusted training error estimators include: AIC, BIC, and Adjusted R-squared
- If you have a limited amount of data, and you want to do model selection, you may want to use one of these instead of a validation set.
- See *HTF* 7.5-7.8.

Model selection strategies

2. Choosing model with lowest test error

The test error for the best model is underestimated.

The underestimation can be quite dramatic if you have many models.

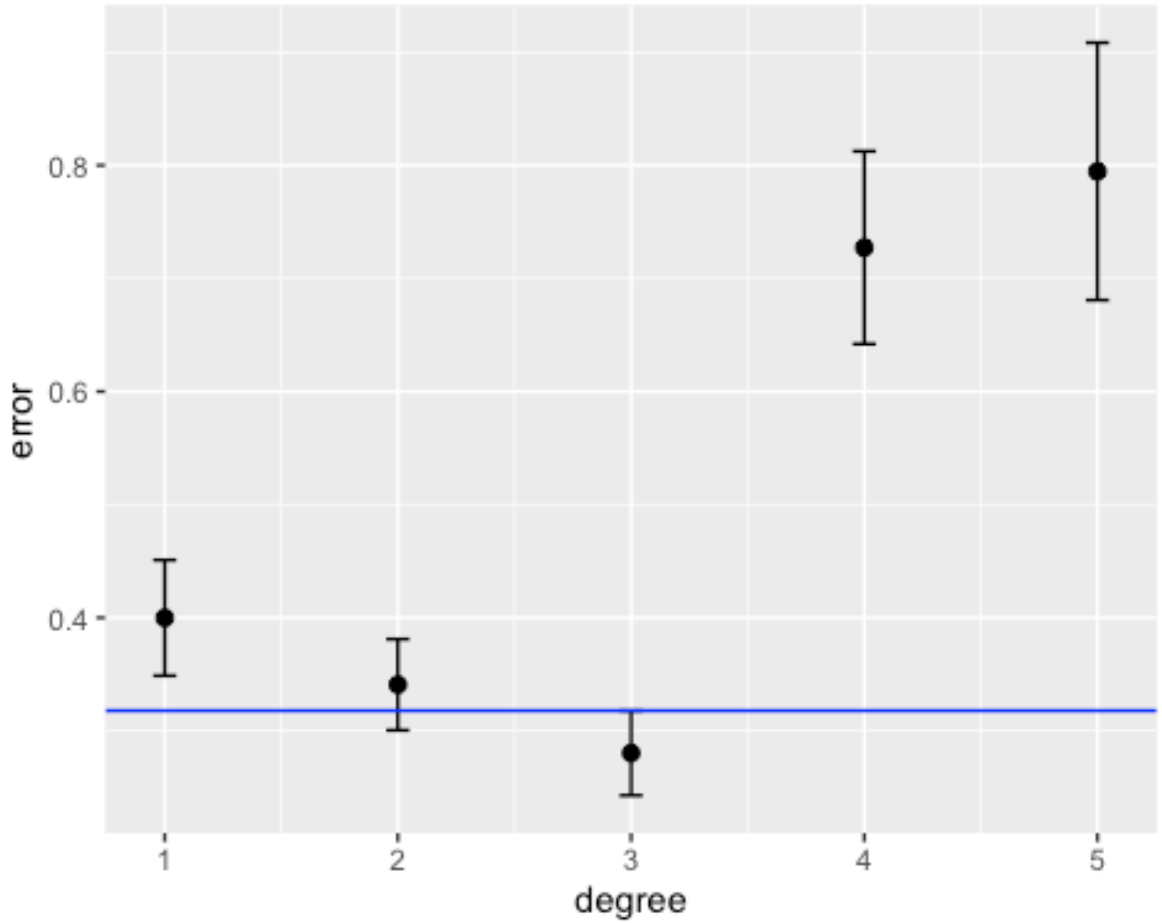
This is another form of “overfitting”

Choosing the model with the lowest test error

Experimental Scenario

- Generate new data set
- Split into training and test
- Choose best model using performance on test set (or using cross-validation)
- Report performance of best model as a predictor for it's expected test error.

Test error on models

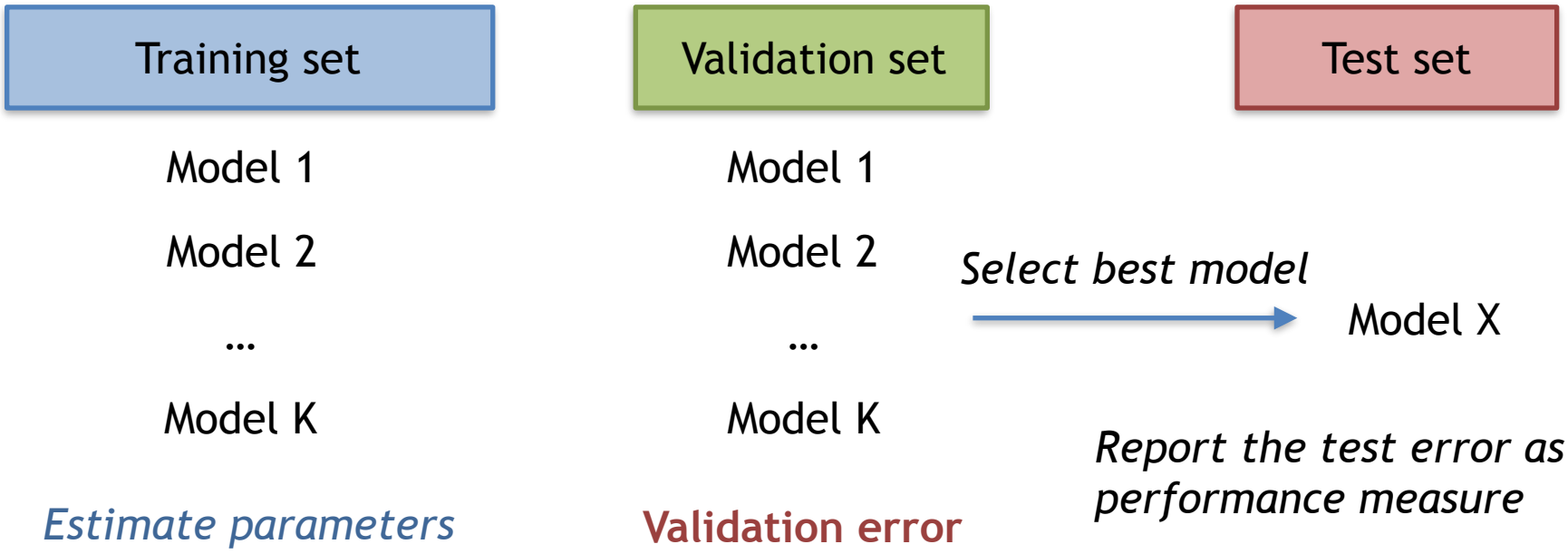


Choosing the model with the lowest test error

- Feature selection: Find the best combination of features
- If you have 100 features, you select from 2^{100} models
- Mortal sin 1: Select the best features on all the data - then calculate CV or test error on that model - and report that error as performance metric (HTF 7.10.2)
- Mortal sin 2: Select the best features on the test (or CV) error - and report that error as performance metric
- Both can lead to dramatic underestimation of prediction error

Model selection strategies

3. Doing model selection on the validation error



This is the standard strategy in machine learning.

Training, Model Selection, and Performance Evaluation

- The data are randomly partitioned into three disjoint subsets:
  - A *training set* used only to find the parameters θ
  - A *validation set* used to find the right model space (e.g., the degree of the polynomial) - you can think of this decision as another set of parameters η
  - A *test set* used to estimate the generalization error of the selected model $M(\eta, \theta)$

Week 5.9
Lab: Model Comparison and Pipelines

Transforms

Feature Expansion:

- `PolynomialFeatures`
- `ColumnTransformer`
- `Feature Union`
- …

Data Manipulation:

- `StandardScaler`
- …

Feature Selection:

- …

`.fit(X[,y])`
`.transform(X[,y])`
`.fit_transform(X[,y])`

coef:η

Estimator

- `LinearModel`
- `LogisticRegression`
- …

`.fit(X,y)`
`.predict(X)`

coef:θ

Encapsulation *of all modelling steps:*

`Pipeline`

`.fit(X,y)`
`.predict(X)`

Model Selection Summary

- The training error decreases with *the complexity (size) of the model space*
- Generalization error decreases at first, then starts increasing
- Model Selection Strategies
  - Single Validation Set
  - Cross-Validation
  - Penalized Training Error
- Evaluation: Held-out test set or (nested) CV
- If you have lots of data, just use held-out validation and test sets.