

Cross-validation for detecting and preventing overfitting

Andrew W. Moore

Professor

School of Computer Science

Carnegie Mellon University

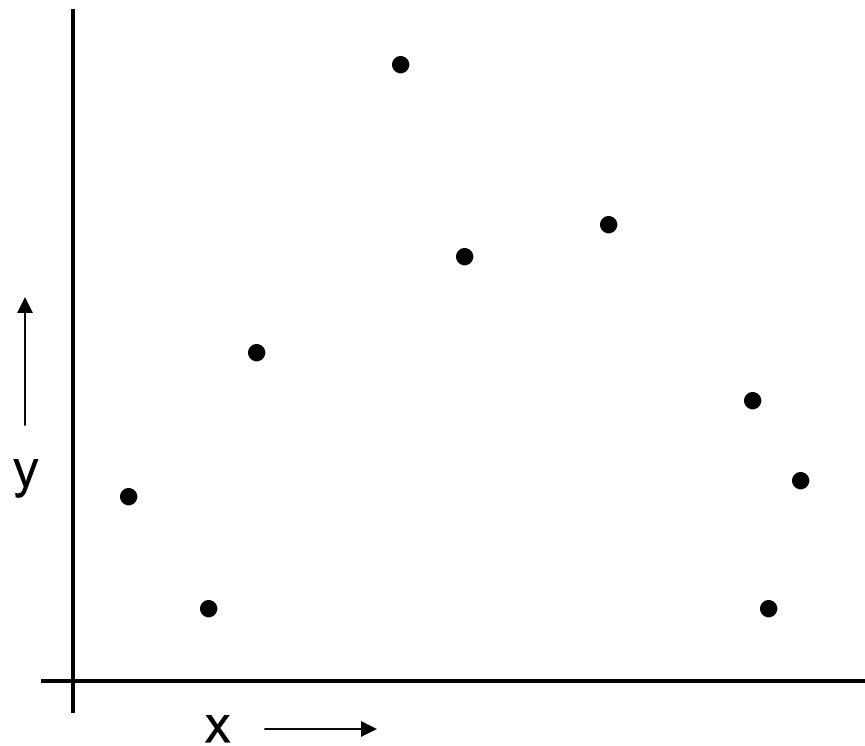
www.cs.cmu.edu/~awm

awm@cs.cmu.edu

412-268-7599

Note to other teachers and users of these slides. Andrew would be delighted if you found this source material useful in giving your own lectures. Feel free to use these slides verbatim, or to modify them to fit your own needs. PowerPoint originals are available. If you make use of a significant portion of these slides in your own lecture, please include this message, or the following link to the source repository of Andrew's tutorials: <http://www.cs.cmu.edu/~awm/tutorials>. Comments and corrections gratefully received.

A Regression Problem

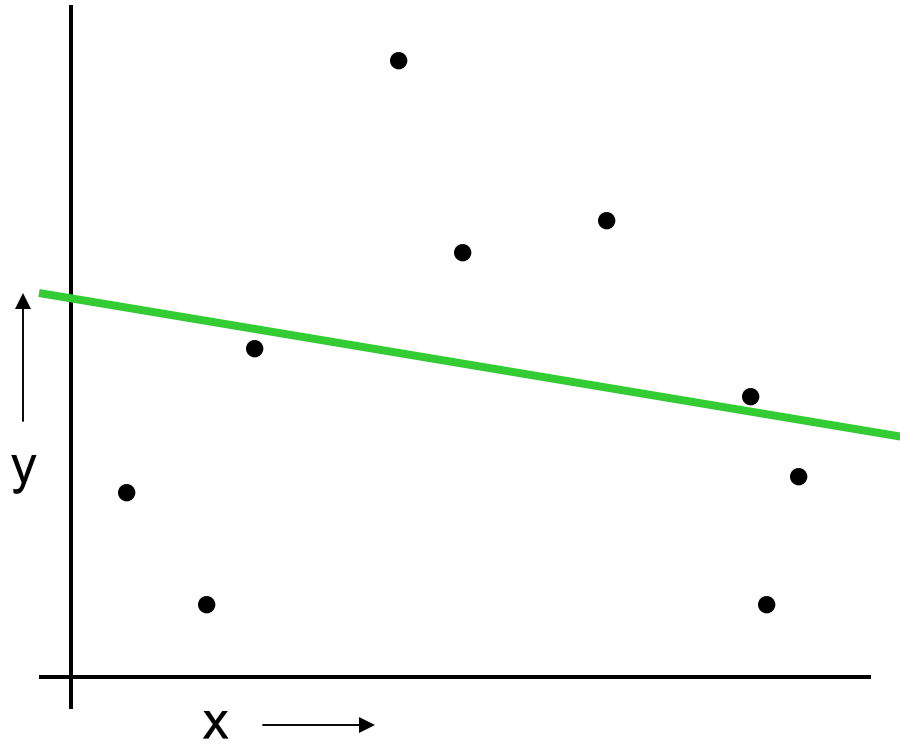


$$y = f(x) + \text{noise}$$

Can we learn f from this data?

Let's consider three methods...

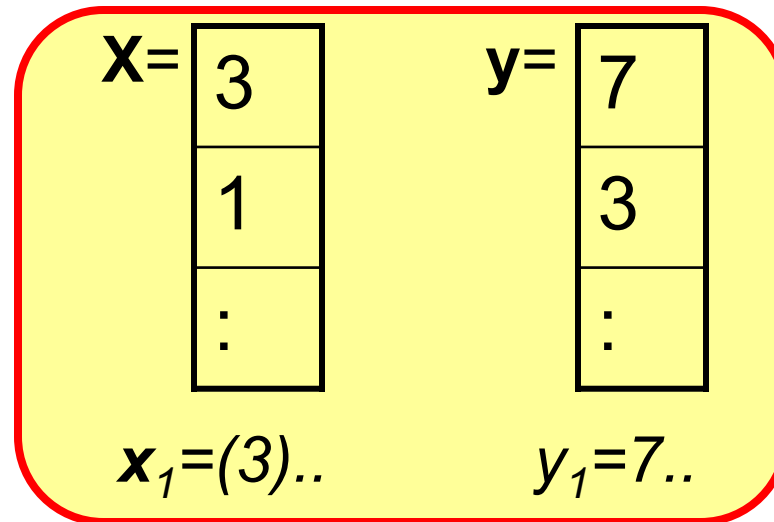
Linear Regression



Linear Regression

Univariate Linear regression with a constant term:

X	Y
3	7
1	3
\vdots	\vdots



Originally
discussed in the
previous Andrew
Lecture: "Neural
Nets"

Linear Regression

Univariate Linear regression with a constant term:

X	Y
3	7
1	3
⋮	⋮

$$\mathbf{X} = \begin{bmatrix} 3 \\ 1 \\ \vdots \end{bmatrix}$$

$$\mathbf{y} = \begin{bmatrix} 7 \\ 3 \\ \vdots \end{bmatrix}$$

$$y_1 = 7..$$

$$\mathbf{Z} = \begin{bmatrix} 1 & 3 \\ 1 & 1 \\ \vdots & \vdots \end{bmatrix}$$

$$\mathbf{y} = \begin{bmatrix} 7 \\ 3 \\ \vdots \end{bmatrix}$$

$$\mathbf{z}_1 = (1, 3)..$$

$$y_1 = 7..$$

$$\mathbf{z}_k = (1, x_k)$$

Linear Regression

Univariate Linear regression with a constant term:

X	Y
3	7
1	3
⋮	⋮

$\mathbf{X} = \begin{bmatrix} 3 \\ 1 \\ \vdots \end{bmatrix}$

$\mathbf{y} = \begin{bmatrix} 7 \\ 3 \\ \vdots \end{bmatrix}$

$y_1 = 7..$

$\mathbf{Z} = \begin{bmatrix} 1 & 3 \\ 1 & 1 \\ \vdots & \vdots \end{bmatrix}$

$\mathbf{y} = \begin{bmatrix} 7 \\ 3 \\ \vdots \end{bmatrix}$

$\mathbf{z}_1 = (1, 3)..$

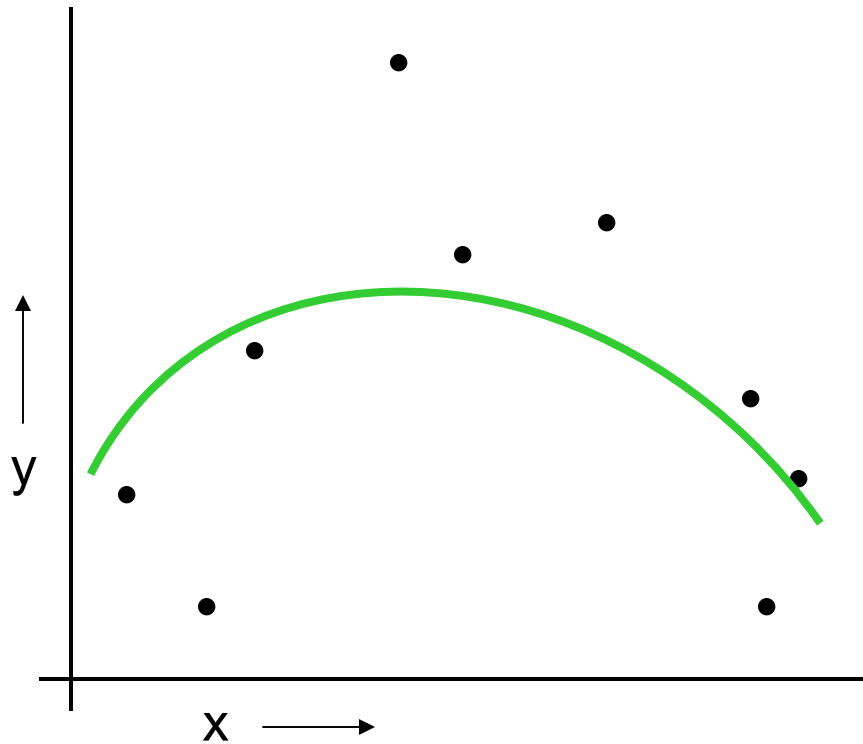
$y_1 = 7..$

$\mathbf{z}_k = (1, x_k)$

$$\boldsymbol{\beta} = (\mathbf{Z}^T \mathbf{Z})^{-1} (\mathbf{Z}^T \mathbf{y})$$

$$y^{est} = \beta_0 + \beta_1 x$$

Quadratic Regression



Quadratic Regression

X	Y
3	7
1	3
⋮	⋮

X=

3
1
⋮

y=

7
3
⋮

$y_1=7..$

Z=

1	3	9
1	1	1
⋮		

$z=(1, x, x^2)$

y=

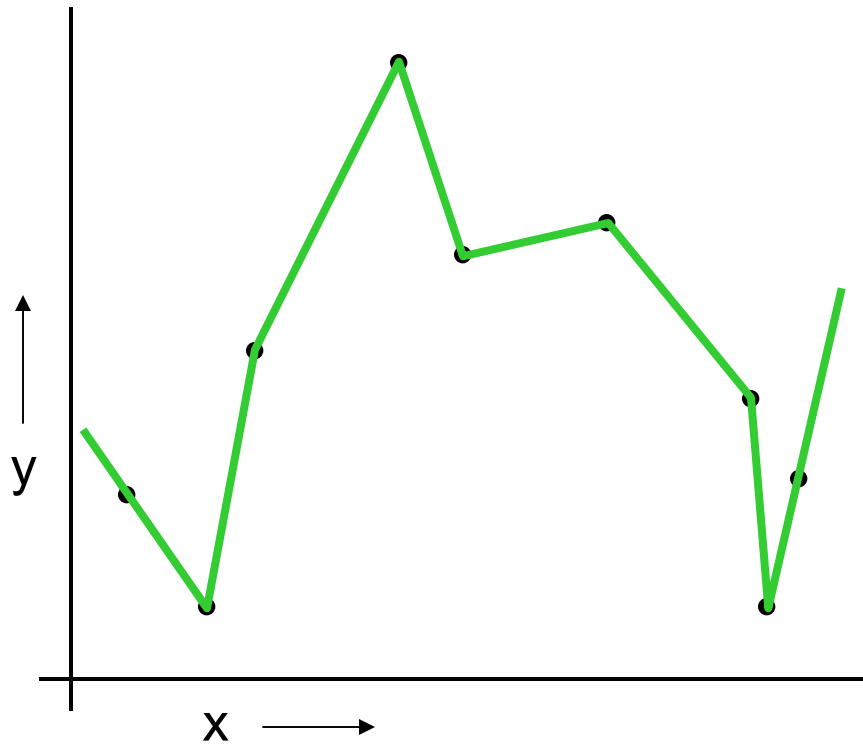
7
3
⋮

$$\beta = (\mathbf{Z}^T \mathbf{Z})^{-1} (\mathbf{Z}^T \mathbf{y})$$

$$y^{est} = \beta_0 + \beta_1 x + \beta_2 x^2$$

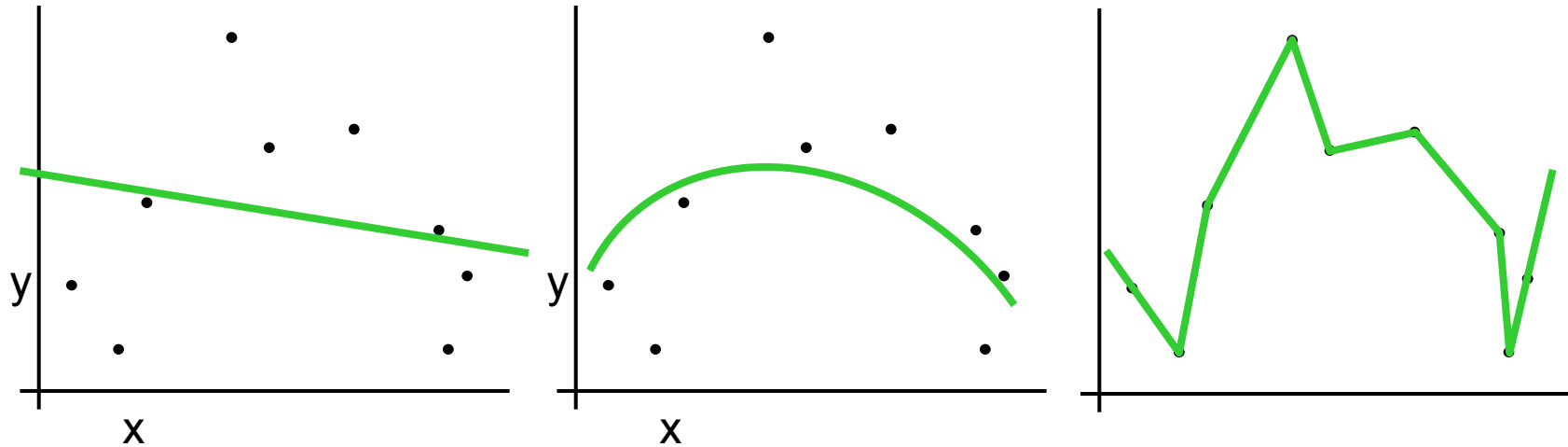
Much more about
this in the future
Andrew Lecture:
“Favorite
Regression
Algorithms”

Join-the-dots



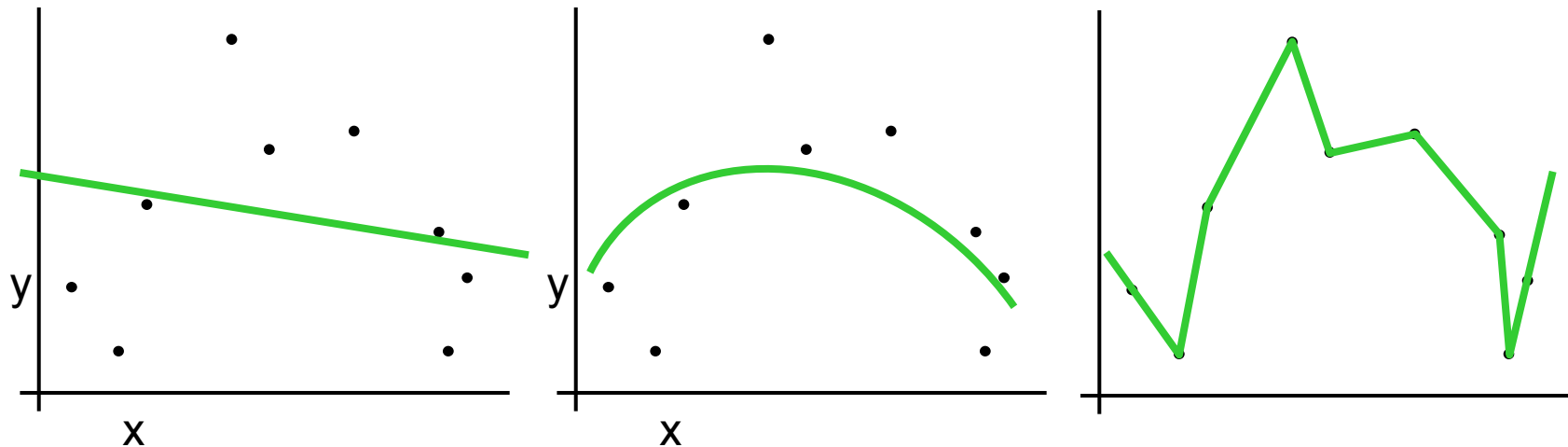
Also known as **piecewise linear nonparametric regression** if that makes you feel better

Which is best?



Why not choose the method with the best fit to the data?

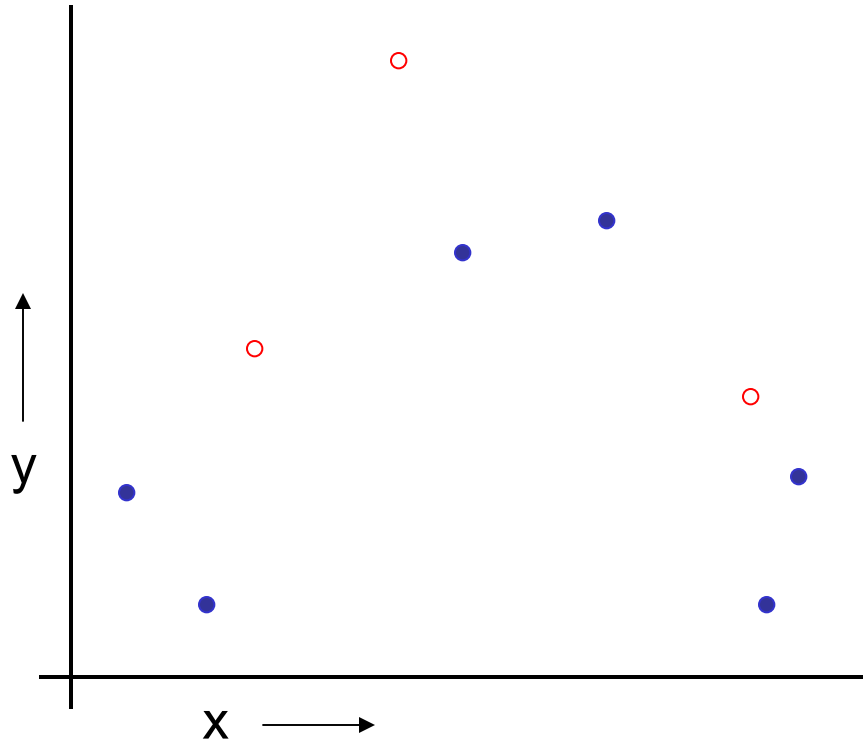
What do we really want?



Why not choose the method with the best fit to the data?

“How well are you going to predict future data drawn from the same distribution?”

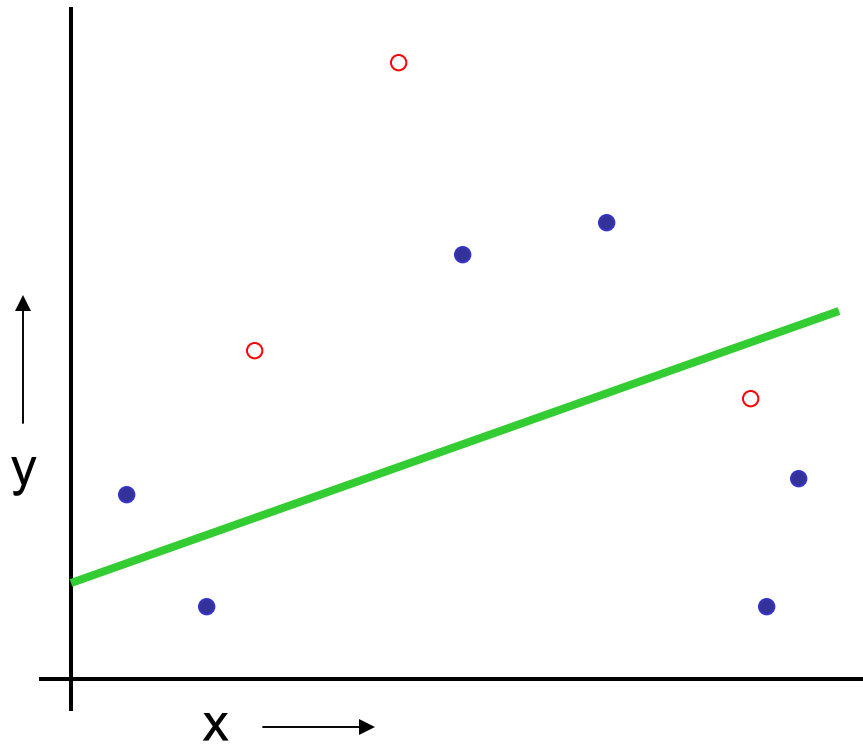
The test set method



1. Randomly choose
30% of the data to be in a
test set

2. The remainder is a
training set

The test set method



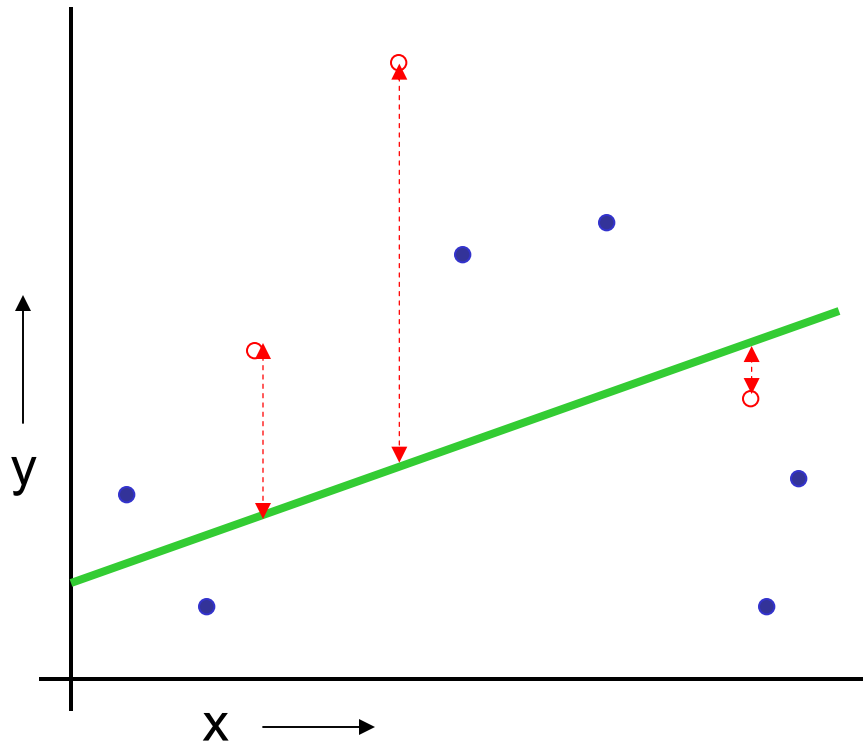
1. Randomly choose 30% of the data to be in a **test set**

2. The remainder is a **training set**

3. Perform your regression on the training set

(Linear regression example)

The test set method

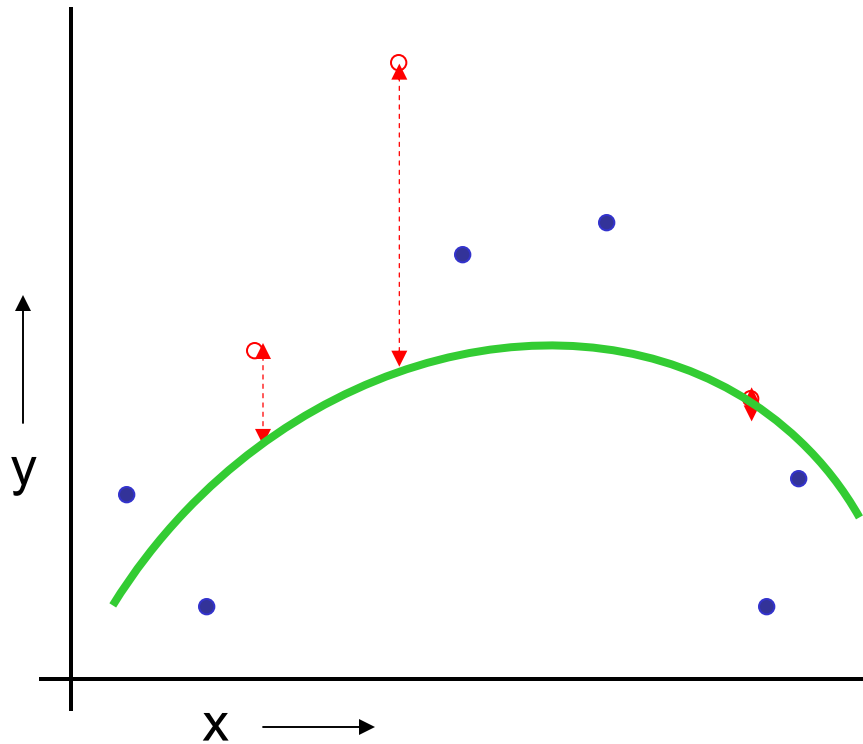


(Linear regression example)

Mean Squared Error = 2.4

1. Randomly choose 30% of the data to be in a **test set**
2. The remainder is a **training set**
3. Perform your regression on the training set
4. Estimate your future performance with the **test set**

The test set method



(Quadratic regression example)

Mean Squared Error = 0.9

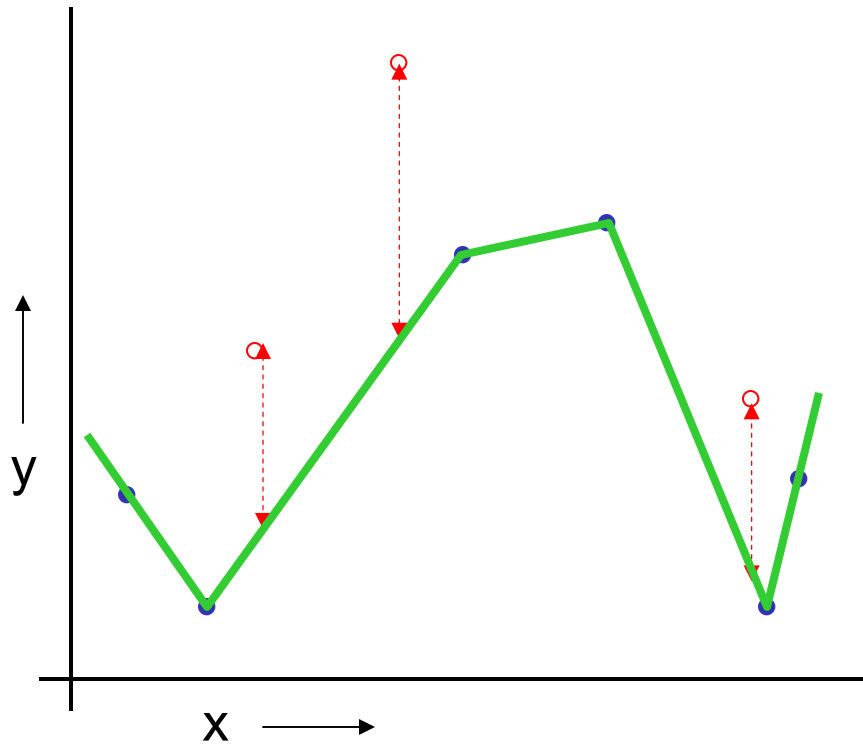
1. Randomly choose
30% of the data to be in a
test set

2. The remainder is a
training set

3. Perform your
regression on the training
set

4. Estimate your future
performance with the test
set

The test set method



(Join the dots example)

Mean Squared Error = 2.2

1. Randomly choose 30% of the data to be in a **test set**
2. The remainder is a **training set**
3. Perform your regression on the training set
4. Estimate your future performance with the **test set**

The test set method

Good news:

- Very very simple
- Can then simply choose the method with the best test-set score

Bad news:

- What's the downside?

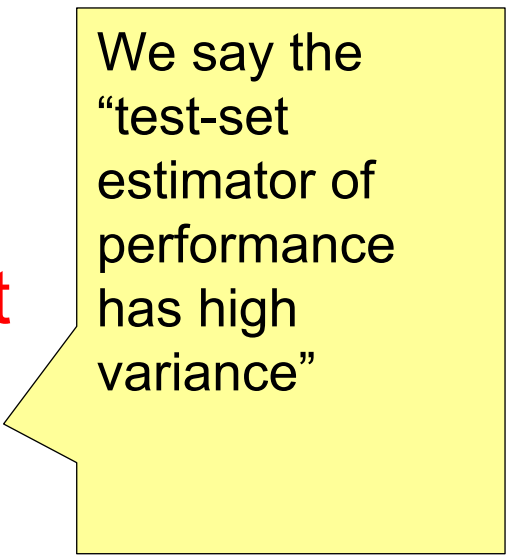
The test set method

Good news:

- Very very simple
- Can then simply choose the method with the best test-set score

Bad news:

- Wastes data: we get an estimate of the best method to apply to 30% less data
- If we don't have much data, our test-set might just be lucky or unlucky

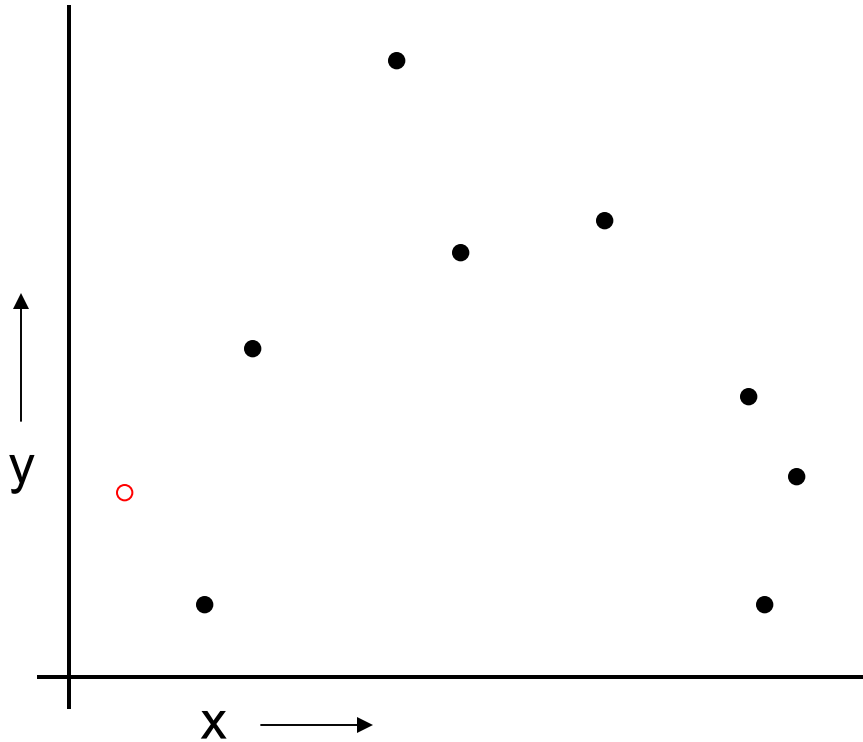


We say the “test-set estimator of performance has high variance”

LOOCV (Leave-one-out Cross Validation)

For $k=1$ to R

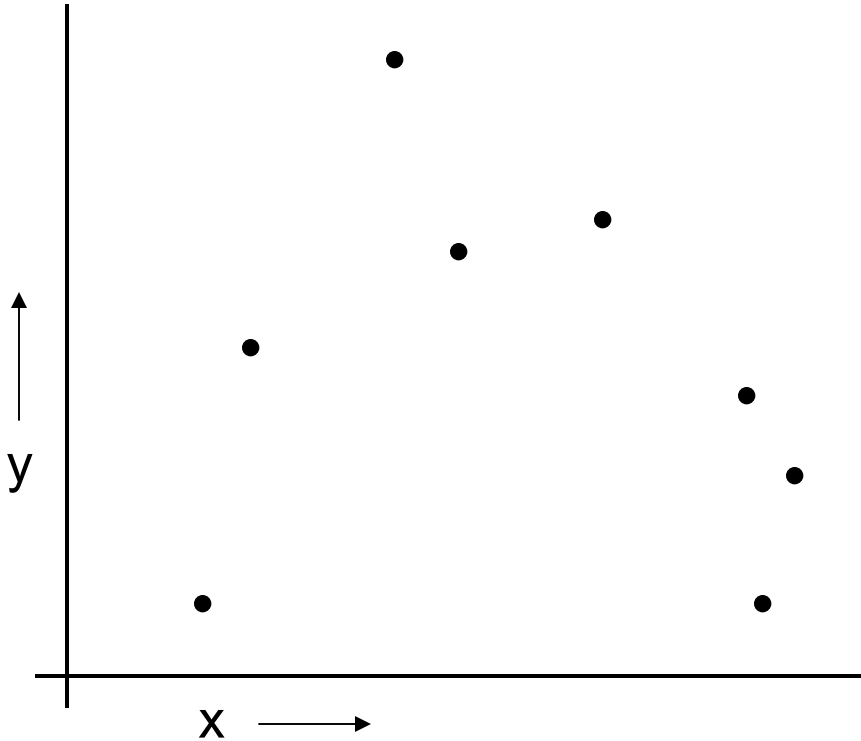
1. Let (x_k, y_k) be the k^{th} record



LOOCV (Leave-one-out Cross Validation)

For $k=1$ to R

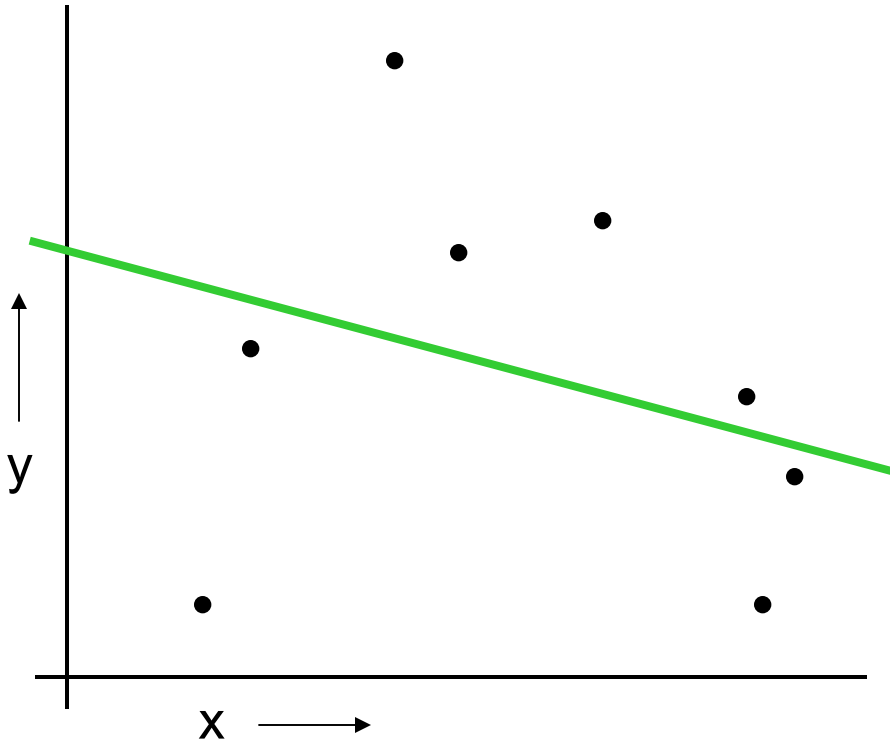
1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset



LOOCV (Leave-one-out Cross Validation)

For $k=1$ to R

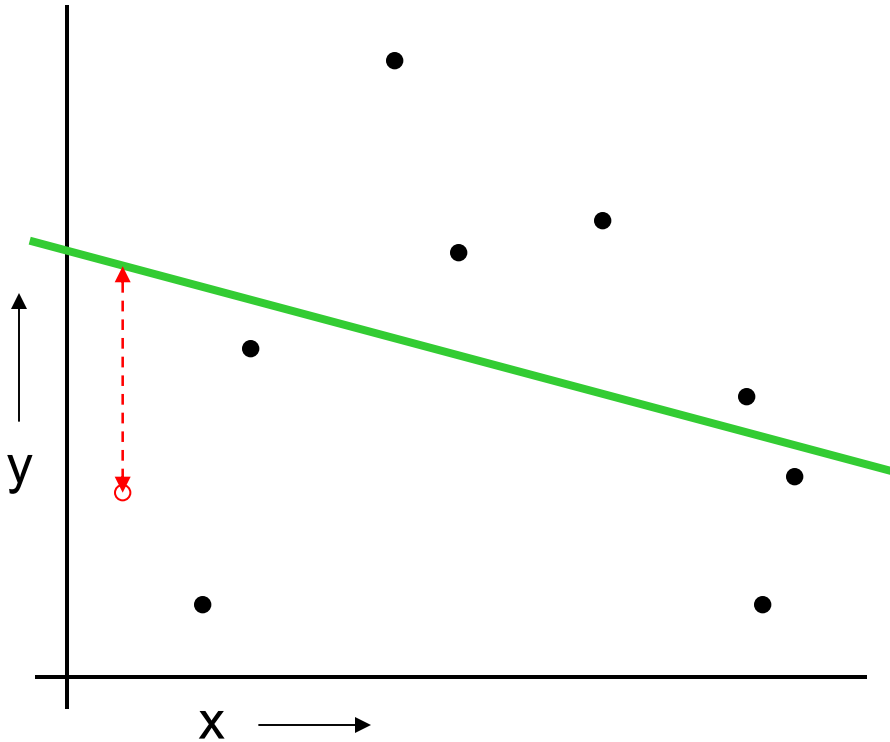
1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset
3. Train on the remaining $R-1$ datapoints



LOOCV (Leave-one-out Cross Validation)

For $k=1$ to R

1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset
3. Train on the remaining $R-1$ datapoints
4. Note your error (x_k, y_k)

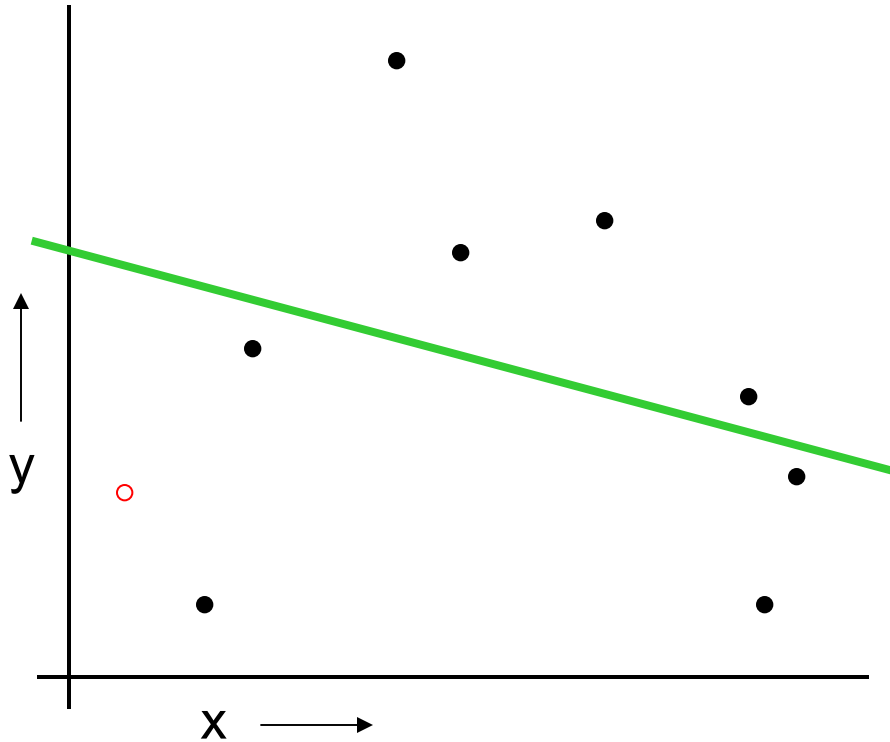


LOOCV (Leave-one-out Cross Validation)

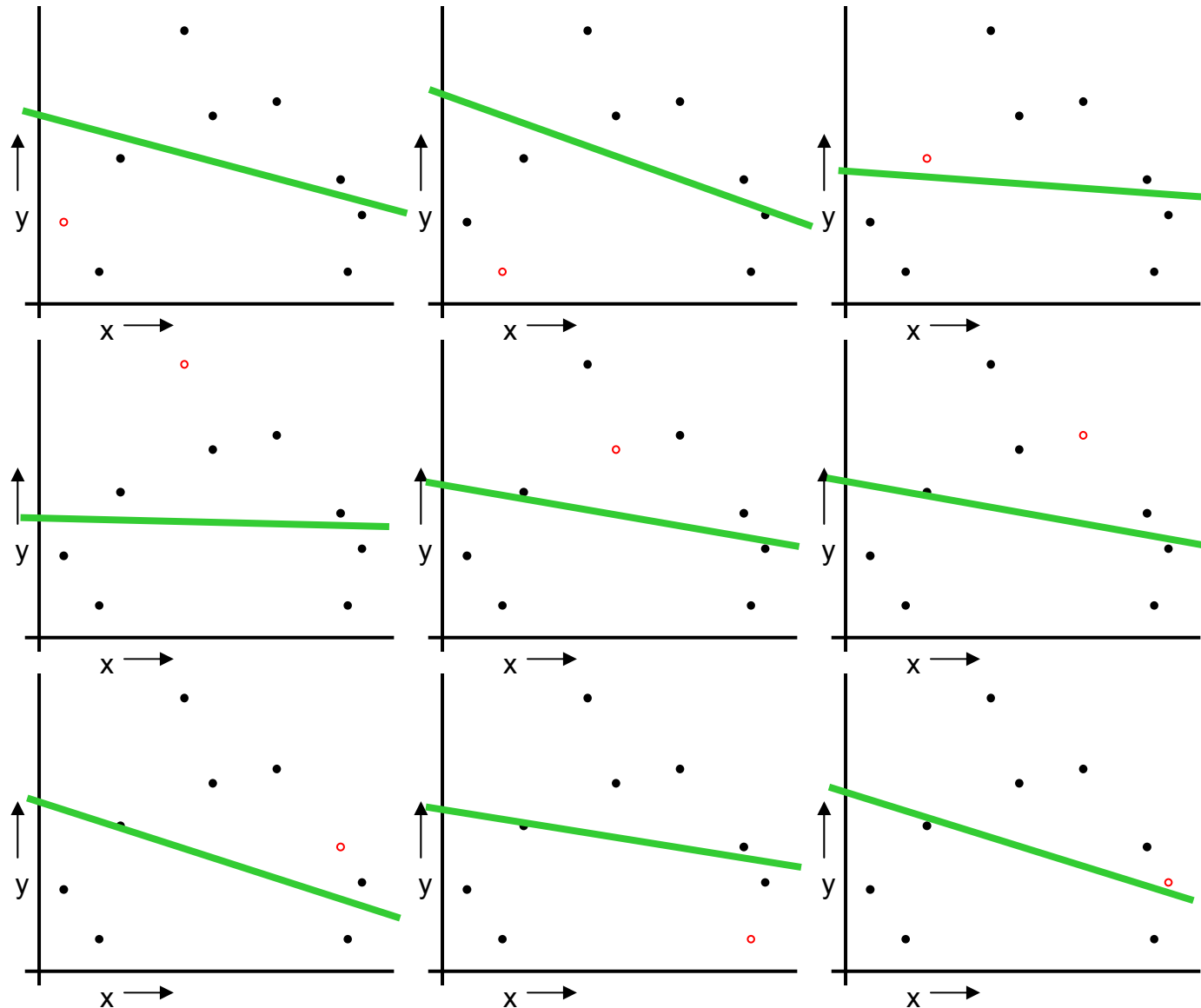
For $k=1$ to R

1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset
3. Train on the remaining $R-1$ datapoints
4. Note your error (x_k, y_k)

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



LOOCV (Leave-one-out Cross Validation)



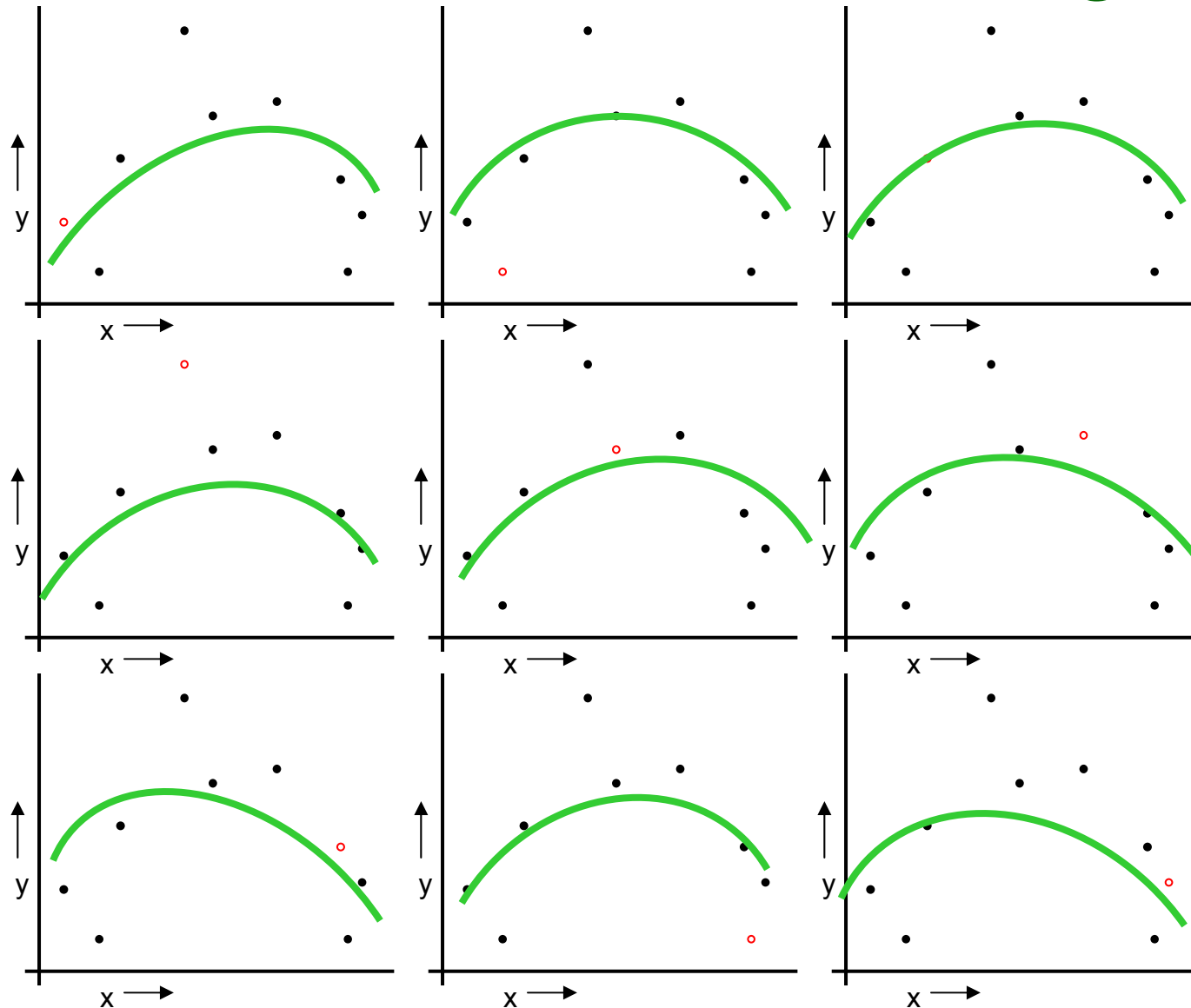
For $k=1$ to R

1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset
3. Train on the remaining $R-1$ datapoints
4. Note your error (x_k, y_k)

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

$$MSE_{LOOCV} = 2.12$$

LOOCV for Quadratic Regression



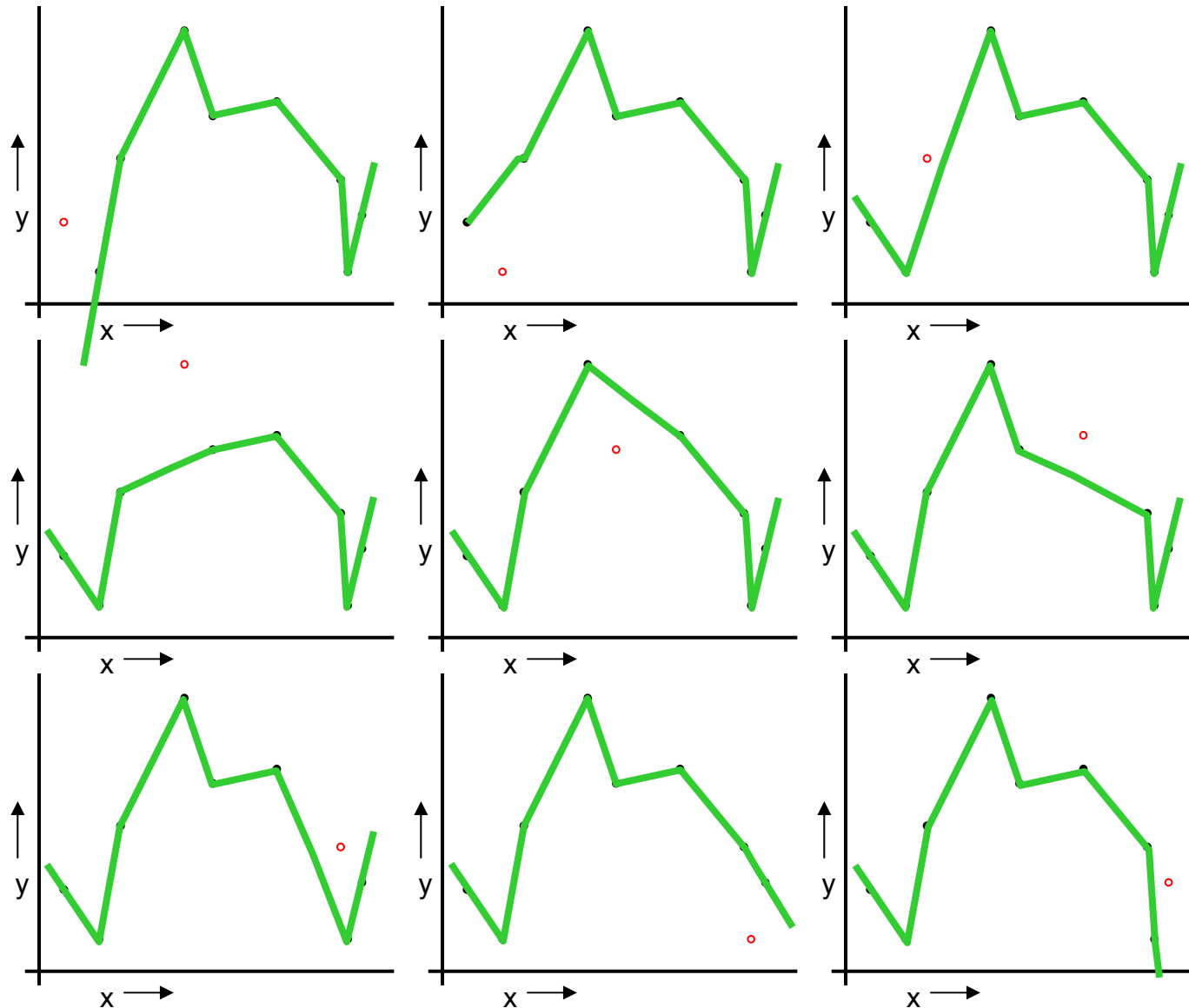
For $k=1$ to R

1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset
3. Train on the remaining $R-1$ datapoints
4. Note your error (x_k, y_k)

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

$$MSE_{\text{LOOCV}} = 0.962$$

LOOCV for Join The Dots



For $k=1$ to R

1. Let (x_k, y_k) be the k^{th} record
2. Temporarily remove (x_k, y_k) from the dataset
3. Train on the remaining $R-1$ datapoints
4. Note your error (x_k, y_k)

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

$$MSE_{LOOCV} = 3.33$$

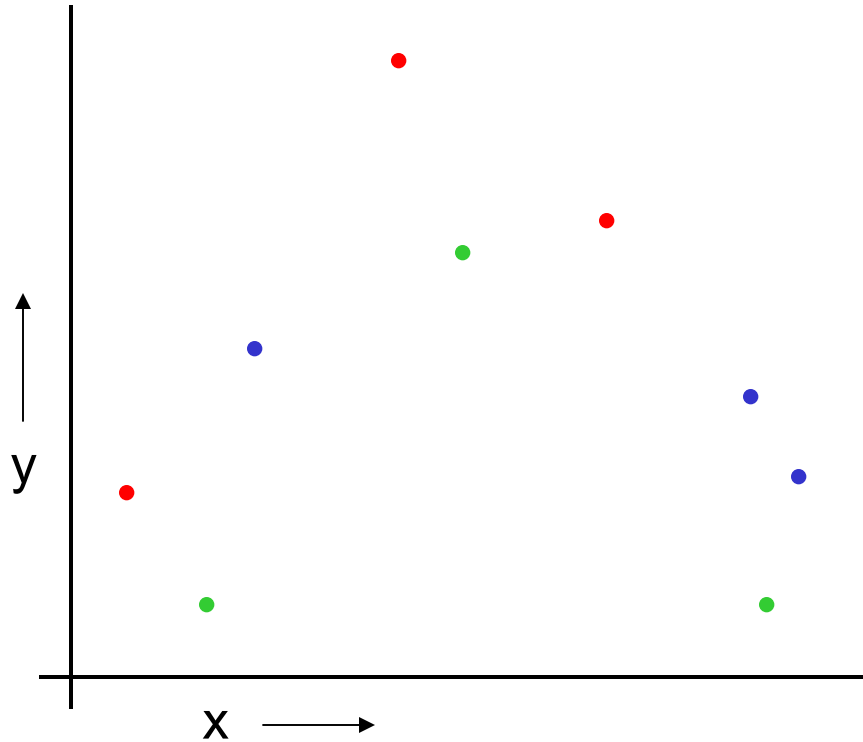
Which kind of Cross Validation?

	Downside	Upside
Test-set	Variance: unreliable estimate of future performance	Cheap
Leave-one-out	Expensive. Has some weird behavior	Doesn't waste data

..can we get the best of both worlds?

k-fold Cross Validation

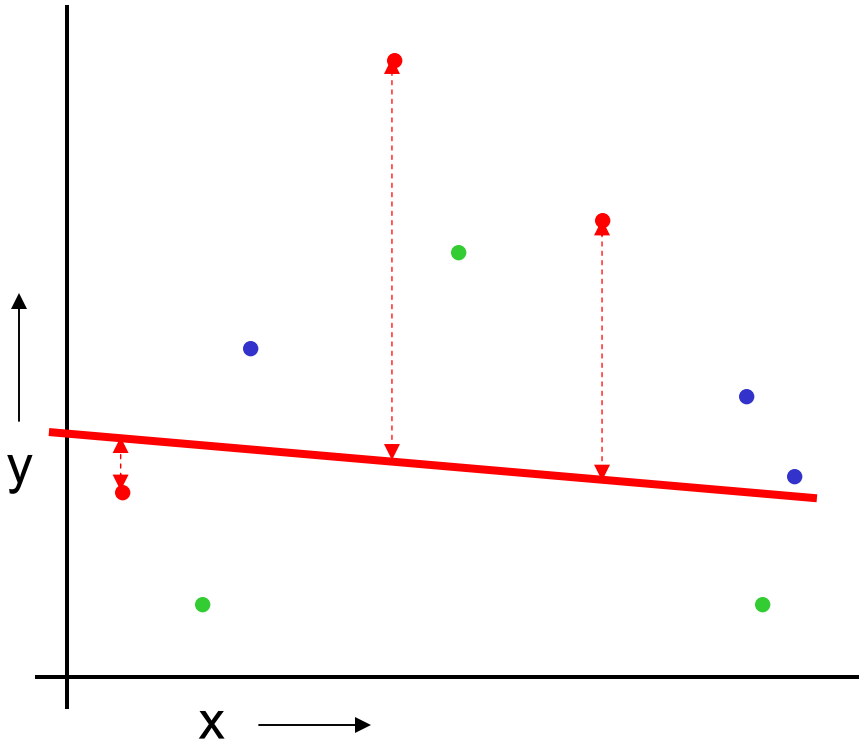
Randomly break the dataset into k partitions (in our example we'll have $k=3$ partitions colored Red Green and Blue)



k-fold Cross Validation

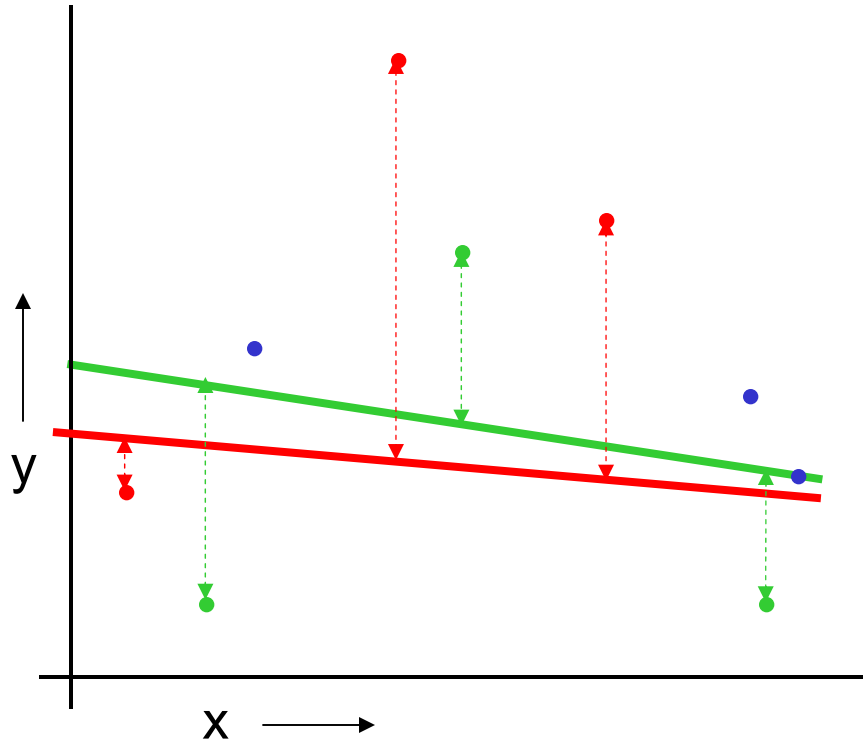
Randomly break the dataset into k partitions (in our example we'll have $k=3$ partitions colored Red Green and Blue)

For the red partition: Train on all the points not in the red partition. Find the test-set sum of errors on the red points.



k-fold Cross Validation

Randomly break the dataset into k partitions (in our example we'll have $k=3$ partitions colored Red Green and Blue)

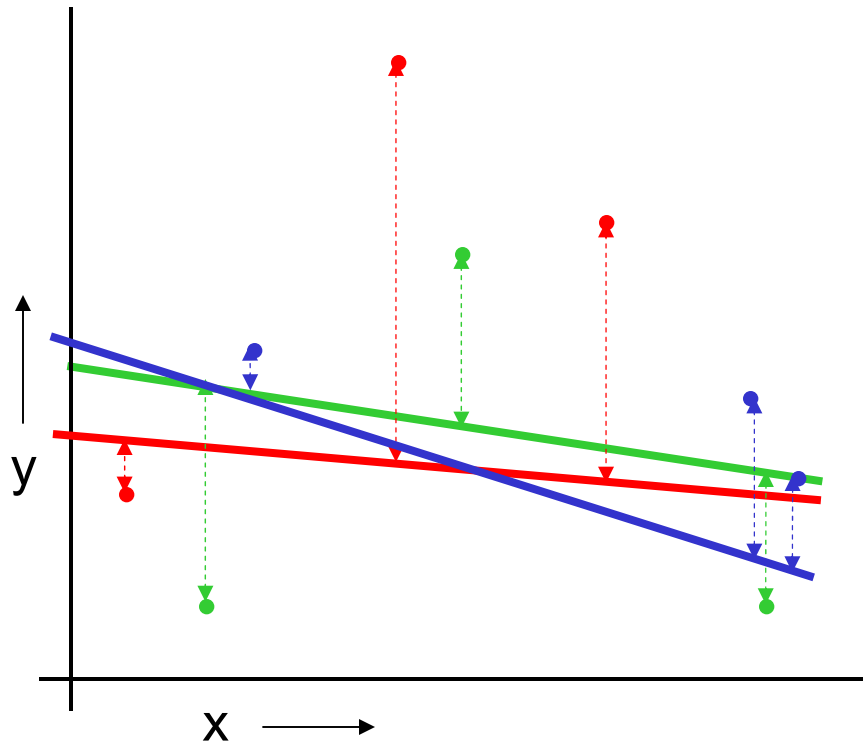


For the red partition: Train on all the points not in the red partition. Find the test-set sum of errors on the red points.

For the green partition: Train on all the points not in the green partition. Find the test-set sum of errors on the green points.

k-fold Cross Validation

Randomly break the dataset into k partitions (in our example we'll have $k=3$ partitions colored Red Green and Blue)



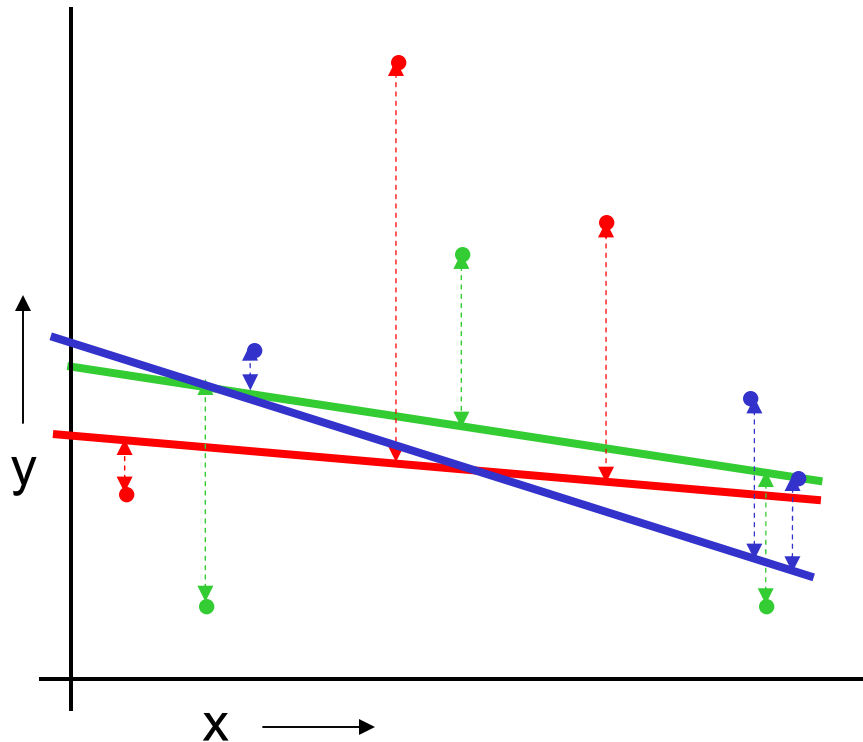
For the red partition: Train on all the points not in the red partition. Find the test-set sum of errors on the red points.

For the green partition: Train on all the points not in the green partition. Find the test-set sum of errors on the green points.

For the blue partition: Train on all the points not in the blue partition. Find the test-set sum of errors on the blue points.

k-fold Cross Validation

Randomly break the dataset into k partitions (in our example we'll have k=3 partitions colored Red Green and Blue)



Linear Regression

$$MSE_{3FOLD} = 2.05$$

For the red partition: Train on all the points not in the red partition. Find the test-set sum of errors on the red points.

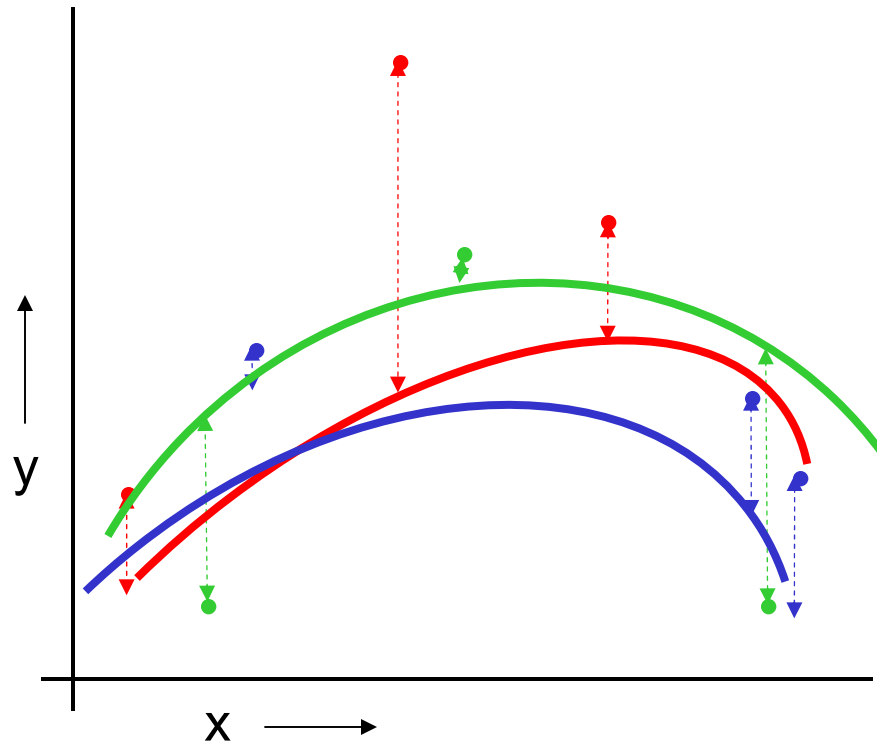
For the green partition: Train on all the points not in the green partition. Find the test-set sum of errors on the green points.

For the blue partition: Train on all the points not in the blue partition. Find the test-set sum of errors on the blue points.

Then report the mean error

k-fold Cross Validation

Randomly break the dataset into k partitions (in our example we'll have k=3 partitions colored Red Green and Blue)



For the red partition: Train on all the points not in the red partition. Find the test-set sum of errors on the red points.

For the green partition: Train on all the points not in the green partition. Find the test-set sum of errors on the green points.

For the blue partition: Train on all the points not in the blue partition. Find the test-set sum of errors on the blue points.

Quadratic Regression

$$MSE_{3FOLD} = 1.11$$

Then report the mean error

k-fold Cross Validation

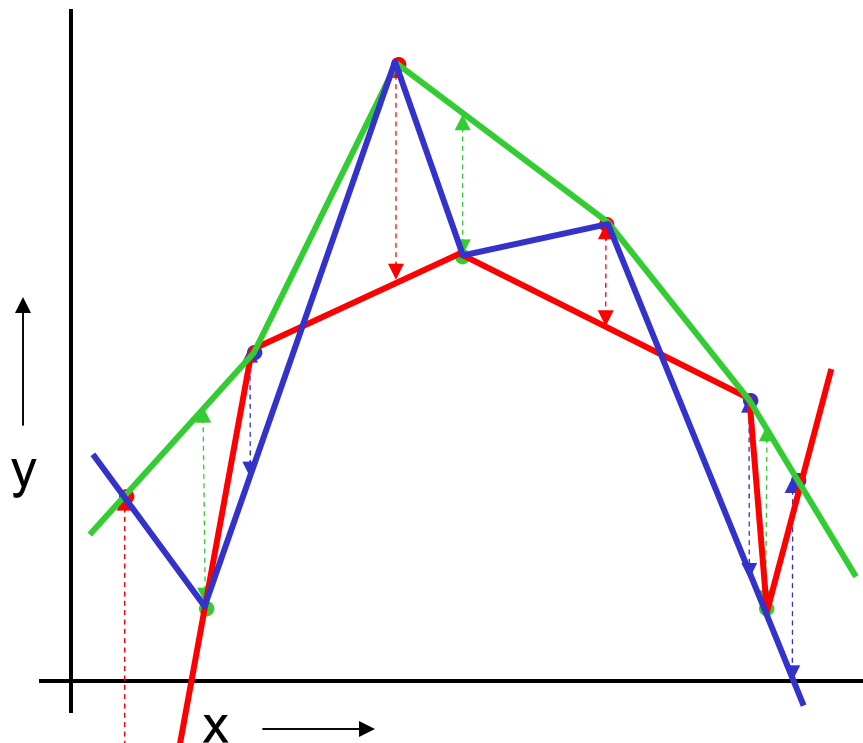
Randomly break the dataset into k partitions (in our example we'll have k=3 partitions colored Red Green and Blue)

For the red partition: Train on all the points not in the red partition. Find the test-set sum of errors on the red points.

For the green partition: Train on all the points not in the green partition. Find the test-set sum of errors on the green points.

For the blue partition: Train on all the points not in the blue partition. Find the test-set sum of errors on the blue points.

Then report the mean error
















Joint-the-dots
 $MSE_{3FOLD}=2.93$

Which kind of Cross Validation?

	Downside	Upside
Test-set	Variance: unreliable estimate of future performance	Cheap
Leave-one-out	Expensive. Has some weird behavior	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 R times.
3-fold	Wastier than 10-fold. Expensivier than test set	Slightly better than test-set
R-fold	Identical to Leave-one-out	













CV-based Model Selection

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

i	f_i	TRAINERR	10-FOLD-CV-ERR	Choice
1	f_1			
2	f_2			
3	f_3			
4	f_4			
5	f_5			
6	f_6			

CV-based Model Selection












- Example: Choosing number of hidden units in a one-hidden-layer neural net.
- Step 1: Compute 10-fold CV error for six different model classes:

Algorithm	TRAINERR	10-FOLD-CV-ERR	Choice
0 hidden units			
1 hidden units			
2 hidden units			ⓧ
3 hidden units			
4 hidden units			
5 hidden units			

- Step 2: Whichever model class gave best CV score: train it with all the data, and that's the predictive model you'll use.

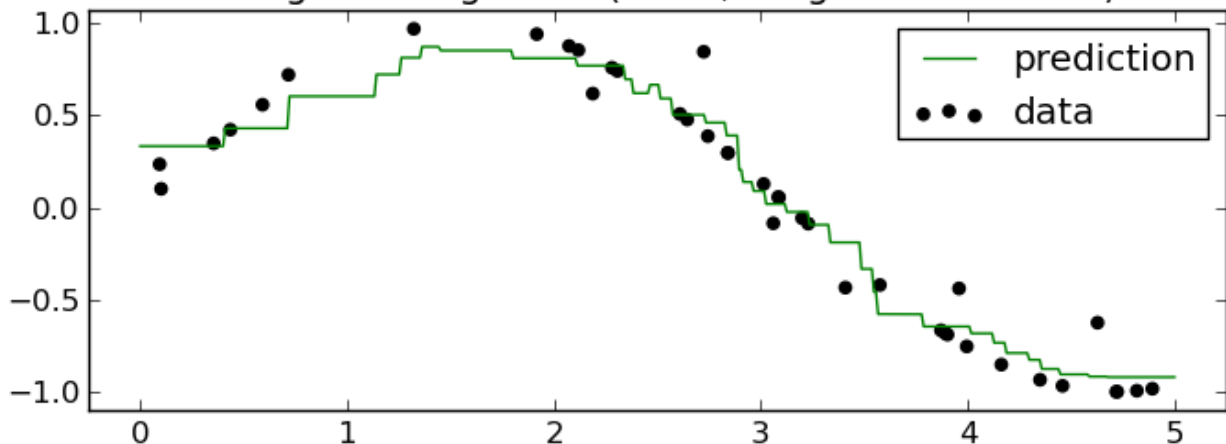
CV-based Model Selection

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

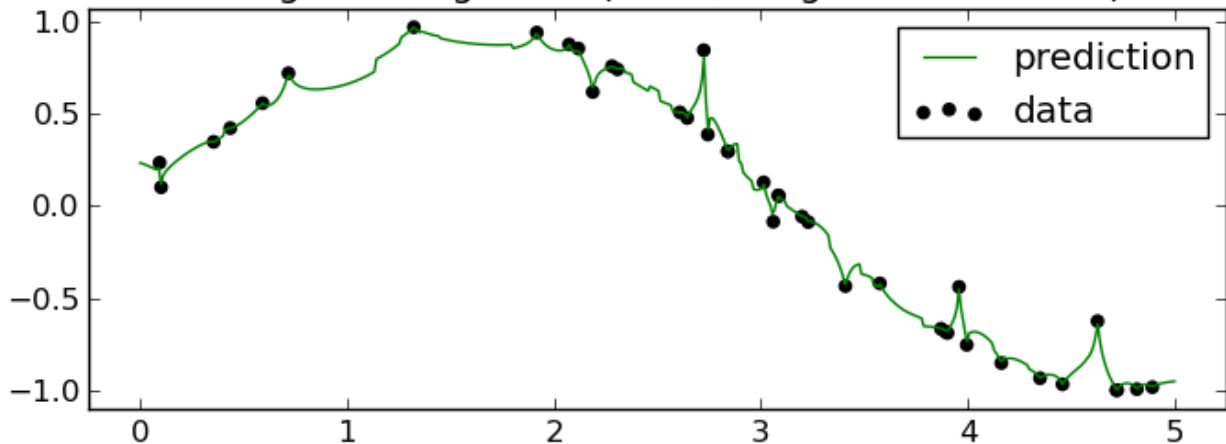
Algorithm	TRAINERR	10-fold-CV-ERR	Choice
$K=1$			
$K=2$			
$K=3$			
$K=4$			⊠
$K=5$			
$K=6$			

- Step 2: Whichever model class gave best CV score: train it with all the data, and that's the predictive model you'll use.

KNeighborsRegressor (k = 5, weights = 'uniform')



KNeighborsRegressor (k = 5, weights = 'distance')



CV-based Model Selection

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

Algorithm	Training Error
$K=1$	
$K=2$	
$K=3$	
$K=4$	
$K=5$	
$K=6$	

Why did we use 10-fold-CV for neural nets and LOOCV for k-nearest neighbor?

And why stop at $K=6$

Are we guaranteed that a local optimum of K vs LOOCV will be the global optimum?

What should we do if we are depressed at the expense of doing LOOCV for $K=1$ through 1000?

The reason is Computational. For k-NN (and all other nonparametric methods) LOOCV happens to be as cheap as regular predictions.

No good reason, except it looked like things were getting worse as K was increasing

Sadly, no. And in fact, the relationship can be very bumpy.













Idea One: $K=1, K=2, K=4, K=8, K=16, K=32, K=64 \dots K=1024$

Idea Two: Hillclimbing from an initial guess at K

- Step 2: Whichever model class gave best CV score: train it with all the data, and that's the predictive model you'll use.

CV-based Algorithm Choice

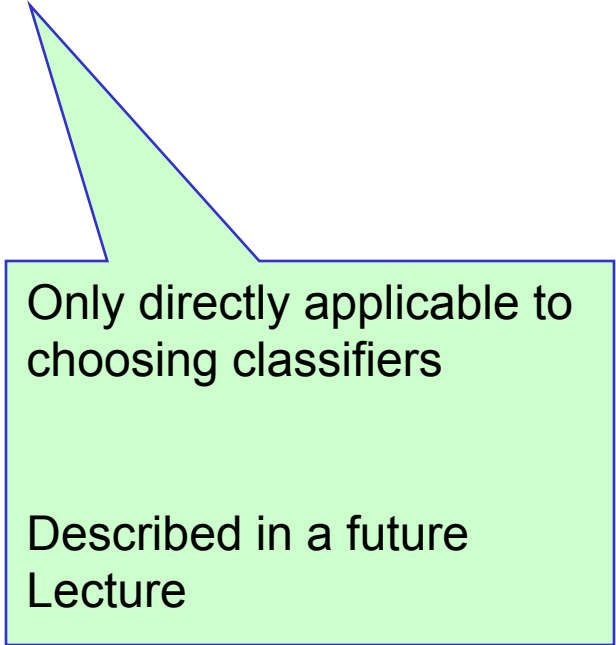
- Example: Choosing which regression algorithm to use
- Step 1: Compute 10-fold-CV error for six different model classes:

Algorithm	TRAINERR	10-fold-CV-ERR	Choice
1-NN			
10-NN			
Linear Reg'n			
Quad reg'n			
LWR, KW=0.1			
LWR, KW=0.5			

- Step 2: Whichever algorithm gave best CV score: train it with all the data, and that's the predictive model you'll use.

Alternatives to CV-based model selection

- Model selection methods:
 1. Cross-validation
 2. AIC (Akaike Information Criterion)
 3. BIC (Bayesian Information Criterion)
 4. VC-dimension (Vapnik-Chervonenkis Dimension)



Only directly applicable to
choosing classifiers

Described in a future
Lecture

Which model selection method is best?

1. (CV) Cross-validation
 2. AIC (Akaike Information Criterion)
 3. BIC (Bayesian Information Criterion)
 4. (SRMVC) Structural Risk Minimize with VC-dimension
- AIC, BIC and SRMVC advantage: you only need the training error.
 - CV error might have more variance
 - SRMVC is wildly conservative
 - Asymptotically AIC and Leave-one-out CV should be the same
 - Asymptotically BIC and carefully chosen k-fold should be same
 - You want BIC if you want the best structure instead of the best predictor (e.g. for clustering or Bayes Net structure finding)
 - Many alternatives---including proper Bayesian approaches.
 - It's an emotional issue.

Other Cross-validation issues

- Can do “leave all pairs out” or “leave-all-ntuples-out” if feeling resourceful.
- Some folks do k-folds in which each fold is an independently-chosen subset of the data
- Do you know what AIC and BIC are?

If so...

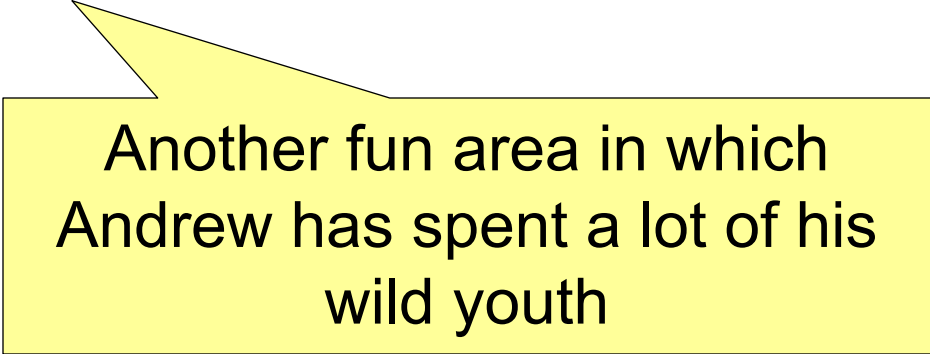
- LOOCV behaves like AIC asymptotically.
- k-fold behaves like BIC if you choose k carefully

If not...

- Nyardely nyardely nyoo nyoo

Feature Selection

- Suppose you have a learning algorithm LA and a set of input attributes $\{X_1, X_2 \dots X_m\}$
- You expect that LA will only find some subset of the attributes useful.
- Question: How can we use cross-validation to find a useful subset?
- Four ideas:
 - Forward selection
 - Backward elimination
 - Hill Climbing
 - Stochastic search (Simulated Annealing or GAs)



Another fun area in which
Andrew has spent a lot of his
wild youth

Very serious warning

- Intensive use of cross validation can overfit.
- How?

- What can be done about it?

Very serious warning

- Intensive use of cross validation can overfit.
- How?
 - Imagine a dataset with 50 records and 1000 attributes.
 - You try 1000 linear regression models, each one using one of the attributes.
- What can be done about it?

Very serious warning

- Intensive use of cross validation can overfit.
- How?
 - Imagine a dataset with 50 records and 1000 attributes.
 - You try 1000 linear regression models, each one using one of the attributes.
 - The best of those 1000 looks good!
- What can be done about it?

Very serious warning

- Intensive use of cross validation can overfit.
- How?
 - Imagine a dataset with 50 records and 1000 attributes.
 - You try 1000 linear regression models, each one using one of the attributes.
 - The best of those 1000 looks good!
 - But you realize it would have looked good even if the output had been purely random!
- What can be done about it?
 - Hold out an additional testset before doing any model selection. Check the best model performs well even on the additional testset.
 - Or: Randomization Testing

What you should know

- Why you can't use "training-set-error" to estimate the quality of your learning algorithm on your data.
- Why you can't use "training set error" to choose the learning algorithm
- Test-set cross-validation
- Leave-one-out cross-validation
- k-fold cross-validation
- Feature selection methods
- CV for classification, regression & densities