

Model Selection and Tuning

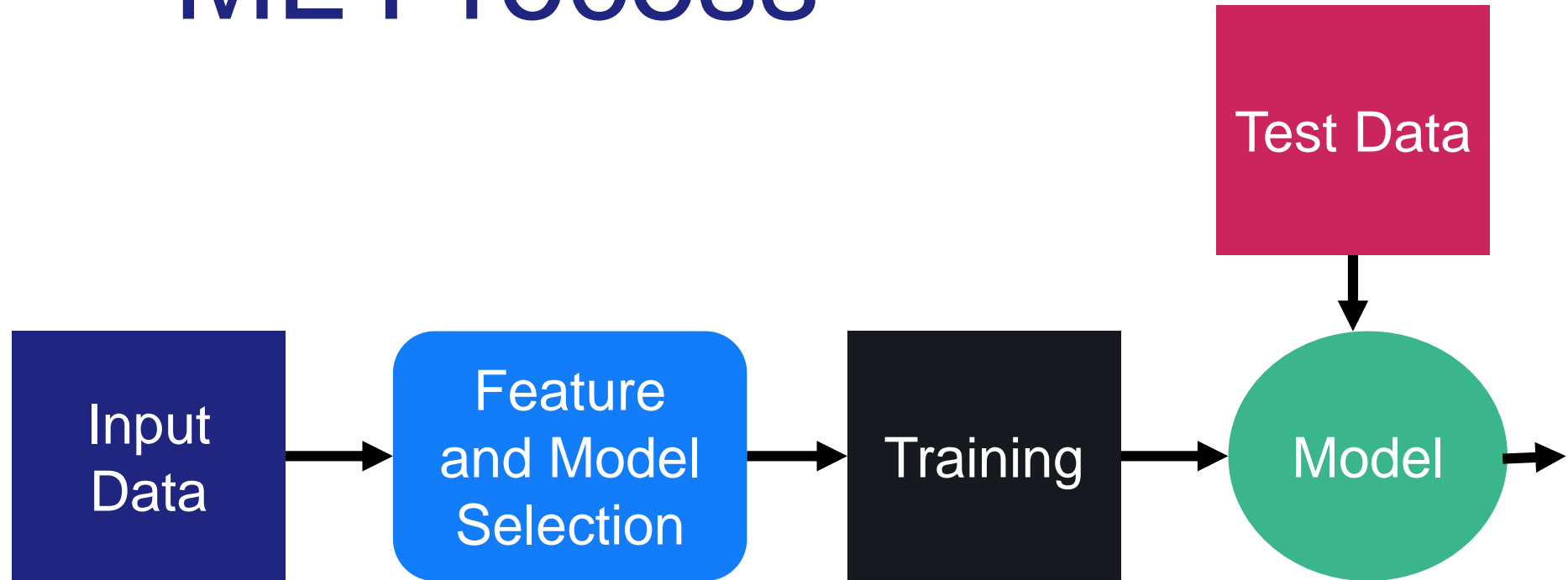
Outline

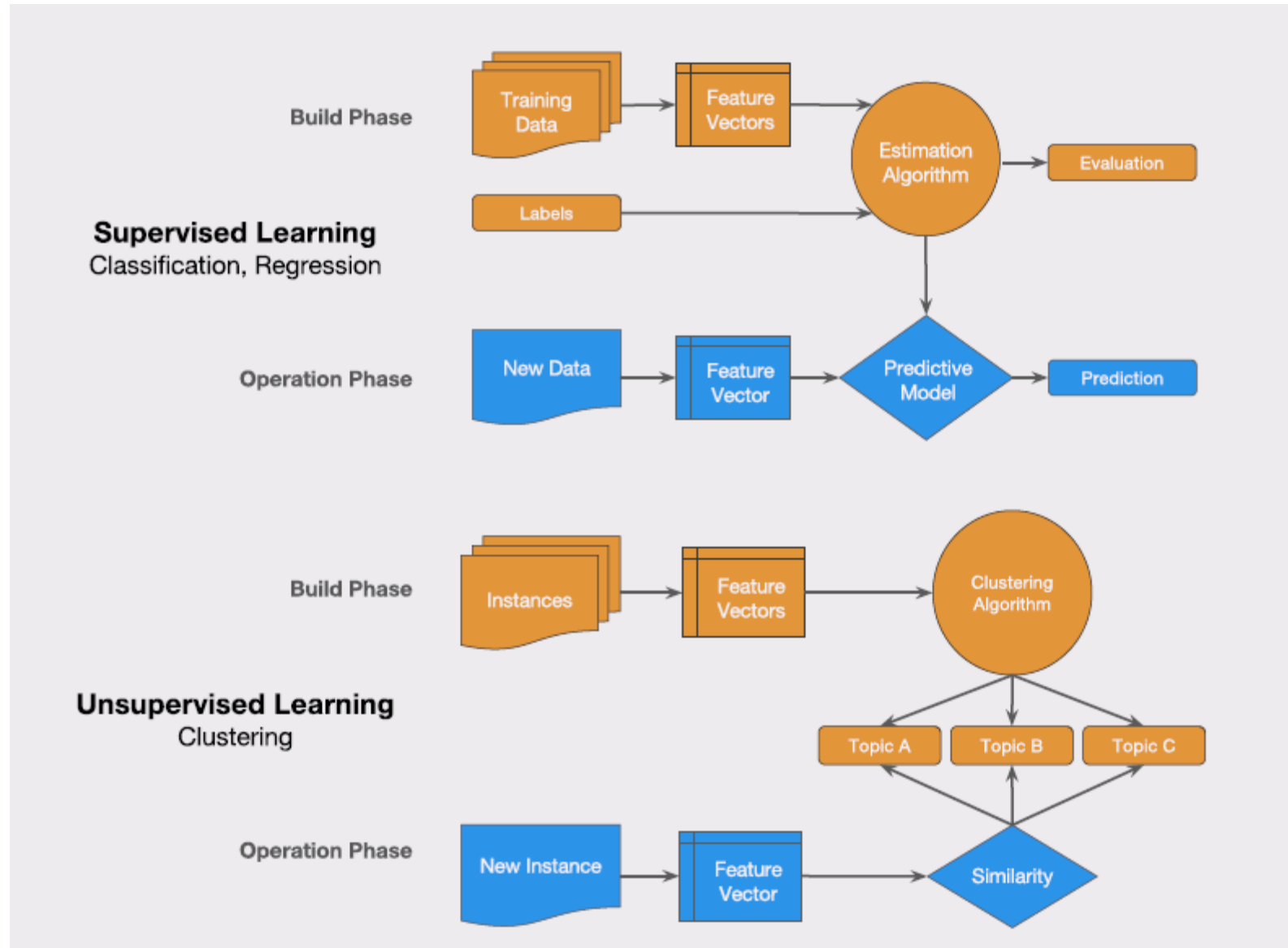
- Model Selection
- Training Set Error vs Test Set Error
- LOOCV
- K-fold Cross Validation
- ROC or Receiver Operating Curve

What is Model Selection?

- Given a set of models $M = \{M_1, M_2, \dots, M_R\}$, choose the model that is expected to do the best on the test data.
- M may consist of
 - Same learning model with different complexities or hyperparameters
 - Nonlinear Regression: Polynomials with different degrees
 - K-Nearest Neighbors: Different choices of K
 - Decision Trees: Different choices of the number of levels / leaves
 - ... and almost any learning model

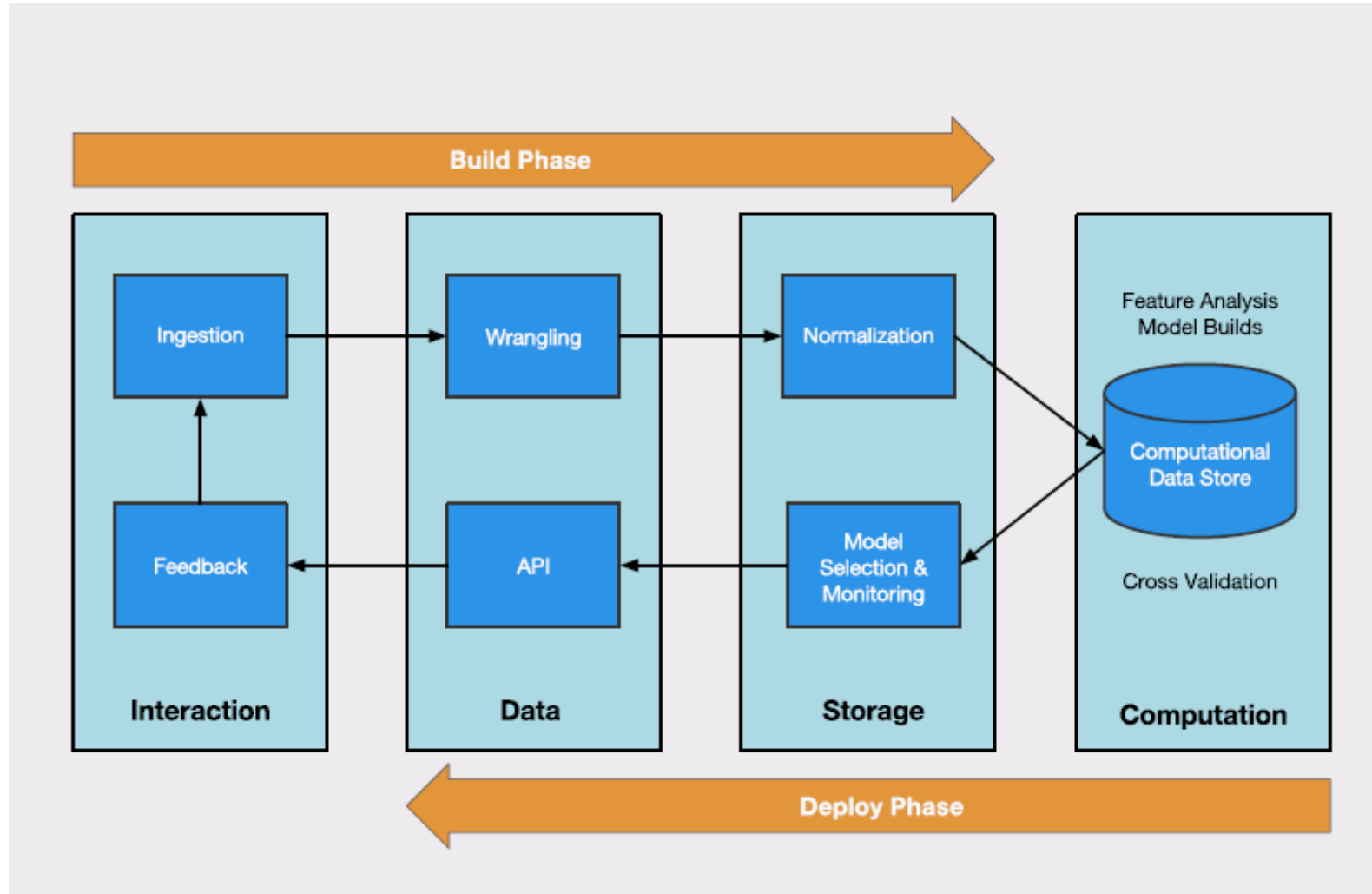
ML Process

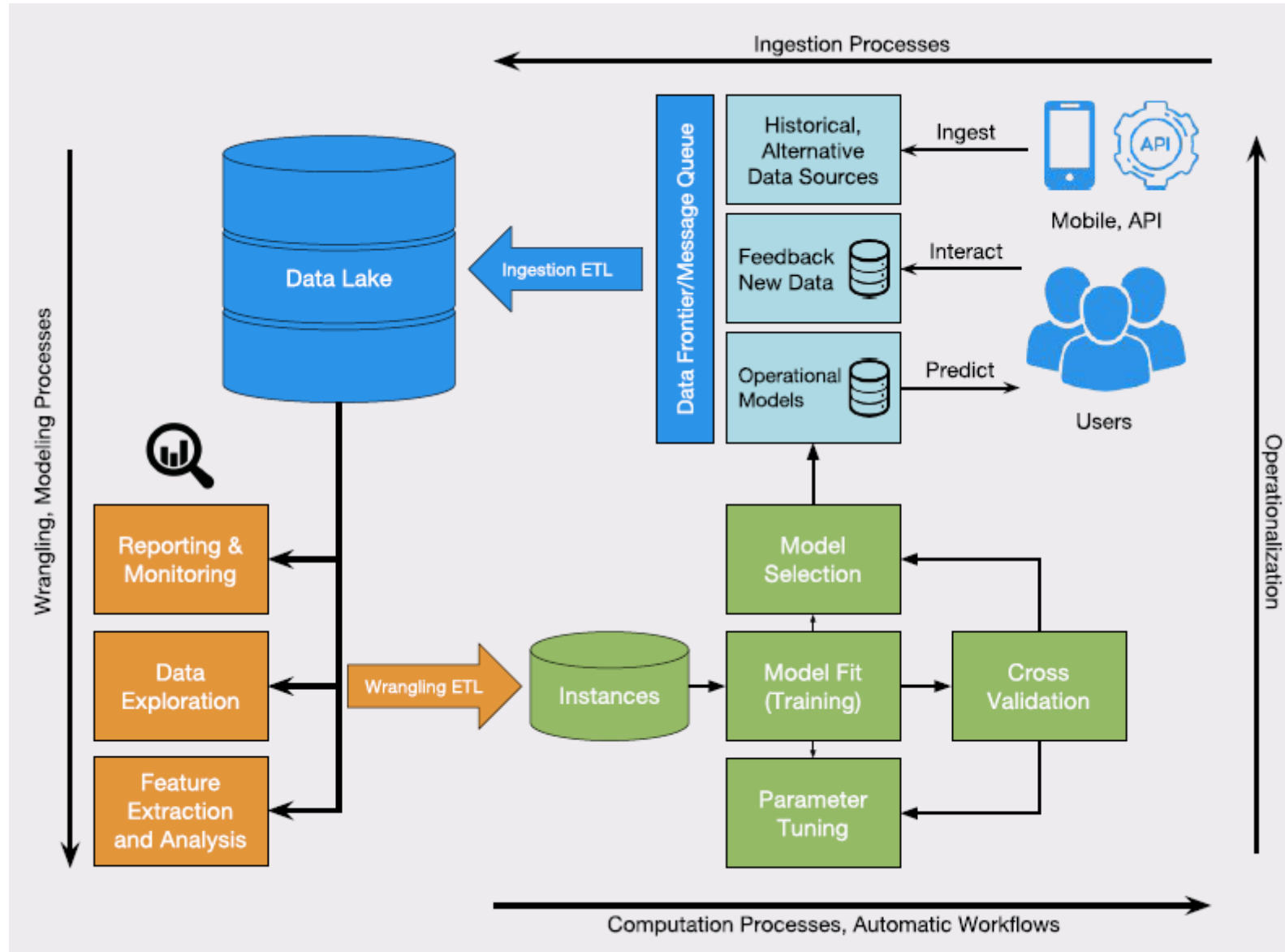




Without Feedback Models are Disconnected

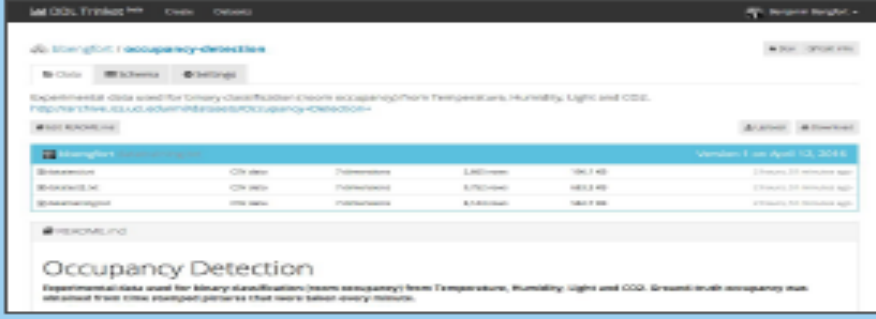
They cannot adapt, tune, or react.





Data Management

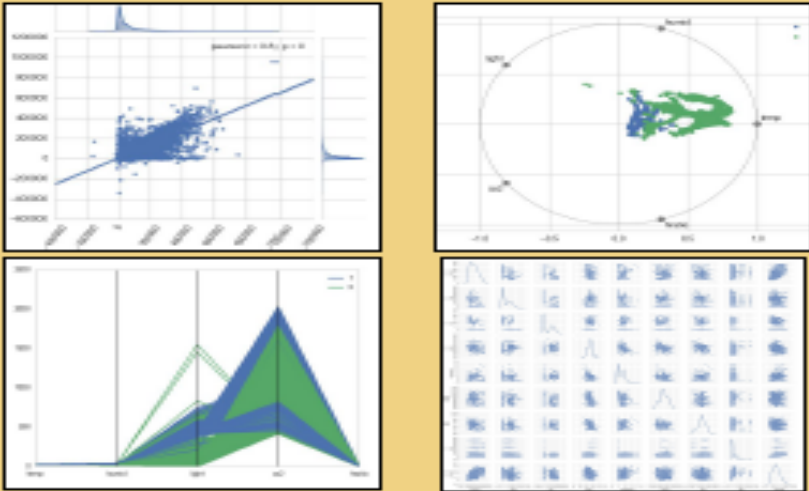
Wrangling
Standardization
Normalization
Selection & Joins



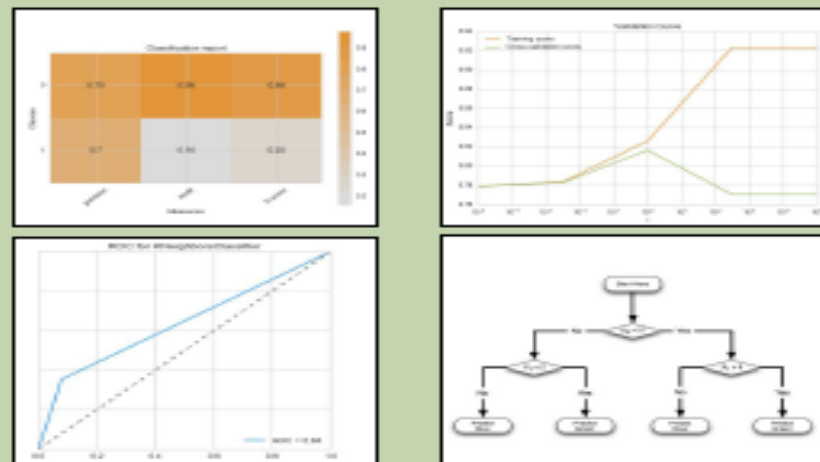
Occupancy Detection

Experimental data used for binary classification (from occupancy) from Temperature, Humidity, Light and CO2. Ground truth occupancy was obtained from raw timestamped data that were taken every minute.

Feature Analysis




Model Evaluation + Hyperparameter Tuning



Model Selection

Linear Models Nearest Neighbors SVM

Ensemble Trees Bayes



Feature Analysis

Feature Selection

Model Storage

Initial Model

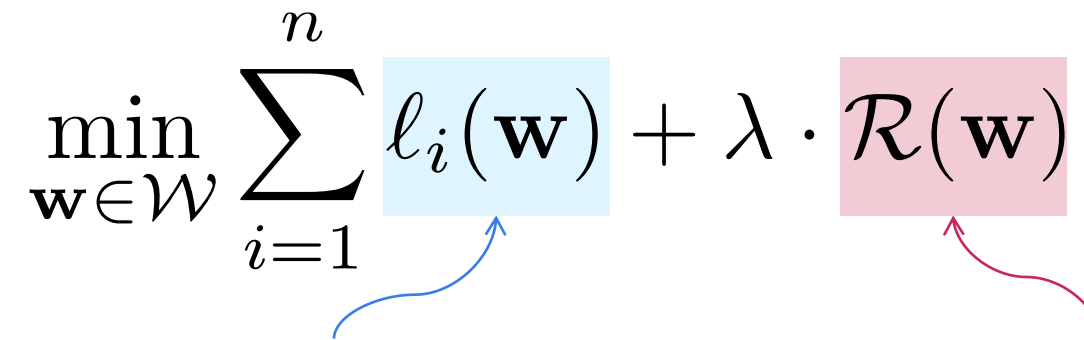
Iterate!

Revisit Features

Model Selection

Why Optimization?

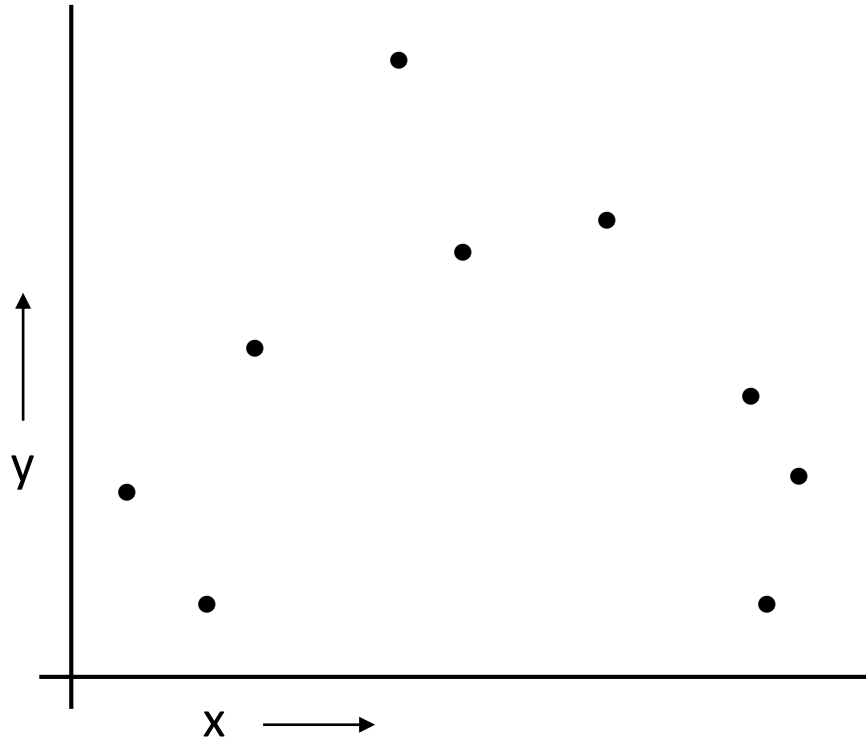
OPT at the heart of ML

$$\min_{\mathbf{w} \in \mathcal{W}} \sum_{i=1}^n \ell_i(\mathbf{w}) + \lambda \cdot \mathcal{R}(\mathbf{w})$$
The equation is displayed with a light blue box around the loss term $\ell_i(\mathbf{w})$ and a light red box around the regularization term $\mathcal{R}(\mathbf{w})$. A blue arrow points from the blue box to the text 'Measures model fit for data point i (avoids under-fitting)'. A red arrow points from the red box to the text 'Measures model “complexity” (avoids over-fitting)'.

Measures model fit
for data point i
(*avoids under-fitting*)

Measures model
“complexity”
(*avoids over-fitting*)

A Regression Problem

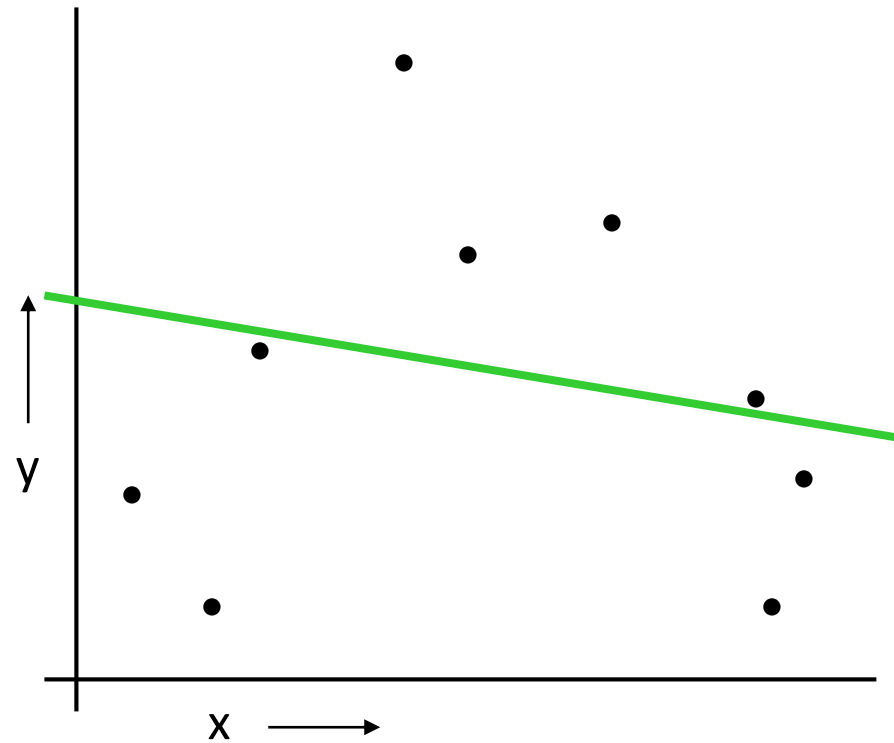


$$y = f(x) + \text{noise}$$

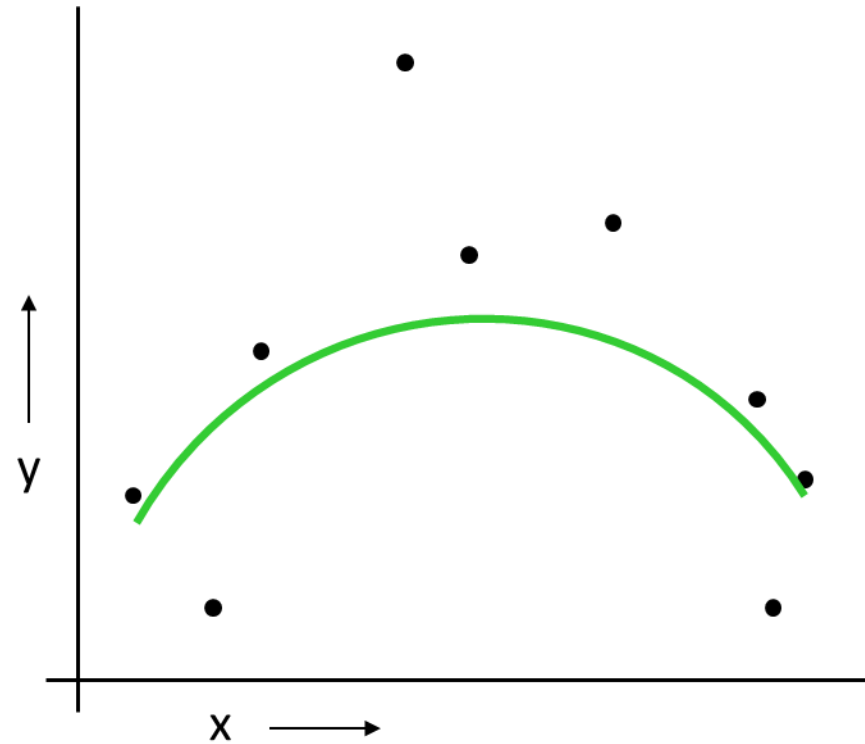
Can we learn f from this data?

Let's consider three methods...

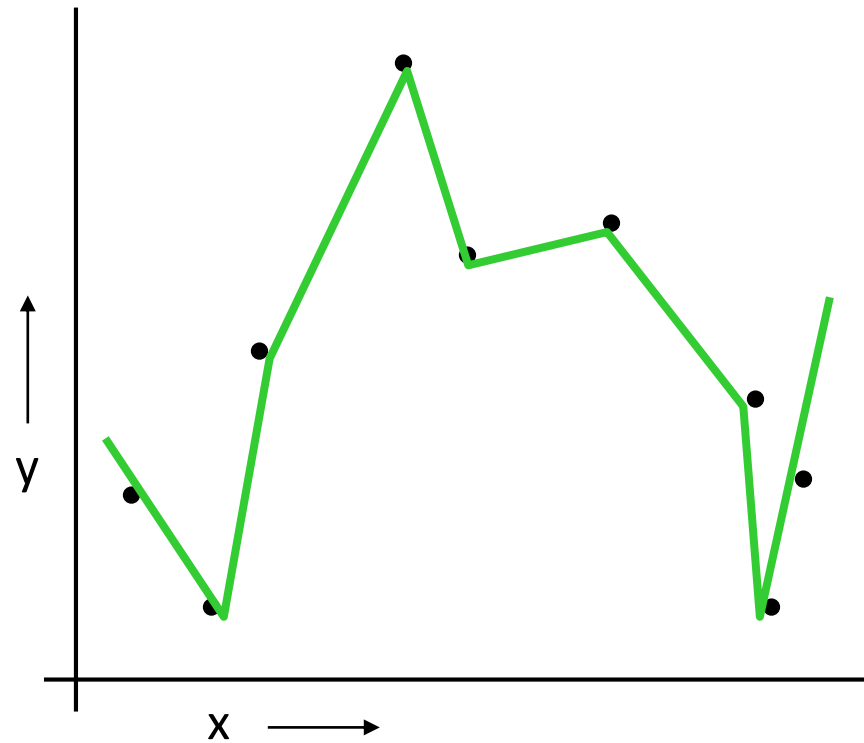
First Method: Linear Regression



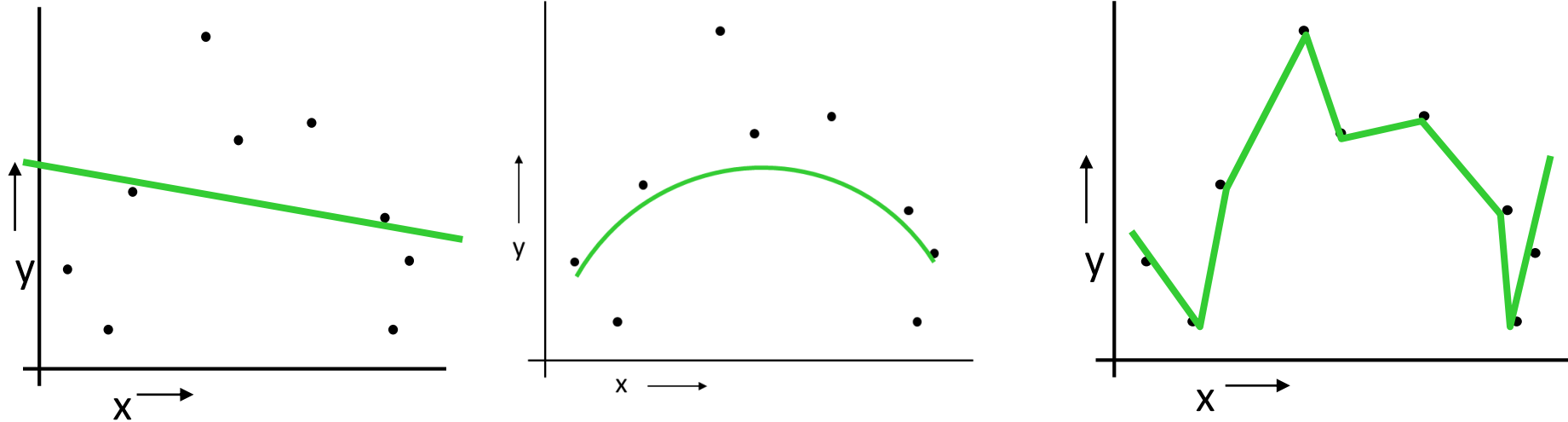
Second Method: Quadratic Regression



Third Method: Join the Dots

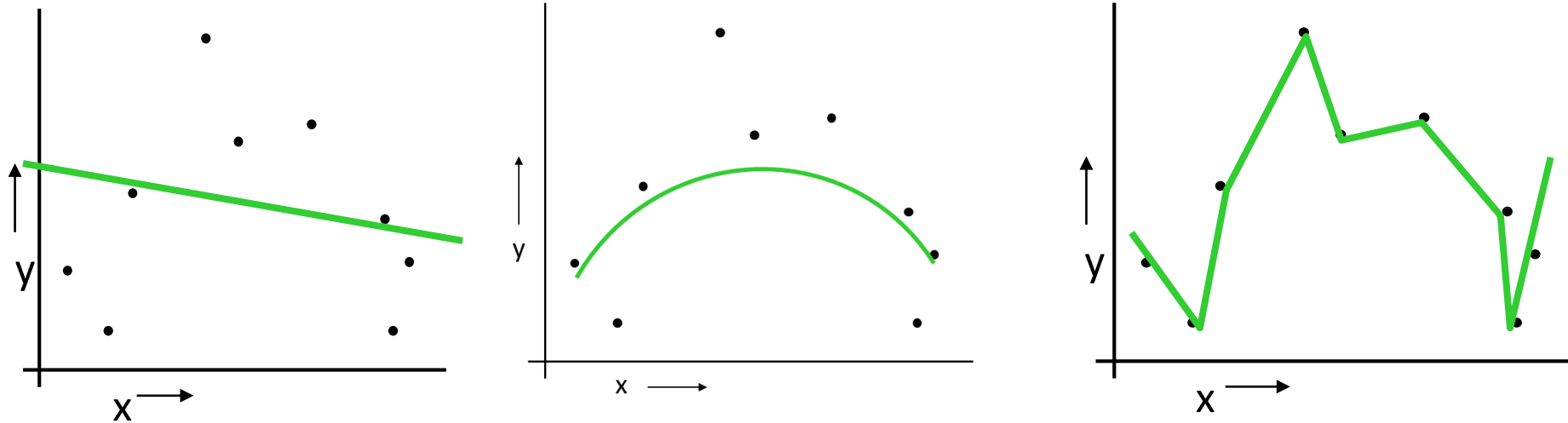


Which is best?



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

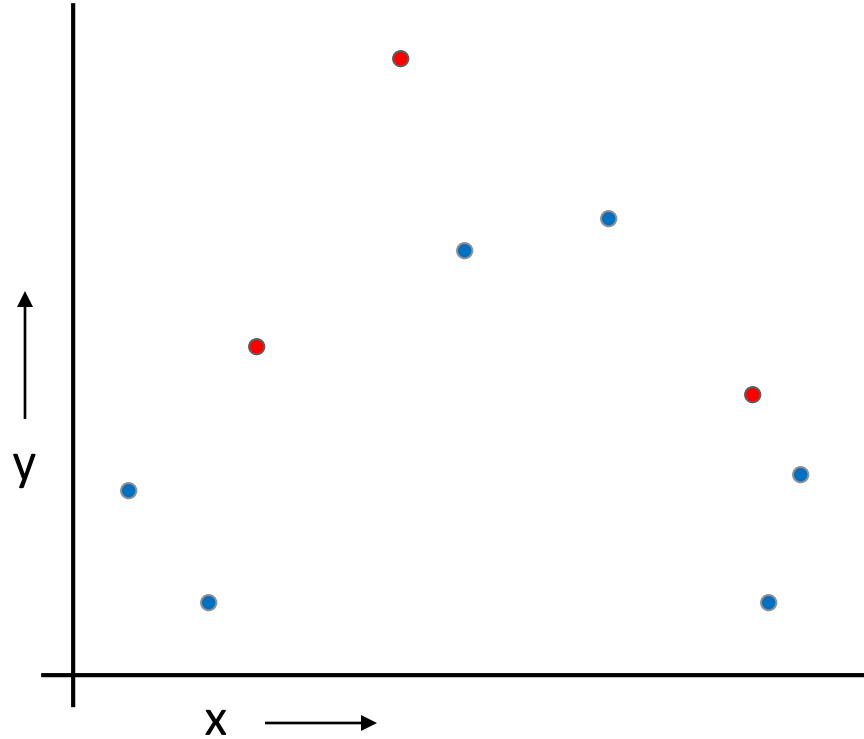
Which is best?



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

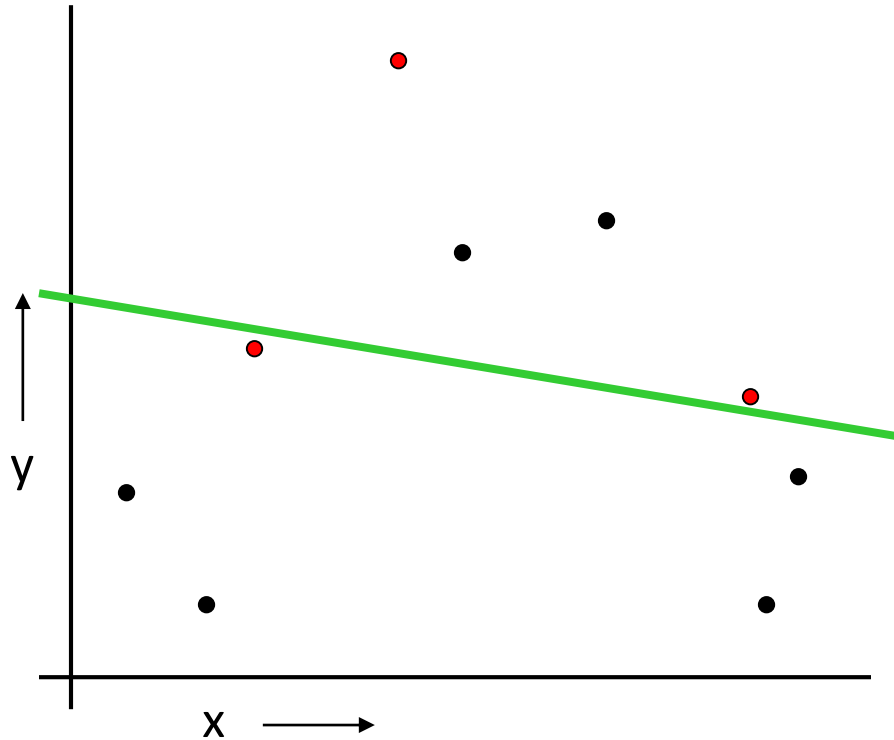
“How well are you going to predict future data drawn from the same distribution?”

The Test Set Method



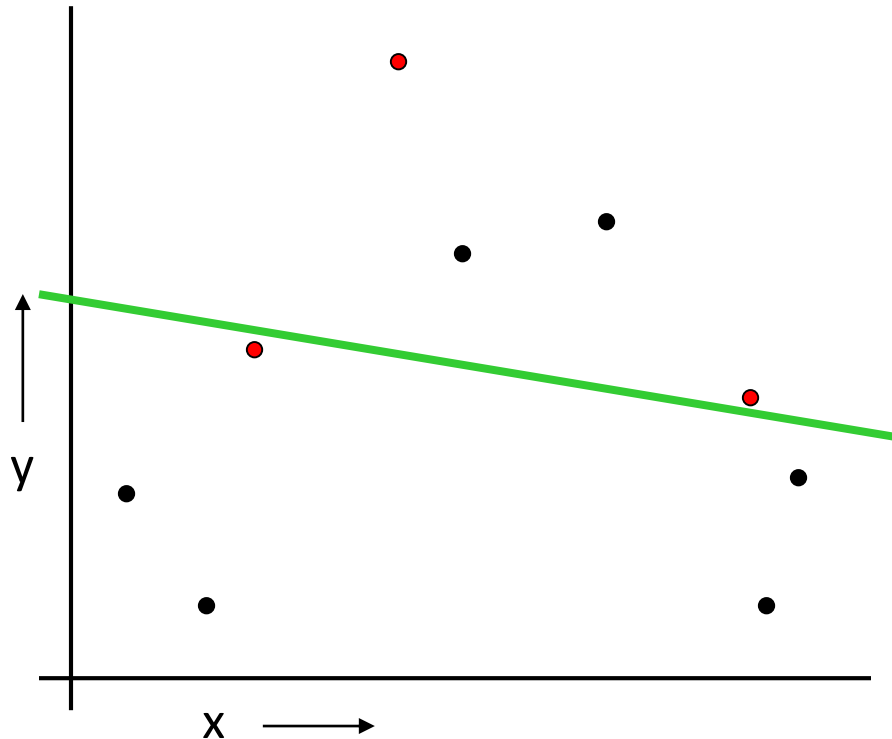
1. Randomly choose 30% of the data to be in a **test set**
2. The remainder is a **training set**

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**

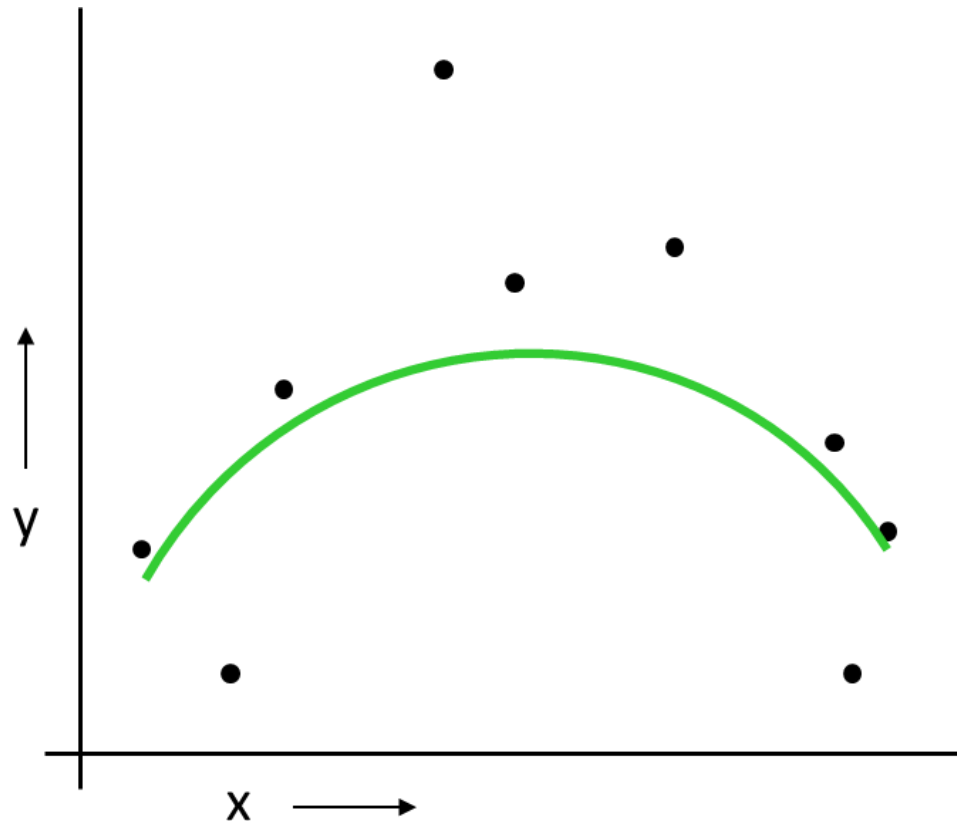
The Test Set Method



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**
4. Estimate your future performance with the test set

(Linear regression example)
Mean Squared Error = 2.4

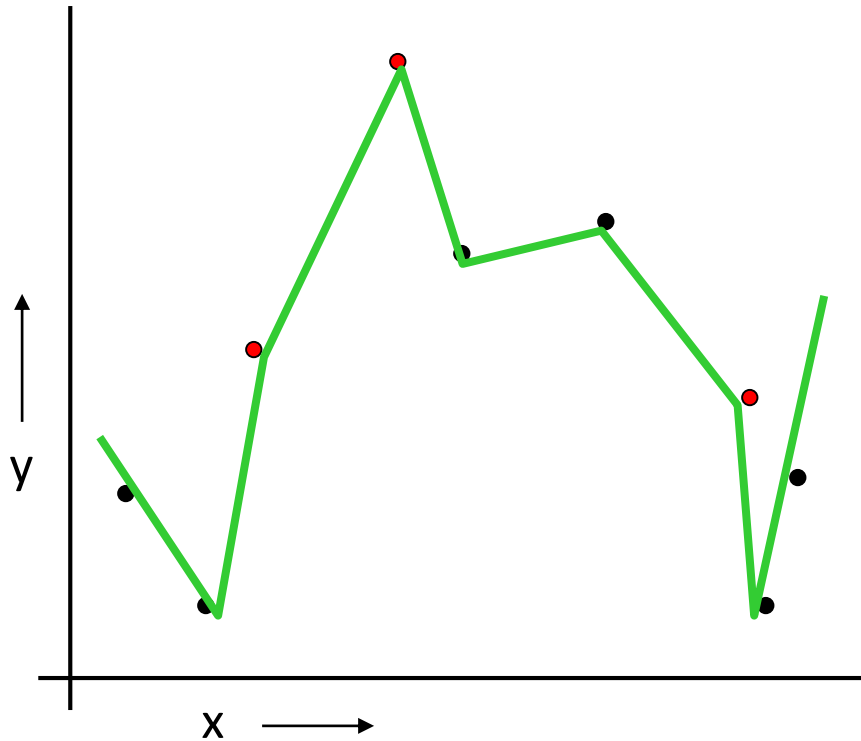
The Test Set Method



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**
4. Estimate your future performance with the test set

(Quadratic regression example)
Mean Squared Error = 0.9

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**
4. Estimate your future performance with the test set

(Join the Dots example)
Mean Squared Error = 2.2

The Test Set Method

- Good News
 - Very Simple
 - Can we then choose the method with the best test score?
- Is there a downside?

The Test Set Method

- Good News
 - Very Simple
 - Can we then choose the method with the best test score?
- Is there a downside?
 - Yes. It wastes 30% of data which can be critical when there is sparsity of data.
- What are the alternatives?
 - Validation set approach
 - LOOCV or Leave One Out Cross Validation
 - K-fold cross validation

Resampling methods

- Cross –Validation
 - Used to estimate the test error associated with a given statistical learning method in order to evaluate its performance, or to select the appropriate level of flexibility
 - Model assessment: The process of evaluating a model's performance
 - Model selection: The process of selecting the proper level of flexibility for a model
- Bootstrap
 - Provide a measure of accuracy of a parameter estimate or of a given statistical learning method

Cross-Validation

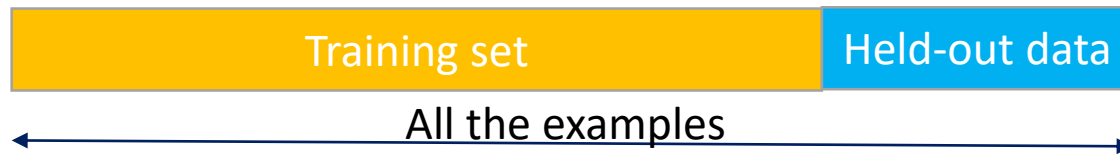
- Test error rate
 - The average error that results from using a statistical learning method to predict the response on a new observation, i.e., a measurement that was not used in training the method
 - Given a data set, the use of a particular statistical learning method is warranted if it results in a low test error
- Training error
 - Calculated by applying the statistical learning method to the observations used in its training
 - Training error rate often is quite different from the test error rate, and in particular the former can dramatically underestimate the latter

Cross-Validation

- In the absence of a very large designated test set that can be used to directly estimate the test error rate, a number of techniques can be used to estimate this quantity using the available training data
- We study a class of methods that estimate the test error rate by *holding out* a subset of the training observations from the fitting process, and then applying the statistical learning method to those held out observations

Held-out Data

- Set aside a fraction (say 10% to 20%) of the training data
- This part becomes held-out data
 - Other names: validation / development data



- Remember: Held-out data is not the test data
- Train each model using the remaining training data
- Evaluate error on the held-out data
- Choose the model with the smallest held-out error
- Problems:
 - Wastes training data, so typically used when we have plenty of training data
 - Held-out data may not be good if there was an unfortunate split

Validation Set Approach

- Suppose that we would like to estimate the test error associated with fitting a particular statistical learning method on a set of observations
- The *validation set approach* is a very simple strategy for this task
- It involves randomly dividing the available set of observations into two parts, a *training set* and a *validation set* or *hold-out set*
- The model is fit on the training set, and the fitted model is used to predict the responses for the observations in the validation set
- The resulting validation set error rate provides an estimate of the test error rate

Validation Set Approach: *Auto* data set

- Analyze Auto data set
 - There appears to be a non-linear relationship between *mpg* and *horsepower*
 - A model that predicts *mpg* using *horsepower* and *horsepower*² gives better results than a model that uses only a linear term
 - **Exercise:**
 - Load Auto data
 - Fit linear regression model
 - Fit quadratic regression model
 - Analyse the models performance

Validation Set Approach: *Auto* data set

- It is natural to wonder whether a cubic or higher-order fit might provide even better results
- This could be answered using validation set approach
- Procedure:
 - Randomly split the 392 observations into two sets
 - A training set containing 196 of the data points
 - A validation set containing the remaining 196 observations
 - The validation set error rates that result from fitting various regression models on the training sample and evaluating their performance on the validation sample, using MSE as a measure of validation set error
 - Fit different regression models with varying degrees of polynomials
 - Plot the validation set errors
 - The validation set MSE for the quadratic fit is considerably smaller than for the linear fit
 - Validation set MSE for the cubic fit is actually slightly larger than for the quadratic fit
- **This implies that including a cubic term in the regression does not lead to better prediction than simply using a quadratic term**

Validation Set Approach: *Auto* data set

- Plot MSE for different degree polynomials

Validation Set Approach: *Auto* data set

- Observe that, for the previous analysis, we randomly divided the data set into two parts
 - A training set
 - A validation set
- If we repeat the process of randomly splitting the sample set into two parts, we will get a somewhat different estimate for the test MSE
- Calculate MSEs for ten different random splits of the observations into training and validation sets

Validation Set Approach: *Auto* data set

- Plot MSE for different degree polynomials using different random sets for training and validation sets

Validation Set Approach: *Auto* data set

- All the curves indicate that the model with a quadratic term has a dramatically smaller validation set MSE than the model with only a linear term
- All ten curves indicate that there is not much benefit in including cubic or higher-order polynomial terms in the model
- It is worth noting that each of the ten curves results in a different test MSE estimate for each of the ten regression models considered
- There is no consensus among the curves as to which model results in the smallest validation set MSE
- Based on the variability among these curves, all that we can conclude with any confidence is that the linear fit is not adequate for this data

Validation Set Approach

- The validation set approach is conceptually simple and is easy to implement
- It has two potential drawbacks:
 - The validation estimate of the test error rate can be highly variable, depending on precisely which observations are included in the training set and which observations are included in the validation set
 - In the validation set approach, only a subset of the observations – those that are included in the training set rather than in the validation set – are used to fit the model. Since statistical methods tend to perform worse when trained on fewer observations, this suggests that the validation set error rate may tend to overestimate the test error rate for the model fit on the fewer observations

We now present cross-validation, a refinement of the validation set approach that addresses these two issues

Leave-One-Out (LOO) Cross-Validation

- Special case of K-fold CV when $K=N$ (number of training examples)
- Each partition is now as example
- Train using $N-1$ examples, validate on the remaining example
- Repeat the same N times, each with a different validation Example



- Finally, choose the model with smallest average validation error
- Can be expensive for large N . Typically used when N is small

Leave-One-Out Cross-Validation(LOOCV)

- LOOCV is closely related to the *validation set approach* but it attempts to address that method's drawback
- LOOCV involves splitting the set of observations into two parts
- Instead of creating two subsets of comparable size, a single observation (x_1, y_1) is used for the validation set, and the remaining observations $\{(x_2, y_2), \dots, (x_n, y_n)\}$ make up the training set
- The statistical learning method is fit on the $n - 1$ training observations, and a prediction \hat{y}_1 is made for the excluded observation, using its value x_1
- Since (x_1, y_1) was not used in the fitting process, $MSE_1 = (y_1 - \hat{y}_1)^2$ provides an approximately unbiased estimate for the test error
- But even though MSE_1 is unbiased for the test error, it is a poor estimate because it is highly variable, since it is based upon a single observation (x_1, y_1)

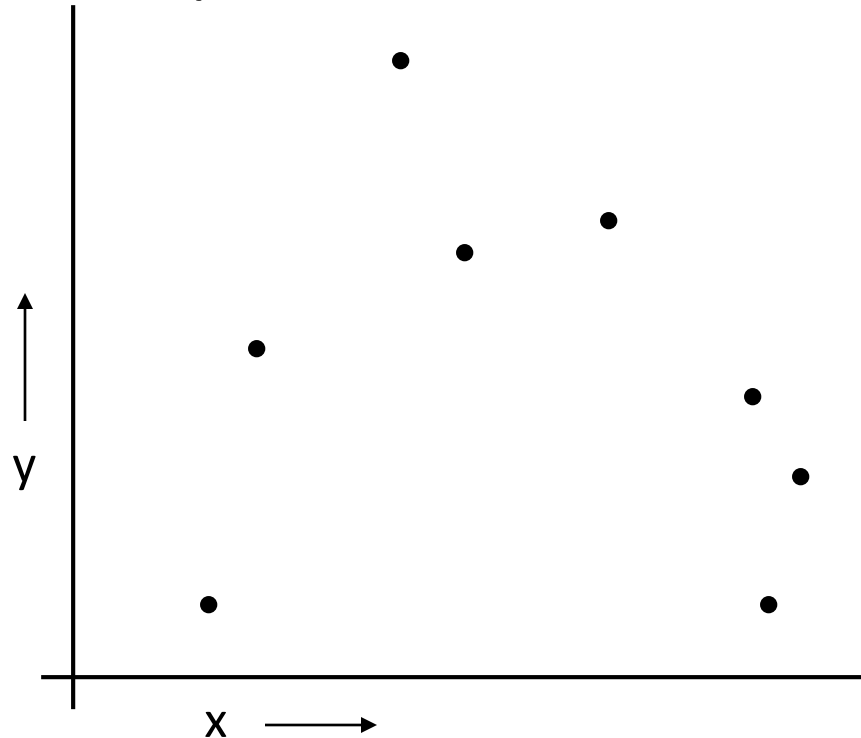
Leave-One-Out Cross-Validation(LOOCV)

- We can repeat the procedure by selecting (x_2, y_2) for the validation data, training the statistical learning procedure on the $n-1$ observations $\{(x_1, y_1), (x_3, y_3), \dots, (x_n, y_n)\}$, and $MSE_2 = (y_2 - \hat{y}_2)^2$
- Repeating this approach n times produces n squared errors, $MSE_1, MSE_2, \dots, MSE_n$
- The LOOCV estimate for the test MSE is: $CV_n = \frac{1}{n} \sum_i MSE_i$

Leave-One-Out Cross-Validation(LOOCV)

- LOOCV has a couple of a major advantages over the validation set approach
 - LOOCV has less bias
 - In contrast to the validation approach which will yield different results when applied repeatedly due to randomness in the training/validation set splits, performing LOOCV multiple times will always yield the same result: there is no randomness in the training/validation set splits
- LOOCV has the potential to be expensive to implement, since the model has to be fit n times
- This can be very time consuming if n is large, and if each individual model is slow to fit

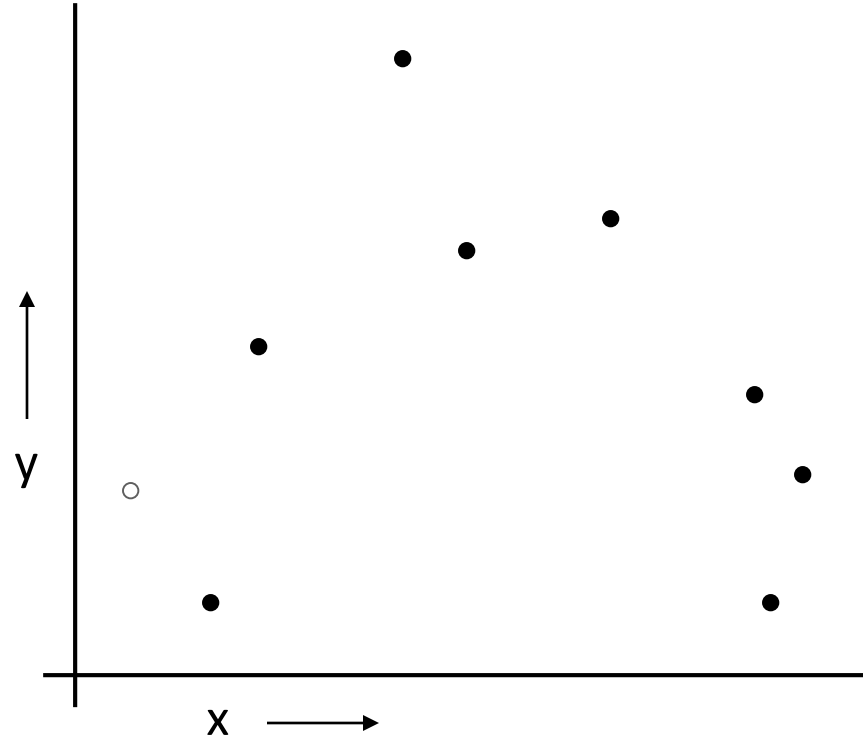
LOOCV (Leave-one-out Cross Validation)



For $k=1$ to n

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

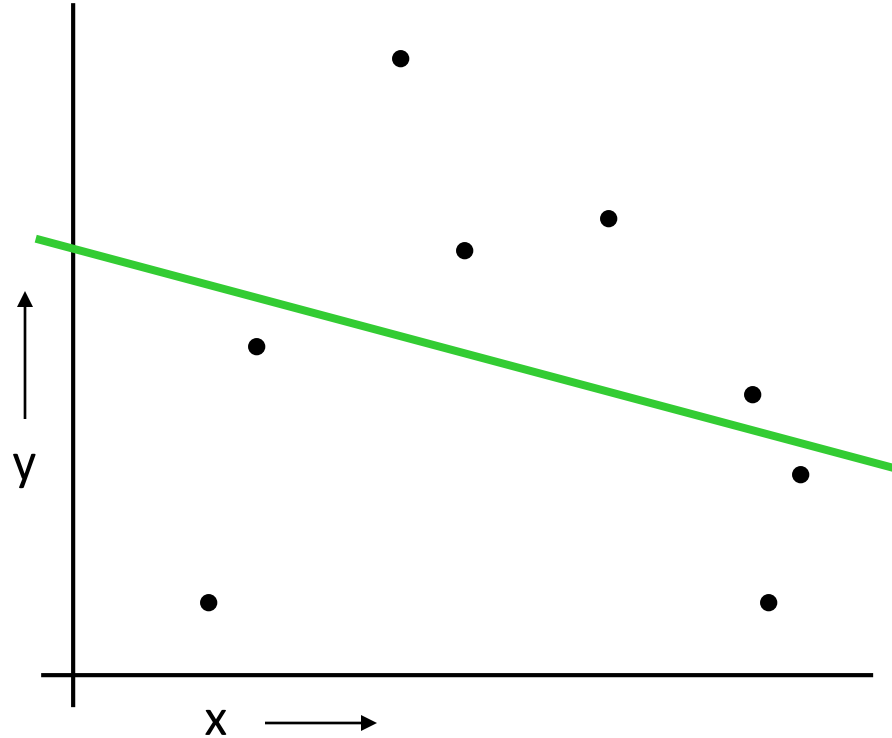
LOOCV (Leave-one-out Cross Validation)



For $k=1$ to n

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

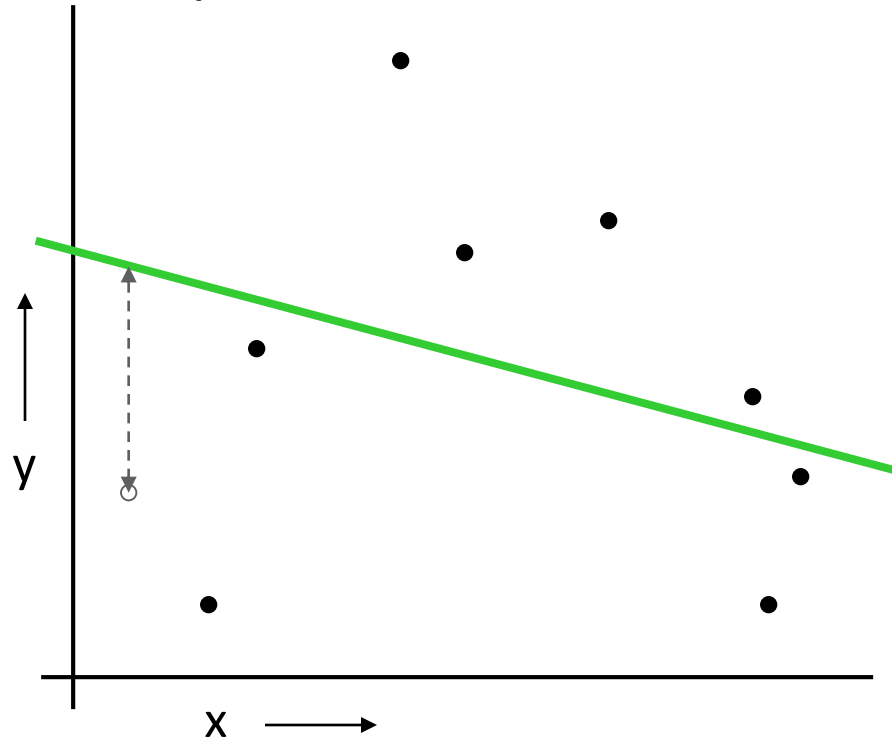
LOOCV (Leave-one-out Cross Validation)



For $k=1$ to n

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 $n-1$ datapoints

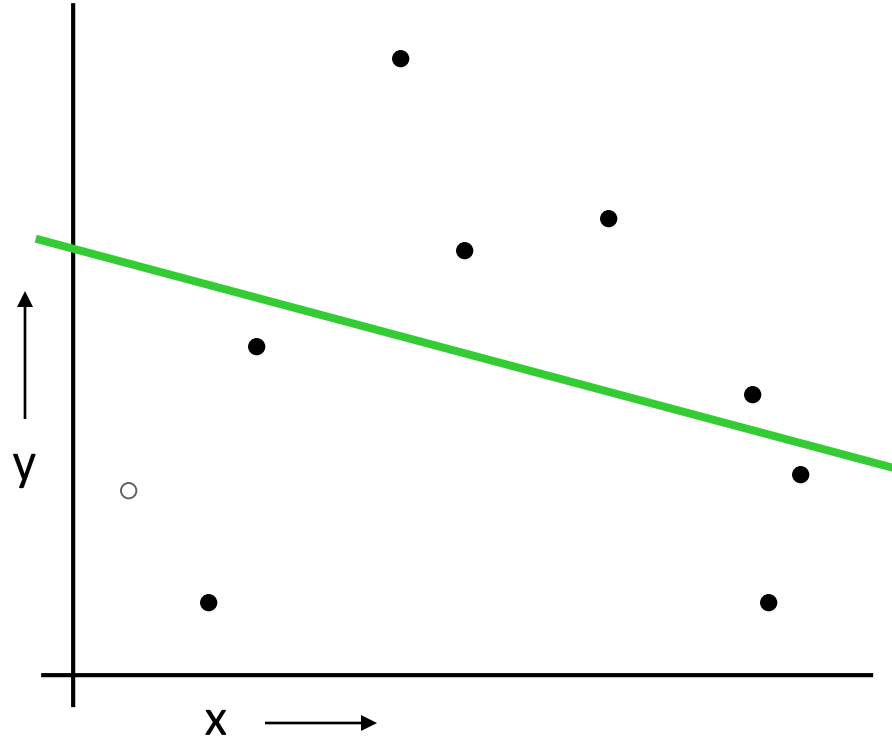
LOOCV (Leave-one-out Cross Validation)



For $k=1$ to n

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 $n-1$ datapoints
4. Note your error (x_k, y_k)

LOOCV (Leave-one-out Cross Validation)

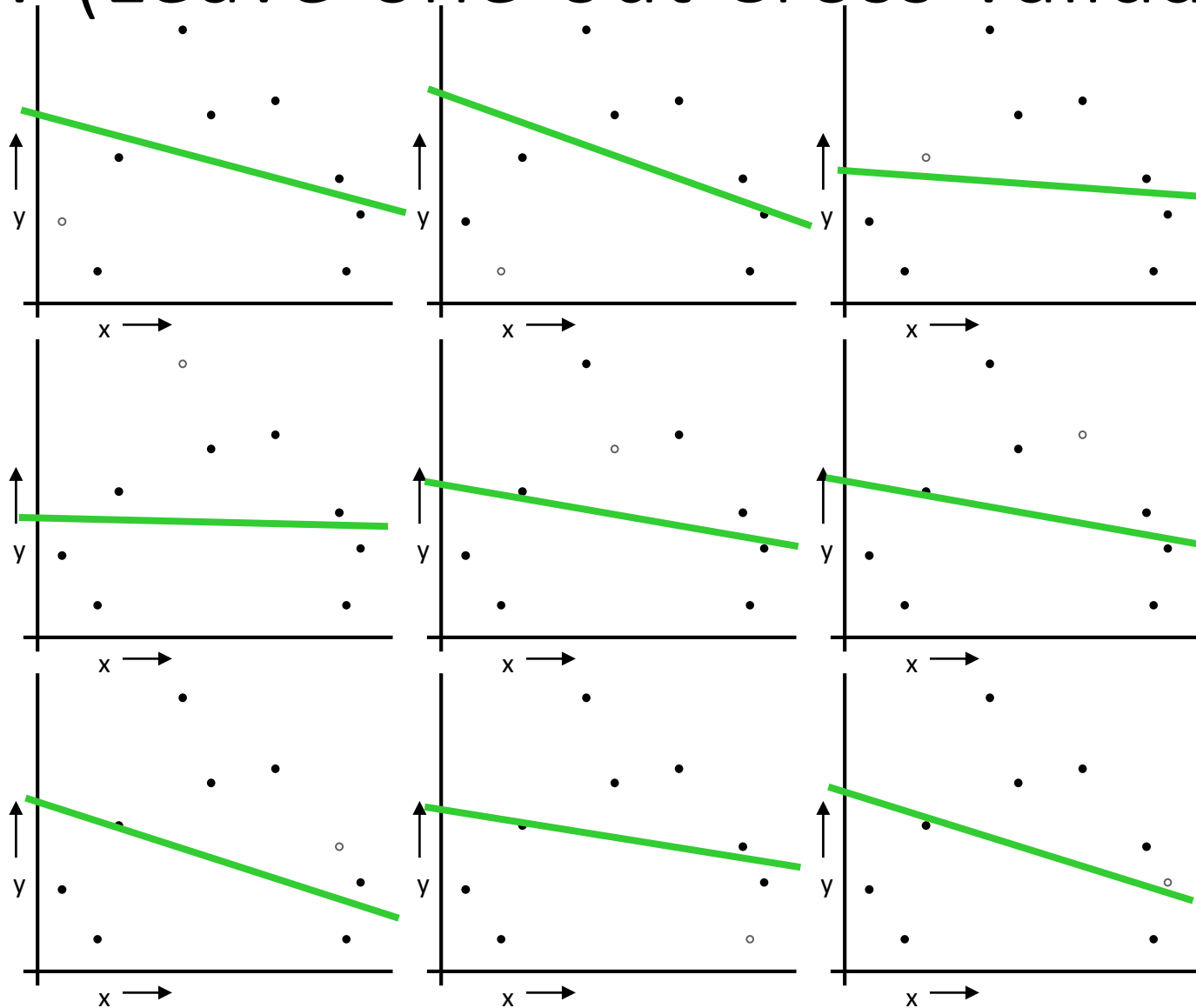


For $k=1$ to n

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 $n-1$ 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)



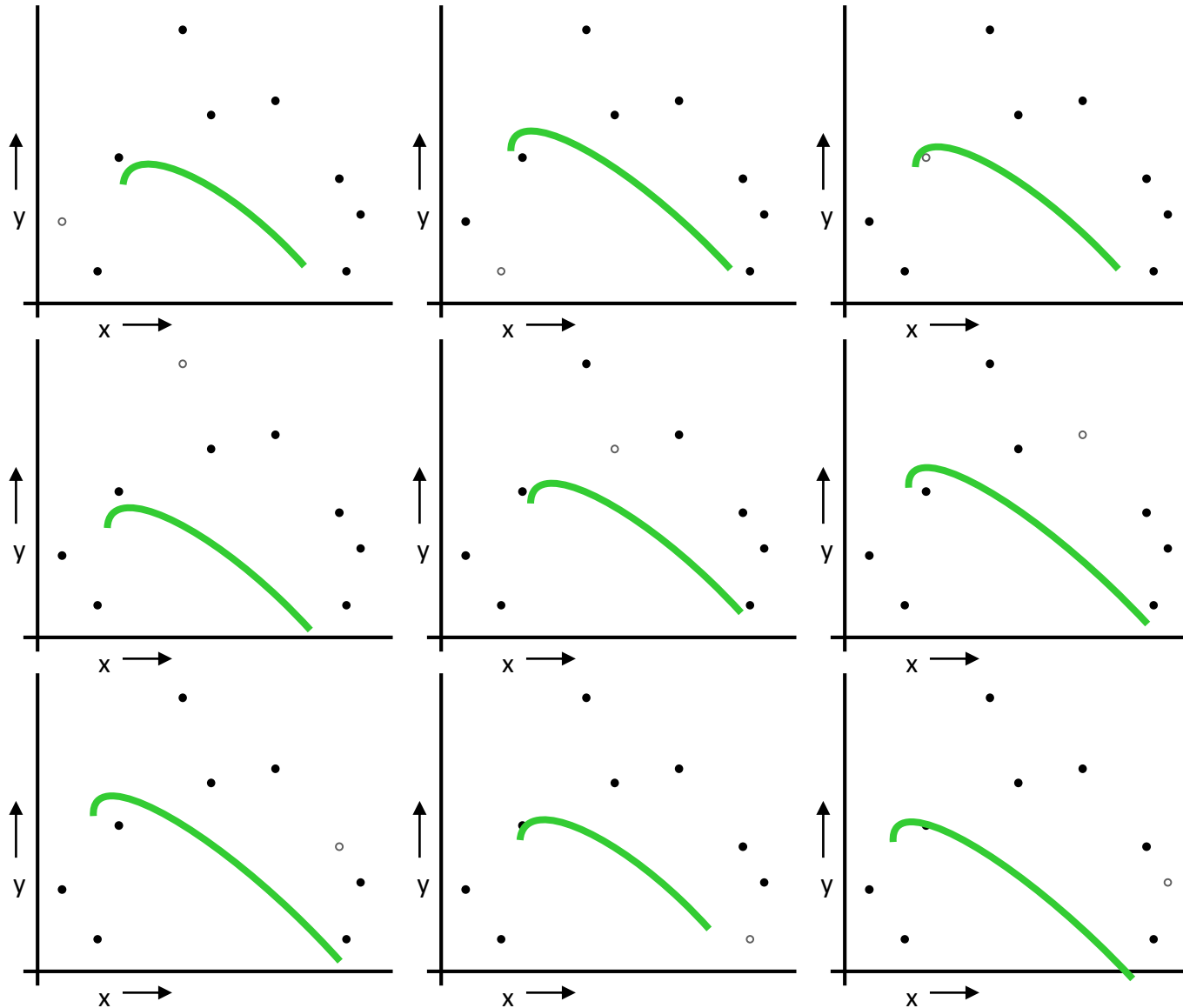
For $k=1$ to n

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 $n-1$ 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



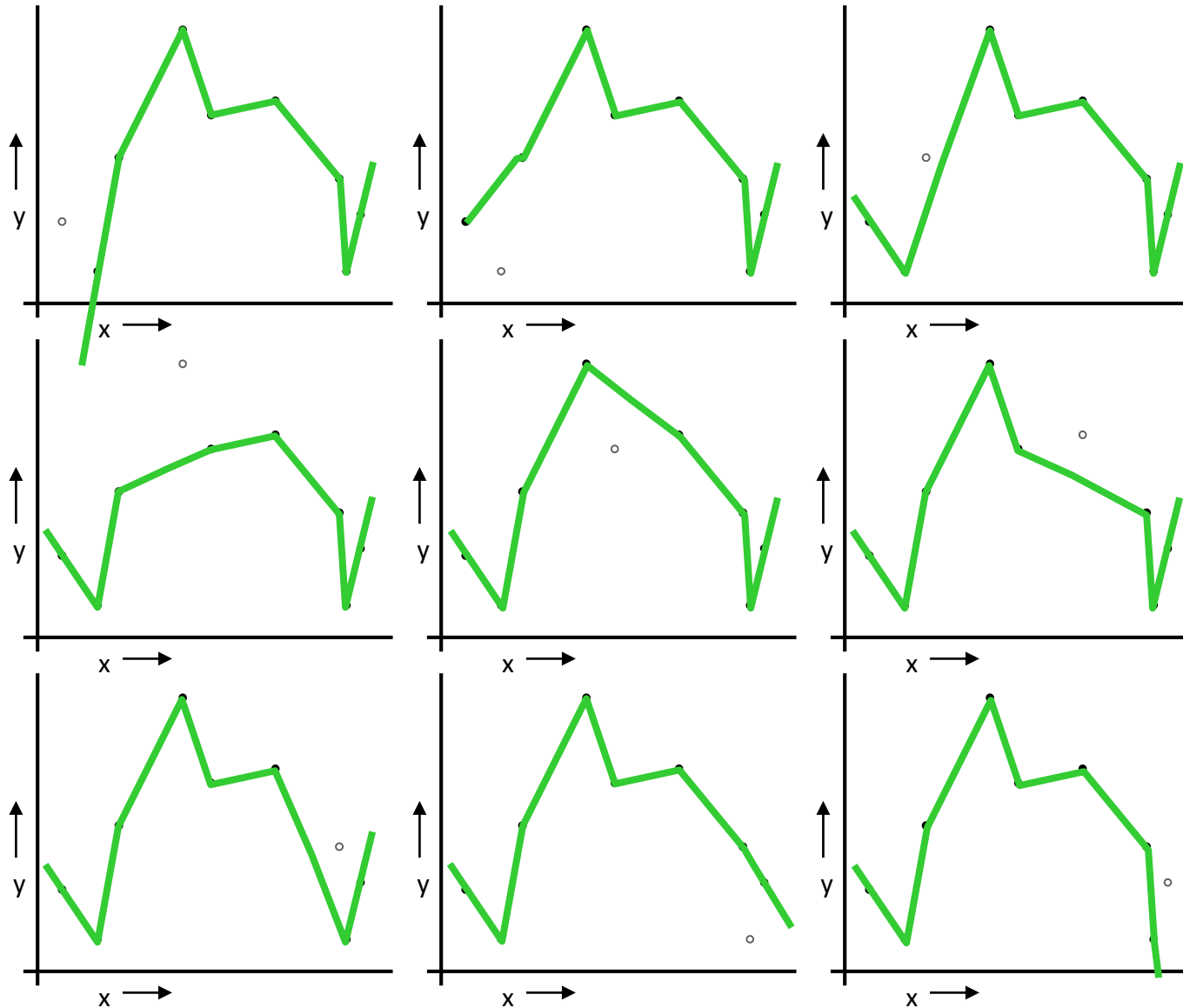
For $k=1$ to n

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 $n-1$ datapoints
4. Note your error (x_k, y_k)

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

$$MSE_{LOOCV} = 0.962$$

LOOCV for Join The Dots



For $k=1$ to n

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 $n-1$ datapoints
4. Note your error (x_k, y_k)

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

$$MSE_{LOOCV} = 3.33$$

LOOCV: *Auto* data set and Boston data set

- Calculate LOOCV metric

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?

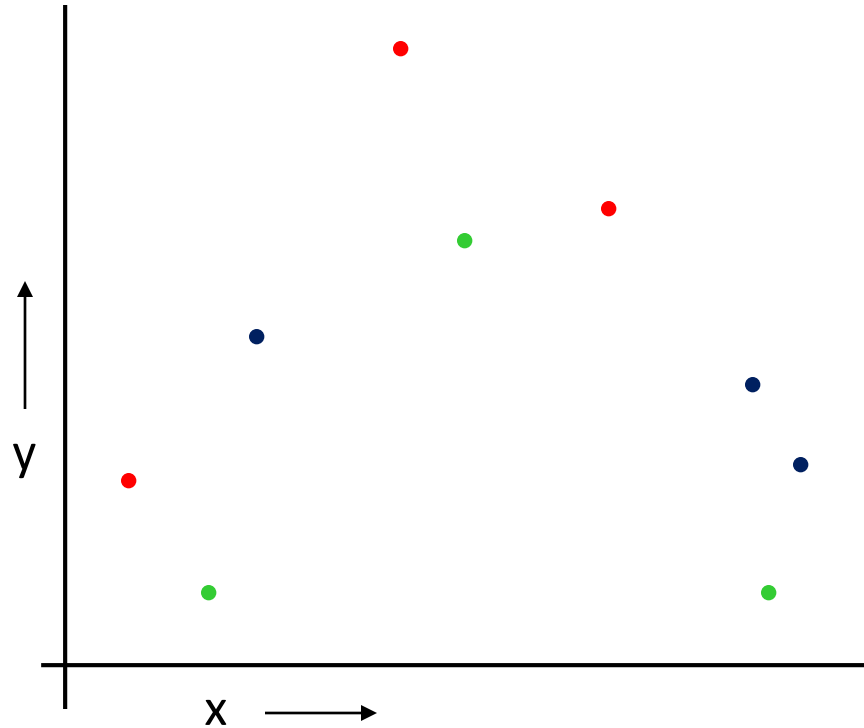
K-Fold Cross Validation

- Create K equal sized partitions of the training data
- Each partition has $\frac{N}{K}$ examples
- Train using $K - 1$ partitions, validate on the remaining partition
- Repeat the same K times, each with a different validation portion



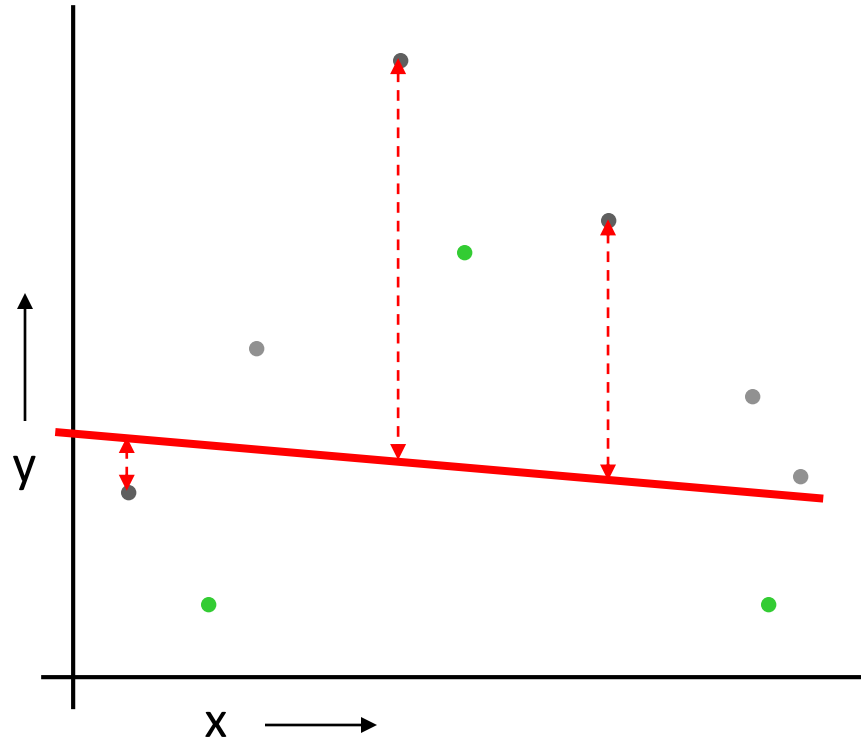
- Finally choose the model with smallest average validation error
- Usually K is chosen as 10

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)

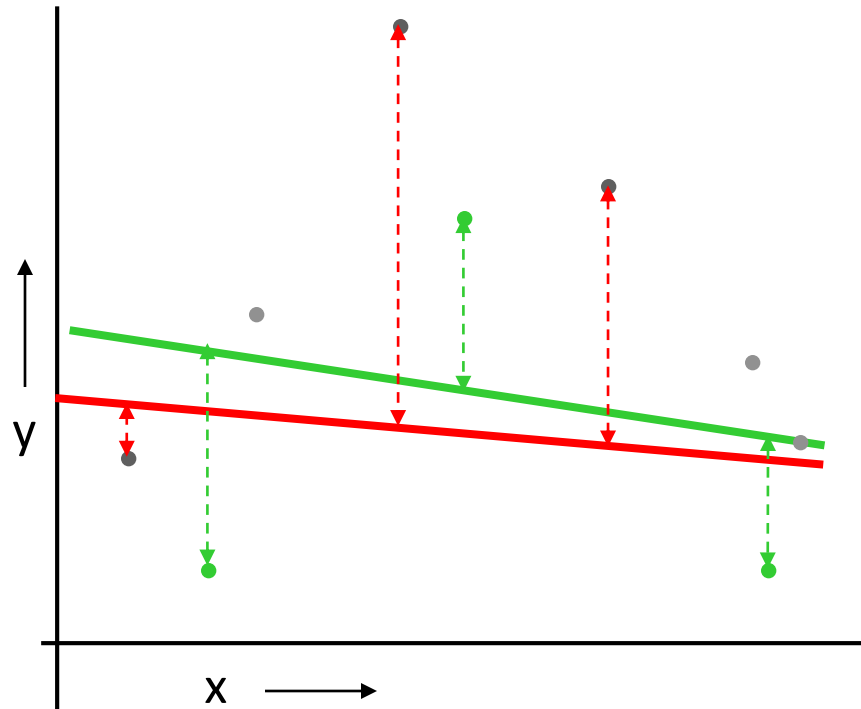
k-fold Cross Validation



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

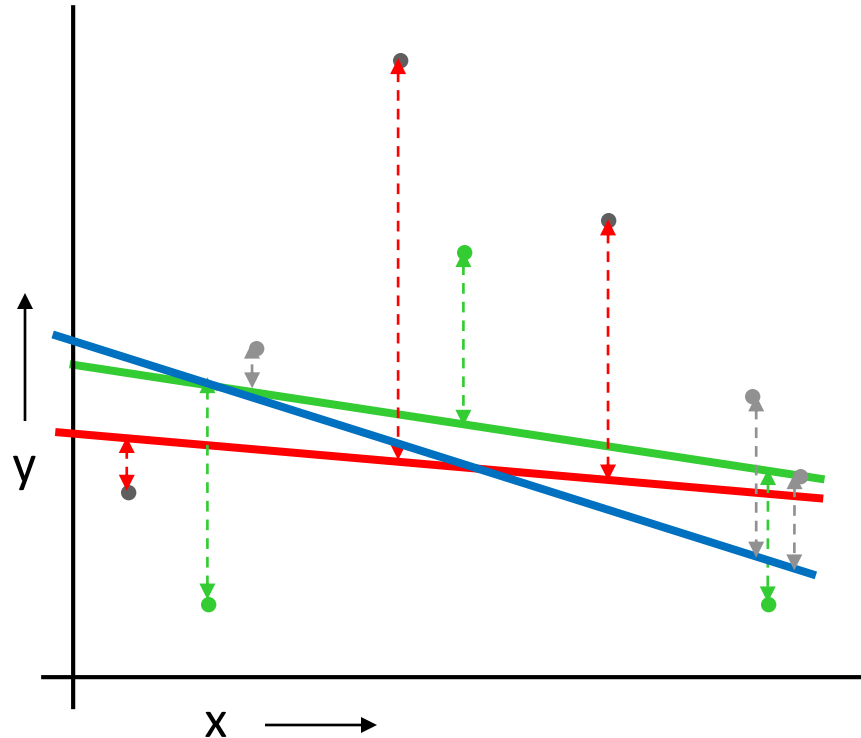


Randomly break the dataset into k partitions (in our example we'll have $k=3$ partitions colored Red Green and Blue)

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



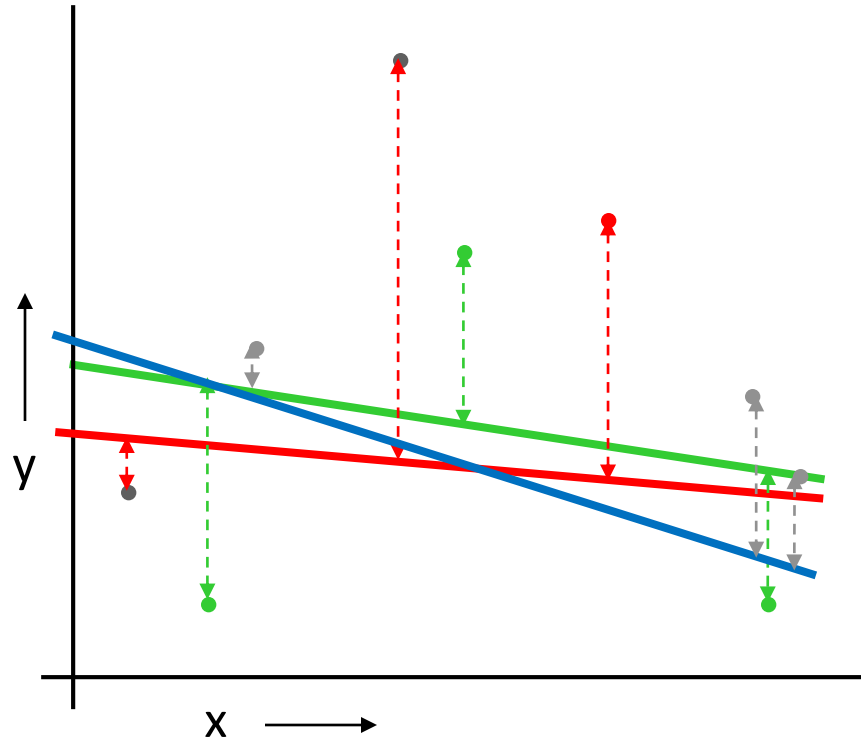
Randomly break the dataset into k partitions (in our example we'll have $k=3$ partitions colored Red Green and Blue)

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



Linear Regression $MSE_{3FOLD}=2.05$

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

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 n times.
3-fold	Wastier than 10-fold. Expensier 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	TRAINER	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

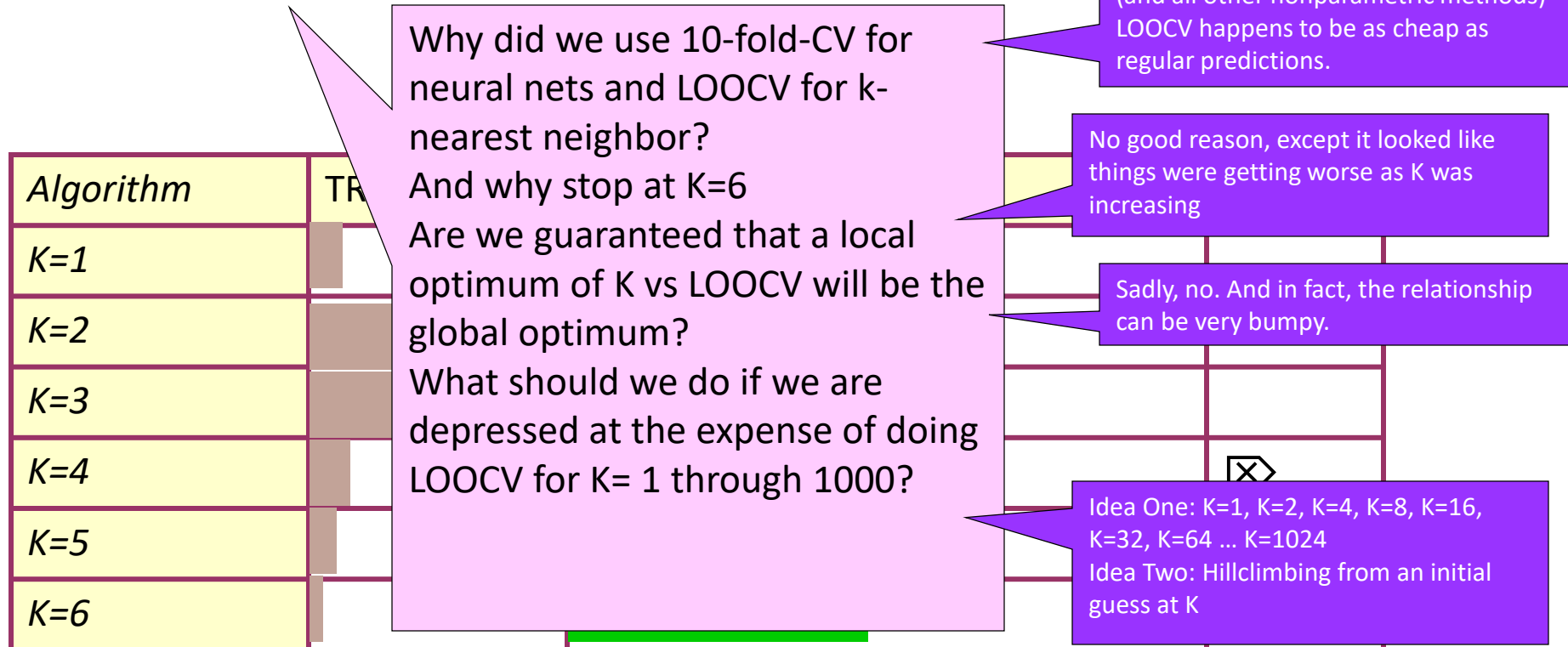
- Example: Choosing “k” for a k-nearest-neighbor regression.
- Step 1: Compute LOOCV error for six different model classes:

Algorithm	TRAINERR	10-fold-CV-ERR	Choice
$K=1$		<div></div>	
$K=2$	<div></div>	<div></div>	
$K=3$	<div></div>	<div></div>	
$K=4$	<div></div>	<div></div>	⊗
$K=5$	<div></div>	<div></div>	
$K=6$	<div></div>	<div></div>	

- 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



- 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

- Can you think of other decisions we can ask Cross Validation to make for us, based on other machine learning algorithms in the class so far?

CV-based Model Selection

- Can you think of other decisions we can ask Cross Validation to make for us, based on other machine learning algorithms in the class so far?
 - Degree of polynomial in polynomial regression
 - Whether to use full, diagonal or spherical Gaussians in a Gaussian Bayes Classifier.
 - The Kernel Width in Kernel Regression
 - The Kernel Width in Locally Weighted Regression
 - The Bayesian Prior in Bayesian Regression

These involve choosing the value of a real-valued parameter. What should we do?

CV-based Model Selection

- Can you think of other decisions we can ask Cross Validation to make for us, based on other machine learning algorithms in the class so far?
 - Degree of polynomial in polynomial regression
 - Whether to use full, diagonal or spherical Gaussians in a Gaussian Bayes Classifier.
 - The Kernel Width in Kernel Regression
 - The Kernel Width in Locally Weighted Regression
 - The Bayesian Prior in Bayesian Regression

These involve choosing the value of a real-valued parameter. What should we do?

Idea One: Consider a discrete set of values (often best to consider a set of values with exponentially increasing gaps, as in the K-NN example).

Idea Two: Compute $\frac{\partial \text{LOOCV}}{\partial \text{Parameter}}$ and then do gradient descent.

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		<div></div>	
10-NN	<div></div>	<div></div>	
Linear Reg'n	<div></div>	<div></div>	
Quad reg'n	<div></div>	<div></div>	⊗
LWR, KW=0.1	<div></div>	<div></div>	
LWR, KW=0.5	<div></div>	<div></div>	

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

Cross-Validation for regression

- Choosing a polynomial degree
- Choosing which regressor to use
- Feature selection (will cover later later)

Classification

Cross-validation for classification

- Instead of computing the sum squared errors on a test set, you should compute...

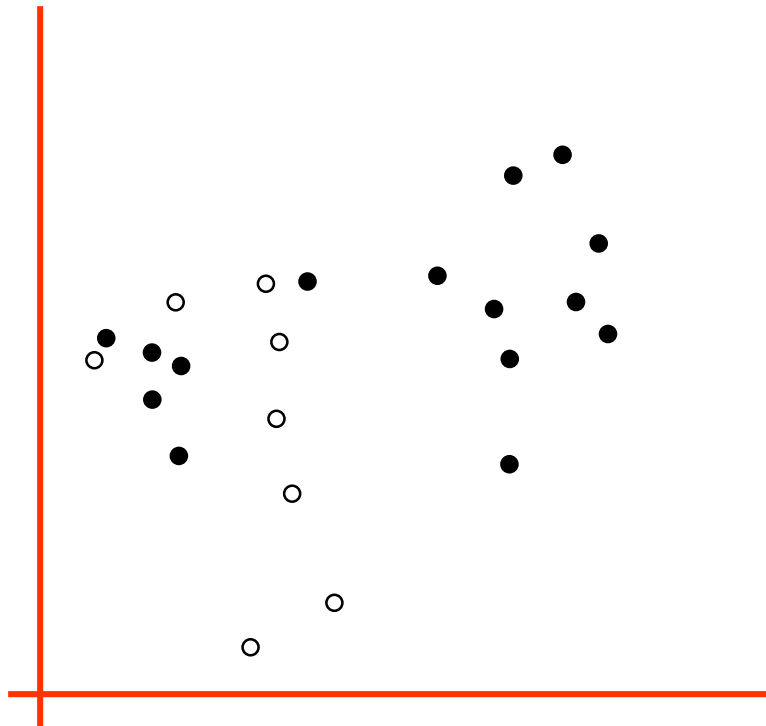
Cross-validation for classification

- Instead of computing the sum squared errors on a test set, you should compute...

The total number of misclassifications on a testset.

Cross-validation for classification

- Instead of computing the sum squared errors on a test set, you should compute...
 - The total number of misclassifications on a testset.



- What's LOOCV of 1-NN?
- What's LOOCV of 3-NN?
- What's LOOCV of 22-NN?

Cross-validation for classification

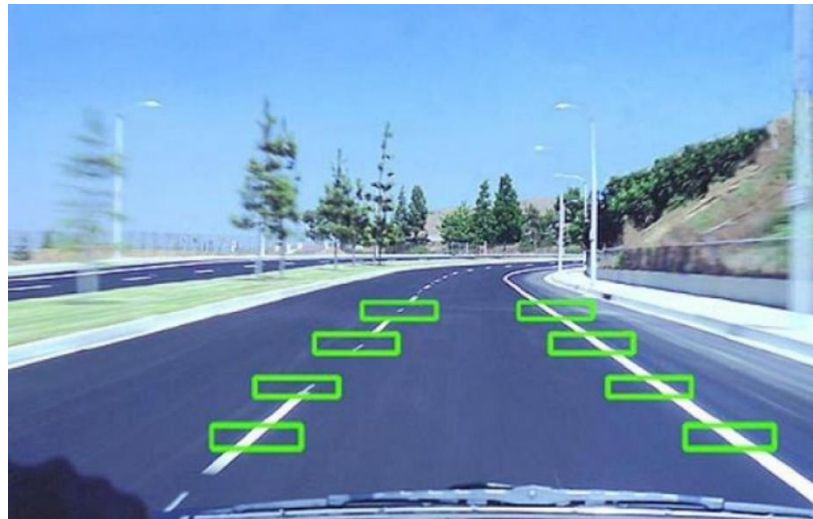
- Instead of computing the sum squared errors on a test set, you should compute...
 - The total number of misclassifications on a testset.
- But there's a more sensitive alternative:
 - Compute
 - $\log P(\text{all test outputs} | \text{all test inputs, your model})$

Cross-Validation for classification

- Choosing the pruning parameter for decision trees
- What kind of Gaussian to use in a Gaussian-based Bayes Classifier
- Choosing which classifier to use

Classification Examples

- Spam Email filtering
- Fraud detection
- Self-piloting automobile



The Classification Problem

- **Given:** a set of n attributes, a set of k classes, and a set of labeled training instances.

- ✓ $F = f_1, f_2, \dots, f_n$

- ✓ $K = \{ \dots k \}$

- ✓ $I = \{ f_1, f_2, \dots, f_n, k \}$

- **Determine** a **classification rule** that predicts the class of any instance based on its attributes.
- 2 classes and more

The Classification Problem

■ Classification V.S. Regression

- ✓ *Classification*: predict the class (ham/spam, image of a cat/not an image of a cat, etc...)
- ✓ *Regression*: predict a value (Housing prices, tomorrows temperature, etc...)

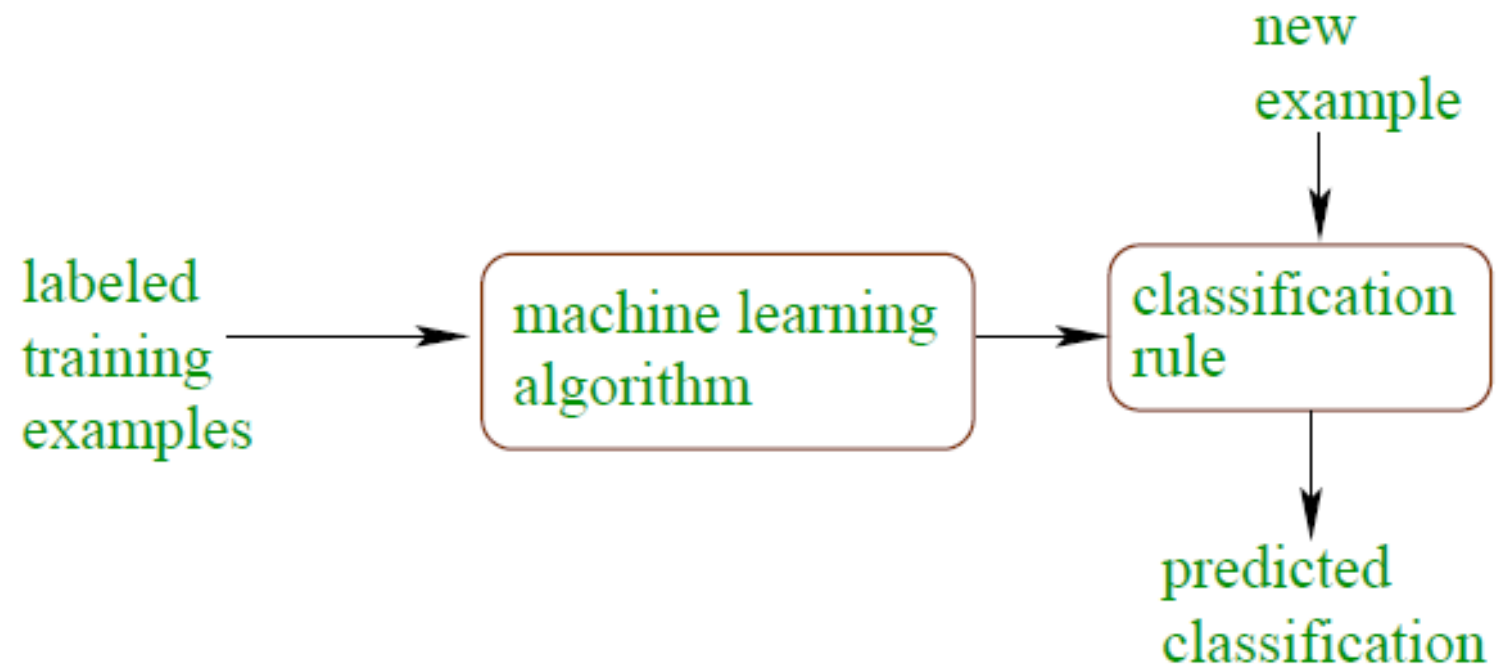
The Classification Problem

- **Classification V.S. Clustering**
 - ✓ *Classification: supervised*, a set of predefined classes
 - taxonomy
 - ✓ *Clustering: unsupervised*, no classes are predefined.
 - detect the relationships between instances

Classic Classifiers

- **Naïve Bayes**
- **Decision Trees**
- **KNN**
- **RandomForest**
- **SVM:** SMO, LibSVM
- **Neural Network**
- **...**

Train Your Classifier



Evaluation Method

- Basic Evaluation Method

- Precision
- Confusion matrix

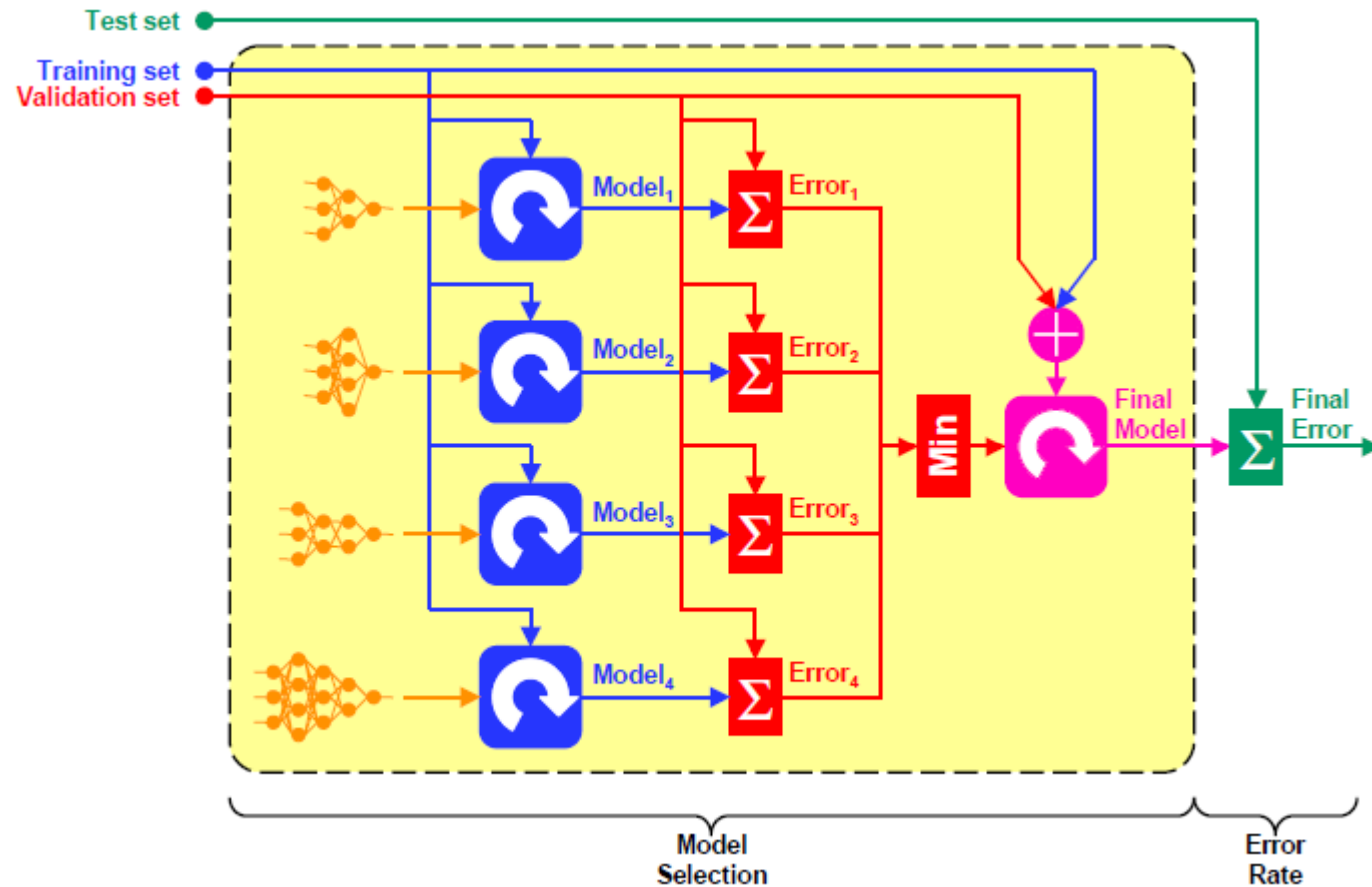
		actual value		
		p	n	total
prediction outcome	p'	True Positive	False Positive	P'
	n'	False Negative	True Negative	N'
total		P	N	

Simple Confusion Matrix

		Condition (as determined by "Gold standard")			
		Condition Positive	Condition Negative		
Test Outcome	Test Outcome Positive	True Positive	False Positive (Type I error)	Positive predictive value = $\frac{\Sigma \text{ True Positive}}{\Sigma \text{ Test Outcome Positive}}$	
	Test Outcome Negative	False Negative (Type II error)	True Negative	Negative predictive value = $\frac{\Sigma \text{ True Negative}}{\Sigma \text{ Test Outcome Negative}}$	
		Sensitivity = $\frac{\Sigma \text{ True Positive}}{\Sigma \text{ Condition Positive}}$	Specificity = $\frac{\Sigma \text{ True Negative}}{\Sigma \text{ Condition Negative}}$		

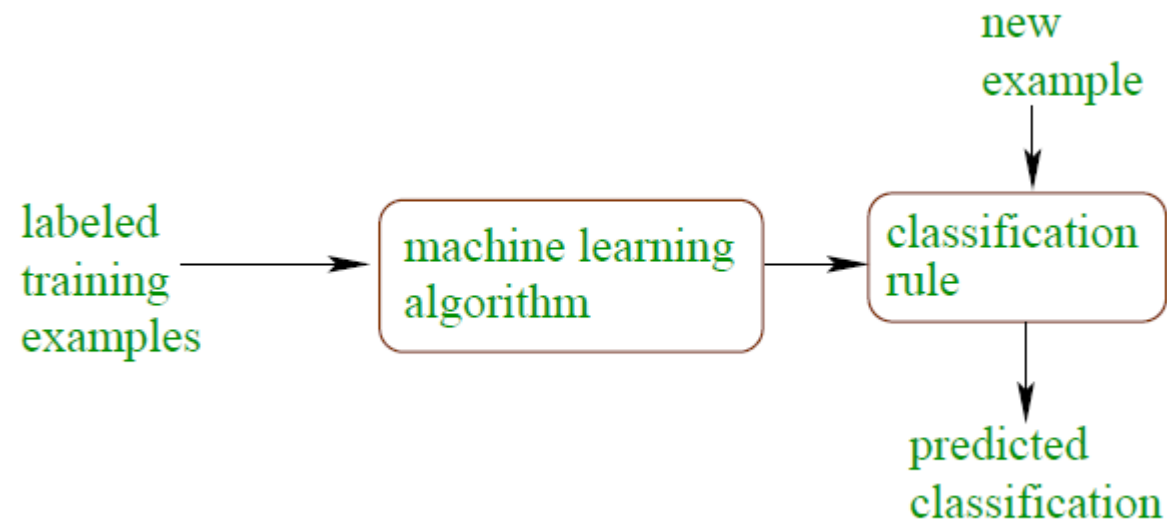
Cross Validation

- Three-way data splits



Apply the Classifier

- Save the Model
- Make the Model dynamic



Model evaluation: quantifying the quality of predictions

Scoring	Function	Comment
Classification		
'accuracy'	<u>metrics.accuracy_score</u>	
'average_precision'	<u>metrics.average_precision_score</u>	
Clustering		
'completeness_score'	<u>metrics.completeness_score</u>	
Regression		
'explained_variance'	<u>metrics.explained_variance_score</u>	
'neg_mean_absolute_error'	<u>metrics.mean_absolute_error</u>	
'neg_mean_squared_error'	<u>metrics.mean_squared_error</u>	
'neg_median_absolute_error'	<u>metrics.median_absolute_error</u>	
'r2'	<u>metrics.r2_score</u>	

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)

Very serious remark

- 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 remark

- 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 remark

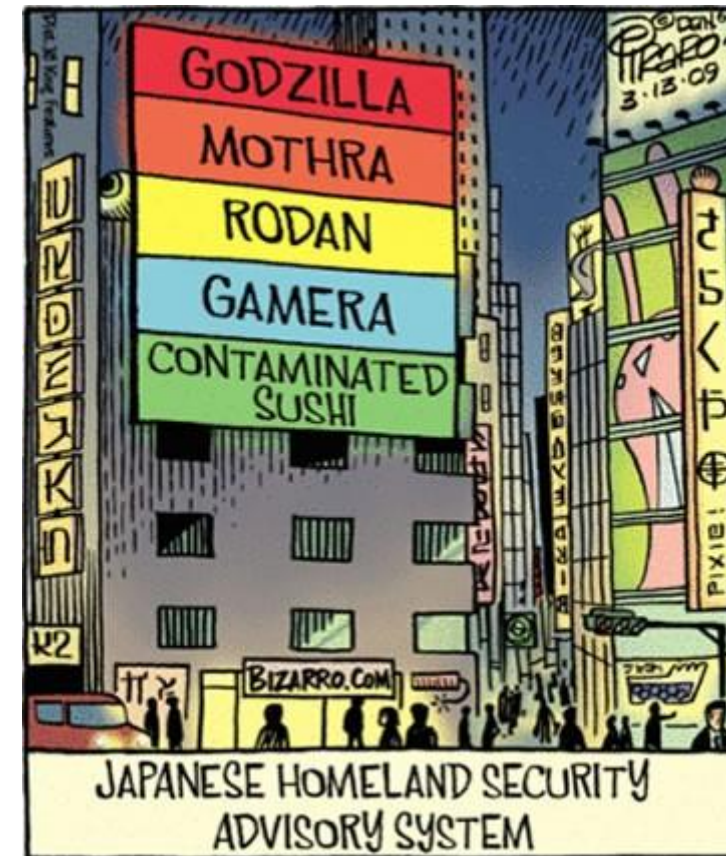
- Intensive use of cross validation can overfit.
- How?
 - Imagine a dataset with 50 records and 1000 attributes.
 - You try 1000 linear regression models, each one using one of the attributes.
 - The best of those 1000 looks good!
 - But you realize it would have looked good even if the output had been 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

Beyond Error Rate

- Predicting security risk
 - Predicting “low risk” for a terrorist, is far worse than predicting “high risk” for an innocent bystander (but maybe not 5 million of them)
- Searching for images
 - Returning irrelevant images is worse than omitting relevant ones

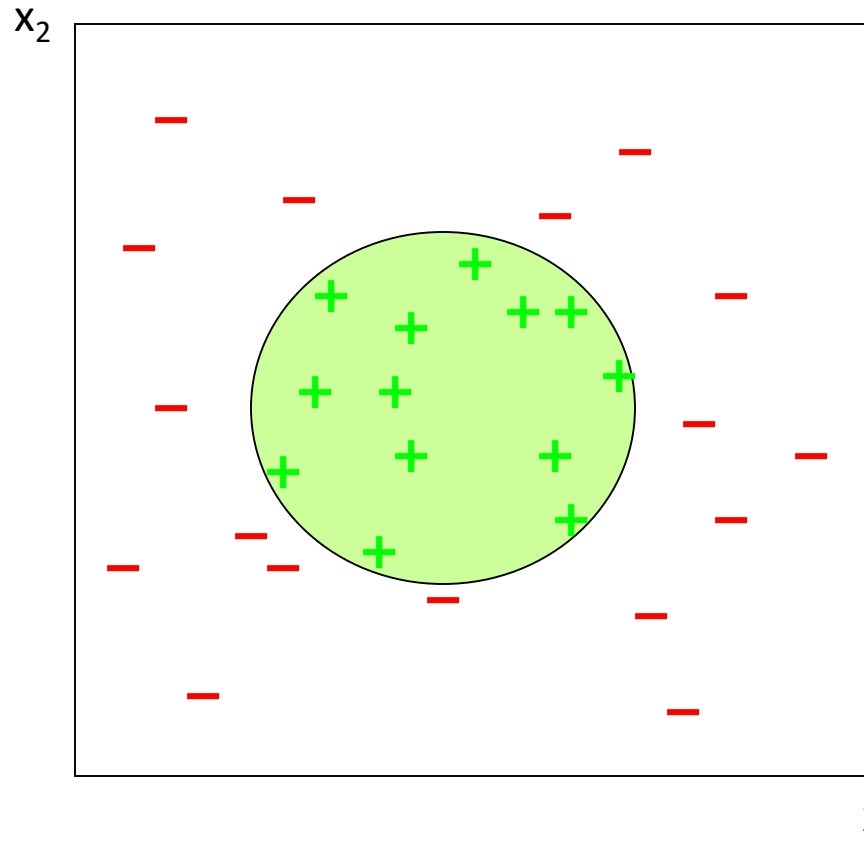


Biased Sample Sets

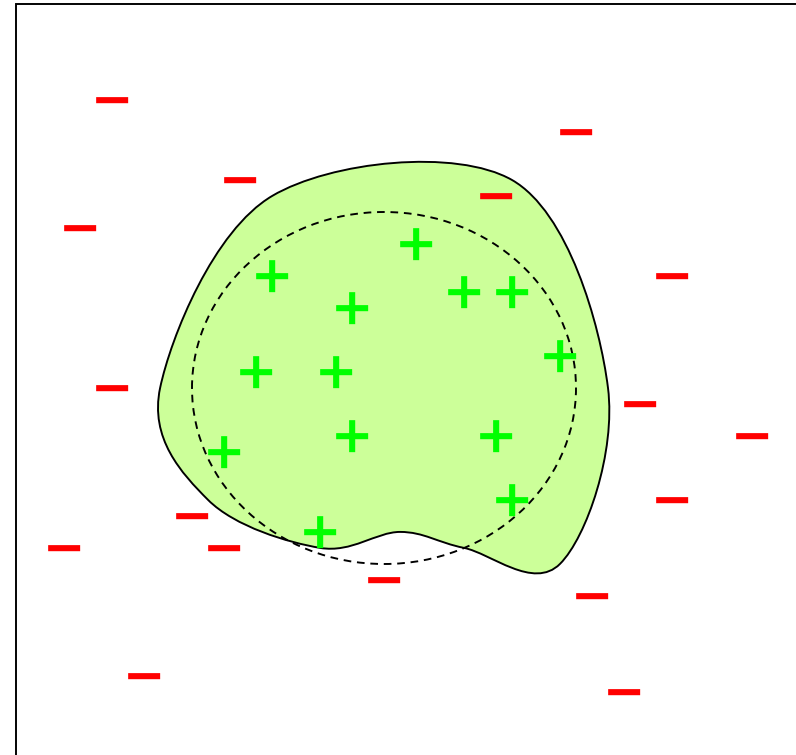
- Often there are orders of magnitude more negative examples than positive
- E.g., all images of Kris on Facebook
- If I classify all images as “not Kris” I’ll have >99.99% accuracy
- Examples of Kris should count much more than non-Kris!

False Positives

True concept



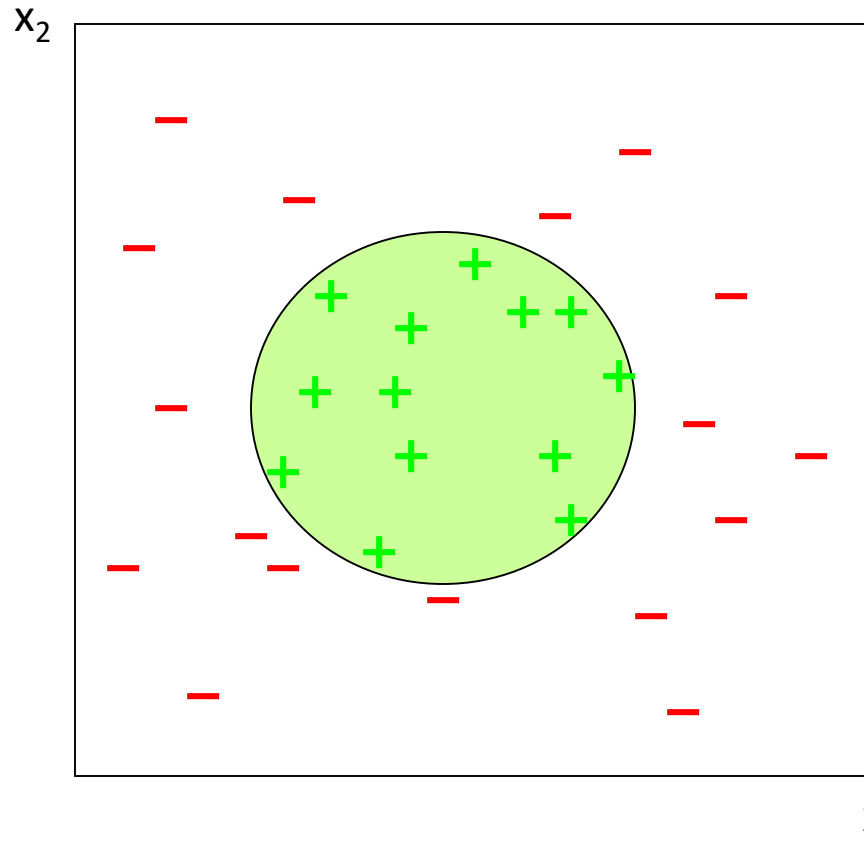
Learned concept



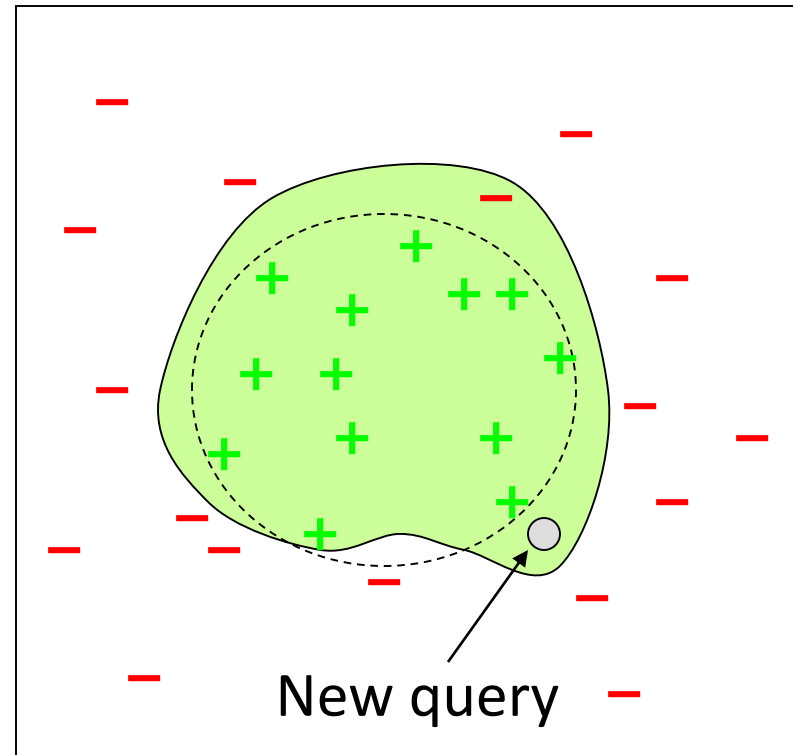
False Positives

An example **incorrectly** predicted to be positive

True concept



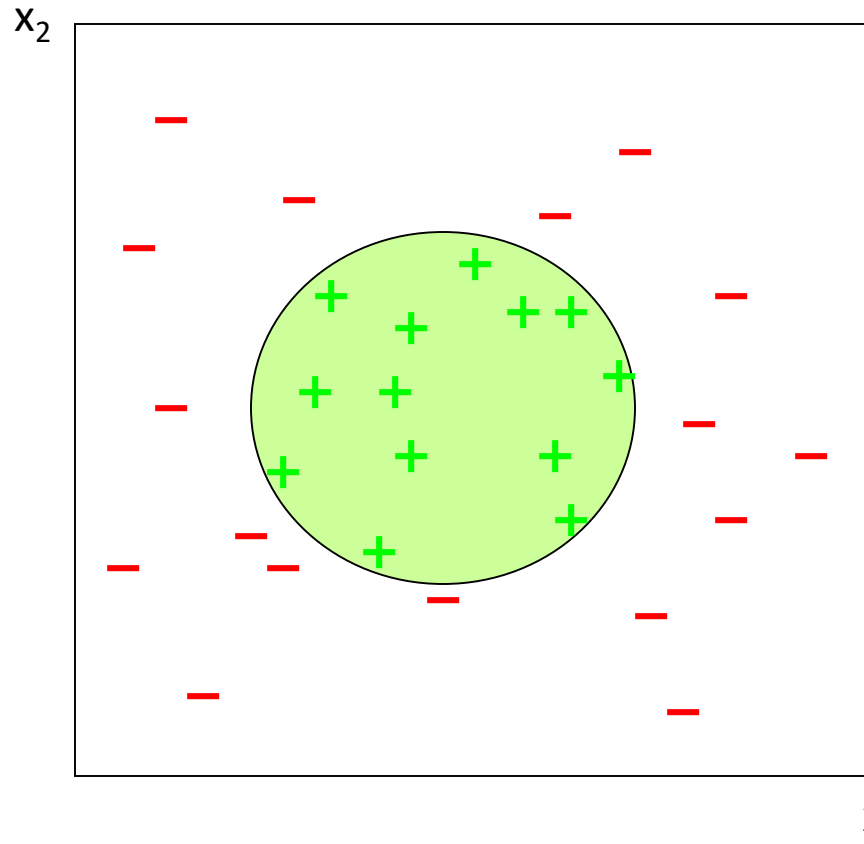
Learned concept



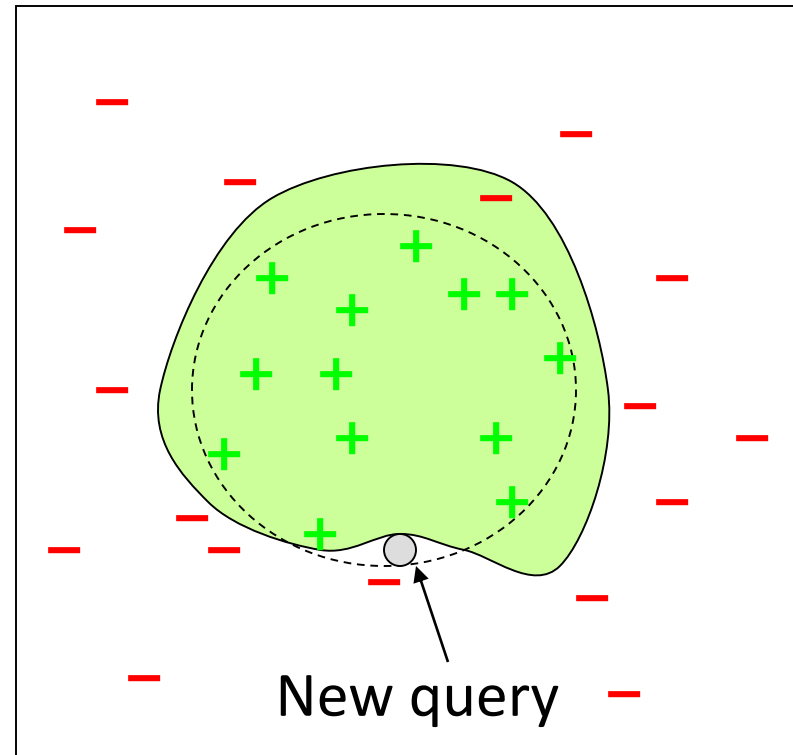
False Negatives

An example **incorrectly** predicted to be negative

True concept



Learned concept



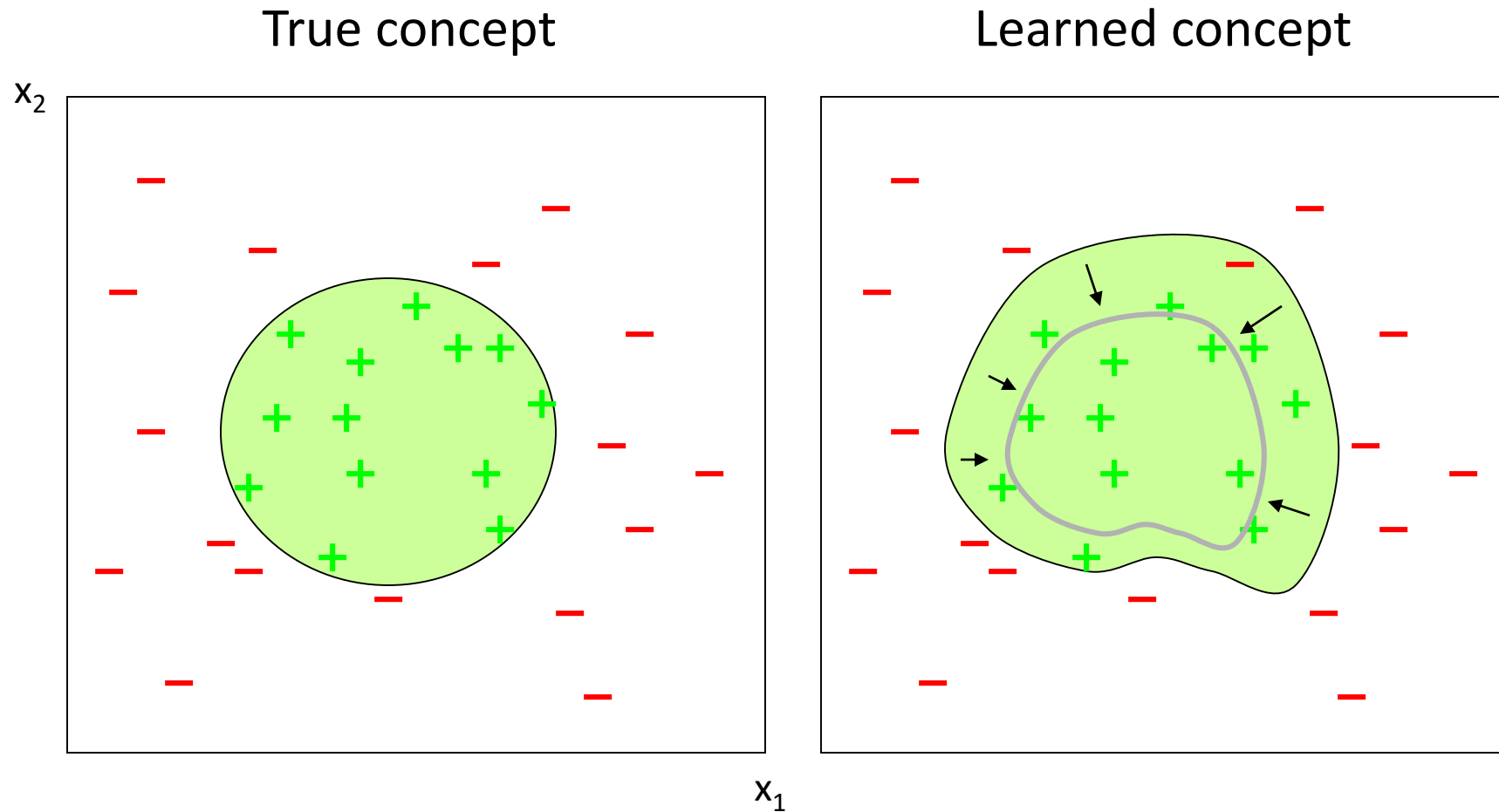
Precision vs. Recall

- Precision
 - $\frac{\text{\# of relevant documents retrieved}}{\text{\# of total documents retrieved}}$
- Recall
 - $\frac{\text{\# of relevant documents retrieved}}{\text{\# of total relevant documents}}$
- Numbers between 0 and 1

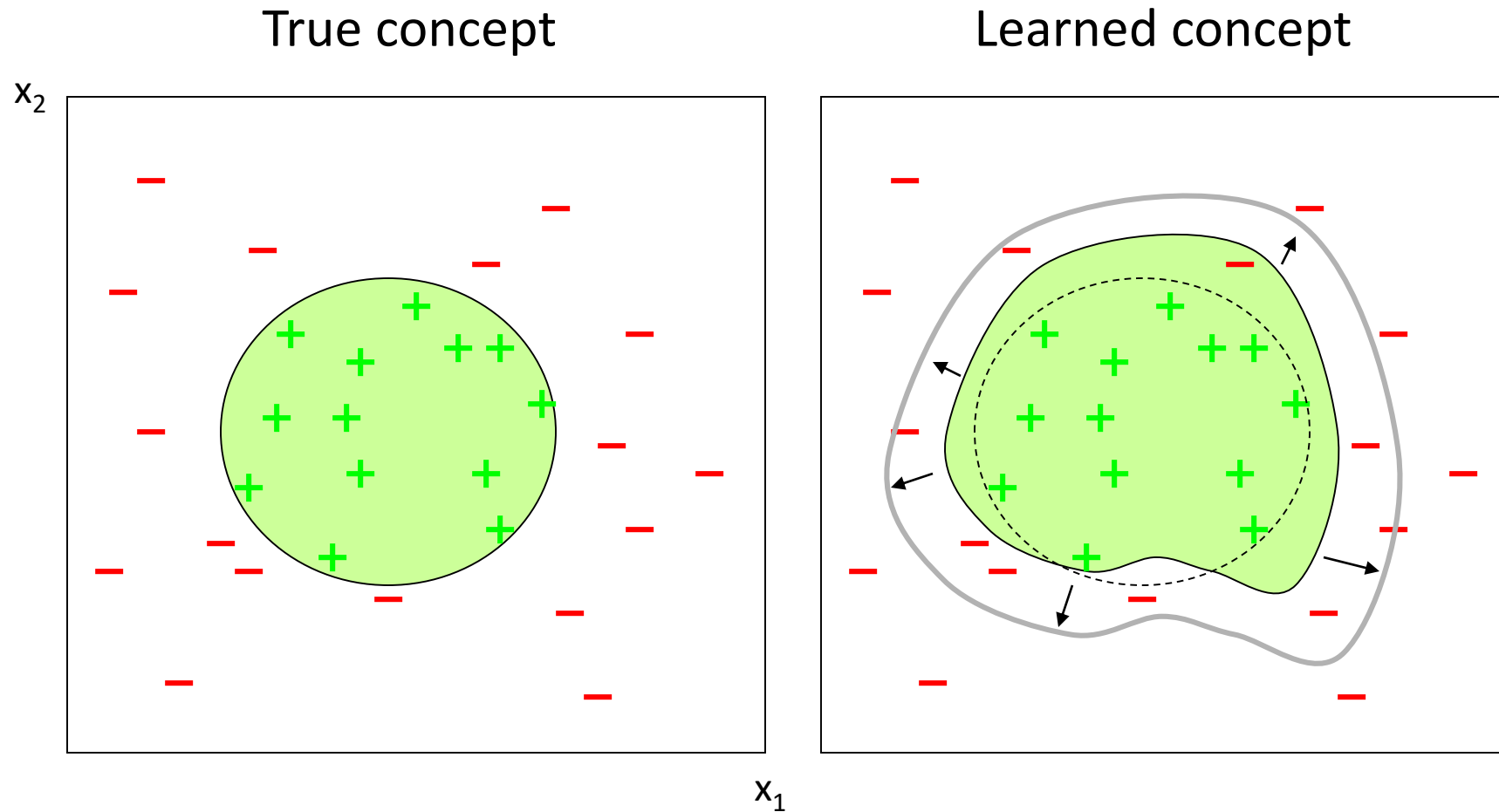
Precision vs. Recall

- Precision
 - $\# \text{ of true positives} / (\# \text{ true positives} + \# \text{ false positives})$
- Recall
 - $\# \text{ of true positives} / (\# \text{ true positives} + \# \text{ false negatives})$
- A precise classifier is selective
- A classifier with high recall is inclusive

Reducing False Positive Rate

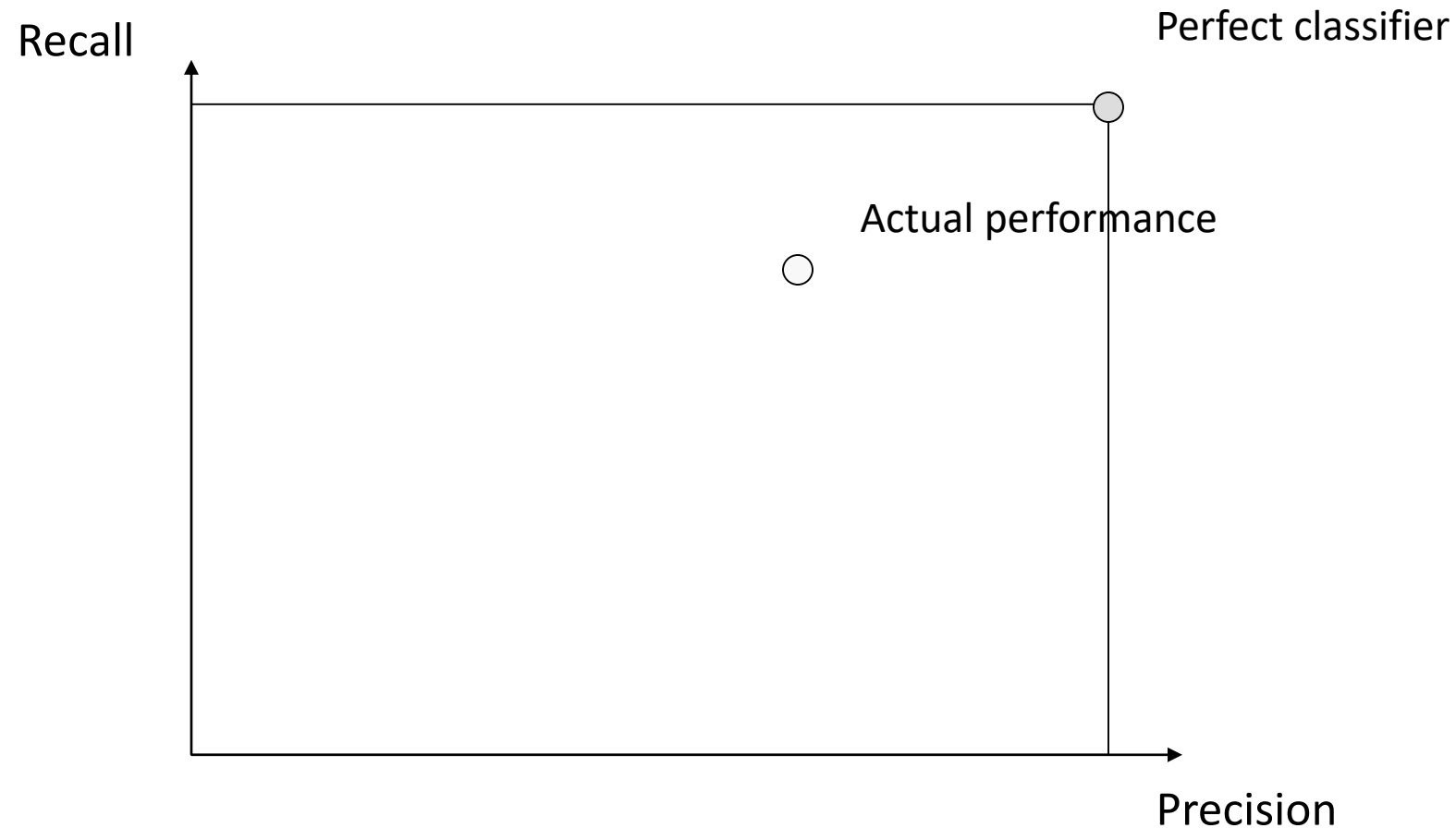


Reducing False Negative rate



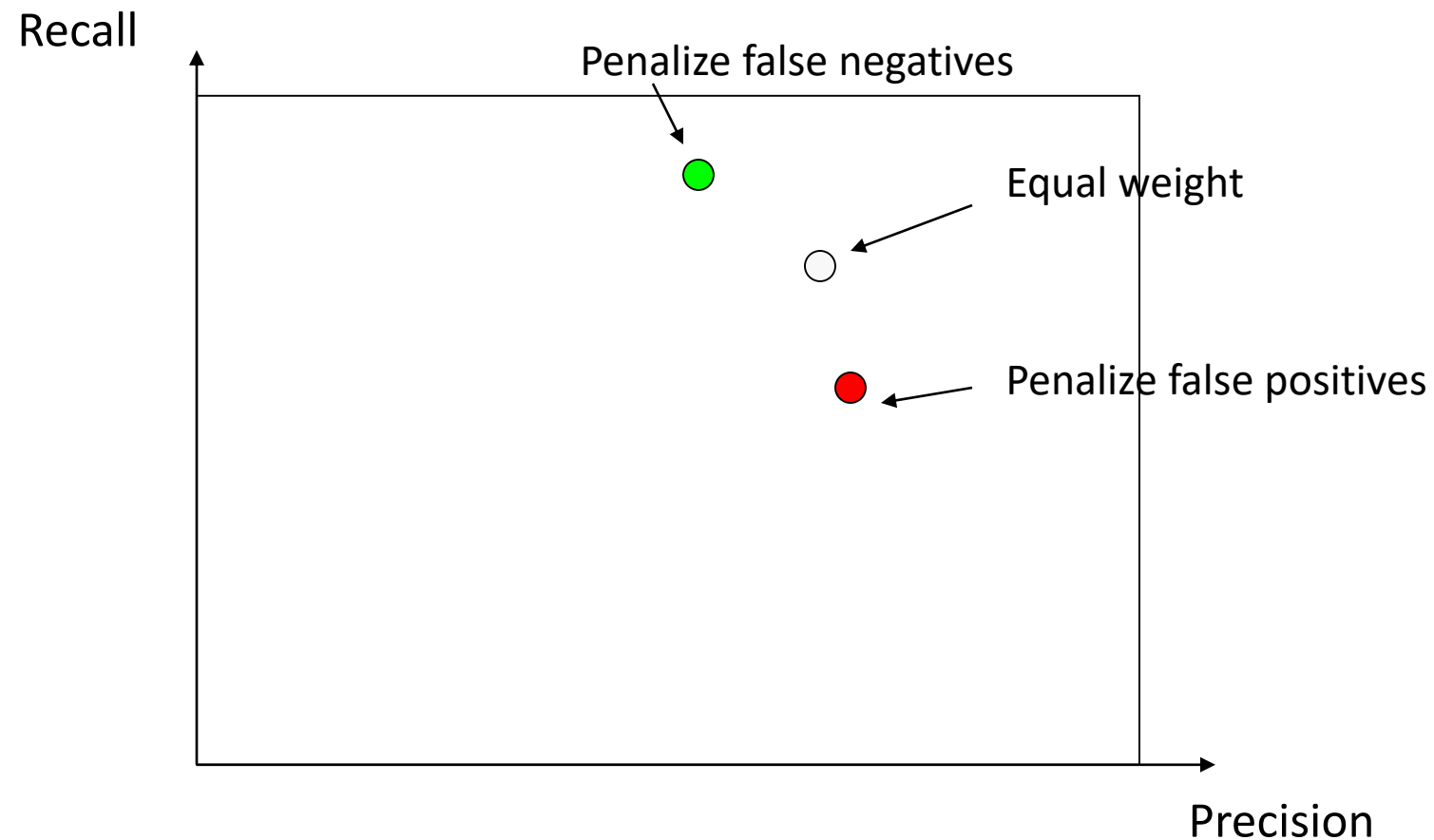
Precision-Recall curves

Measure Precision vs Recall as the classification boundary is tuned



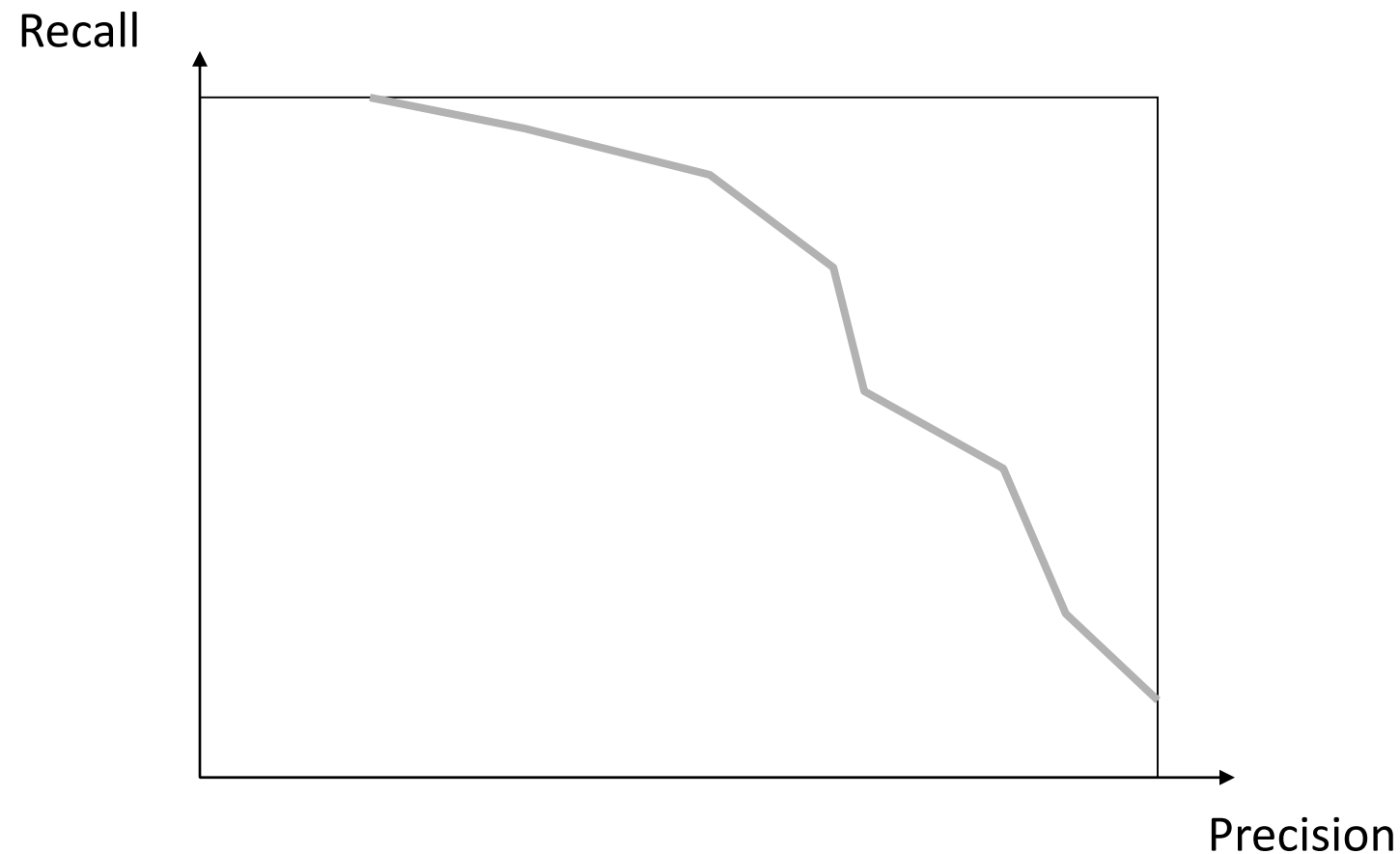
Precision-Recall curves

Measure Precision vs Recall as the classification boundary is tuned



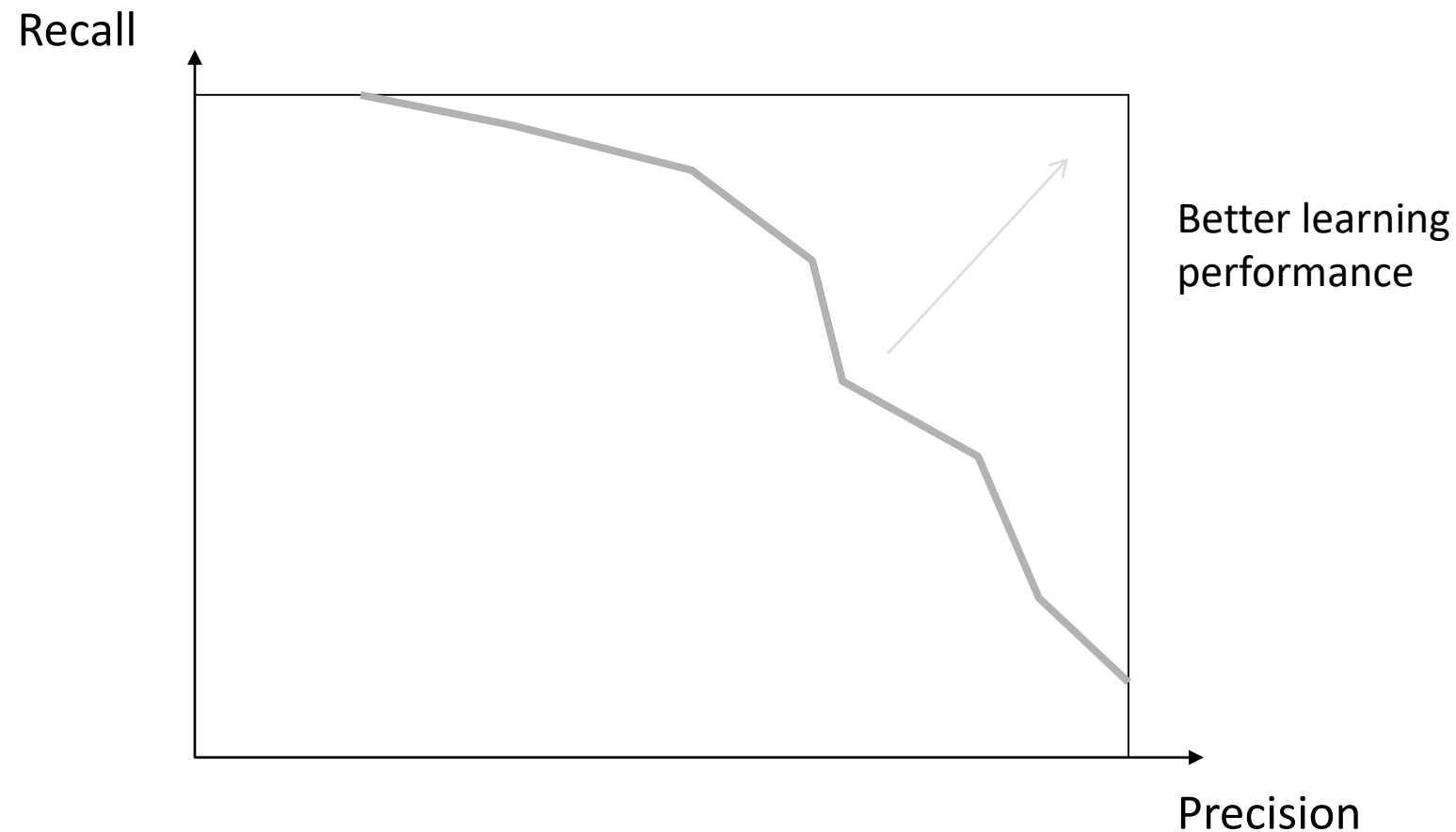
Precision-Recall curves

Measure Precision vs Recall as the classification boundary is tuned



Precision-Recall curves

Measure Precision vs Recall as the classification boundary is tuned



ROC (Receiver Operating Characteristic)

http://en.wikipedia.org/wiki/Receiver_operating_characteristic

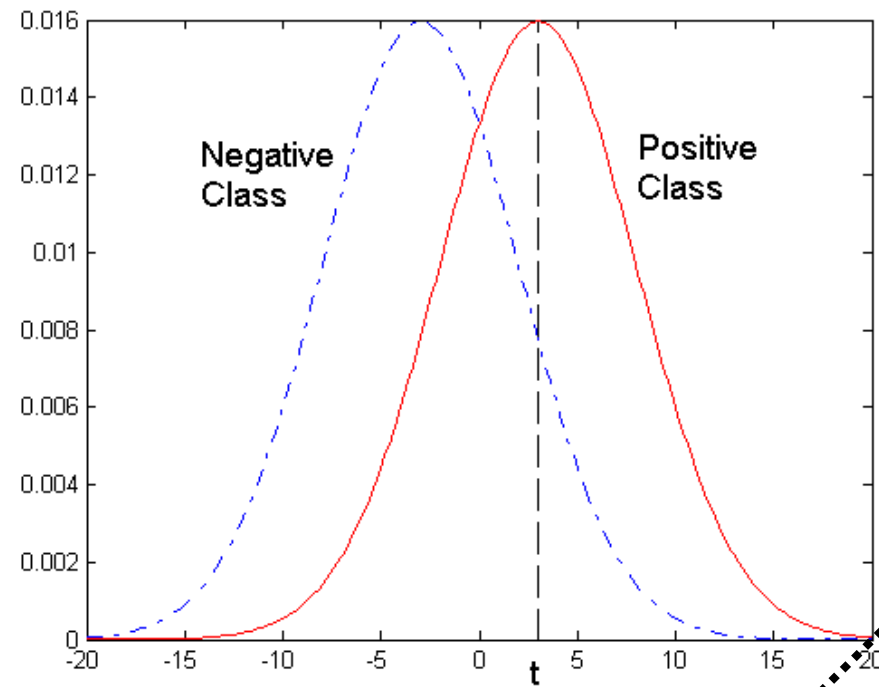
- Developed in 1950s for signal detection theory to analyze noisy signals
 - Characterize the trade-off between positive hits and false alarms
- ROC curve plots TP (on the y-axis) against FP (on the x-axis)
- Performance of each classifier represented as a point on the ROC curve
 - changing the threshold of algorithm, sample distribution or cost matrix changes the location of the point

What are ROC curves?

- In statistics, a receiver operating characteristic curve, i.e. ROC curve, is a graphical plot that illustrates the diagnostic ability of a binary classifier system as its discrimination threshold is varied.
- The ROC curve is created by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. The true-positive rate is also known as sensitivity, recall or probability of detection[1] in machine learning. The false-positive rate is also known as the fall-out or probability of false alarm[1] and can be calculated as $(1 - \text{specificity})$.

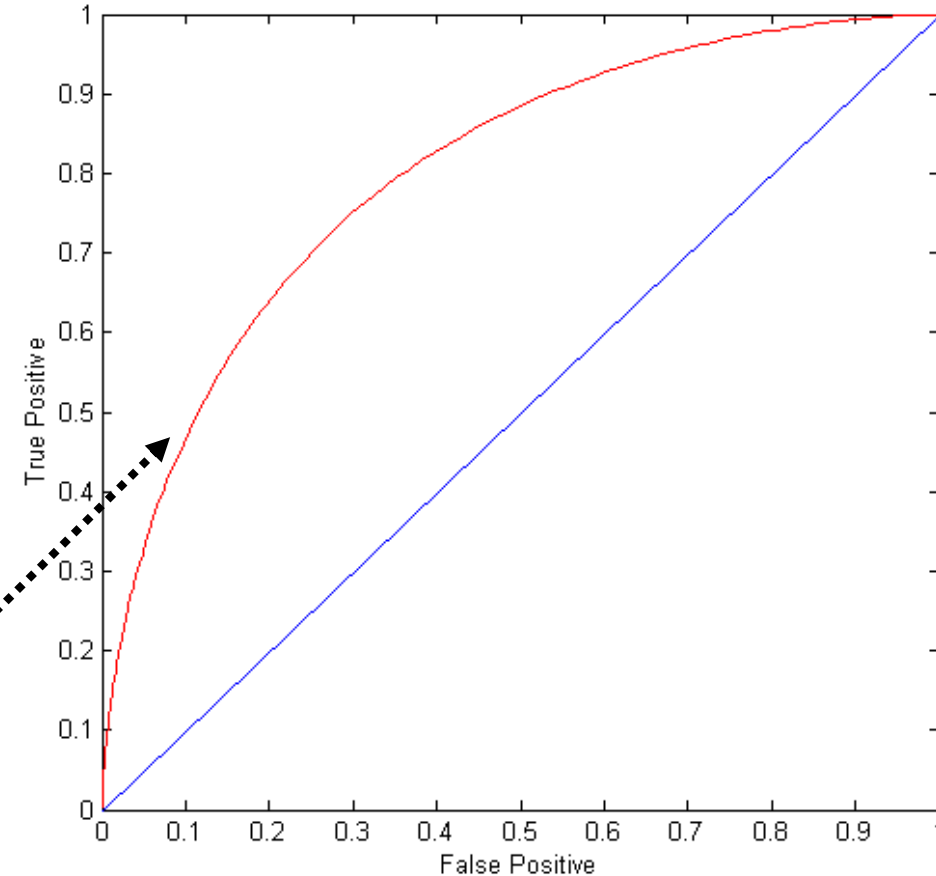
ROC Curve

- 1-dimensional data set containing 2 classes (positive and negative)
- any points located at $x > t$ is classified as positive



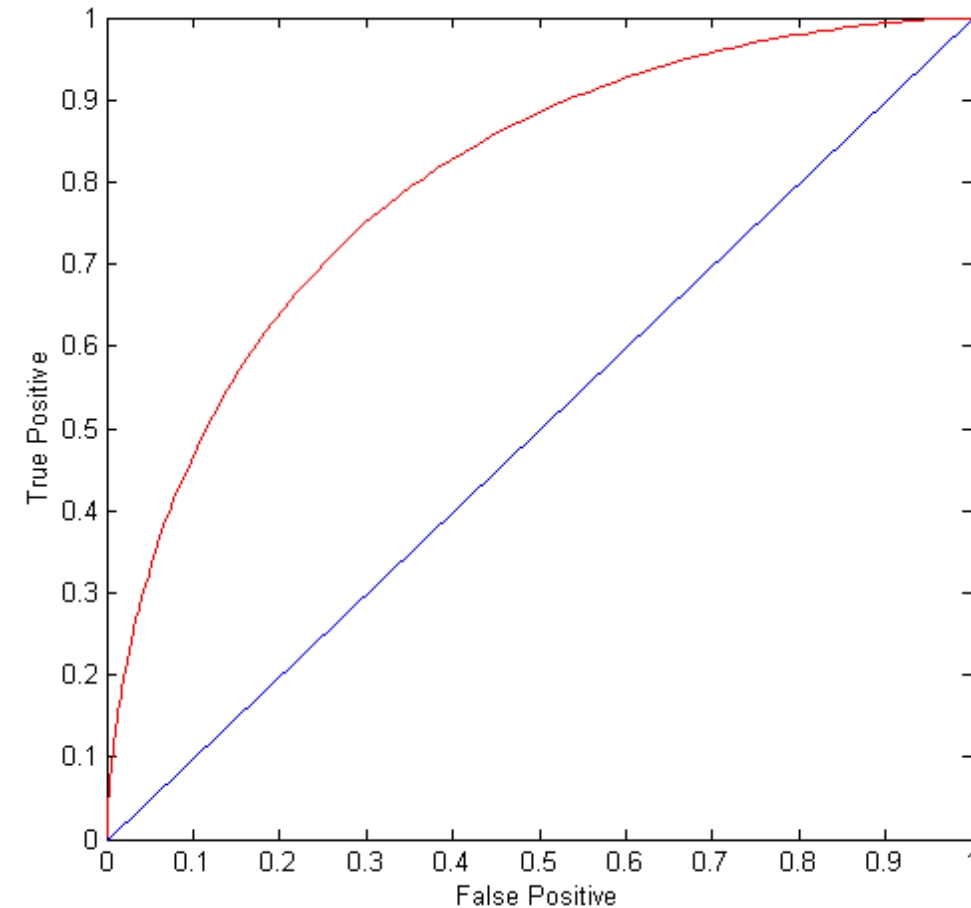
At threshold t :

TP=0.5, FN=0.5, FP=0.12, TN=0.88

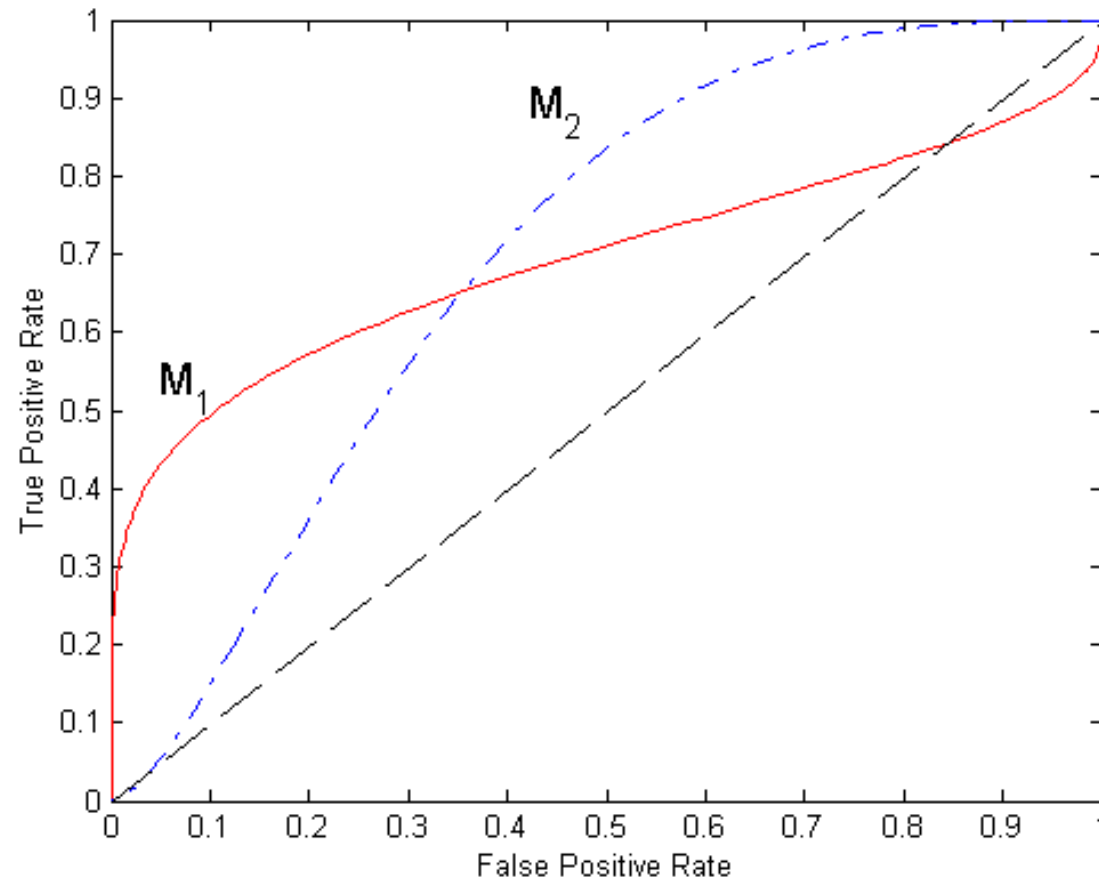


ROC Curve

- (TP,FP):
- (0,0): declare everything to be negative class
- (1,1): declare everything to be positive class
- (1,0): ideal
- Diagonal line:
 - Random guessing
 - Below diagonal line:
 - prediction is opposite of the true class



Using ROC for Model Comparison



- No model consistently outperform the other
 - M_1 is better for small FPR
 - M_2 is better for large FPR
- Area Under the ROC curve
 - Ideal: Area = 1
 - Random guess: Area = 0.5

How to Construct an ROC curve?

Instance	$P(+ A)$	True Class
1	0.95	+
2	0.93	+
3	0.87	-
4	0.85	-
5	0.85	-
6	0.85	+
7	0.76	-
8	0.53	+
9	0.43	-
10	0.25	+

- Use classifier that produces posterior probability for each test instance $P(+|A)$
- Sort the instances according to $P(+|A)$ in decreasing order
- Apply threshold at each unique value of $P(+|A)$
- Count the number of TP, FP, TN, FN at each threshold
- TP rate, $TPR = TP/(TP+FN)$
- FP rate, $FPR = FP/(FP + TN)$

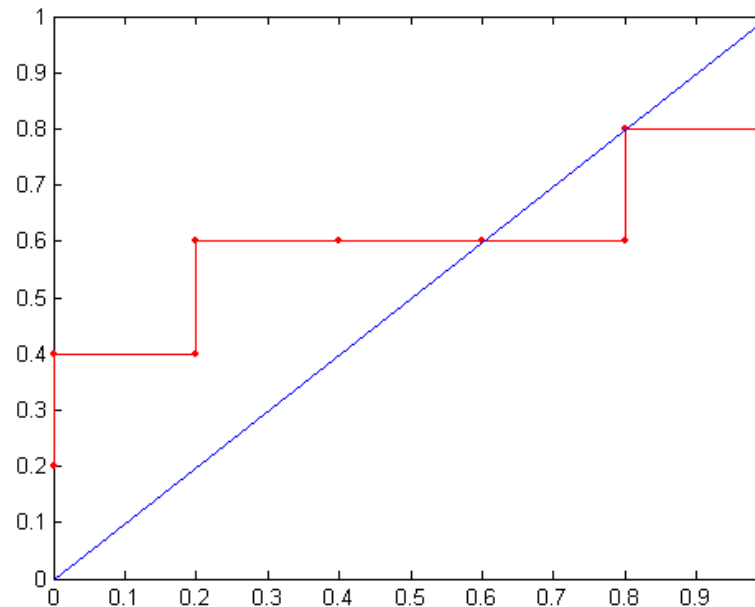
How to construct an ROC Curve?

Threshold \geq

Class	+	-	+	-	-	-	+	-	+	+	
	0.25	0.43	0.53	0.76	0.85	0.85	0.85	0.87	0.93	0.95	1.00
TP	5	4	4	3	3	3	3	2	2	1	0
FP	5	5	4	4	3	2	1	1	0	0	0
TN	0	0	1	1	2	3	4	4	5	5	5
FN	0	1	1	2	2	2	2	3	3	4	5
TPR	1	0.8	0.8	0.6	0.6	0.6	0.6	0.4	0.4	0.2	0
FPR	1	1	0.8	0.8	0.6	0.4	0.2	0.2	0	0	0

$TPR = TP / (TP + FN)$
 $FPR = FP / (FP + TN)$

ROC Curve:



Reverse of above order

$+$ $+$ $-$ $+$ $-$ $-$ $-$ $+$ $-$ $+$
 \uparrow
 $+$
 \rightarrow $-$