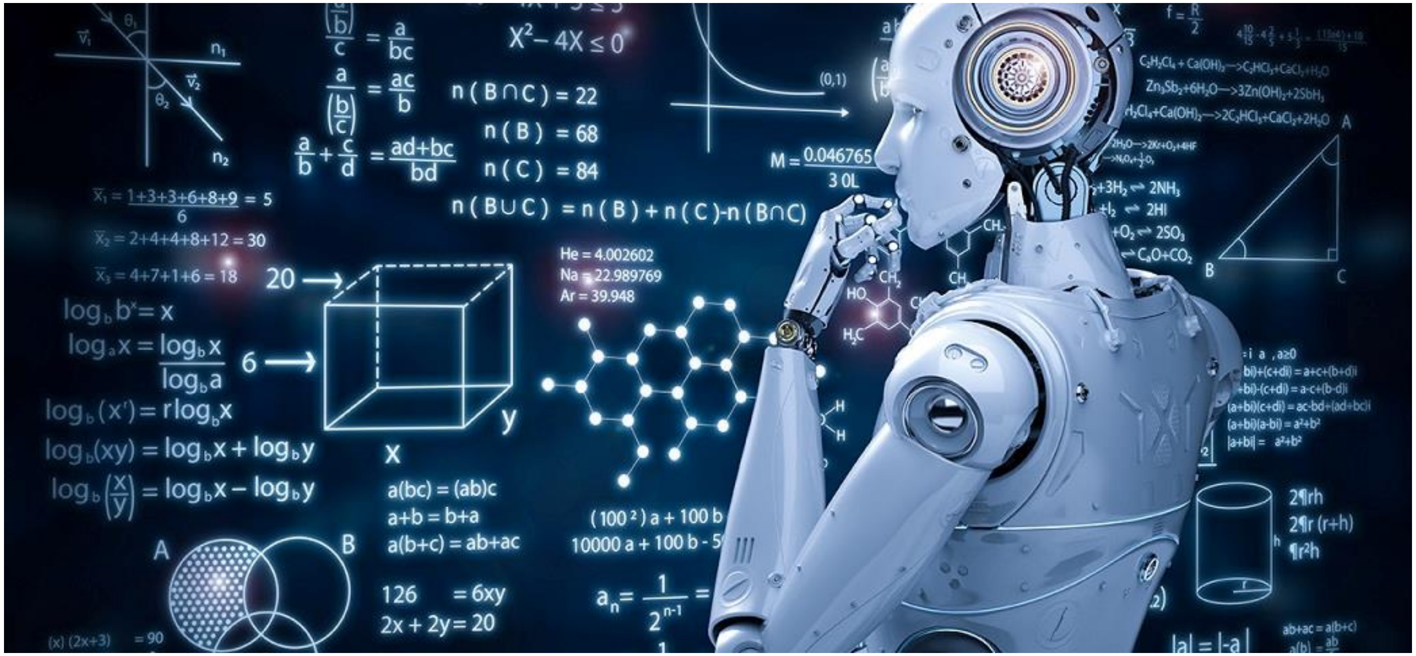


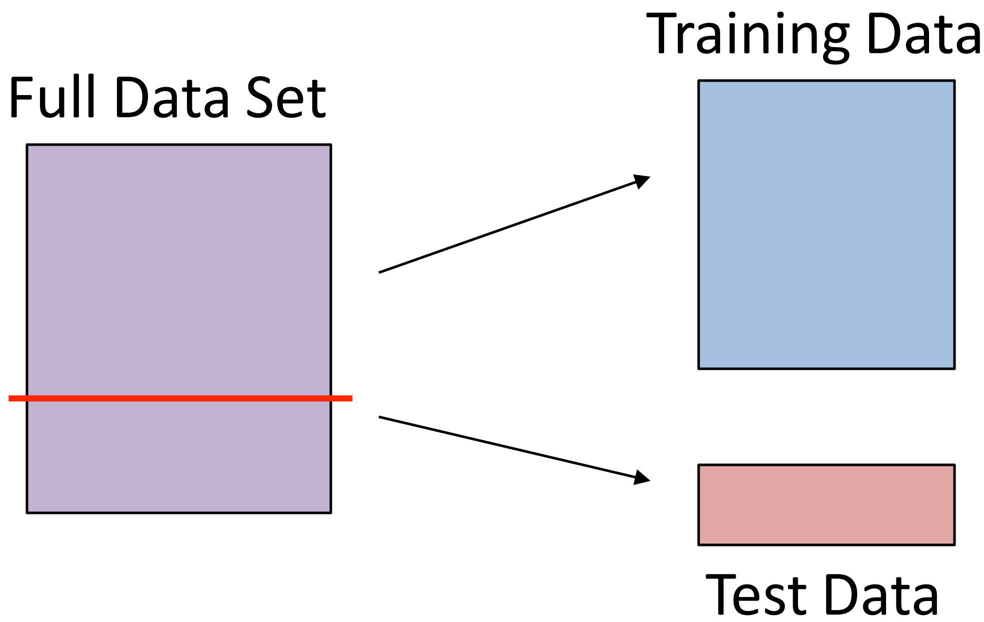
BBM406: Fundamentals of Machine Learning

Machine Learning Methodology

Reminder-Basic ML Terminology

- Example/Sample: an object, instance of the data used.
- Features: the set of attributes, often represented as a vector, associated to an example (e.g., height and weight for gender prediction).
- Labels: in classification, category associated to an object (e.g., positive or negative in binary classification); in regression real value.
- Training data: data used for training learning algorithm (often labeled data).
- Test data: data used for testing learning algorithm (unlabeled data).

Training and Test Data



General Approach:

- Train each model on the "training data"...
- ...and then test each model's accuracy on the test data

Classification Metrics

$$\text{accuracy} = \frac{\#\ \text{correct predictions}}{\#\ \text{test instances}}$$

$$\text{error} = 1 - \text{accuracy} = \frac{\#\ \text{incorrect predictions}}{\#\ \text{test instances}}$$

Confusion Matrix

- Given a dataset of P positive instances and N negative instances:

		Actual Class	
		Yes	No
Predicted Class	Yes	TP	FP
	No	FN	TN

$$P = TP + FN$$

$$N = FP + TN$$

TP (True positive) : Actual class is positive and the model predicted as positive

FN (False negative) : Actual class is positive but the model predicted as negative

TN (True negative) : Actual class is negative and the model predicted as negative

FP (False positive) : Actual class is negative but the model predicted as positive

Confusion Matrix

- Given a dataset of P positive instances and N negative instances:

		Actual Class	
		Yes	No
Predicted Class	Yes	TP	FP
	No	FN	TN

$$\text{accuracy} = \frac{TP + TN}{P + N}$$

- Imagine using classifier to identify positive cases (i.e., for information retrieval)

$$\text{precision} = \frac{TP}{TP + FP}$$

Probability that a randomly selected result is relevant

$$\text{recall} = \frac{TP}{TP + FN}$$

Probability that a randomly selected relevant document is retrieved

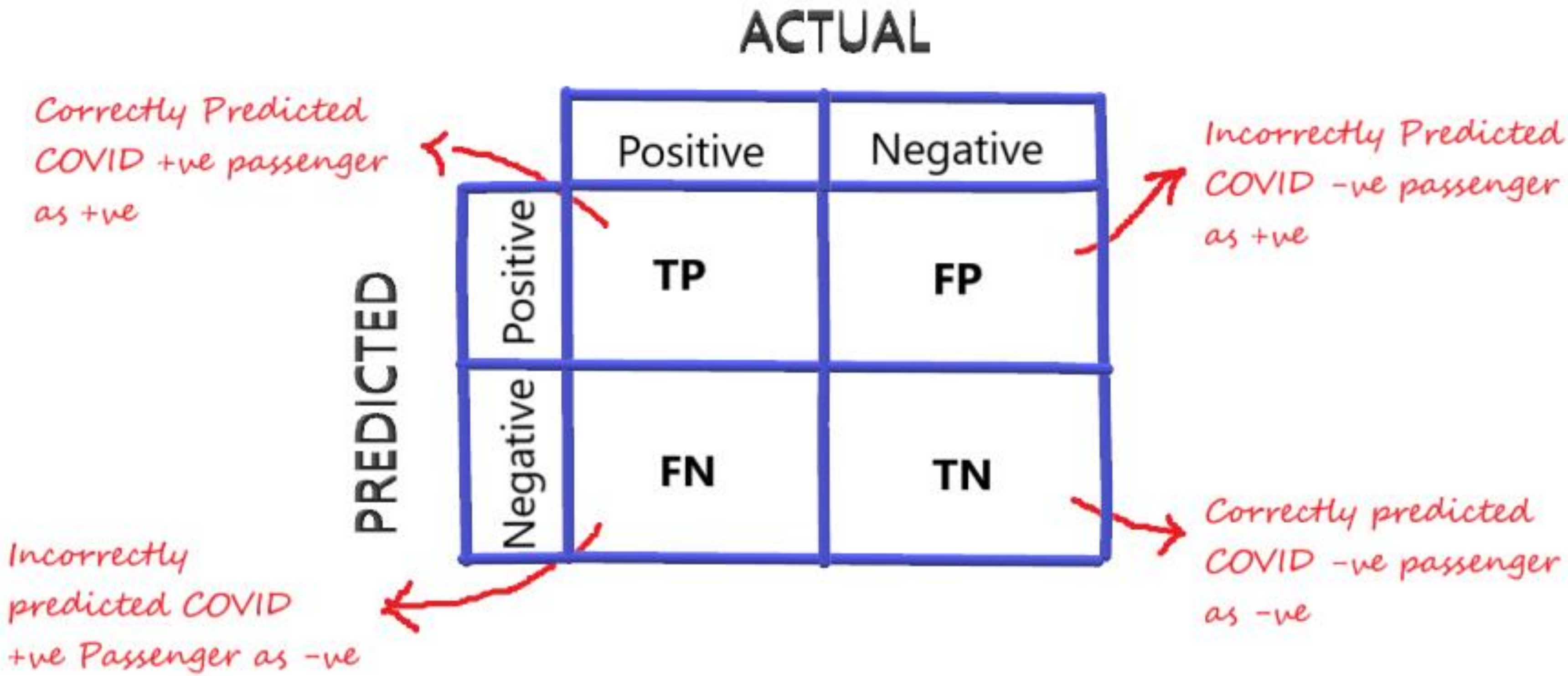
Classification Metrics - Example

- Assume that we have a machine learning model classifying passengers as COVID positive and negative.

  - **True Positive (TP):** A passenger who is classified as COVID positive and is actually positive.
  - **False Negative(FN):** A passenger who is classified as not COVID positive (negative) and is actually COVID positive.
  - **True Negative (TN):** A passenger who is classified as not COVID positive (negative) and is actually not COVID positive (negative).
  - **False Positive (FP)**: A passenger who is classified as COVID positive and is actually not COVID positive (negative).

Based on blog post of Muskan Gulati

Classification Metrics - Example



Based on blog post of Muskan Gulati

Classification Metrics – Example1

- Let’s consider 50,000 passengers travel per day on average. Out of which, 10 are actually COVID positive.
- One of the easy ways to increase accuracy is to **classify every passenger as COVID negative**. So our confusion matrix looks like:

PREDICTED \ ACTUAL	Positive	Negative
Positive	TP = 0	FP = 0
Negative	FN =10	TN 50,000 - 10 = 49,990

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$

Accuracy =49,990/50,000 = 0.9998 or **99.98**%

According to Accuracy value, this model is very good. However, it is quite bad since it classifies every passenger as negative.

Based on blog post of Muskan Gulati

Classification Metrics – Example1

- However, if we check the recall value, we will see the problem!

	ACTUAL Positive	ACTUAL Negative
PREDICTED Positive	**TP** = 0	**FP** = 0
PREDICTED Negative	**FN** =10	**TN** 50,000 - 10 = 49,990

$$\textbf{Recall} = \frac{\overbrace{\textbf{TP}}^{\text{Correctly predicted as COVID +ve}}}{\underbrace{\textbf{TP + FN}}_{\text{Total COVID +ve Passengers}}}$$

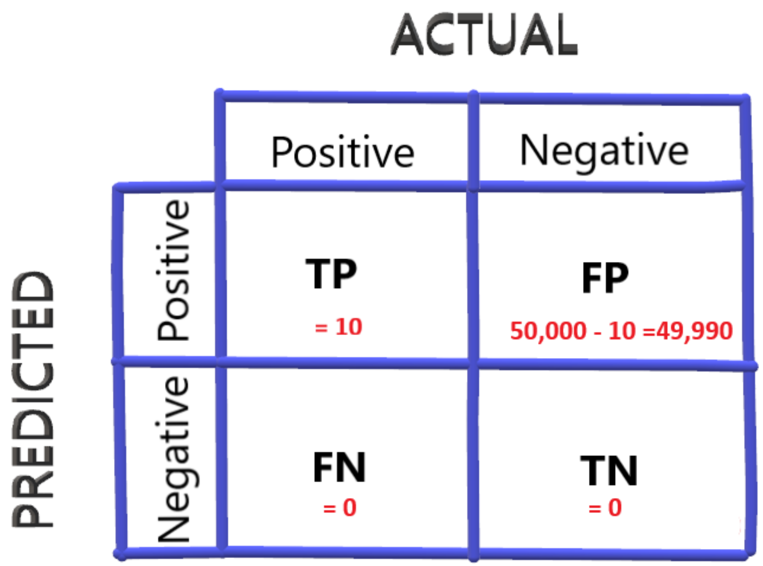
Recall=0/10 = 0

Recall value says that this model is very bad since it can not find any of the positive samples.

Based on blog post of Muskan Gulati

Classification Metrics – Example2

- We want to maximize the recall. Consider another scenario of **classifying every passenger as COVID positive**.
- Then, the confusion matrix will look like:



Recall for this case would be:

Recall = 10/(10+0) = 1

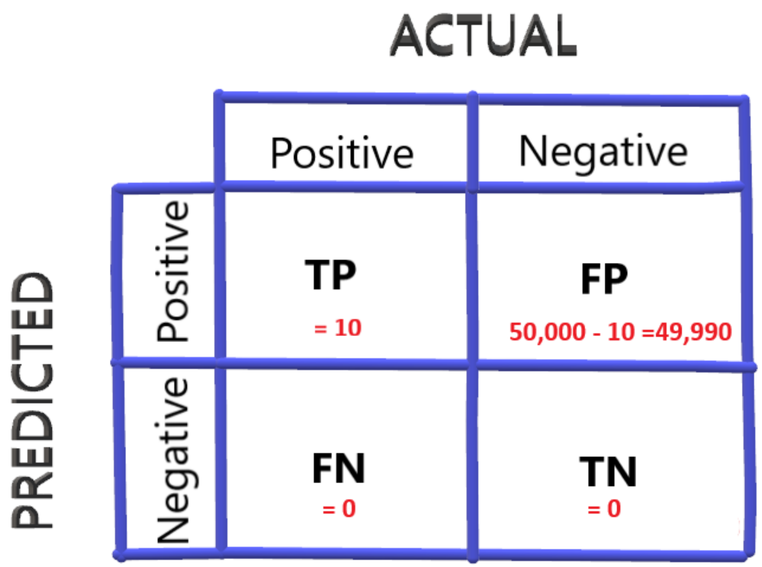
However, all passengers are classified as positive and we need to apply extra procedures (PCR tests, etc.) to identify the real situation.

This increases operation costs and trouble for passengers.

Based on blog post of Muskan Gulati

Classification Metrics – Example2

- However, if we check the Precision, we will see that classifying every passenger as COVID positive is not a good idea.



Correctly Predicted as COVID +ve

$$\text{Precision} = \frac{TP}{\underbrace{TP + FP}_{\text{Total Predicted as COVID +ve}}}$$

Although Recall = 10/(10+0) = 1,

Precision = 10 / (10 + 49990) = 0.0002

Thus, the model is bad since it has low precision value.

Based on blog post of Muskan Gulati

Classification Metrics – Example3

- Consider below confusion matrix.

		ACTUAL	
		Positive	Negative
PREDICTED	Positive	TP = 1	FP = 0
	Negative	FN = 9	TN 50,000 - 9 = 49,991

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

Precision = 1 / (1 +0) = 1
Recall = 1/(1+9) = 0.1

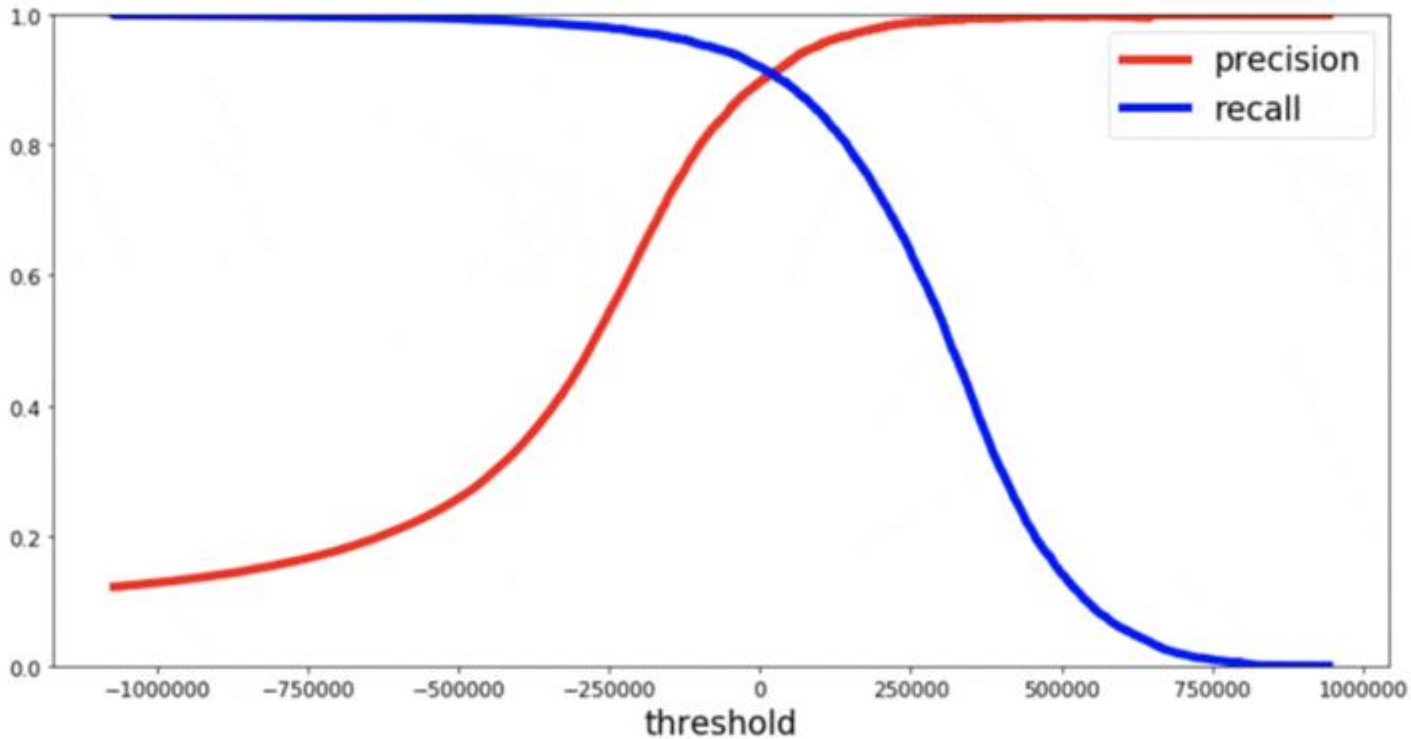
Thus, the model is bad since it has low recall value.

Precision or Recall values are not enough alone! We need to check both values.

Based on blog post of Muskan Gulati

Precision-Recall Tradeoff

For most classifiers, there is going to be a trade-off between recall and precision.



Based on blog post of Muskan Gulati

F1 Metric

- If you need to compare different models with different precision-recall value, it is often convenient to combine precision and recall into a single metric.
- F1 metric helps for that purpose

$$\text{F1 Score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

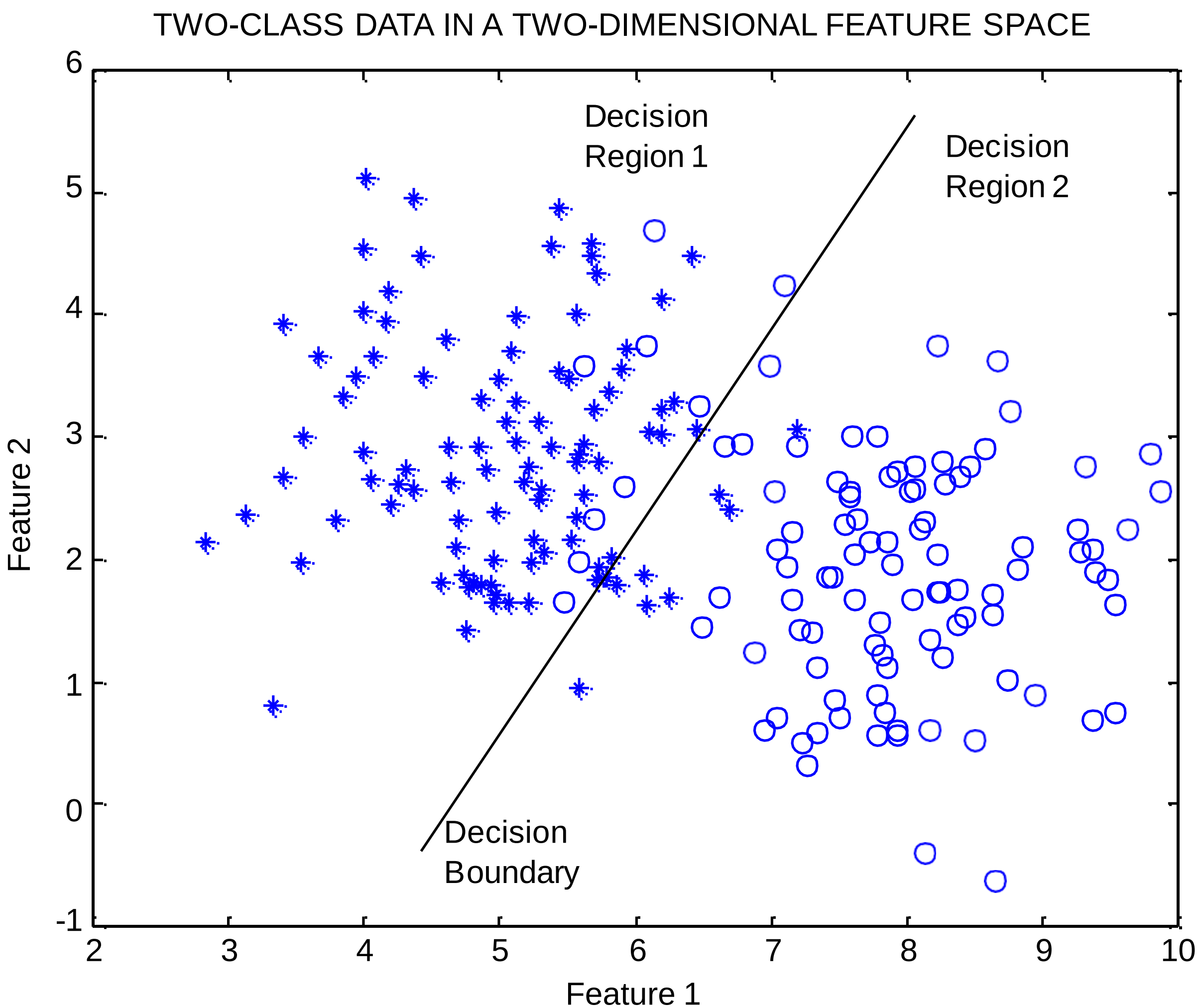
Based on blog post of Muskan Gulati

Training Data and Test Data

- Training data: data used to build the model
- Test data: new data, not used in the training process

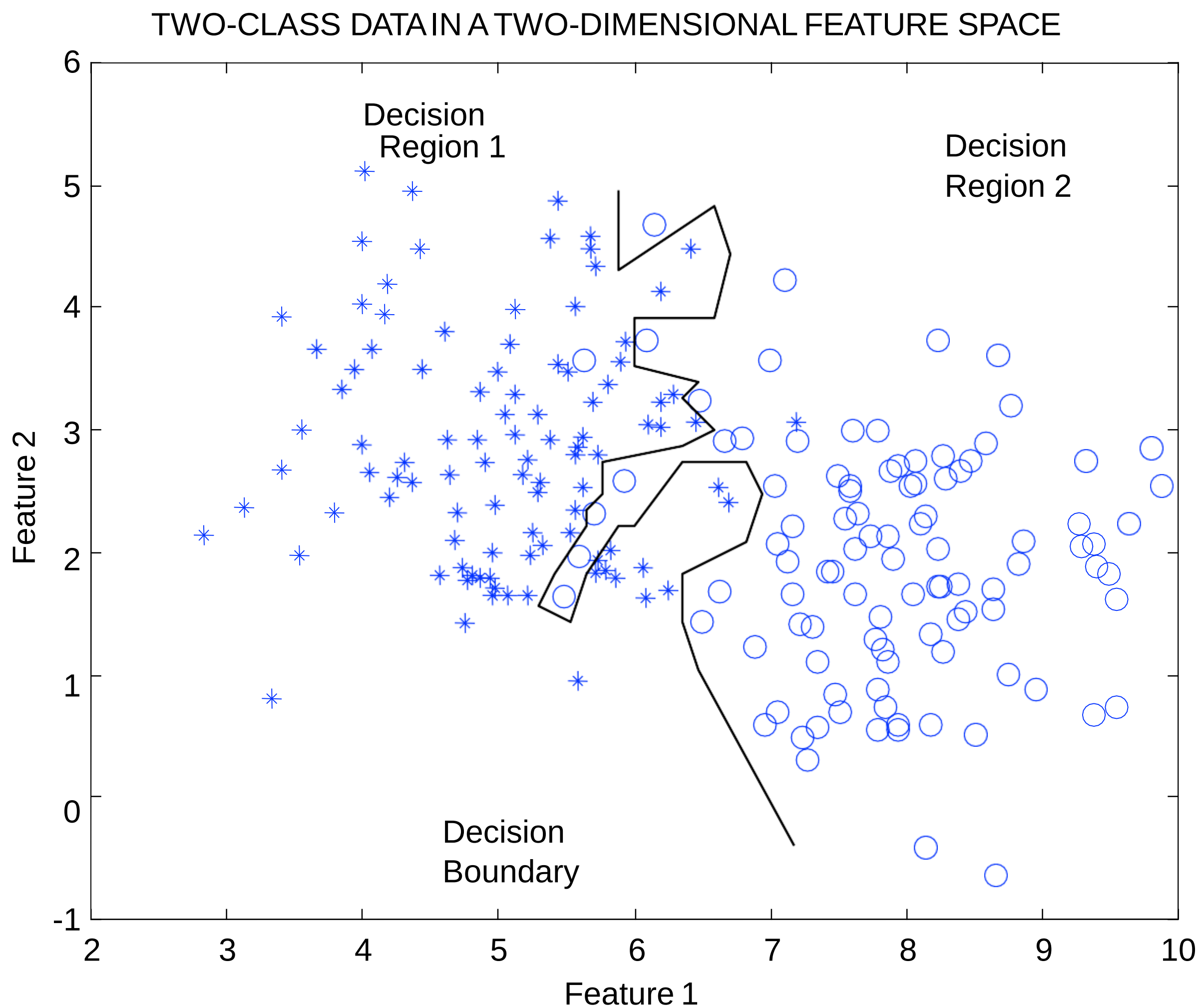
- Training performance is often a poor indicator of generalization performance
  - Generalization is what we <u>really</u> care about in ML
  - Easy to overfit the training data
  - Performance on test data is a good indicator of generalization performance
  - i.e., test accuracy is more important than training accuracy

Simple Decision Boundary



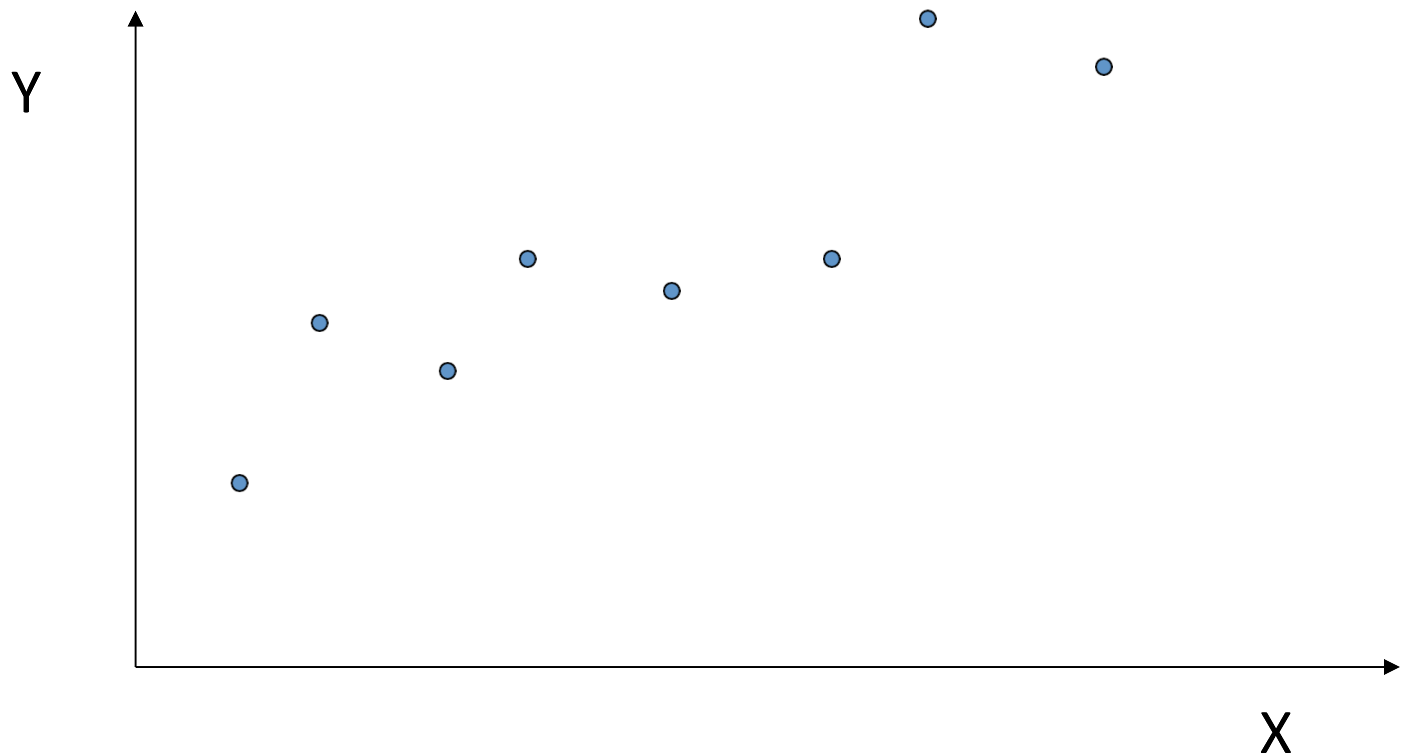
Slide by Padhraic Smyth, UCIrvine

More Complex Decision Boundary



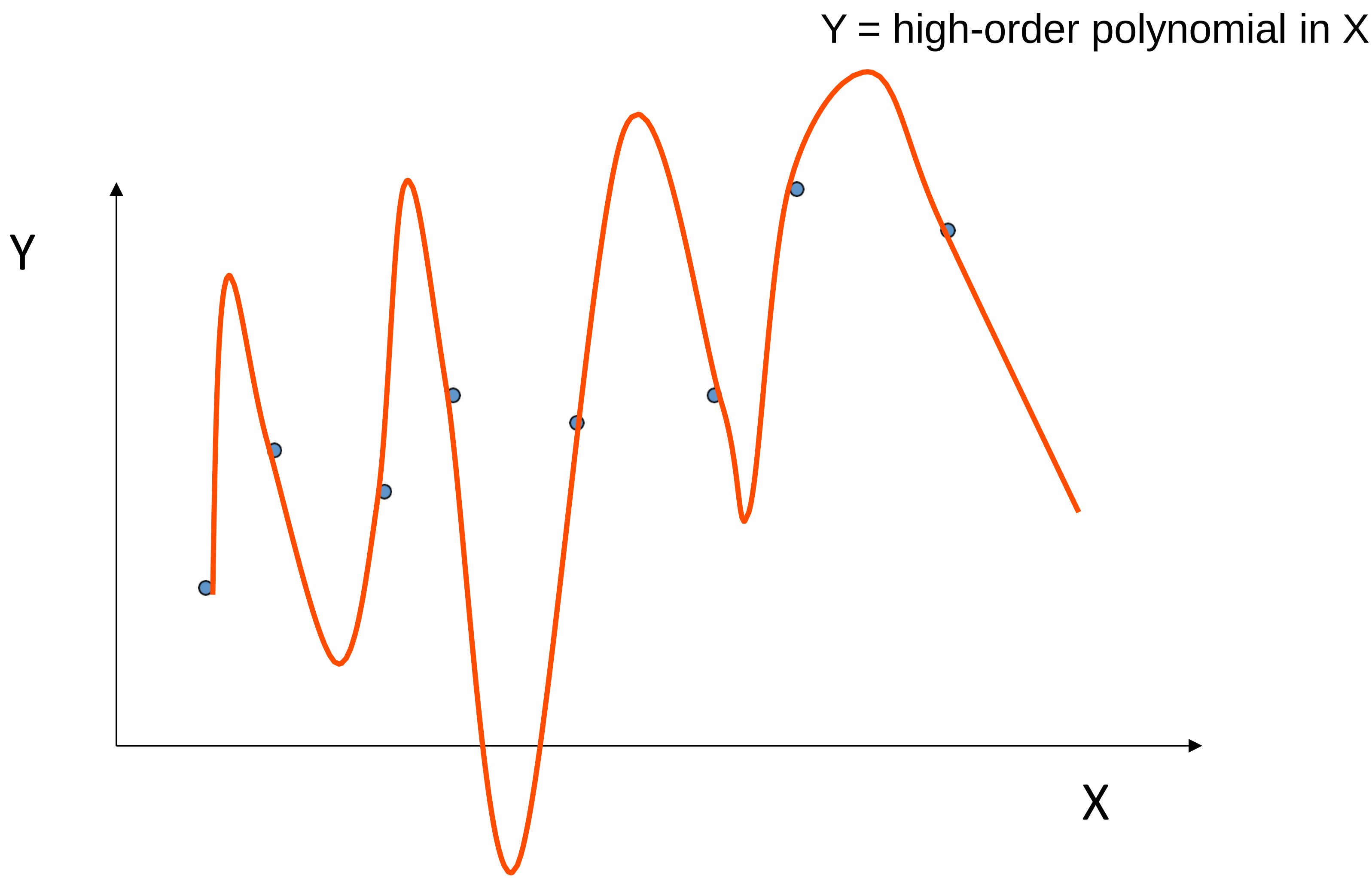
Slide by Padhraic Smyth, UCIrvine

Example: The Overfitting Phenomenon

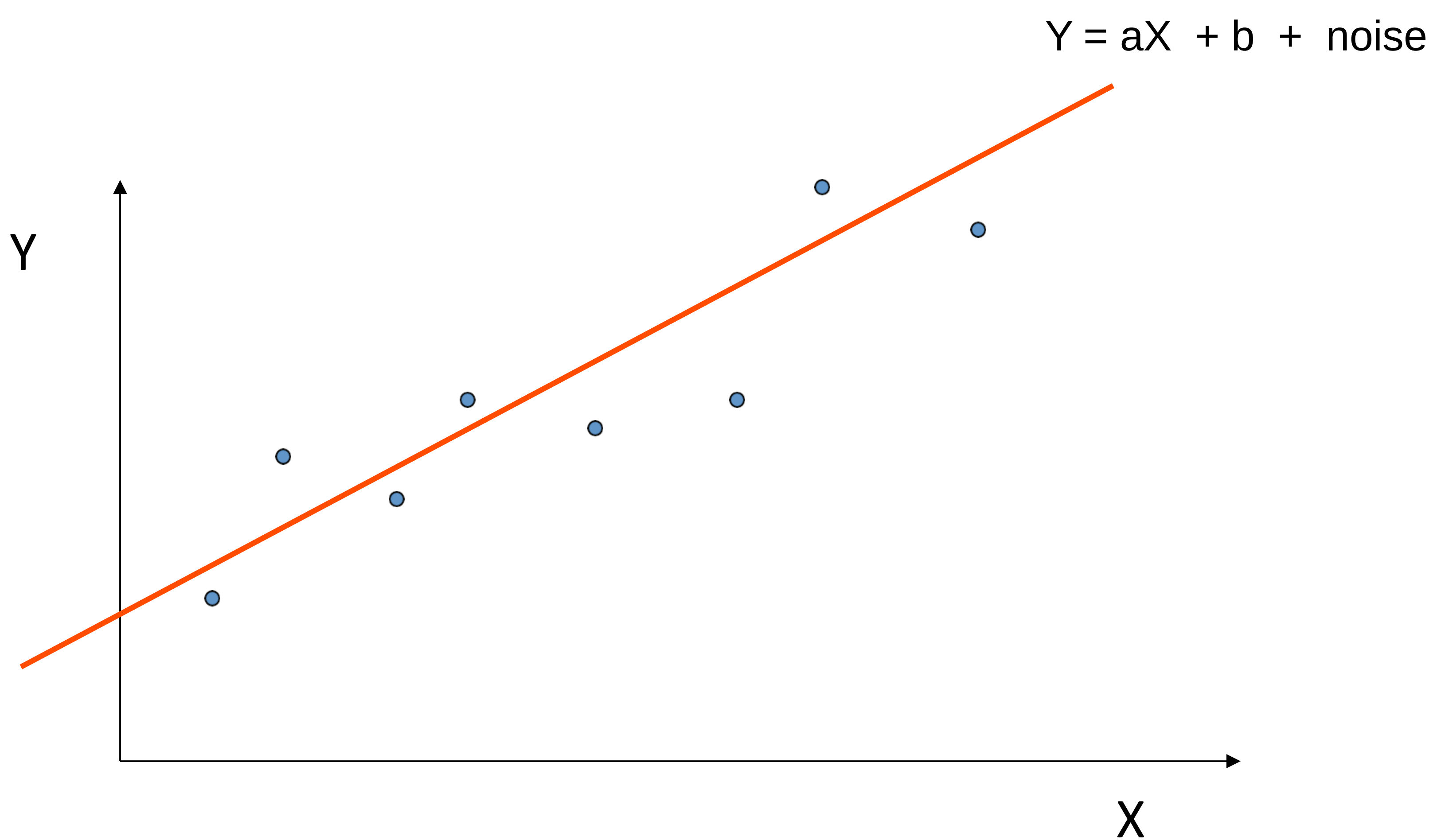


Slide by Padhraic Smyth, UCIrvine

A Complex Model

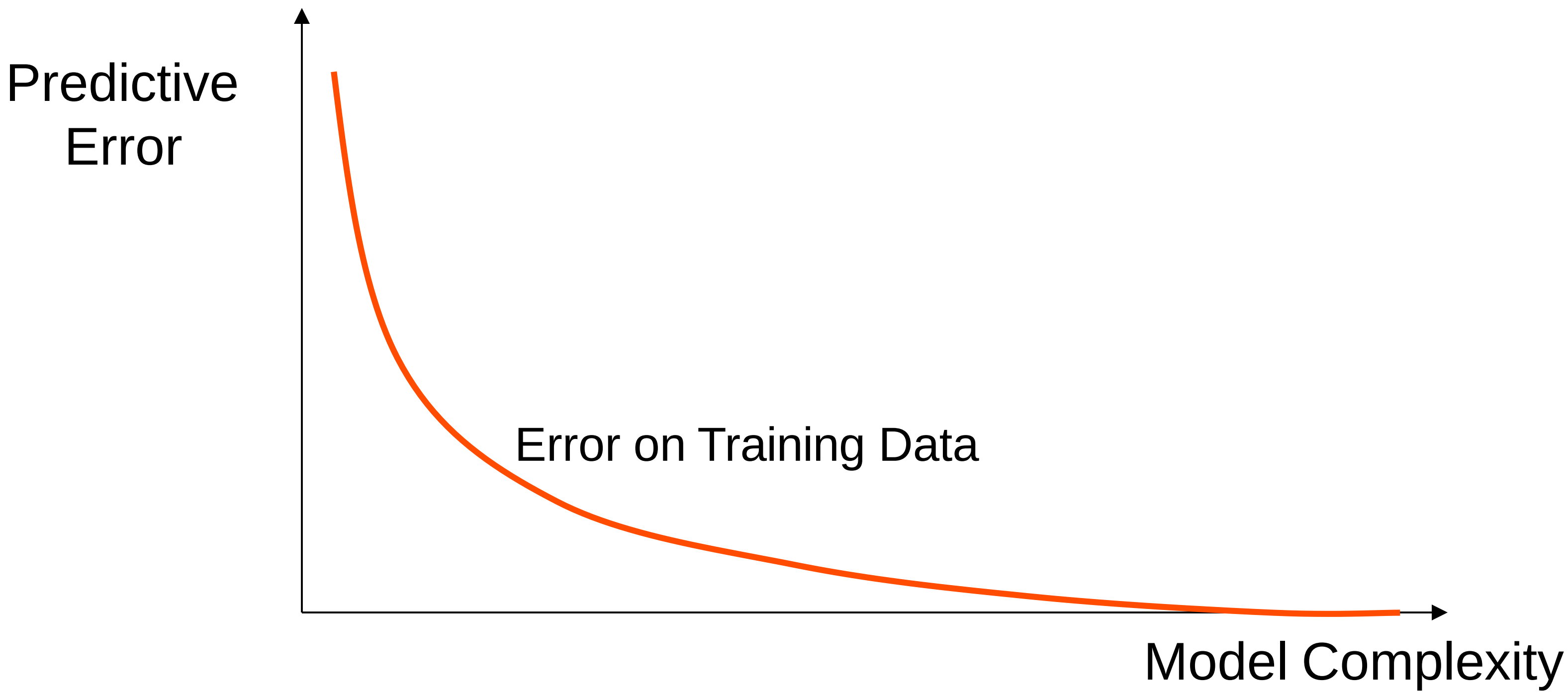


A More Simpler (better) Model



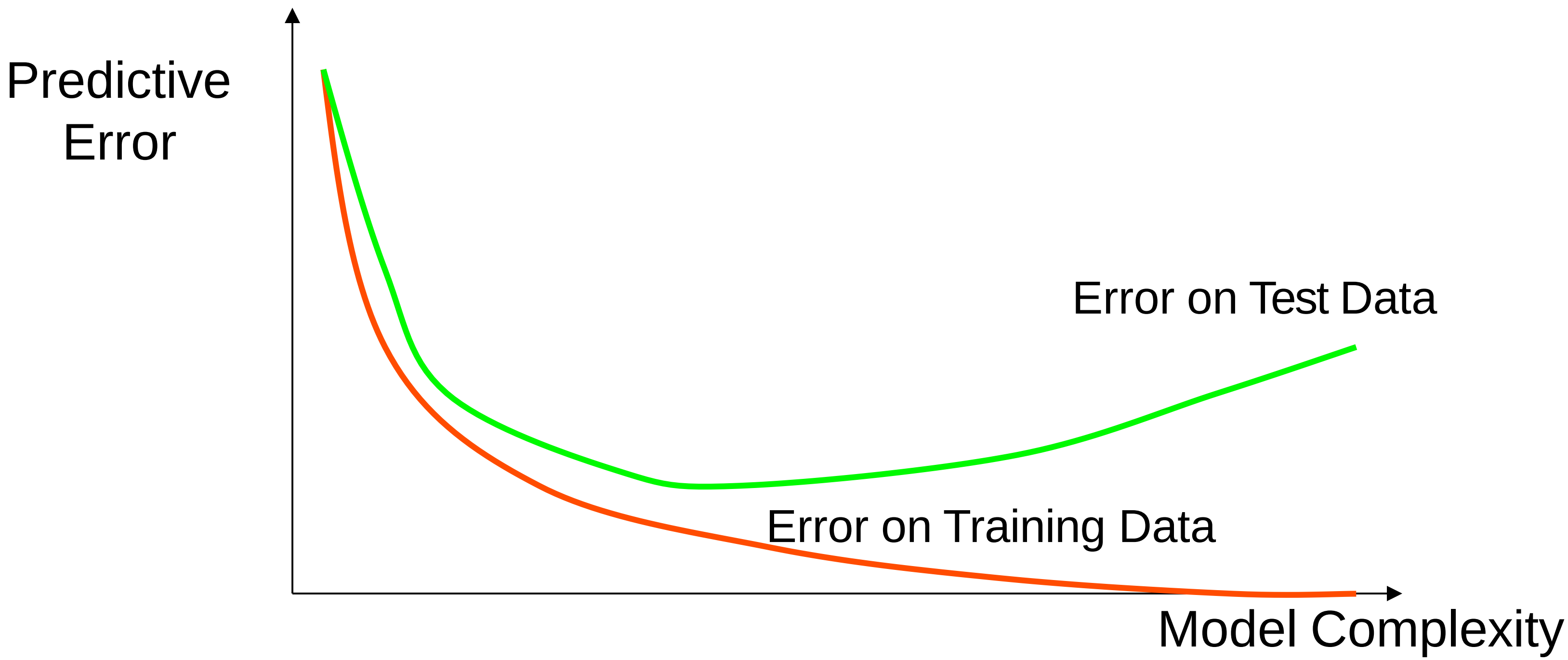
Slide by Padhraic Smyth, UCIrvine

How Overfitting Affects Prediction



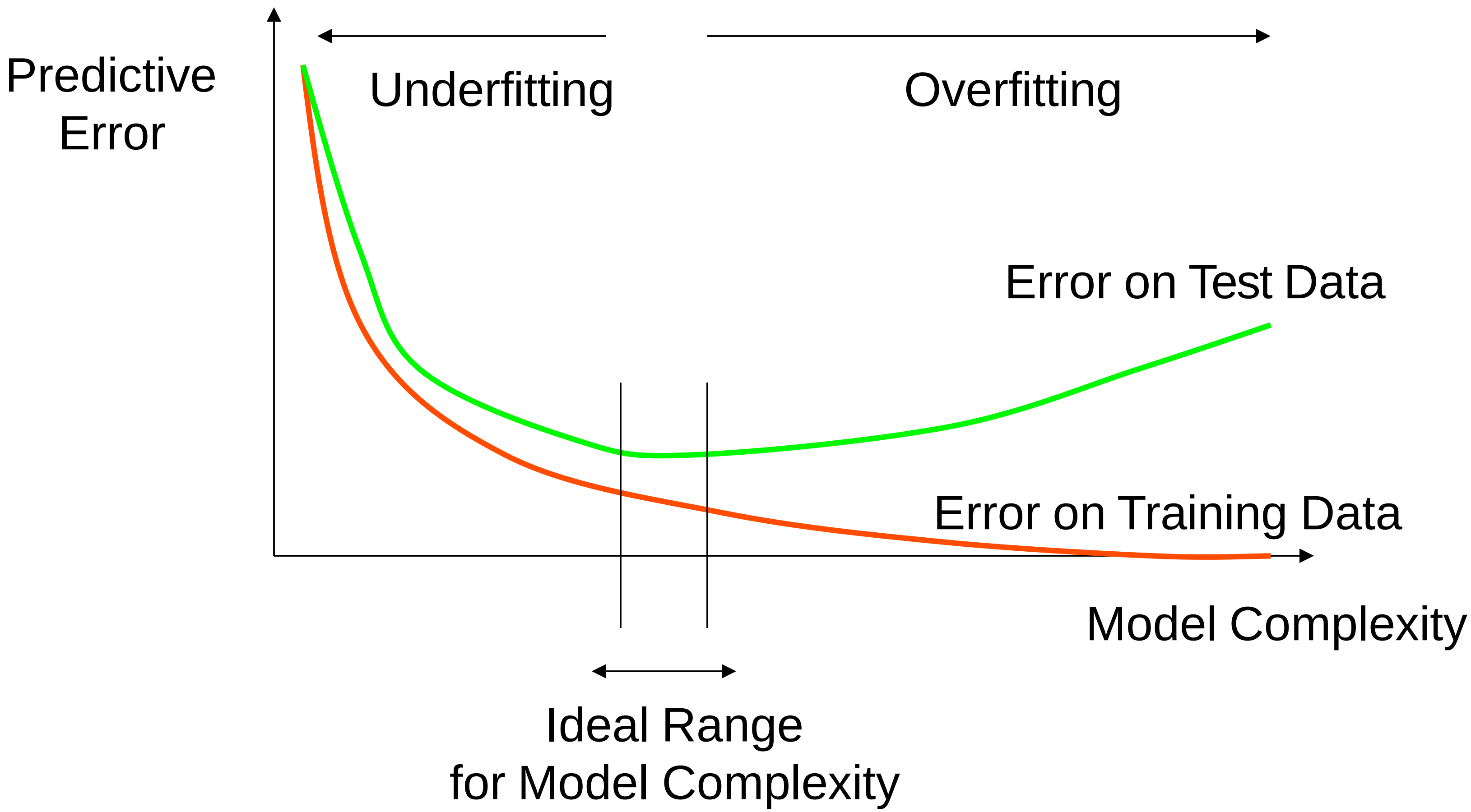
Slide by Padhraic Smyth, UCIrvine

How Overfitting Affects Prediction



Slide by Padhraic Smyth, UCIrvine

How Overfitting Affects Prediction



Slide by Padhraic Smyth, UCIrvine

Splittting Training/Test Data

- Splitting data in training/test sets
  - training data is used to fit parameters
  - test data is used to assess how classifier generalizes to new data

- But, how we can split the data?

slide by Olga Veksler

Validation data

- Same question when choosing among several classifiers
  - our polynomial degree example can be looked at as choosing among 3 classifiers (degree 1, 2, or 3)
- Solution: split the labeled data into three parts

labeled data

Training ≈60%	Validation ≈20%	Test ≈20%
train tunable parameters **w**	train other parameters, or to select classifier	use **only** to assess final performance

slide by Olga Veksler

Training/Validation

labeled data

Training ≈60%	Validation ≈20%	Test ≈20%

Training error: computed on training example

Validation error: computed on validation examples

Test error: computed on test examples

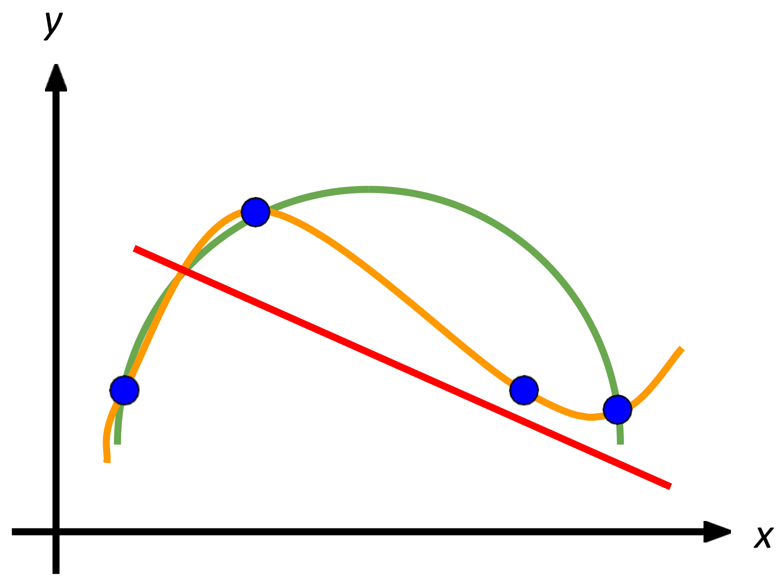
slide by Olga Veksler

Example: Deciding a tunable parameter

- Want to fit a polynomial f (**x**,**w**)

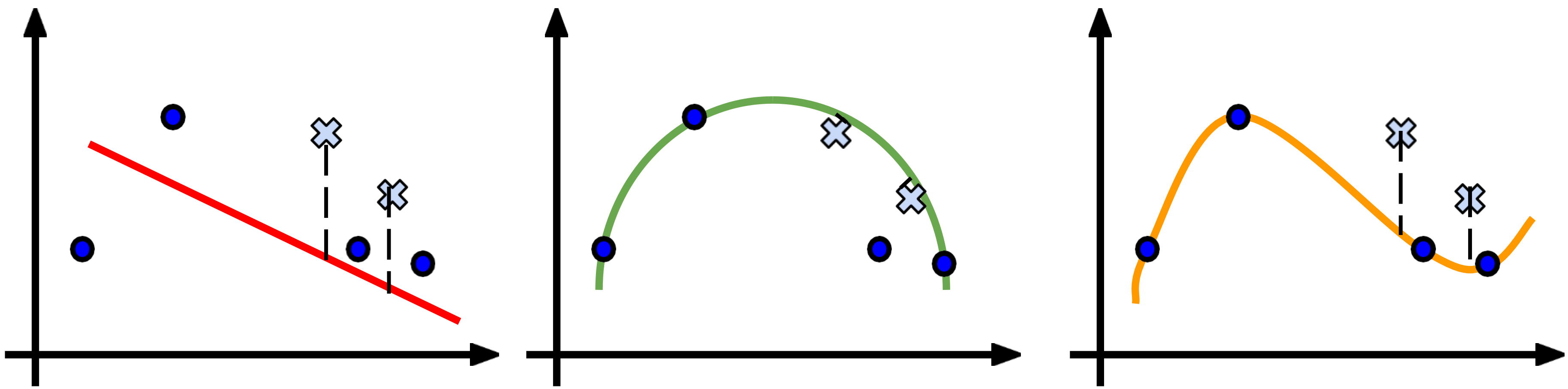
- Instead of fixing polynomial degree, make it parameter **d**
  - learning machine f (**x**,**w**,**d**)

- Consider just three choices for **d**
  - degree 1
  - degree 2
  - degree 3



- Training error is a bad measure to choose **d**
  - degree 3 is the best according to the training error, but overfits the data

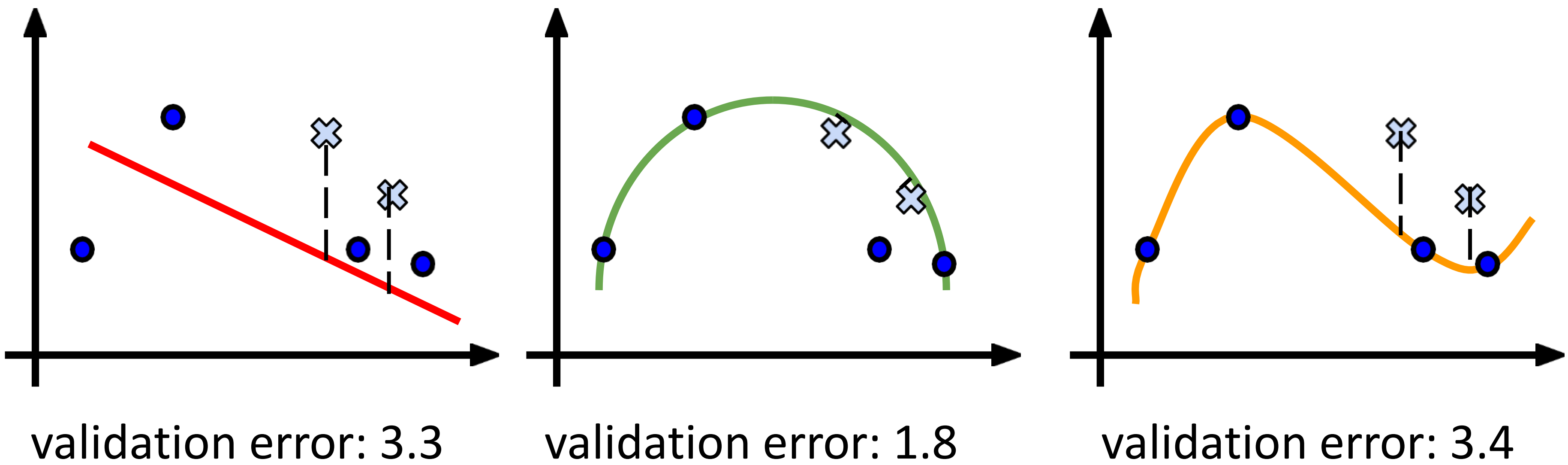
Training/Test Data Split



- Validation error sems appropriate for selecting **d**.
  - degree 2 is the best model according to the test error
- Except what do we report as the test error now?
- Validation error should be computed on data that was **not used for training at all!**
- Here used "validation" data for training, *i.e.* choosing model

slide by Olga Veksler

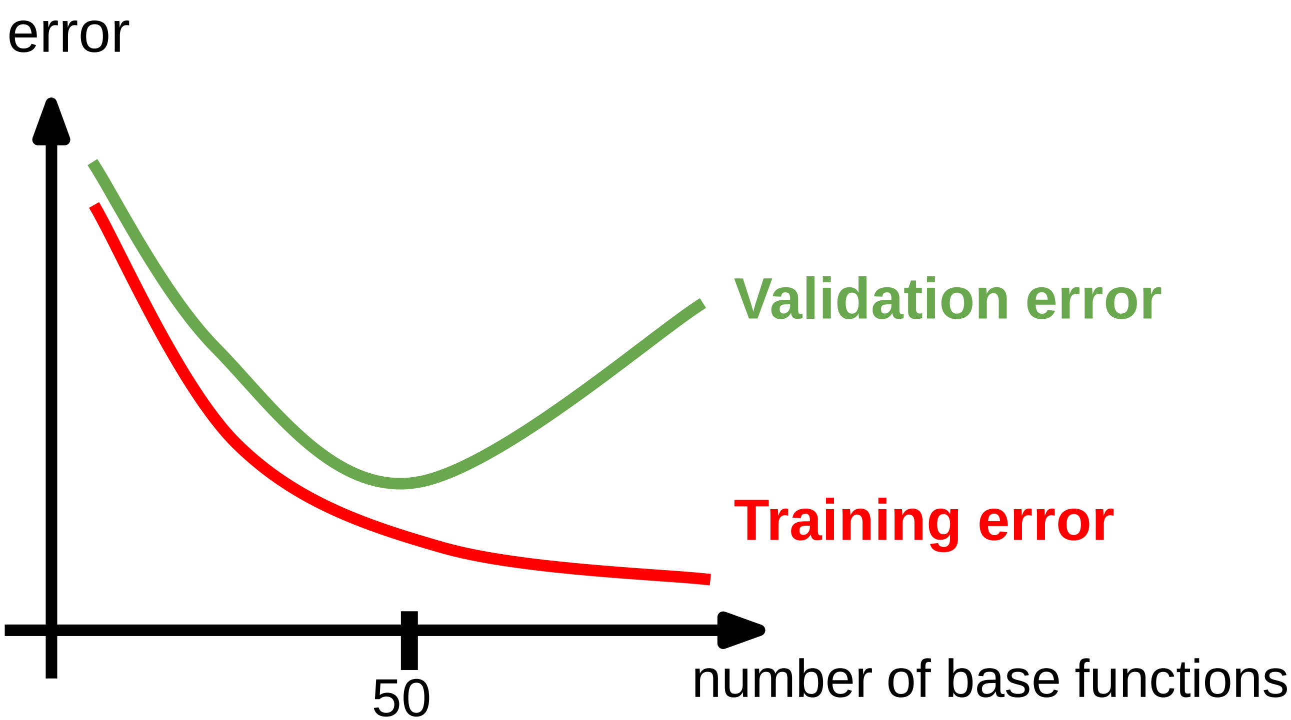
Training/Validation/Test Data



- Training Data
- Validation Data
  - **d** = 2 is chosen
- Test Data
  - 1.3 test error computed for **d** = 2

slide by Olga Veksler

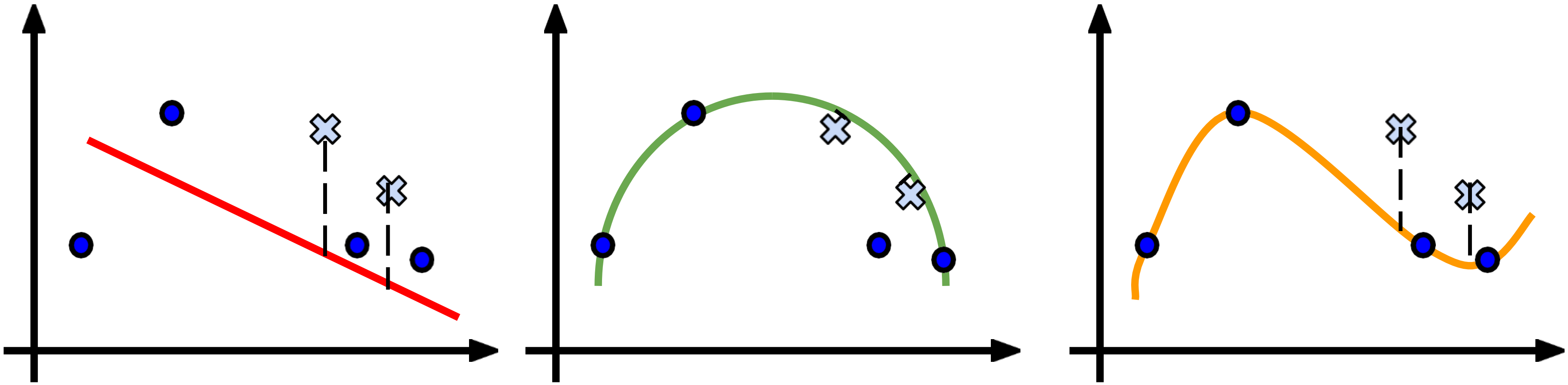
Choosing Parameters: Example



- Need to choose number of hidden units for a Multi Layer Perceptron
  - The more hidden units, the better can fit training data
  - But at some point we overfit the data

slide by Olga Veksler

Diagnosing Underfitting/Overfitting



Underfitting
- large training error
- large validation error

Just Right
- small training error
- small validation error

Overfitting
- small training error
- large validation error

Fixing Underfitting/Overfitting

- Fixing Underfitting
  - getting more training examples will not help
  - get more features
  - try more complex classifier
    - if using MLP, try more hidden units

- Fixing Overfitting
  - getting more training examples might help
  - try smaller set of features
  - Try less complex classifier
    - If using MLP, try less hidden units

Train/Validation/Test Method

- Good news
  - Very simple

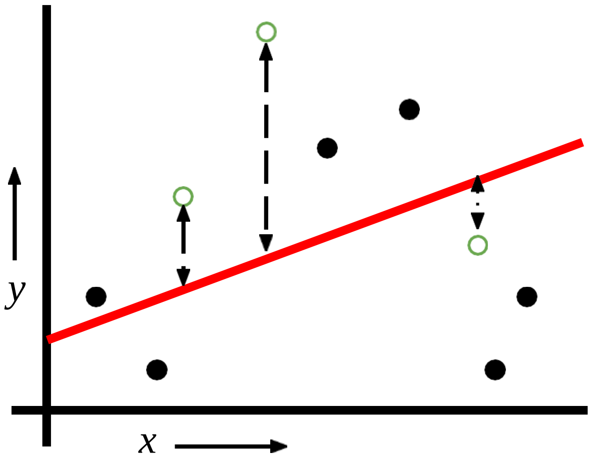
- Bad news:
  - Wastes data
    - in general, the more data we have, the better are the estimated parameters
    - we estimate parameters on 40% less data, since 20% removed for test and 20% for validation data
  - If we have a small dataset our test (validation) set might just be lucky or unlucky

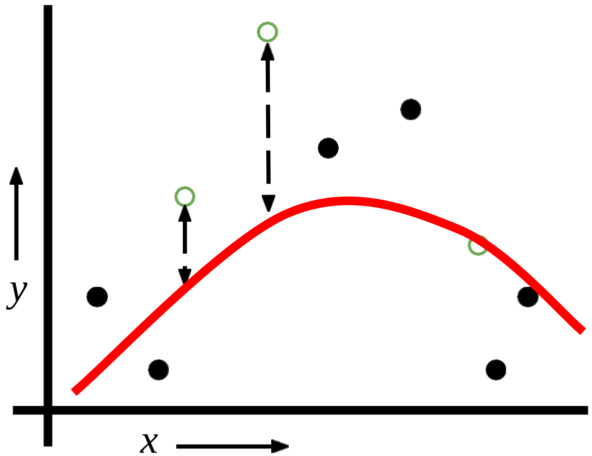
- Cross Validation is a method for performance evaluation that wastes less data

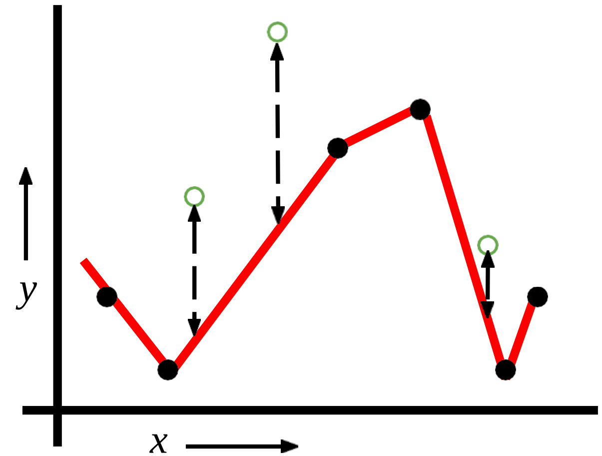
slide by Olga Veksler

Small Dataset

Linear Model: Quadratic Model: Join the dots Model:



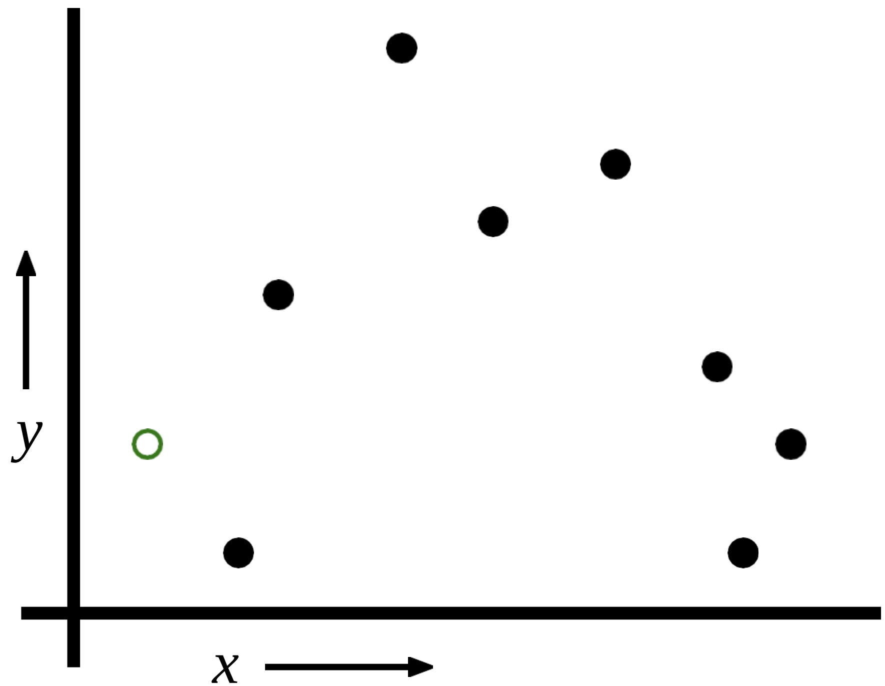




Mean Squared Error = 2.4 Mean Squared Error = 0.9 Mean Squared Error = 2.2

slide by Olga Veksler

LOOCV (Leave-one-out Cross Validation)

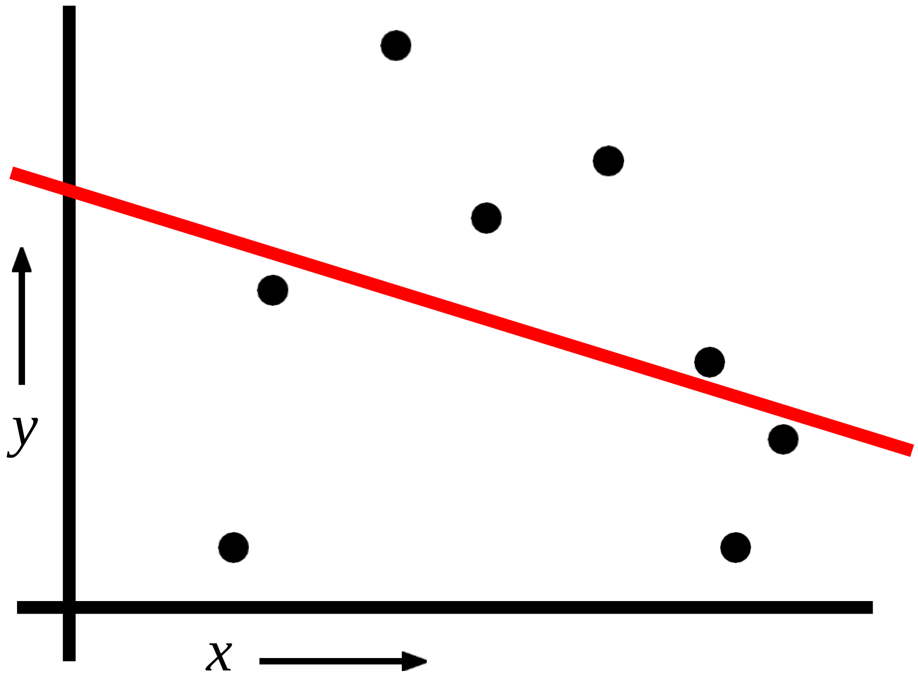


For k=1 to n

1. Let $(\mathbf{x}^{(k)},\mathbf{y}^{(k)})$ be the k^{th} example
2. Temporarily remove $(\mathbf{x}^{(k)},\mathbf{y}^{(k)})$ from the dataset

slide by Olga Veksler

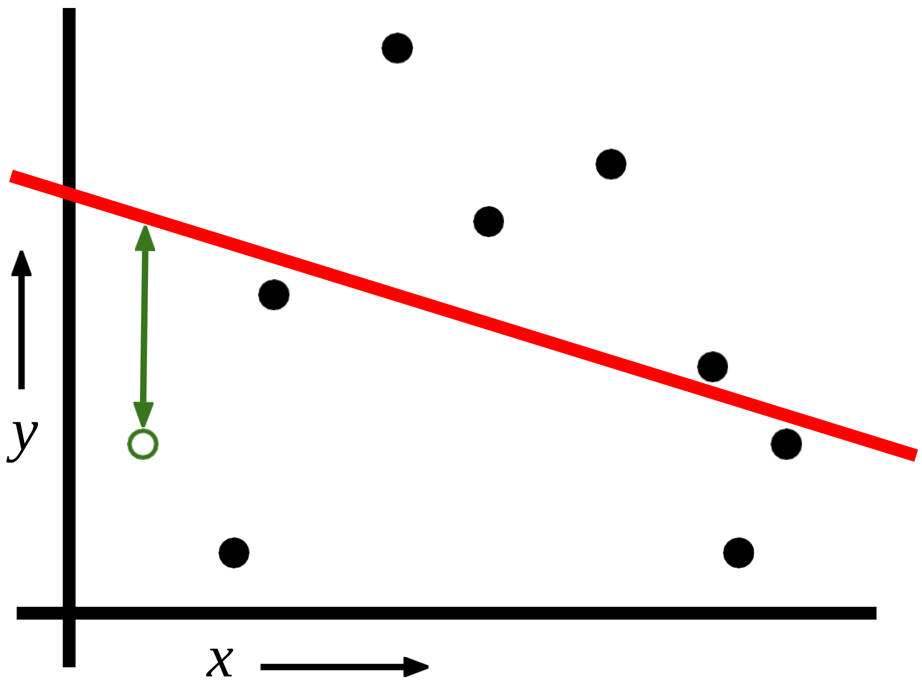
LOOCV (Leave-one-out Cross Validation)



For k=1 to n

1. Let $(\mathbf{x}^{(k)},\mathbf{y}^{(k)})$ be the k^{th} example
2. Temporarily remove $(\mathbf{x}^{(k)},\mathbf{y}^{(k)})$ from the dataset
3. Train on the remaining n-1 examples

LOOCV (Leave-one-out Cross Validation)

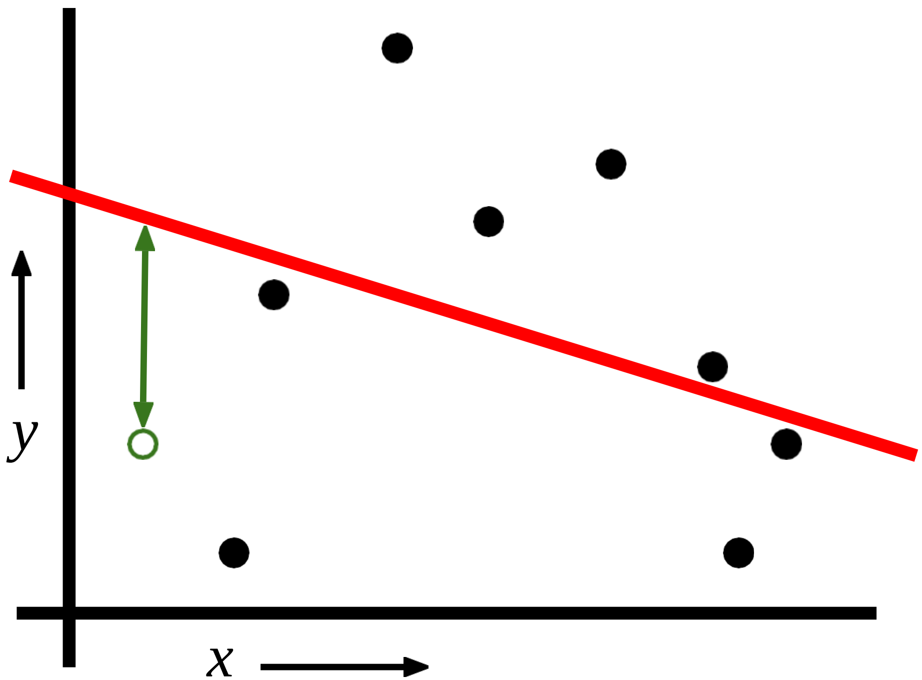


For k=1 to n

1. Let $(\mathbf{x}^{(k)},\mathbf{y}^{(k)})$ be the k^{th} example
2. Temporarily remove $(\mathbf{x}^{(k)},\mathbf{y}^{(k)})$ from the dataset
3. Train on the remaining n-1 examples
4. Note your error on $(\mathbf{x}^{(k)},\mathbf{y}^{(k)})$

slide by Olga Veksler

LOOCV (Leave-one-out Cross Validation)



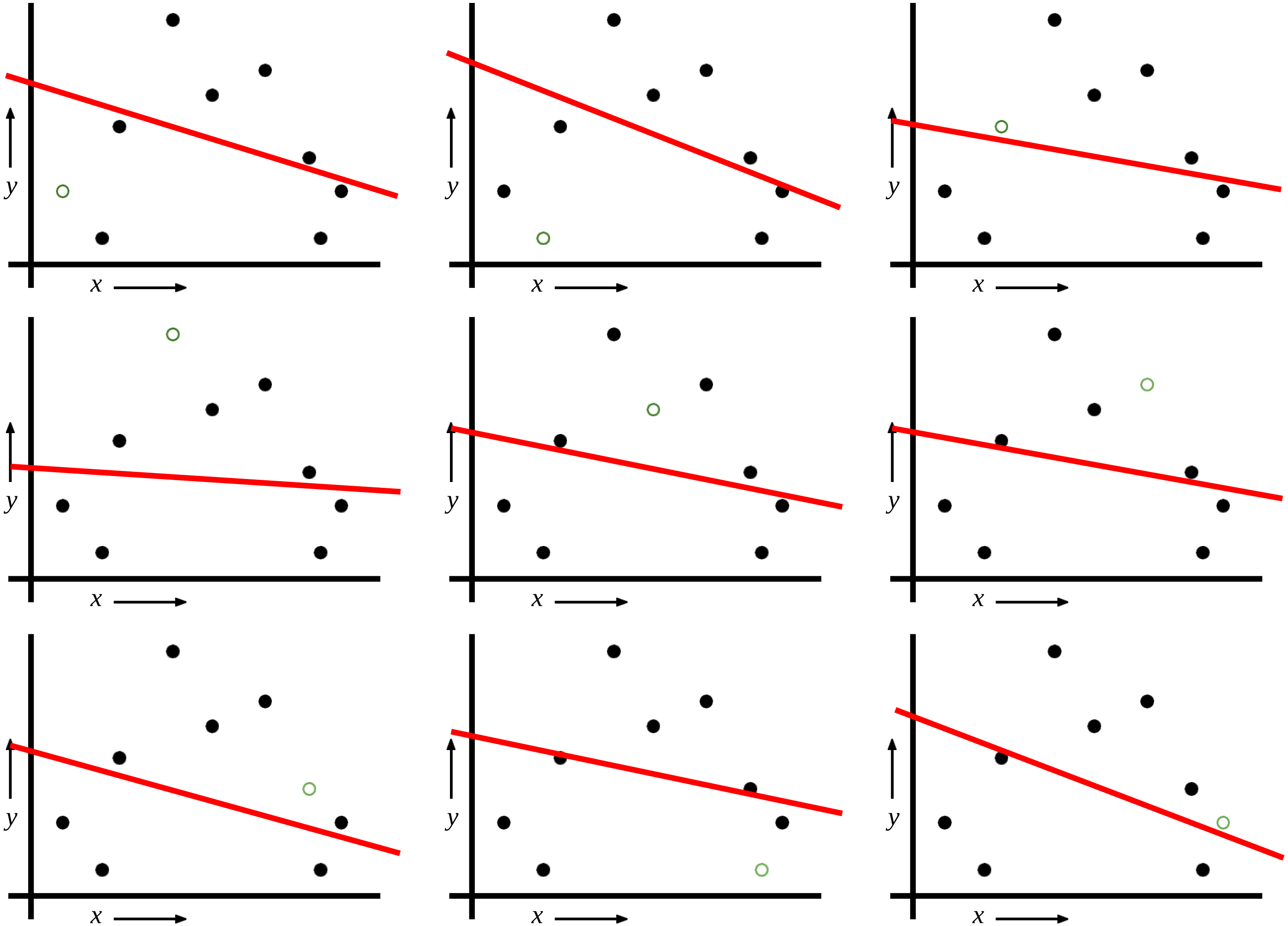
For k=1 to n

1. Let $(\mathbf{x}^{(k)},\mathbf{y}^{(k)})$ be the kth example
2. Temporarily remove $(\mathbf{x}^{(k)},\mathbf{y}^{(k)})$ from the dataset
3. Train on the remaining n-1 examples
4. Note your error on $(\mathbf{x}^{(k)},\mathbf{y}^{(k)})$

When you've done all points, report the mean error

slide by Olga Veksler

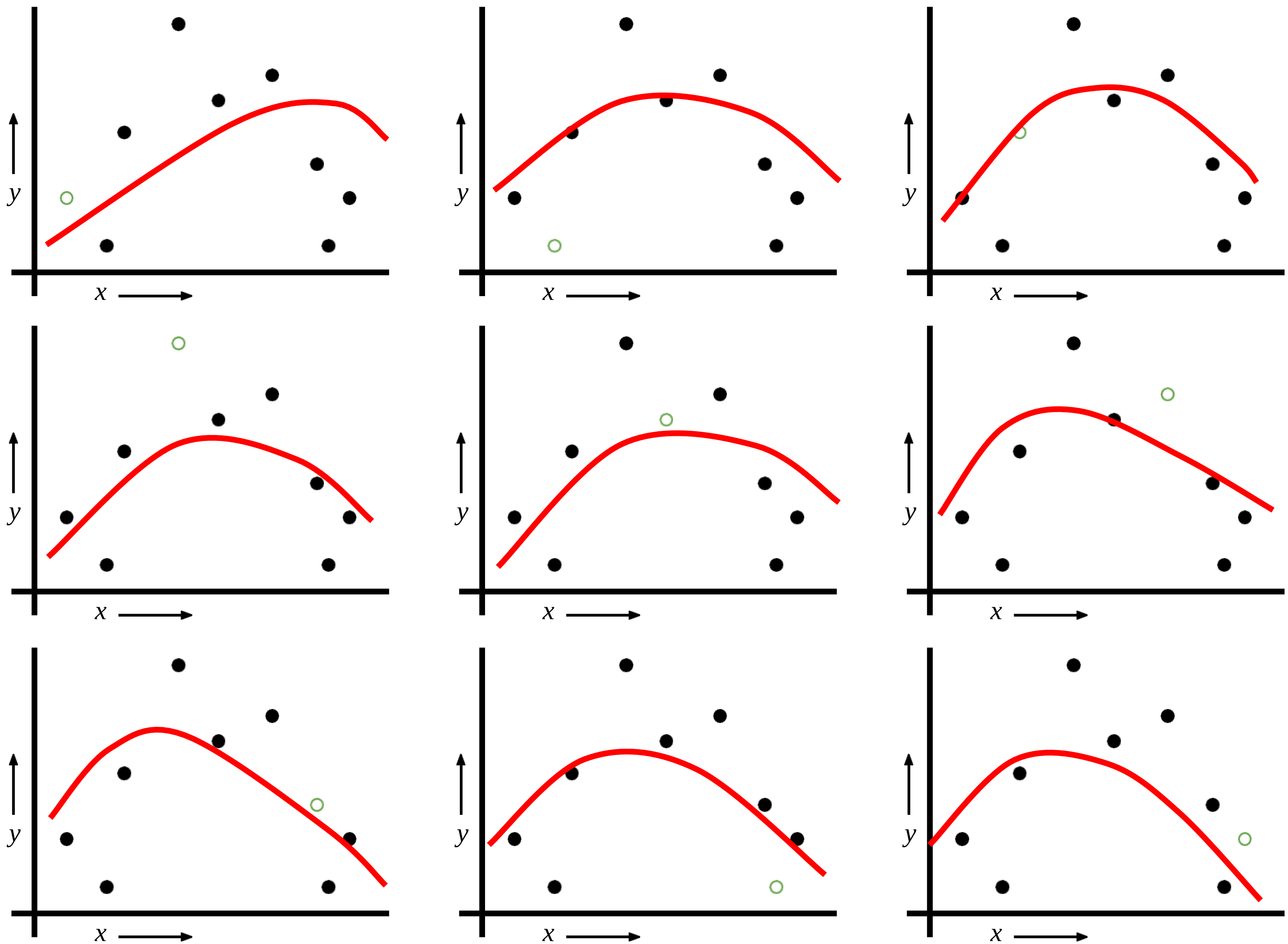
LOOCV (Leave-one-out Cross Validation)



MSE$_{LOOCV}$ = 2.12

slide by Olga Veksler

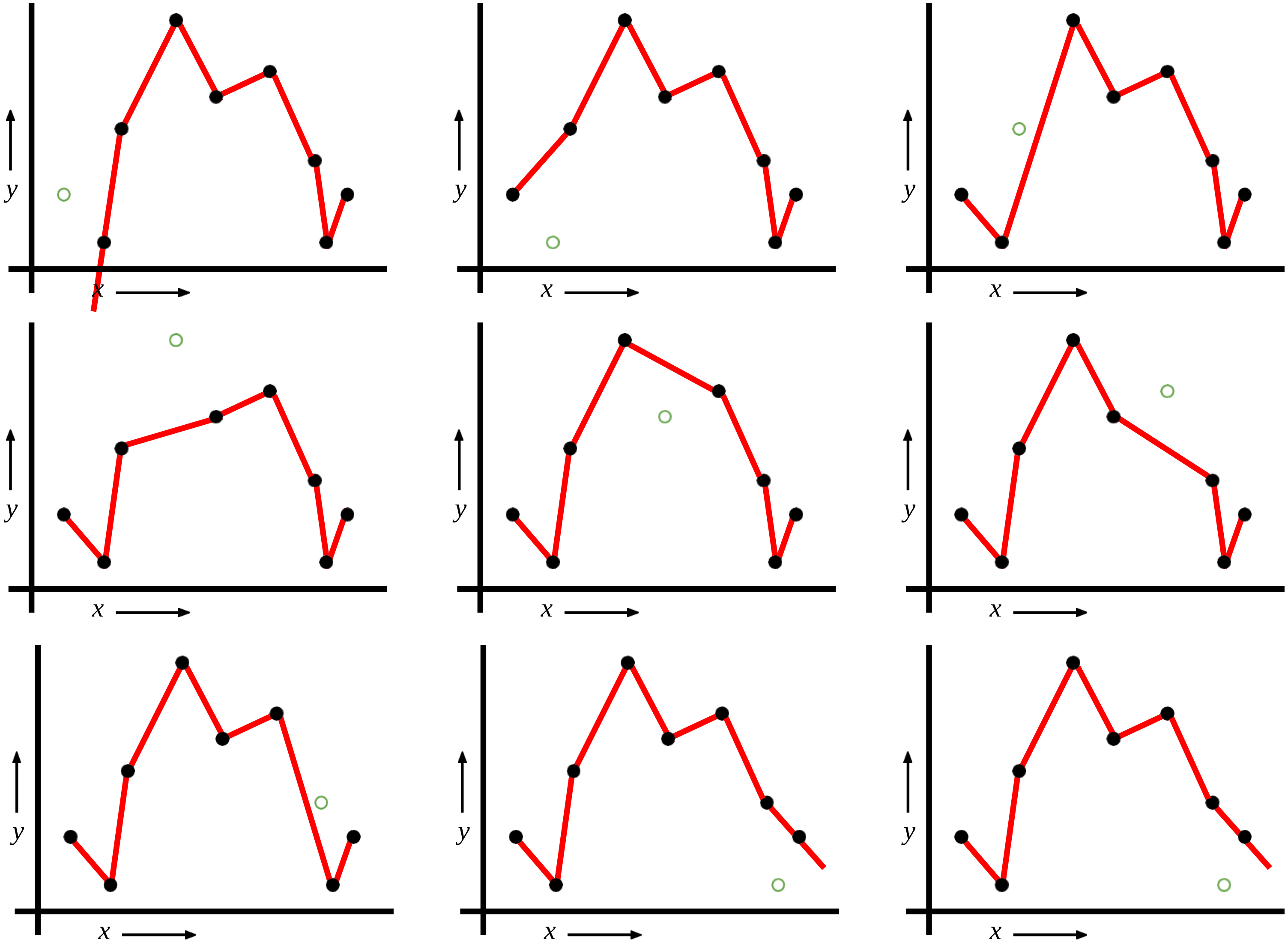
LOOCV for Quadratic Regression



$MSE_{LOOCV} = 0.96$

slide by Olga Veksler

LOOCV for Joint The Dots



MSE_{LOOCV}
$= 3.33$

slide by Olga Veksler

Which kind of validation?

	Downside	Upside
Test-set	may give unreliable estimate of future performance	cheap
Leave-one-out	expensive	doesn't waste data

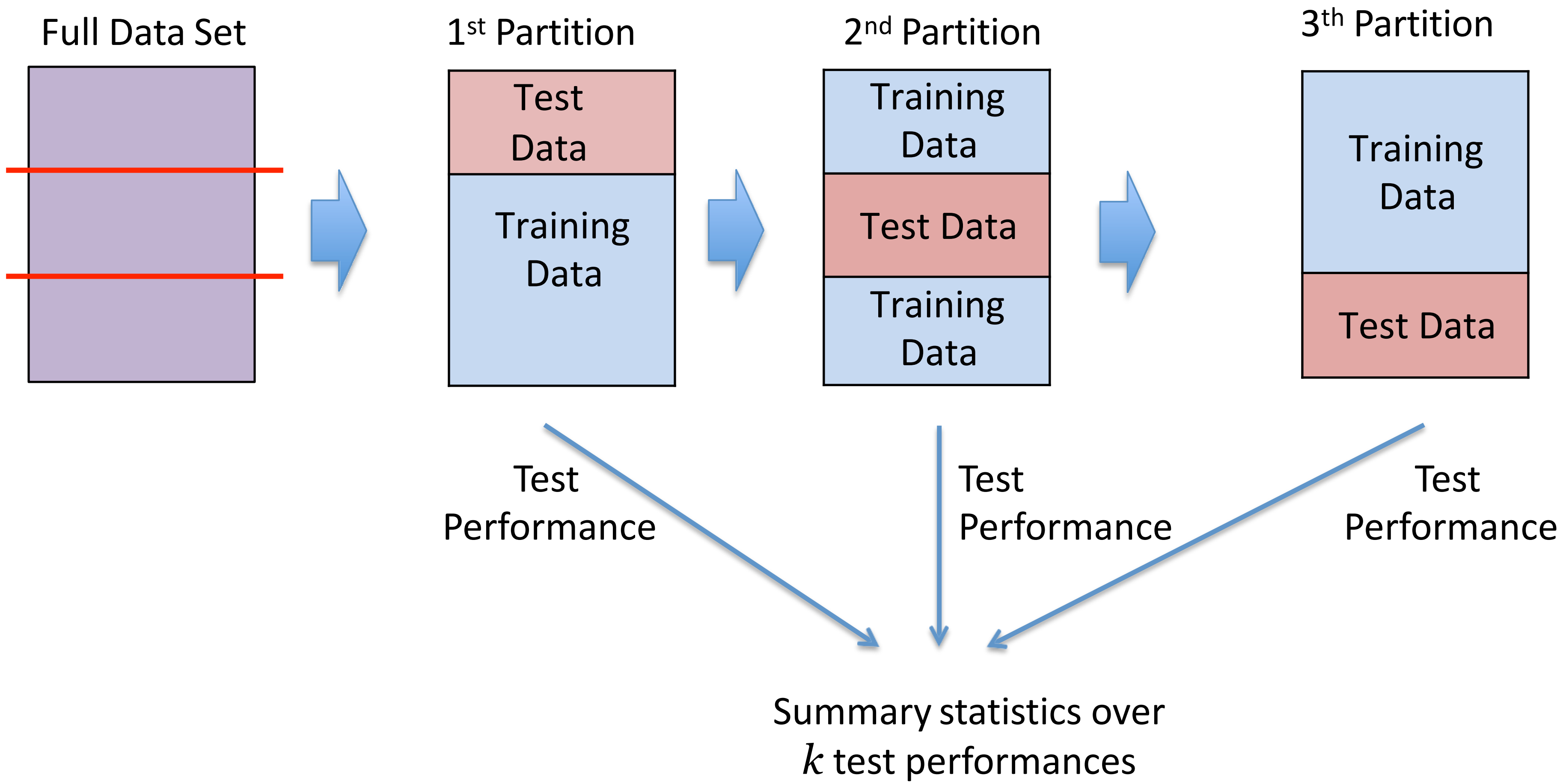
- Can we get the best of both worlds?

k-fold Cross-Validation

- Why just choose one particular "split" of the data?
  - In principle, we should do this multiple times since performance may be different for each split

- k-fold Cross-Validation (e.g., k=10)
  - randomly partition full data set of n instances into *k* disjoint subsets (each roughly of size n/k)
  - Choose each subset (fold) in turn as the test set; train model on the other folds and evaluate
  - Compute statistics over *k* test performances, or choose best of the *k* models
  - Can also do "leave-one-out CV" where k = n

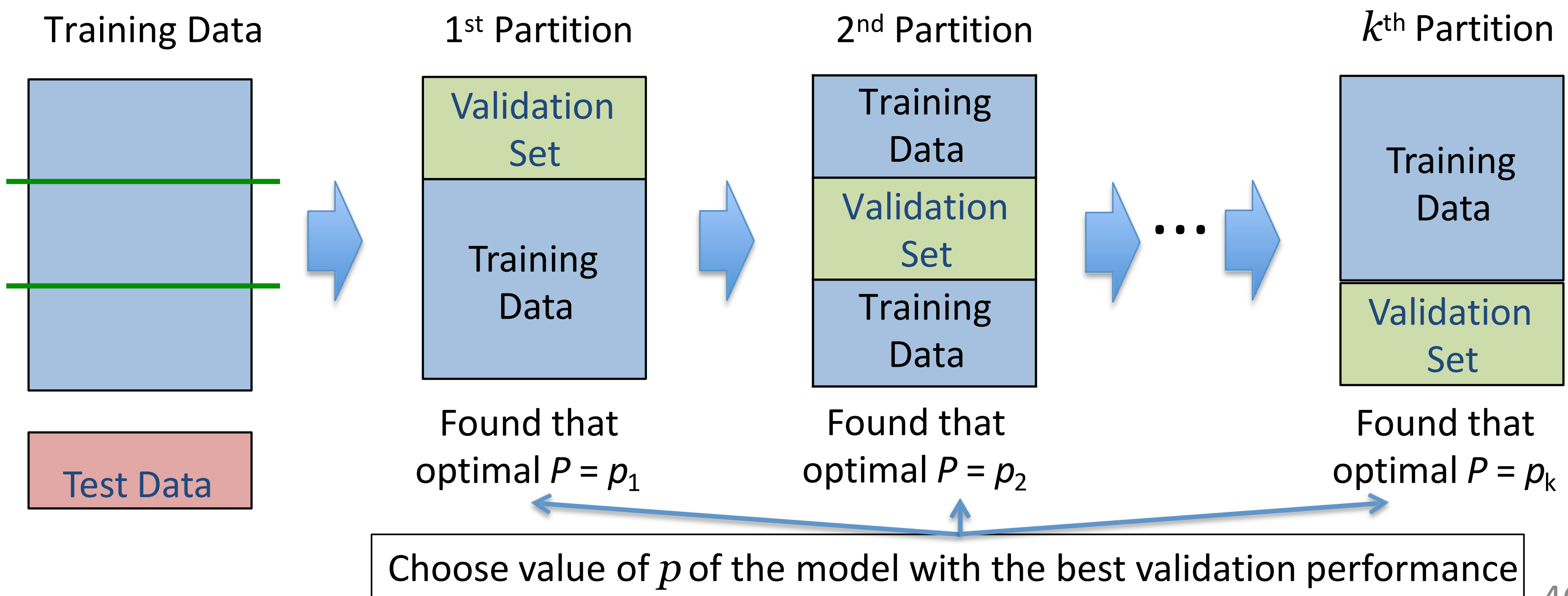
Example 3-Fold Cross-Validation (CV)



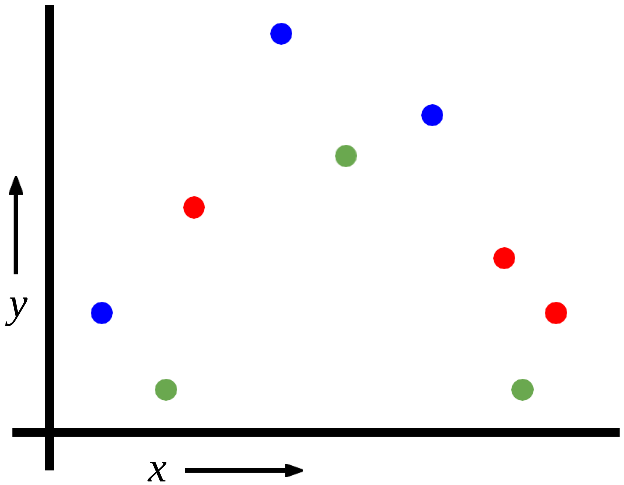
Optimizing Model Parameters

Can also use CV to choose value of model parameter *P*

- Search over space of parameter values
  - Evaluate model with $P = p_i$ on validation set
- Choose value p' with highest validation performance
- Learn model on full training set with $P = p'$



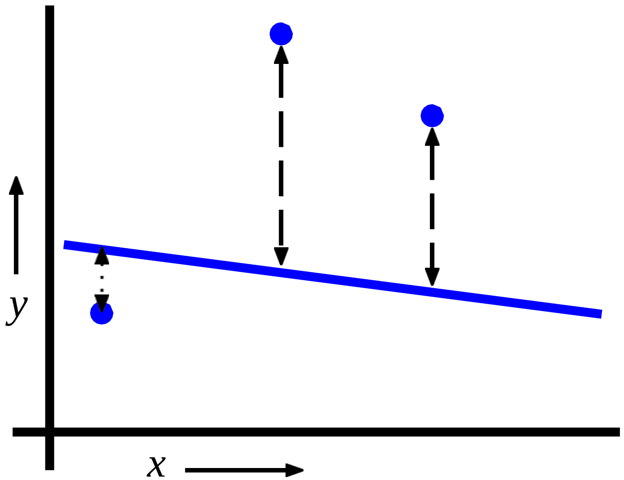
k-Fold Cross Validation for Regression



- Randomly break the dataset into k partitions
- In this example, we have k=3 partitions colored red green and blue

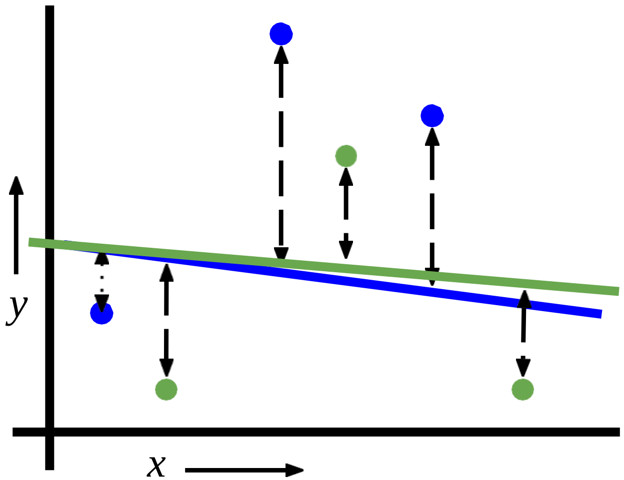
slide by Olga Veksler

k-Fold Cross Validation for Regression



- Randomly break the dataset into k partitions
- In this example, we have k=3 partitions colored red green and blue
- For the blue partition:
    - Train on all points not in the blue partition.
    - Find test-set sum of errors on blue points

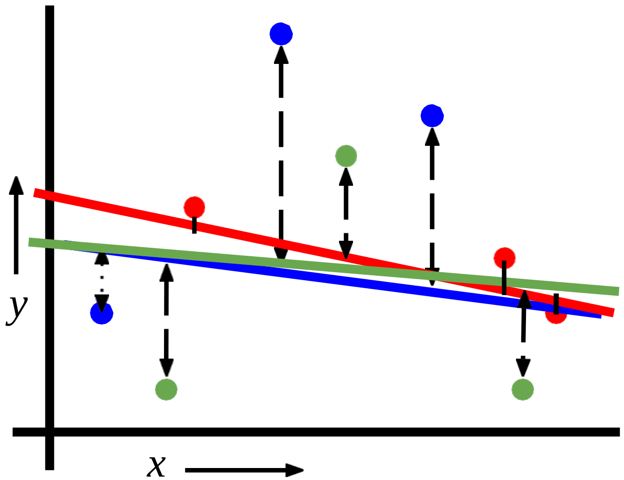
k-Fold Cross Validation for Regression



- Randomly break the dataset into k partitions
- In this example, we have k=3 partitions colored red green and blue
- For the blue partition:
  - Train on all points not in the blue partition.
  - Find test-set sum of errors on blue points
- For the green partition:
  - Train on all points not in the green partition.
  - Find test-set sum of errors on green points

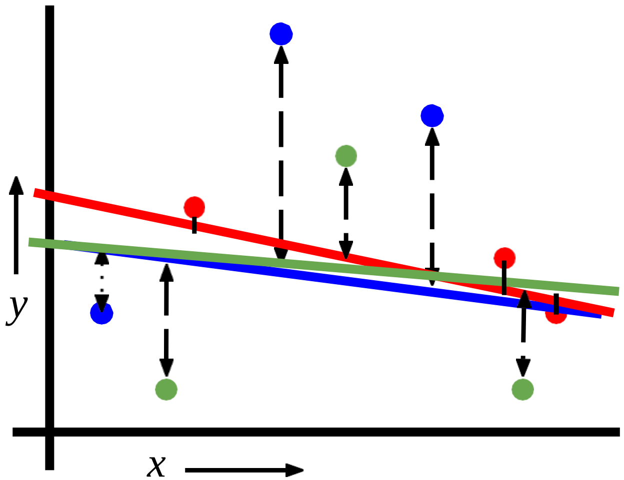
slide by Olga Veksler

k-Fold Cross Validation for Regression



- Randomly break the dataset into k partitions
- In this example, we have k=3 partitions colored red green and blue
- For the blue partition:
  - Train on all points not in the blue partition.
  - Find test-set sum of errors on blue points
- For the green partition:
  - Train on all points not in the green partition.
  - Find test-set sum of errors on green points
- For the red partition:
  - Train on all points not in the red partition.
  - Find test-set sum of errors on red points

k-Fold Cross Validation for Regression



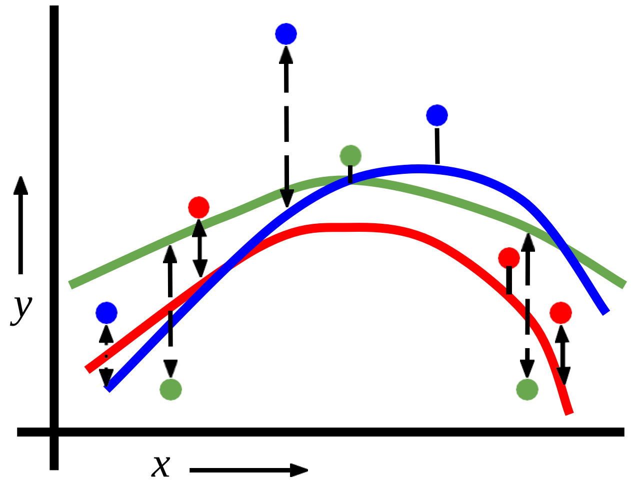
Linear Regression

$MSE_{3FOLD} = 2.05$

- Randomly break the dataset into k partitions
- In this example, we have k=3 partitions colored red green and blue
- For the blue partition:
  - Train on all points not in the blue partition.
  - Find test-set sum of errors on blue points
- For the green partition:
  - Train on all points not in the green partition.
  - Find test-set sum of errors on green points
- For the red partition:
  - Train on all points not in the red partition.
  - Find test-set sum of errors on red points
- Report the mean error

slide by Olga Veksler

k-Fold Cross Validation for Regression

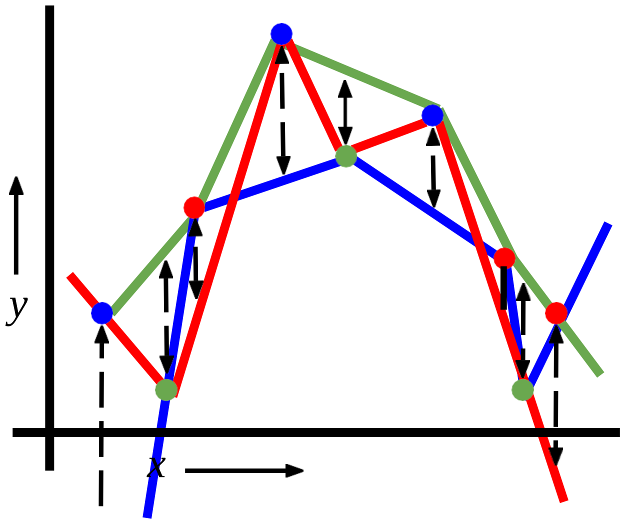


Quadratic Regression
$MSE_{3FOLD} = 1.1$

- Randomly break the dataset into k partitions
- In this example, we have k=3 partitions colored red green and blue
- For the blue partition:
  - Train on all points not in the blue partition.
  - Find test-set sum of errors on blue points
- For the green partition:
  - Train on all points not in the green partition.
  - Find test-set sum of errors on green points
- For the red partition:
  - Train on all points not in the red partition.
  - Find test-set sum of errors on red points
- Report the mean error

slide by Olga Veksler

k-Fold Cross Validation for Regression



Join the dots
$MSE_{3FOLD} = 2.93$

- Randomly break the dataset into k partitions
- In this example, we have k=3 partitions colored red green and blue
- For the blue partition:
  - Train on all points not in the blue partition.
  - Find test-set sum of errors on blue points
- For the green partition:
  - Train on all points not in the green partition.
  - Find test-set sum of errors on green points
- For the red partition:
  - Train on all points not in the red partition.
  - Find test-set sum of errors on red points
- Report the mean error

slide by Olga Veksler

Which kind of Cross Validation?

	Downside	Upside
Test-set	may give unreliable estimate of future performance	cheap
Leave-one-out	expensive	doesn't waste data
10-fold	wastes 10% of the data, 10 times more expensive than test set	only wastes 10%, only 10 times more expensive instead of **n** times
3-fold	wastes more data than 10–fold, more expensive than test set	slightly better than test–set
N-fold	Identical to Leave–one–out	

slide by Olga Veksler

Cross-validation (CV) for classification

- In regresssion, we computed the sum squared errors to understand how successful is the model.

- In classification, instead of computing the sum squared errors on a test set, we should compute…

The total number of misclassifications on a test set

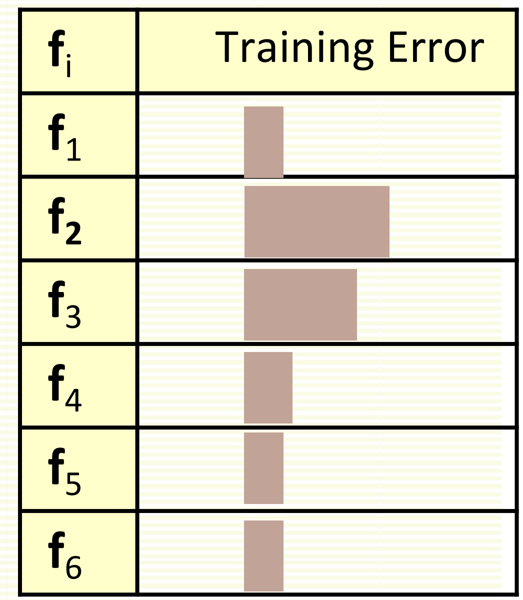
Cross-validation for classification

- Choosing *k* for k-nearest neighbors
- Choosing kernel parameters for SVM
- Any other “free” parameter of a classifier
- Choosing features to use
- Choosing which classifier to use

slide by Andrew Moore

Cross Validation-based Model Selection

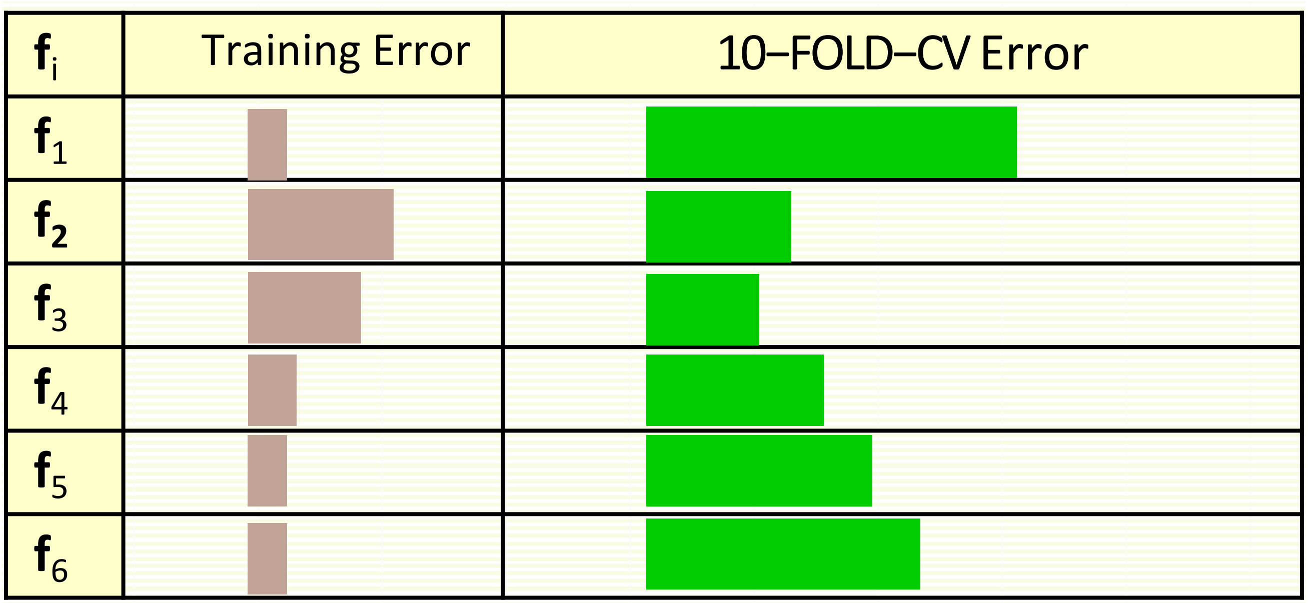
- We're trying to decide which algorithm is best by using Cross Validation (CV).
- We train each machine learning algorithm and make a table...



slide by Olga Veksler

Cross Validation-based Model Selection

- We're trying to decide which algorithm to use.
- We train each machine learning algorithm and make a table...



slide by Olga Veksler

Cross Validation-based Model Selection

- We’re trying to decide which algorithm to use.
- We train each machine learning algorithm and make a table...

f_i	Training Error	10–FOLD–CV Error	Choice
f_1			
f_2			
f_3			✓
f_4			
f_5			
f_6			

CV-based Model Selection-Example: *k*-NN

- Example: Choosing "k" for a k-nearest-neighbor regression.
- Step 1: Compute LOOCV error for six different model classes:

Algorithm	Training Error	10–fold–CV Error	Choice
k=1			
k=2			
k=3			
k=4			✓
k=5			
k=6			

- Step 2: Choose model that gave the best CV score
- Train with all the data, and that's the final model you'll use

slide by Olga Veksler

CV-based Model Selection-Example: *k*-NN

- Why stop at k=6?
  - No good reason, except it looked like things were getting worse as K was increasing

- Are we guaranteed that a local optimum of K vs LOOCV will be the global optimum?
  - No, in fact the relationship can be very bumpy

- What should we do if we are depressed at the expense of doing LOOCV for k = 1 through 1000?
  - Try: k=1, 2, 4, 8, 16, 32, 64, ... ,1024
  - Then do hillclimbing from an initial guess at k

slide by Olga Veksler

More on Cross-Validation

- Cross-validation generates an approximate estimate of how well the classifier will do on “unseen” data
  - As $k \rightarrow n$, the model becomes more accurate (more training data)
  - but, CV becomes more computationally expensive
  - Choosing $k < n$ is a compromise

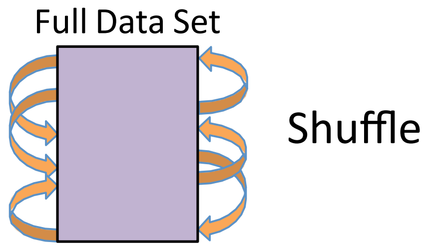
- Averaging over different partitions is more robust than just a single train/validate partition of the data

- It is an even better idea to do CV repeatedly!

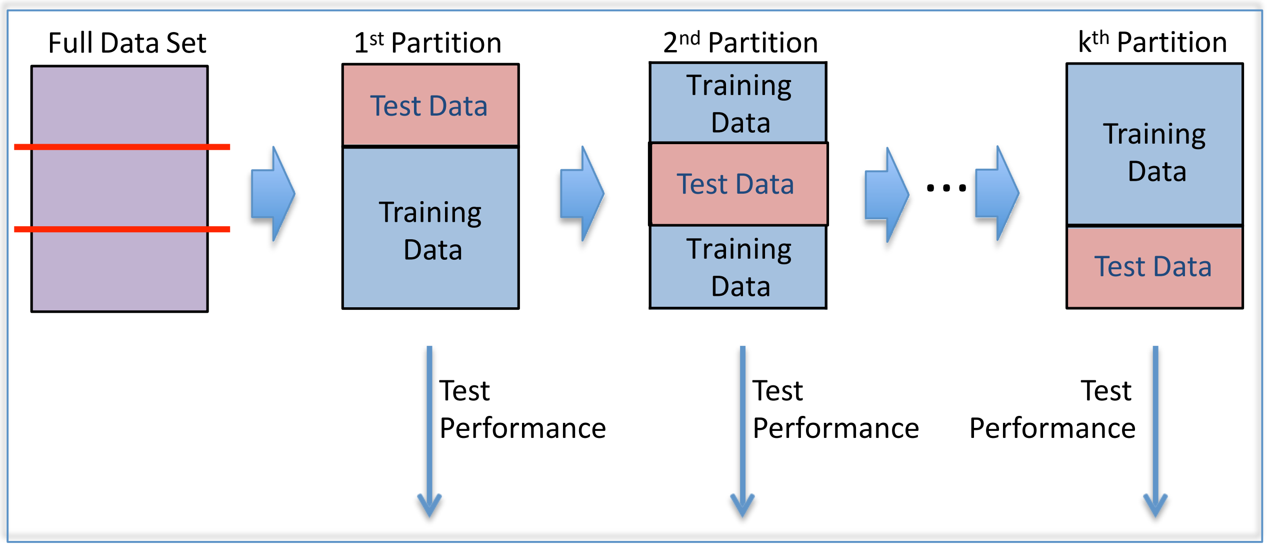
Multiple Trials of *k*-Fold Cross Validation

1.) Loop for *t* trials:

a.) Randomize Data Set



b.) Perform *k*-fold CV



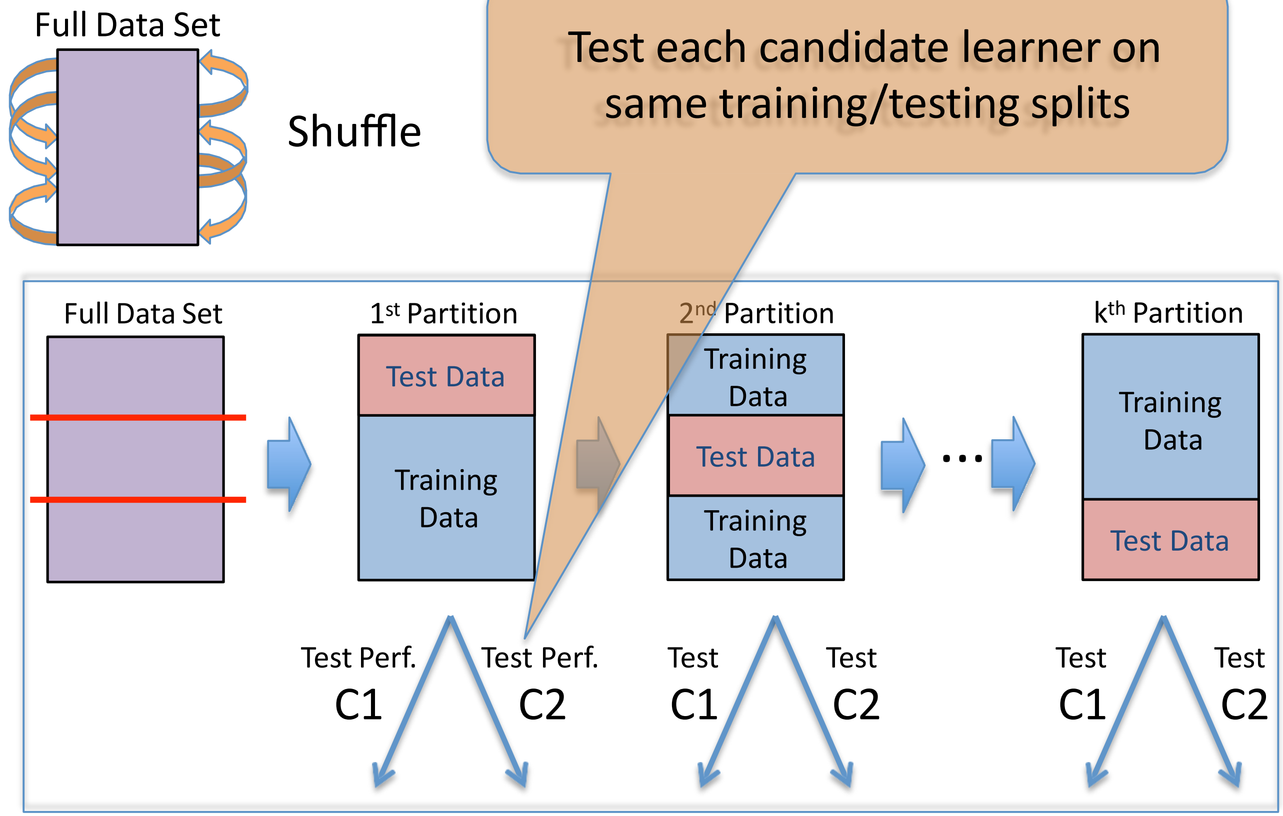
2.) Compute statistics over *t* x *k* test performances and take average

Comparing Multiple Classifiers

1.) Loop for *t* trials:

a.) Randomize Data Set

b.) Perform *k*-fold CV



2.) Compute statistics over t x k test performances

Allows us to do paired summary statistics (e.g., paired t–test)

Thanks for listening!

Questions?