GENERAL ASSEMBLY

INTRO TO DATA SCIENCE

MODEL EVALUATION

DATA SCIENCE IN THE NEWS

DATA SCIENCE IN THE NEWS

Introducing Kaggle Datasets

Ben Hamner | 01.19.2016

At Kaggle, we want to help the world learn from data. This sounds bold and grandiose, but the biggest barriers to this are incredibly simple. **It's tough to access data. It's tough to understand what's in the data once you access it.** We want to change this. That's why we've created a home for high quality public datasets, Kaggle Datasets.

Kaggle Datasets has four core components:

- **Access:** simple, consistent access to the data with clear licensing
- **Analysis:** a way to explore the data without downloading it
- **Results:** visibility to the previous work that's been created on the data
- **Conversation:** forums and comments for discussing the nuances of the data

http://blog.kaggle.com/2016/01/19/introducing-kaggle-datasets/

LAST TIME:

I. DATA FORMATS
II. APIS

EXERCISES:

III. EXTENDED HANDS-ON LAB

QUESTIONS?

WHAT WAS THE MOST INTERESTING THING YOU LEARNT?

WHAT WAS THE HARDEST TO GRASP?

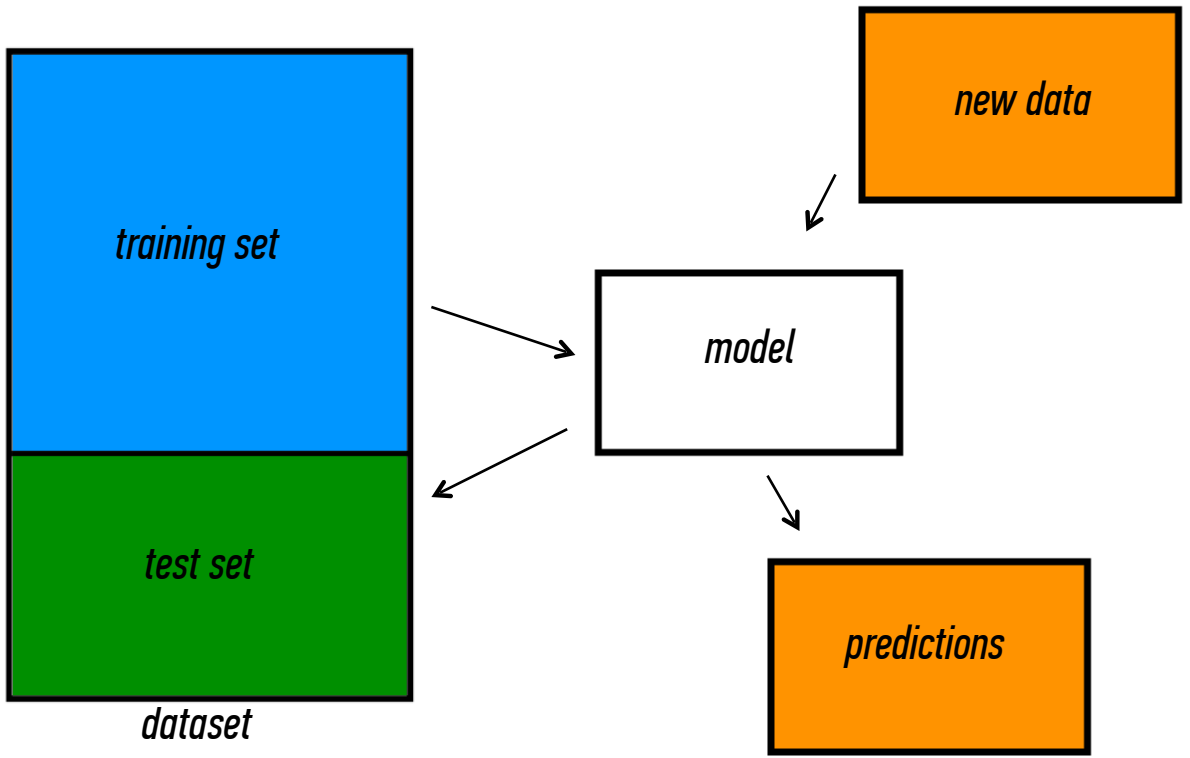
AGENDA

I. REVIEW: EVALUATION SO FAR, CROSS VALIDATION
II. ERROR RATES & CONFUSION MATRIX
III. ROC CURVES
IV. IMBALANCED CLASSES

REVIEW

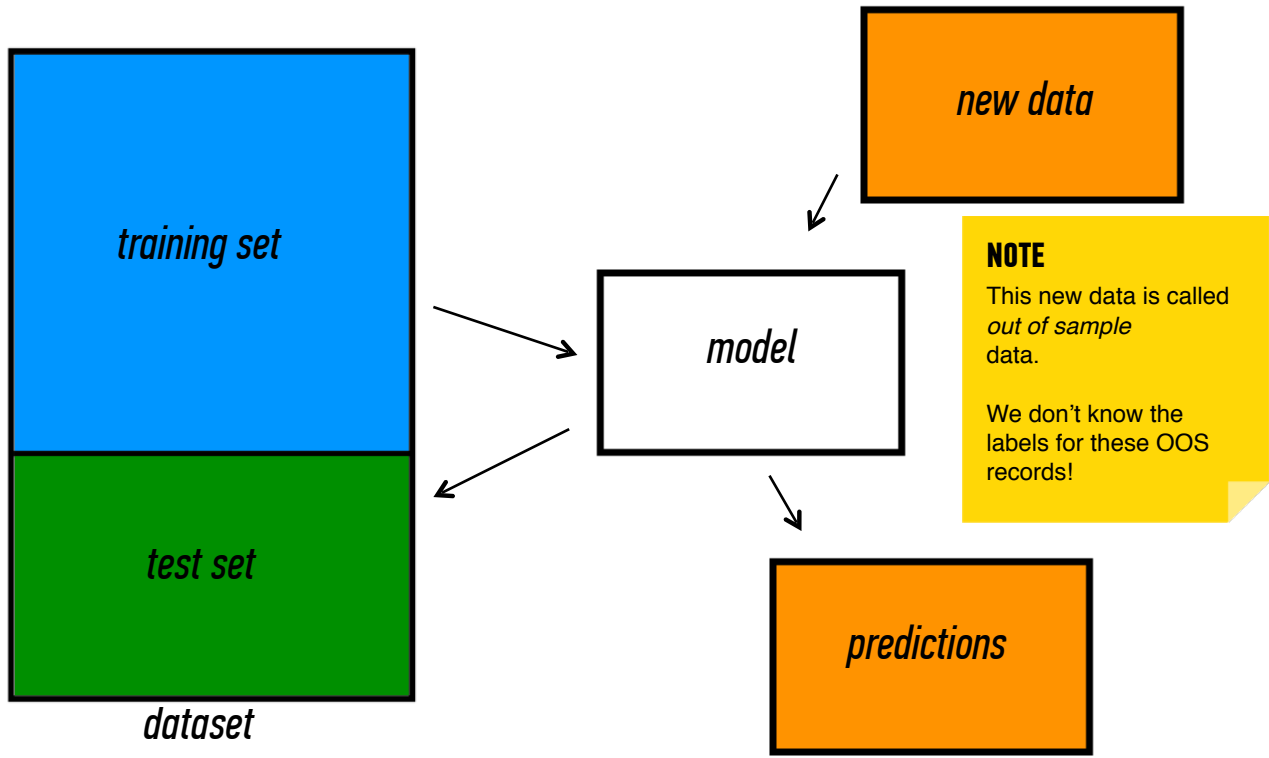
Q: What steps does a classification problem require?

1) split dataset
2) train model
3) test model
4) make predictions

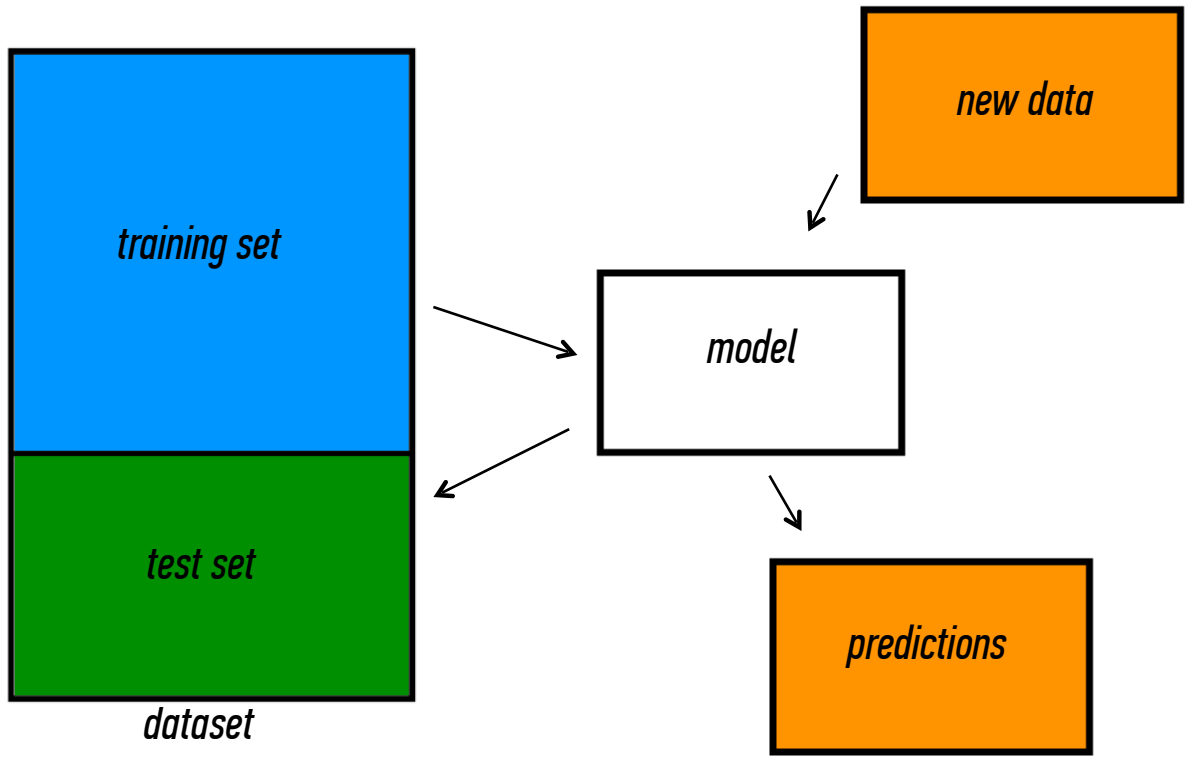


Q: What steps does a classification problem require?

1) split dataset
2) train model
3) test model
4) make predictions

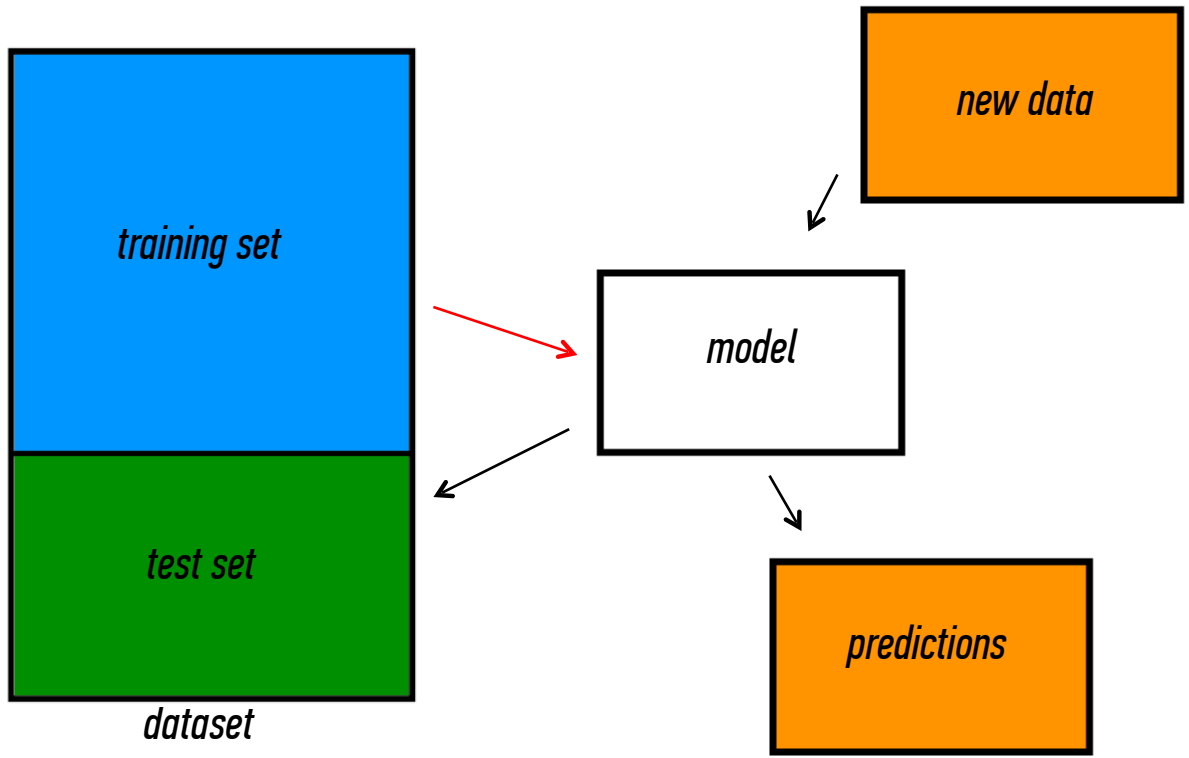


Q: What types of prediction error will we run into?



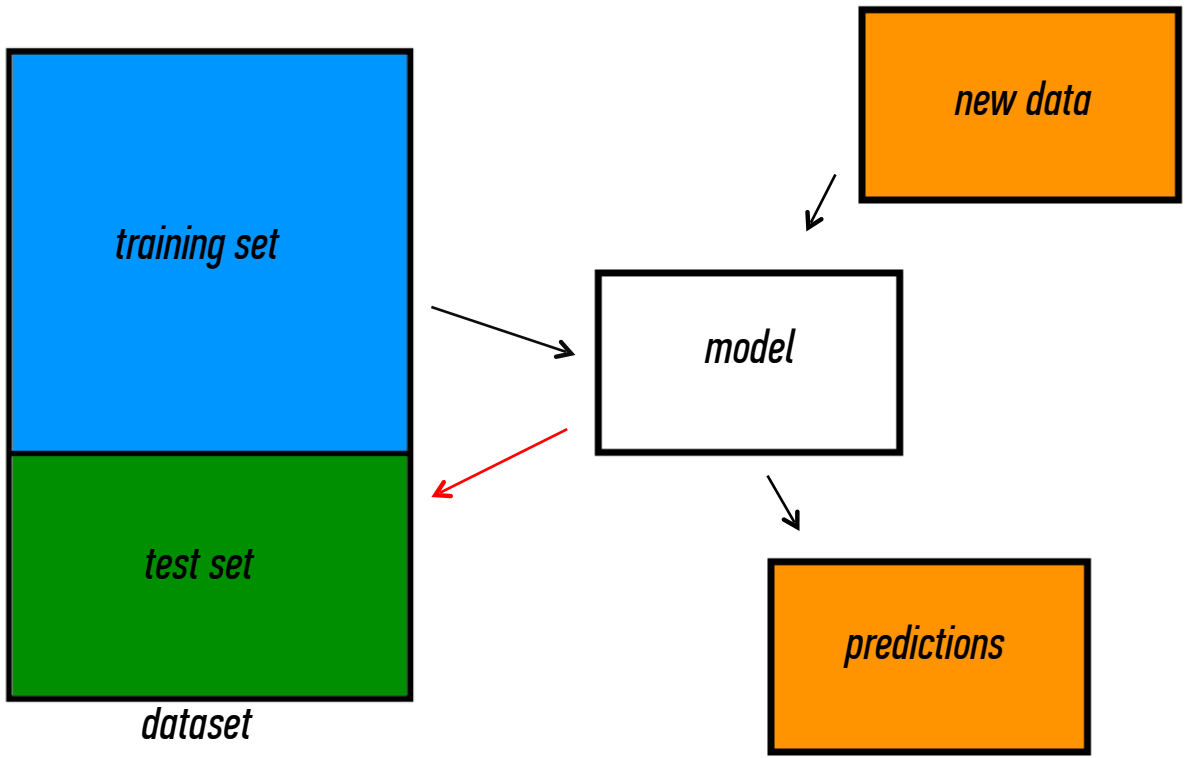
Q: What types of prediction error will we run into?

1) training error



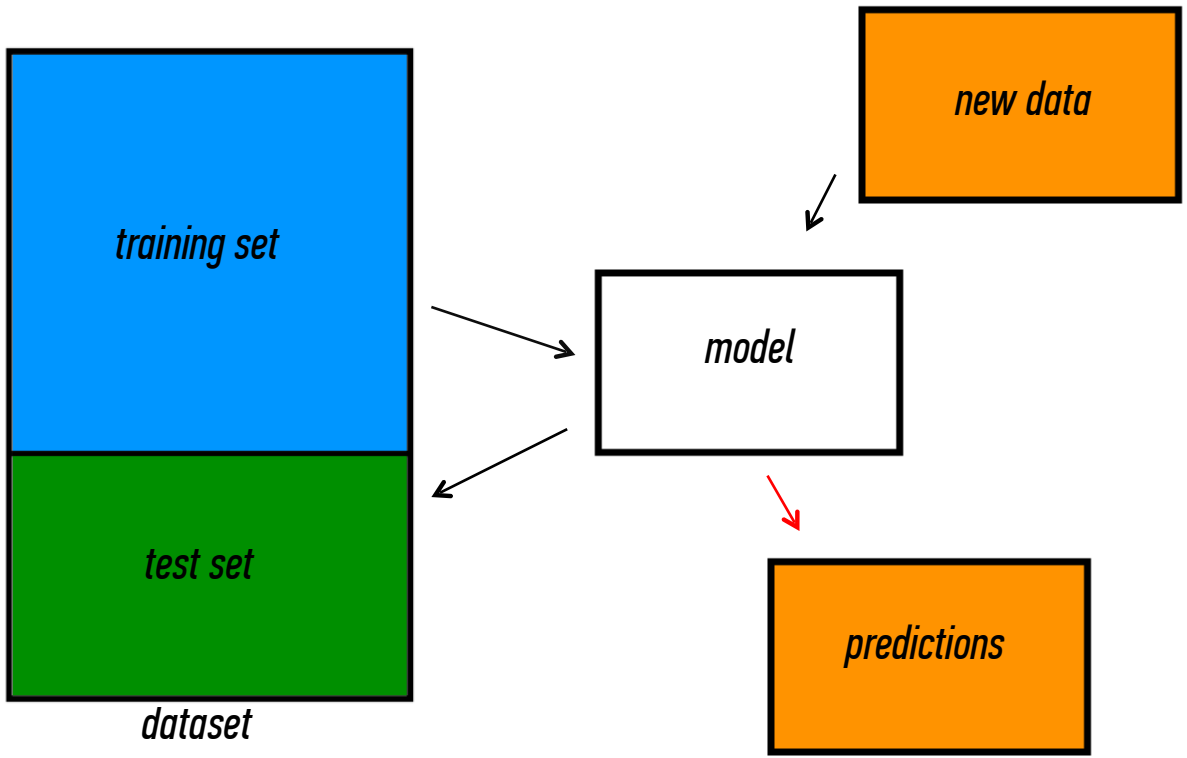
Q: What types of prediction error will we run into?

1) training error
2) generalization error



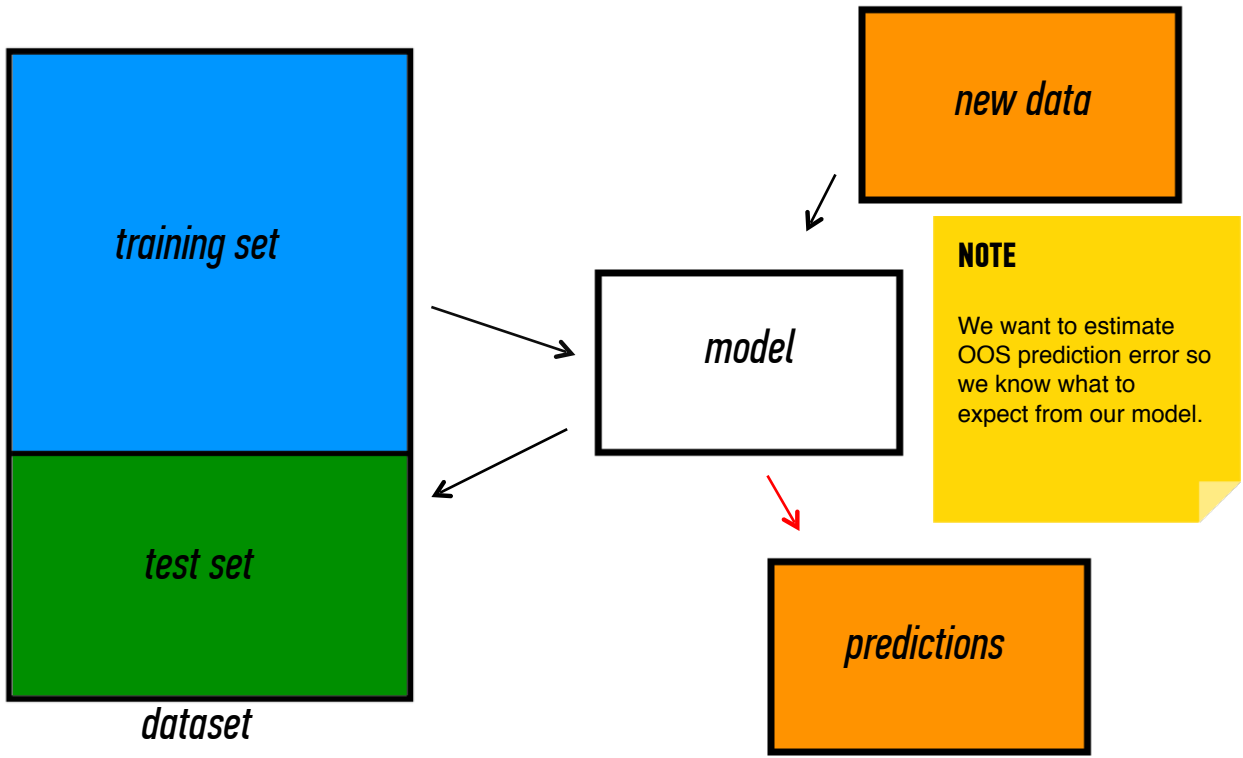
Q: What types of prediction error will we run into?

1) training error
2) generalization error
3) OOS error



Q: What types of prediction error will we run into?

1) training error
2) generalization error
3) OOS error



III. CROSS VALIDATION

Steps for n-fold cross-validation:

Steps for n-fold cross-validation:

1) Randomly split the dataset into n equal partitions.

Steps for n-fold cross-validation:

1) *Randomly split the dataset into n equal partitions.*
2) *Use partition 1 as test set & union of other partitions as training set.*

Steps for n-fold cross-validation:

1) *Randomly split the dataset into n equal partitions.*
2) *Use partition 1 as test set & union of other partitions as training set.*
3) *Find generalization error.*

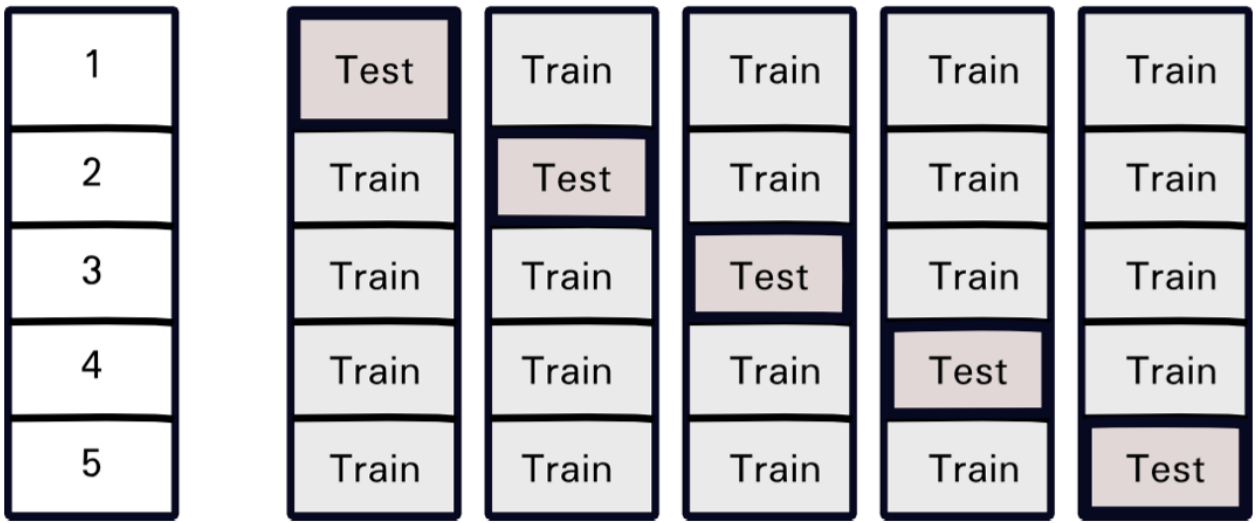
Steps for n-fold cross-validation:

1) *Randomly split the dataset into n equal partitions.*
2) *Use partition 1 as test set & union of other partitions as training set.*
3) *Find generalization error.*
4) *Repeat steps 2-3 using a different partition as the test set at each iteration.*

Steps for n-fold cross-validation:

1) *Randomly split the dataset into n equal partitions.*
2) *Use partition 1 as test set & union of other partitions as training set.*
3) *Find generalization error.*
4) *Repeat steps 2-3 using a different partition as the test set at each iteration.*
5) *Take the average generalization error as the estimate of OOS accuracy.*

CROSS-VALIDATION: 5-FOLD EXAMPLE



Accuracy

k_1 %

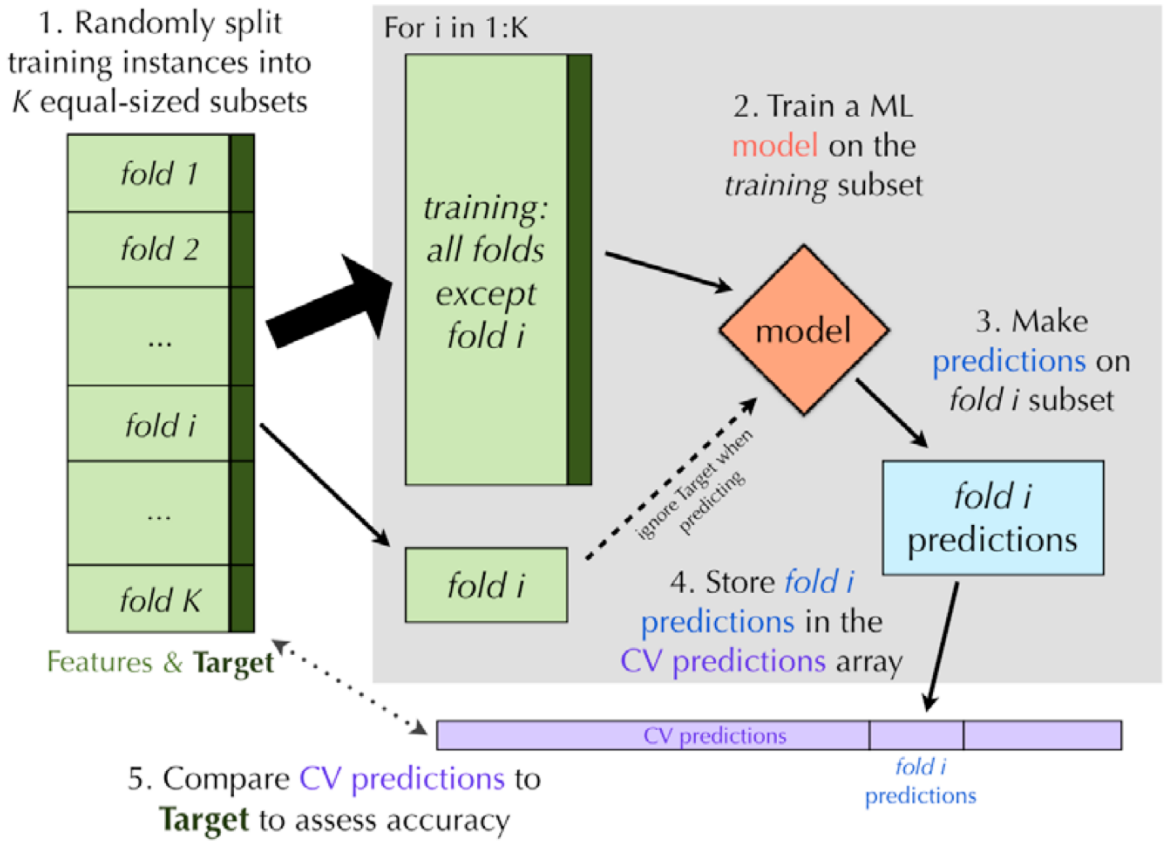
k_2 %

k_3 %

k_4 %

k_5 %

5-Fold Generalization Error = $(k_1 + k_2 + k_3 + k_4 + k_5) / 5$


1. Randomly split training instances into K equal-sized subsets
fold 1
fold 2
...
fold i
...
fold K
Features & Target
For i in 1:K
training: all folds except fold i
fold i
2. Train a ML model on the training subset
model
ignore Target when predicting
3. Make predictions on fold i subset
fold i predictions
4. Store fold i predictions in the CV predictions array
CV predictions
fold i predictions
5. Compare CV predictions to Target to assess accuracy

Features of n-fold cross-validation:

Features of n-fold cross-validation:

1) *More accurate estimate of OOS prediction error.*

Features of n-fold cross-validation:

1) *More accurate estimate of OOS prediction error.*
2) *More efficient use of data than single train/test split.*
    - *Each record in our dataset is used for both training and testing.*

Features of n-fold cross-validation:

1) *More accurate estimate of OOS prediction error.*
2) *More efficient use of data than single train/test split.*
    - *Each record in our dataset is used for both training and testing.*
3) *Presents tradeoff between efficiency and computational expense.*
    - *10-fold CV is 10x more expensive than a single train/test split*

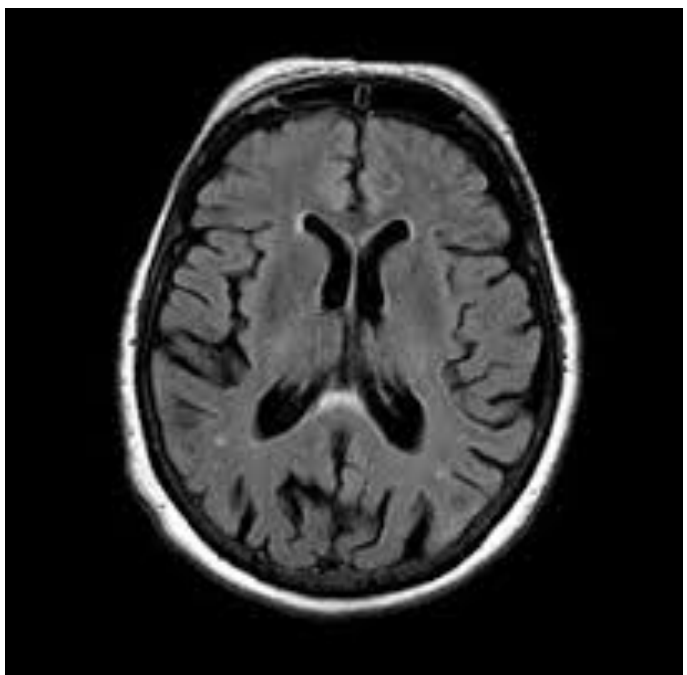
Features of n-fold cross-validation:

1) *More accurate estimate of OOS prediction error.*
2) *More efficient use of data than single train/test split.*
    - *Each record in our dataset is used for both training and testing.*
3) *Presents tradeoff between efficiency and computational expense.*
    - *10-fold CV is 10x more expensive than a single train/test split*
4) *Can be used for model selection.*

A MOTIVATING EXAMPLE

Cancer Screen => classify cancer scans for doctor to review

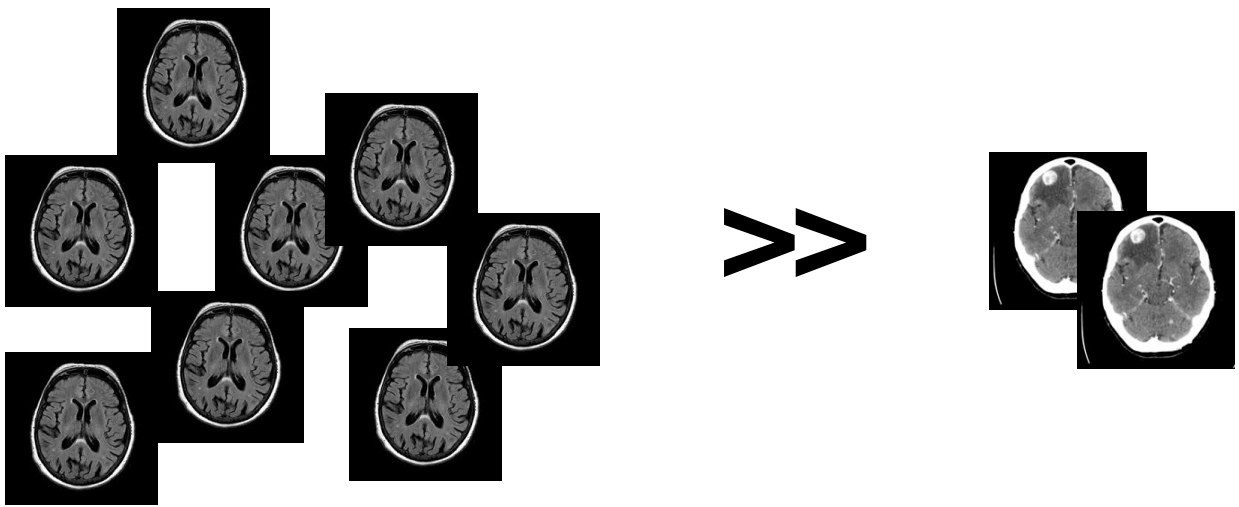
No Cancer



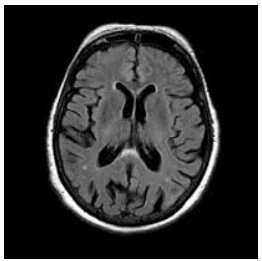
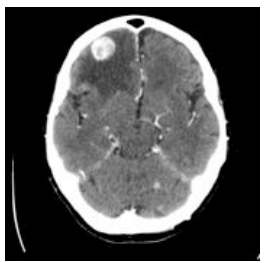
Cancer

ISSUE 1: Many more healthy brain scans

- Imbalance confuses classifiers => only perform well on dominant class
- Situation is very common in other fields (e.g. fraud detection)



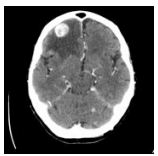
ISSUE 2: Not all errors are equal…

Error 1		Classifier Label: Cancerous	Permissable, because a physician will review it
Error 2		Classifier Label: Non-Cancerous	Not permissable, because this data will be discarded

ERROR RATES

To deal with issue 2 we need a more sophisticated definition of error rates in a binary classification problem

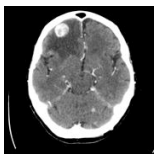
True Positive: An Example that is **positive** and is classified as **positive**



Label: **positive**

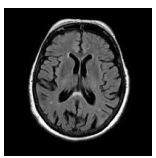
To deal with issue 2 we need a more sophisticated definition of error rates in a binary classification problem

True Positive: An Example that is **positive** and is classified as **positive**



Label: **positive**

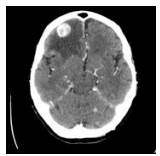
True Negative: An Example that is **negative** and is classified as **negative**



Label: **negative**

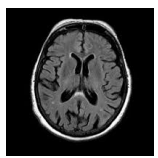
To deal with issue 2 we need a more sophisticated definition of error rates in a binary classification problem

True Positive: An Example that is **positive** and is classified as **positive**



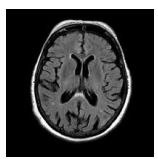
Label: **positive**

True Negative: An Example that is **negative** and is classified as **negative**



Label: **negative**

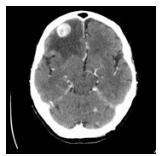
False Positive: An Example that is **negative** and is classified as **positive**



Label: **positive**

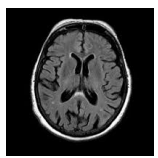
To deal with issue 2 we need a more sophisticated definition of error rates in a binary classification problem

True Positive: An Example that is **positive** and is classified as **positive**



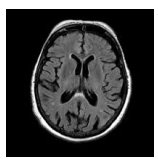
Label: **positive**

True Negative: An Example that is **negative** and is classified as **negative**



Label: **negative**

False Positive: An Example that is **negative** and is classified as **positive**



Label: **positive**

False Negative: An Example that is **positive** and is classified as **negative**

Label: **negative**

Confusion Matrix

	Condition Positive	***Condition Negative***
Test Positive	***TRUE POSITIVE***	***FALSE POSITIVE (Type I error)***
Test Negative	***FALSE NEGATIVE (Type II error)***	***TRUE NEGATIVE***

Confusion Matrix

n = 165	*Condition Positive*	*Condition Negative*
Test Positive	*100*	*10*
Test Negative	*5*	*50*

How many classes are there?
How many patients?
How many times is disease predicted?
How many patients actually have the disease?

Confusion Matrix

		Condition (as determined by "Gold standard")			
	Total population	Condition positive	Condition negative	Prevalence = $\frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	
Test outcome	Test outcome positive	**True positive**	**False positive** (Type I error)	Positive predictive value (PPV), Precision = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Test outcome positive}}$	False discovery rate (FDR) = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Test outcome positive}}$
	Test outcome negative	**False negative** (Type II error)	**True negative**	False omission rate (FOR) = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Test outcome negative}}$	Negative predictive value (NPV) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Test outcome negative}}$
	Accuracy (ACC) = $\frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$	True positive rate (TPR), Sensitivity, Recall = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) = $\frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR−}}$
		False negative rate (FNR), Miss rate = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	True negative rate (TNR), Specificity (SPC) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) = $\frac{\text{FNR}}{\text{TNR}}$	

n = 165	*Condition Positive*	*Condition Negative*
Test Positive	*100*	*10*
Test Negative	*5*	*50*

Accuracy:
Overall, how often is it **correct**?

(TP + TN) / total = 150/165 = 0.91

Precision:
When test is positive, how often is prediction correct?
TP / test yes = 100/110 = 0.91

Sensitivity/Recall/TPR:
When actual value is positive, how often is prediction correct?
TP / actual yes = 100/105 = 0.95

Specificity/TNR:
When actual value is negative, how often is prediction correct?
TN / actual no = 50/60 = 0.83

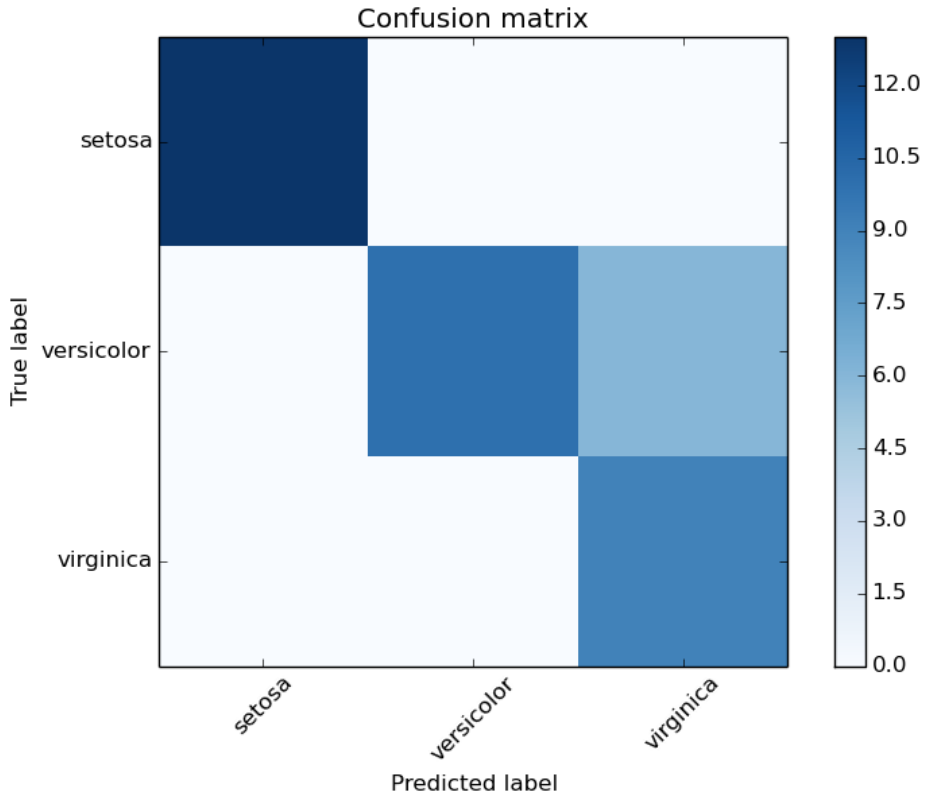
n = 165	*Condition Positive*	*Condition Negative*
Test Positive	*100*	*10*
Test Negative	*5*	*50*

Precision:
When test is positive, how often is prediction correct?
TP / test yes = 100/110 = 0.91

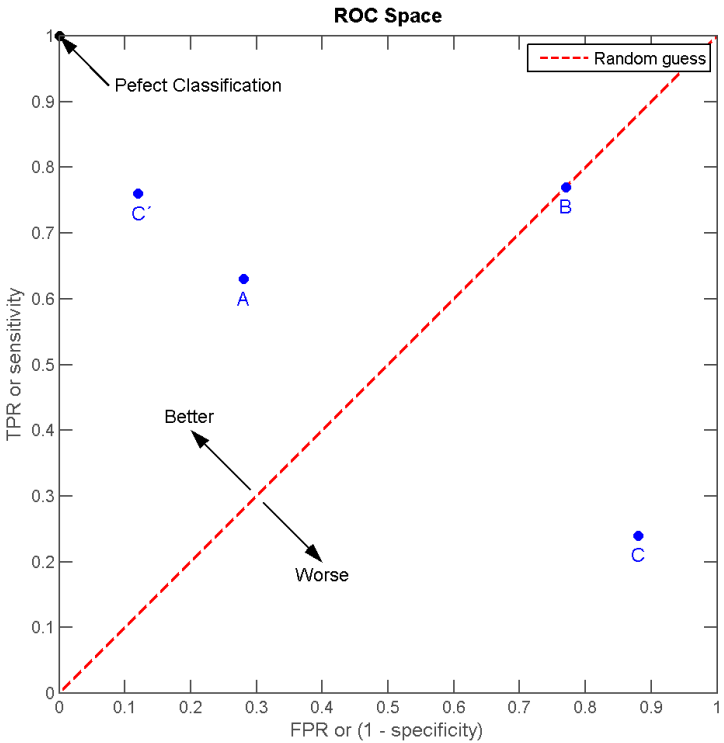
Sensitivity/Recall/TPR:
When actual value is positive, how often is prediction correct?
TP / actual yes = 100/105 = 0.95

F score

$$F = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$


Confusion matrix
setosa
versicolor
virginica
True label
setosa
versicolor
virginica
Predicted label
12.0
10.5
9.0
7.5
6.0
4.5
3.0
1.5
0.0

ROC CURVES



TP Rate = True Positives / All positives

FP Rate = False Positives / All Negatives

Email Score	True Label
0.99	Spam
0.82	Spam
0.65	Spam
0.65	Ham
0.52	Spam
0.22	Spam
0.11	Ham
0.02	Ham

Every email is assigned a "spamminess" score by our classification algorithm. To actually make our predictions, we choose a numeric cutoff for classifying as spam.

An ROC curve will help us to visualize how well our classifier is doing without having to choose a cutoff!

Email Score	True Label	Predicted Label Using 0.5 Cutoff
0.99	Spam	Spam
0.82	Spam	Spam
0.65	Spam	Spam
0.65	Ham	Spam
0.52	Spam	Spam
0.22	Spam	Ham
0.11	Ham	Ham
0.02	Ham	Ham

*Specificity: When true label is **ham**, how often is the prediction **correct**?*

*Sensitivity: When true label is **spam**, how often is the prediction **correct**?*

Email Score	True Label
0.99	Spam
0.82	Spam
0.65	Spam
0.65	Ham
0.52	Spam
0.22	Spam
0.11	Ham
0.02	Ham

Cutoff	Specificity	Sensitivity
1		
0.9		
0.8		
0.6		
0.5	2/3 = 0.66	4/5 = 0.8
0.2		
0.1		
0		

*Specificity: When true label is **ham**, how often is the prediction **correct**?*

*Sensitivity: When true label is **spam**, how often is the prediction **correct**?*

Email Score	True Label
0.99	Spam
0.82	Spam
0.65	Spam
0.65	Ham
0.52	Spam
0.22	Spam
0.11	Ham
0.02	Ham

Cutoff	Specificity	Sensitivity
1	3/3 = 1	0/5 = 0
0.9	3/3 = 1	1/5 = 0.2
0.8	3/3 = 1	2/5 = 0.4
0.6	2/3 = 0.66	3/5 = 0.6
0.5	2/3 = 0.66	4/5 = 0.8
0.2	2/3 = 0.66	5/5 = 1
0.1	1/3 = 0.33	5/5 = 1
0	0/3 = 0	5/5 = 1

*Specificity: When true label is **ham**, how often is the prediction **correct**?*

*Sensitivity: When true label is **spam**, how often is the prediction **correct**?*

Email Score	True Label
0.99	Spam
0.82	Spam
0.65	Spam
0.65	Ham
0.52	Spam
0.22	Spam
0.11	Ham
0.02	Ham

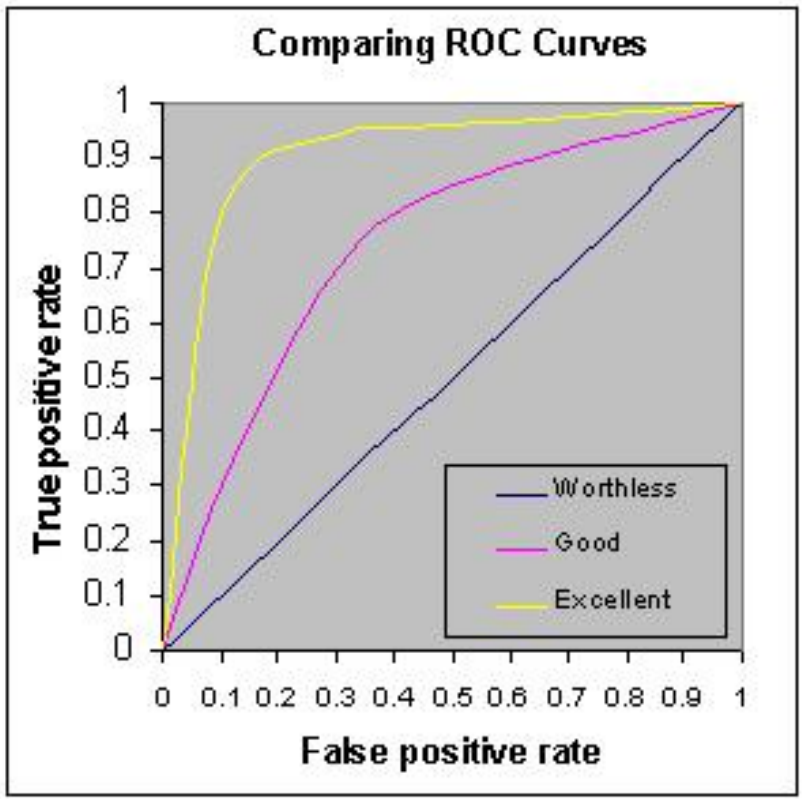
Cutoff	FPR (x-axis)	TPR (y-axis)
1	0	0
0.9	0	0.2
0.8	0	0.4
0.6	0.33	0.6
0.5	0.33	0.8
0.2	0.33	1
0.1	0.66	1
0	1	1

FPR (x-axis) = 1-Specificity

TPR (y-axis) = Sensitivity

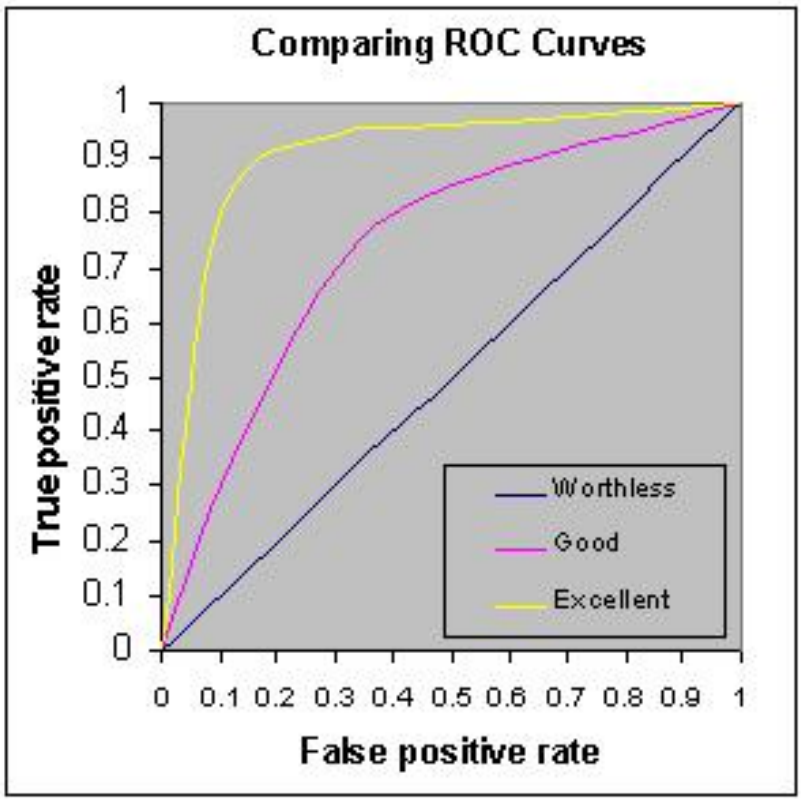
Q: On the ROC curve, can you see the cutoff that was used to generate a point?

A: No, that information is not visible.



ROC Curves show the relationship between the TP Rate and the FP Rate as we vary the decision threshold for the classifier

Cut off	TPR (y)	FPR (x)	Cut off	TPR (y)	FPR (x)
0	1	1	0.50	0.75	0.25
0.05	1	0.75	0.65	0.5	0
0.15	1	0.5	0.85	0.25	0
0.25	1	0.25	1	0	0



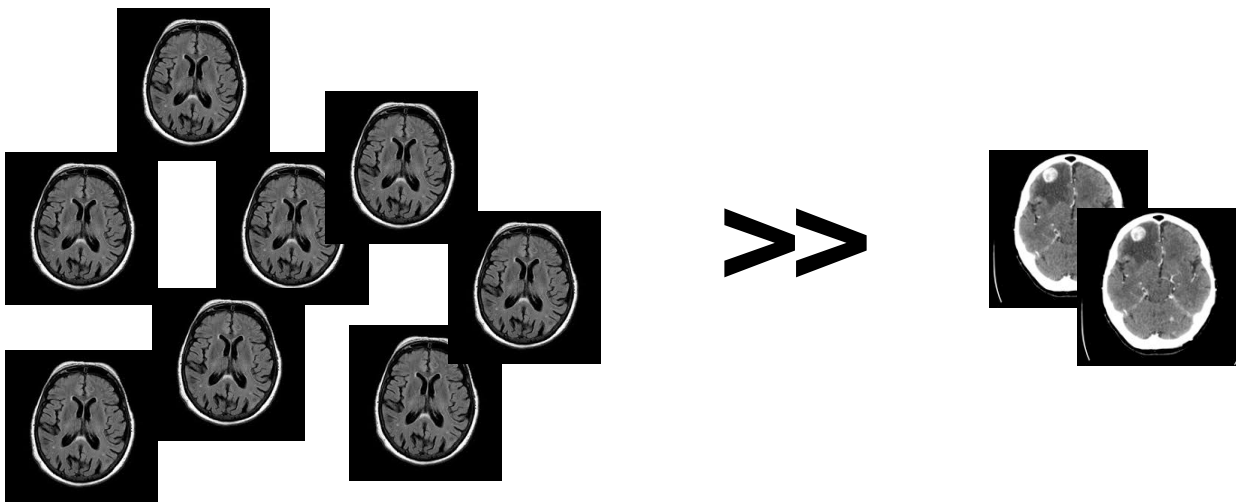
Area Under the Curve (AUC)

We evaluate a classifier by measuring the Area Under the Curve for its ROC curve. The Greater area under the curve, the more effective the classifier.

Then for our chosen classifer, we pick an appropriate decision threshold. In general, we pick the decision threshold that gets us closest to the upper left corner

IMBALANCED CLASSES

Imbalanced classes can be re-balanced in several ways



Imbalanced classes can be re-balanced in several ways

1. **Undersampling** the dominant class - remove some the majority class so it has less weight

Imbalanced classes can be re-balanced in several ways

1. **Undersampling** the dominant class - remove some the majority class so it has less weight

2. **Oversampling** the minority class - add more of the minority class so it has more weight.

Imbalanced classes can be re-balanced in several ways

1. **Undersampling** the dominant class - remove some the majority class so it has less weight

2. **Oversampling** the minority class - add more of the minority class so it has more weight.

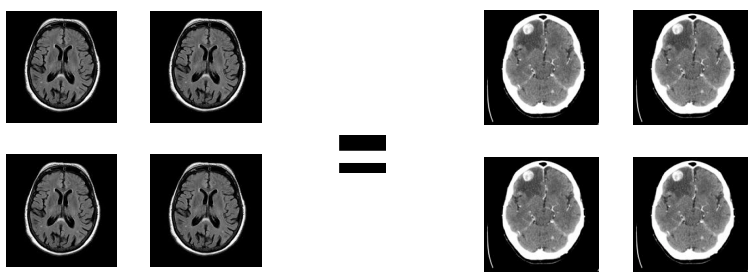
3. **Hybrid** - doing both

Imbalanced classes can be re-balanced in several ways

1. **Undersampling** the dominant class - remove some the majority class so it has less weight

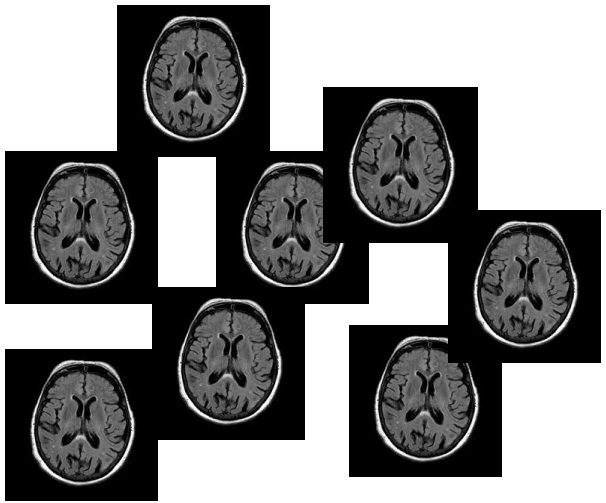
2. **Oversampling** the minority class - add more of the minority class so it has more weight.

3. **Hybrid** - doing both



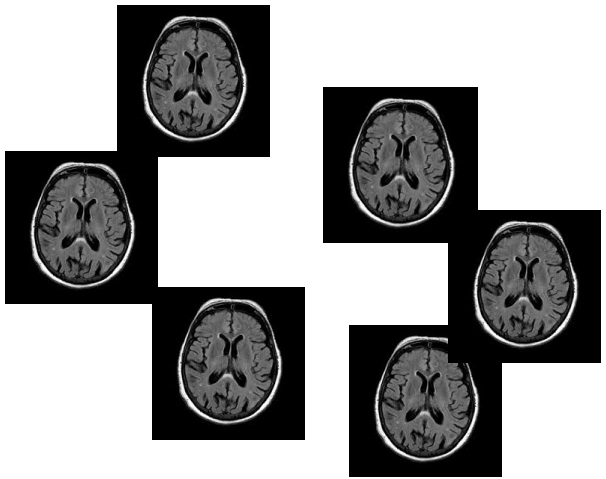
Undersampling

Randomly remove elements from the majority class.



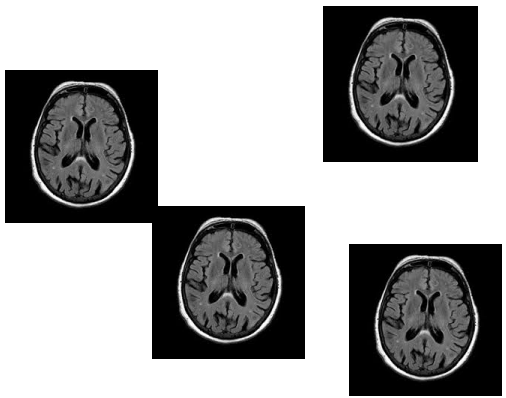
Undersampling

Randomly remove elements from the majority class.



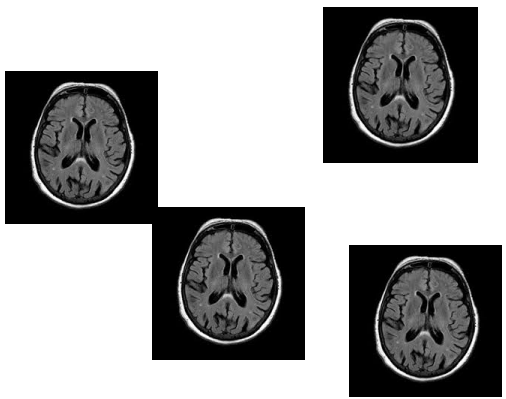
Undersampling

Randomly remove elements from the majority class.



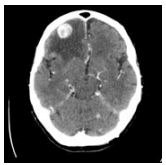
Undersampling

Randomly remove elements from the majority class.



Drawback: Removing data points could lose important information

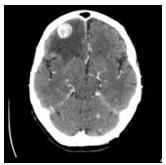
Oversampling

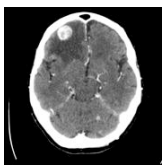


Duplicate elements of your minority class

Oversampling

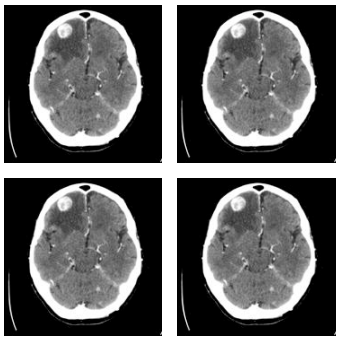
Duplicate elements of your minority class





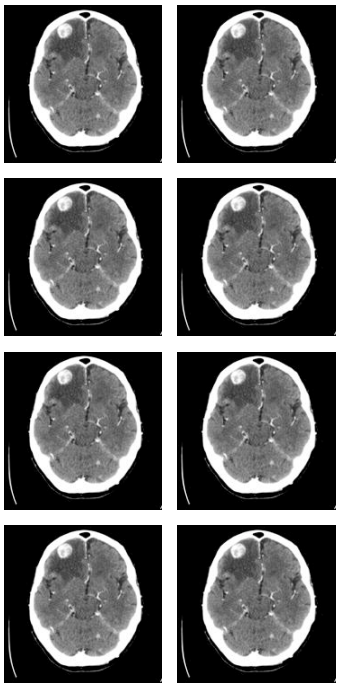
Oversampling

Duplicate elements of your minority class



Oversampling

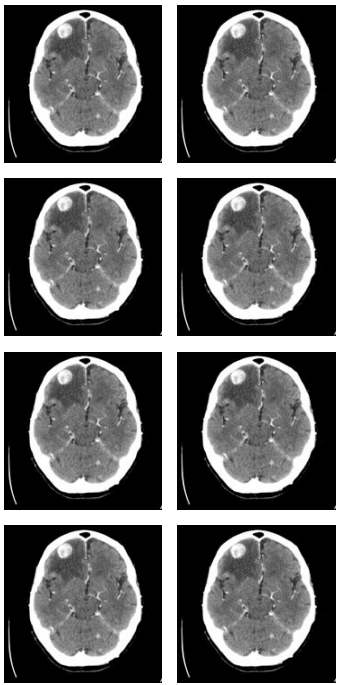
Duplicate elements of your minority class



Oversampling

Duplicate elements of your minority class

Drawback: Just replicating randomly minority classes could cause overfit



OTHER EVALUATION METRICS

REGRESSION METRICS

RMSE

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

- Used for regression problems
- Square root of the mean of the squared errors
- Easily interpretable (in the "y" units)
- "Punishes" larger errors

RMSE

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Example:
y_true = [100, 50, 30]
y_preds = [90, 50, 50]
RMSE = np.sqrt((10**2 + 0**2 + 20**2) / 3) = 12.88

EXPLAINED VARIANCE

$$\texttt{explained\_variance}(y, \hat{y}) = 1 - \frac{Var\{y - \hat{y}\}}{Var\{y\}}$$

Example:
y_true = [3, -0.5, 2, 7]
y_pred = [2.5, 0.0, 2, 8]
explained_variance(y_true, y_pred) = 0.957

Mean Absolute Error

$$\mathrm{MAE}(y, \hat{y}) = \frac{1}{n_{\mathrm{samples}}} \sum_{i=0}^{n_{\mathrm{samples}}-1} |y_i - \hat{y}_i| .$$

Example:
y_true = [3, -0.5, 2, 7]
y_pred = [2.5, 0.0, 2, 8]
mean_absolute_error(y_true, y_pred) = 0.5

Median Absolute Error

$$\text{MedAE}(y, \hat{y}) = \text{median}(| \ y_1 - \hat{y}_1 \ |, \ldots, | \ y_n - \hat{y}_n \ |).$$

Particularly interesting because it's robust to outliers.

Classification Metrics

Accuracy Score

$$\texttt{accuracy}(y, \hat{y}) = \frac{1}{n_{\text{samples}}} \sum_{i=0}^{n_{\text{samples}}-1} 1(\hat{y}_i = y_i)$$

i.e. the relative frequency of accurate predictions.