Evaluating Classifiers

Machine Learning

PHYS 453 – Spring 2022

Dr. Daugherity

Abilene Christian University

Evaluating Classifiers

- How can I measure how well a classifier works?
- Where do I look for ways to improve performance?

Sources:

- https://scikit-learn.org/stable/modules/model_evaluation.html#classification-metrics
- Binary Classification Metrics paper, on canvas or: https://arxiv.org/pdf/1410.5330

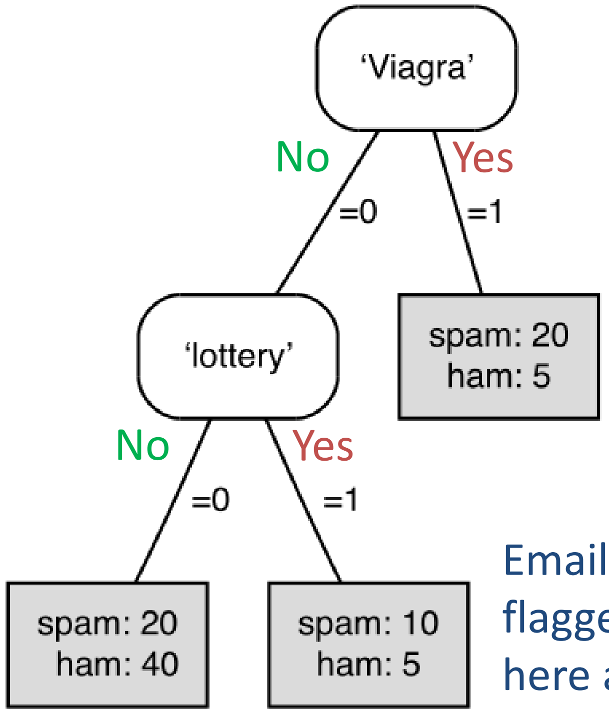
Evaluating Classifiers

CONFUSION MATRIX

Confusion Matrix

		Predicted class	
		P	*N*
Actual Class	*P*	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

Spam detection decision tree



Email that contains word Viagra is flagged as spam. 20 of 25 emails here are actually spam

Email that contains neither is flagged as ham.

Email that contains word lottery is flagged as spam. 10 of 15 emails here are actually spam

	SPAM Predicted ⊕	HAM Predicted ⊖	
Actual ⊕	30	20	50
Actual ⊖	10	40	50
	40	60	100

Accuracy Metrics

		Predicted class	
		P	N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

$$ERR = \frac{FP + FN}{FP + FN + TP + TN} = 1 - ACC$$

Error %

$$ACC = \frac{TP + TN}{FP + FN + TP + TN} = 1 - ERR$$

Accuracy %

$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN}$$

False Positive Rate = (# of FP) / (# actually N)
"what percentage of the real N did I miss?"

$$TPR = \frac{TP}{P} = \frac{TP}{FN + TP}$$

True Positive Rate = (# of TP) / (# actually P)
"what percentage of the real P did I get?"

$$PRE = \frac{TP}{TP + FP}$$

PRECISION = the ability of the classifier not to label as positive a sample that is negative.
Fraction of pos guesses that are right.

$$REC = TPR = \frac{TP}{P} = \frac{TP}{FN + TP}$$

RECALL = the ability of the classifier to find all the positive samples
Fraction of all actual pos we guessed as pos.

$$F_1 = 2 \cdot \frac{PRE \cdot REC}{PRE + REC}$$

F1 Score = combines both into a single number. 1 is perfect.

Challenge: gotta find them all!

	Predicted ⊕	Predicted ⊖	
Actual ⊕	30	20	50
Actual ⊖	10	40	50
	40	60	100

Predicted class

		P	N
Actual Class	P	True Positives (TP)	False Negatives (FN)
	N	False Positives (FP)	True Negatives (TN)

$$ERR = \frac{FP + FN}{FP + FN + TP + TN} = 1 - ACC$$

$$ACC = \frac{TP + TN}{FP + FN + TP + TN} = 1 - ERR$$

$$FPR = \frac{FP}{N} = \frac{FP}{FP + TN}$$

$$TPR = \frac{TP}{P} = \frac{TP}{FN + TP}$$

$$PRE = \frac{TP}{TP + FP}$$

$$REC = TPR = \frac{TP}{P} = \frac{TP}{FN + TP}$$

$$F_1 = 2 \cdot \frac{PRE \cdot REC}{PRE + REC}$$

Measure	Definition	Equal to	Estimates
number of positives	$Pos = \sum_{x \in Te} I[c(x) = \oplus]$		
number of negatives	$Neg = \sum_{x \in Te} I[c(x) = \ominus]$	$\|Te\| - Pos$	
number of true positives	$TP = \sum_{x \in Te} I[\hat{c}(x) = c(x) = \oplus]$		
number of true negatives	$TN = \sum_{x \in Te} I[\hat{c}(x) = c(x) = \ominus]$		
number of false positives	$FP = \sum_{x \in Te} I[\hat{c}(x) = \oplus, c(x) = \ominus]$	$Neg - TN$	
number of false negatives	$FN = \sum_{x \in Te} I[\hat{c}(x) = \ominus, c(x) = \oplus]$	$Pos - TP$	
proportion of positives	$pos = \frac{1}{\|Te\|} \sum_{x \in Te} I[c(x) = \oplus]$	$Pos/\|Te\|$	$P(c(x) = \oplus)$
proportion of negatives	$neg = \frac{1}{\|Te\|} \sum_{x \in Te} I[c(x) = \ominus]$	$1 - pos$	$P(c(x) = \ominus)$
class ratio	$clr = pos/neg$	Pos/Neg	
(*) accuracy	$acc = \frac{1}{\|Te\|} \sum_{x \in Te} I[\hat{c}(x) = c(x)]$		$P(\hat{c}(x) = c(x))$
(*) error rate	$err = \frac{1}{\|Te\|} \sum_{x \in Te} I[\hat{c}(x) \neq c(x)]$	$1 - acc$	$P(\hat{c}(x) \neq c(x))$

Measure	Definition	Equal to	Estimates
true positive rate, sensitivity, recall	$tpr = \frac{\sum_{x \in Te} I[\hat{c}(x)=c(x)=\oplus]}{\sum_{x \in Te} I[c(x)=\oplus]}$	TP/Pos	$P(\hat{c}(x) = \oplus \| c(x) = \oplus)$
true negative rate, specificity	$tnr = \frac{\sum_{x \in Te} I[\hat{c}(x)=c(x)=\ominus]}{\sum_{x \in Te} I[c(x)=\ominus]}$	TN/Neg	$P(\hat{c}(x) = \ominus \| c(x) = \ominus)$
false positive rate, false alarm rate	$fpr = \frac{\sum_{x \in Te} I[\hat{c}(x)=\oplus, c(x)=\ominus]}{\sum_{x \in Te} I[c(x)=\ominus]}$	$FP/Neg = 1 - tnr$	$P(\hat{c}(x) = \oplus \| c(x) = \ominus)$
false negative rate	$fnr = \frac{\sum_{x \in Te} I[\hat{c}(x)=\ominus, c(x)=\oplus]}{\sum_{x \in Te} I[c(x)=\oplus]}$	$FN/Pos = 1 - tpr$	$P(\hat{c}(x) = \ominus \| c(x) = \oplus)$
precision, confidence	$prec = \frac{\sum_{x \in Te} I[\hat{c}(x)=c(x)=\oplus]}{\sum_{x \in Te} I[\hat{c}(x)=\oplus]}$	$TP/(TP + FP)$	$P(c(x) = \oplus \| \hat{c}(x) = \oplus)$

Table : A summary of different quantities and evaluation measures for classifiers on a test set Te. Symbols starting with a capital letter denote absolute frequencies (counts), while lower-case symbols denote relative frequencies or ratios. All except those indicated with (*) are defined only for binary classification.

A slightly different version of the same thing, just in case…

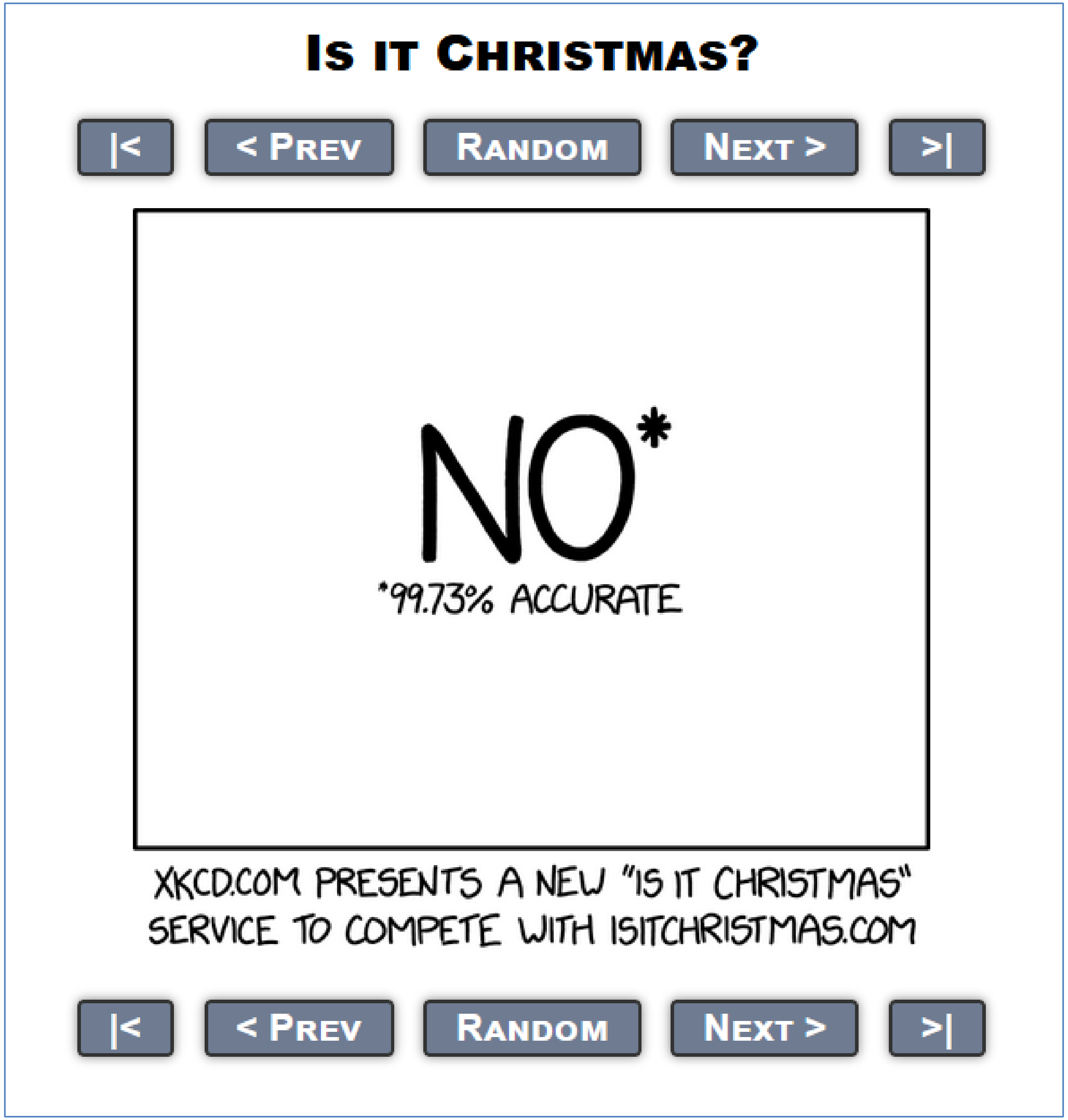
Suppose a classifier's predictions on a test set are as in the following table:

	Predicted ⊕	*Predicted* ⊖	
Actual ⊕	**60**	**15**	75
Actual ⊖	**10**	**15**	25
	70	30	100

From this table, we see that the true positive rate is $tpr = 60/75 = 0.80$ and the true negative rate is $tnr = 15/25 = 0.60$. The overall accuracy is $acc = (60 + 15)/100 = 0.75$, which is no longer the average of true positive and negative rates. However, taking into account the proportion of positives $pos = 0.75$ and the proportion of negatives $neg = 1 - pos = 0.25$, we see that

$$acc = pos \cdot tpr + neg \cdot tnr$$

https://xkcd.com/2236/



https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html

sklearn.metrics.confusion_matrix

`sklearn.metrics.confusion_matrix`(*y_true, y_pred, *, labels=None, sample_weight=None, normalize=None*) [source]

Compute confusion matrix to evaluate the accuracy of a classification.

By definition a confusion matrix C is such that $C_{i,j}$ is equal to the number of observations known to be in group i and predicted to be in group j.

Thus in binary classification, the count of true negatives is $C_{0,0}$, false negatives is $C_{1,0}$, true positives is $C_{1,1}$ and false positives is $C_{0,1}$.

Read more in the User Guide.

Parameters:

y_true : *array-like of shape (n_samples,)*
Ground truth (correct) target values.

y_pred : *array-like of shape (n_samples,)*
Estimated targets as returned by a classifier.

labels : *array-like of shape (n_classes), default=None*
List of labels to index the matrix. This may be used to reorder or select a subset of labels. If `None` is given, those that appear at least once in `y_true` or `y_pred` are used in sorted order.

sample_weight : *array-like of shape (n_samples,), default=None*
Sample weights.

New in version 0.18.

normalize : *{'true', 'pred', 'all'}, default=None*
Normalizes confusion matrix over the true (rows), predicted (columns) conditions or all the population. If None, confusion matrix will not be normalized.

```
>>> from sklearn.metrics import confusion_matrix
>>> y_true = [2, 0, 2, 2, 0, 1]
>>> y_pred = [0, 0, 2, 2, 0, 2]
>>> confusion_matrix(y_true, y_pred)
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]])
```

```
>>> y_true = ["cat", "ant", "cat", "cat", "ant", "bird"]
>>> y_pred = ["ant", "ant", "cat", "cat", "ant", "cat"]
>>> confusion_matrix(y_true, y_pred, labels=["ant", "bird", "cat"])
array([[2, 0, 0],
       [0, 0, 1],
       [1, 0, 2]])
```

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html#sklearn.metrics.classification_report

```
>>> from sklearn.metrics import classification_report
>>> y_true = [0, 1, 2, 2, 2]
>>> y_pred = [0, 0, 2, 2, 1]
>>> target_names = ['class 0', 'class 1', 'class 2']
>>> print(classification_report(y_true, y_pred, target_names=target_names))
              precision    recall  f1-score   support

     class 0       0.50      1.00      0.67         1
     class 1       0.00      0.00      0.00         1
     class 2       1.00      0.67      0.80         3

    accuracy                           0.60         5
   macro avg       0.50      0.56      0.49         5
weighted avg       0.70      0.60      0.61         5
```

Overall accuracy = 0.60

Chapter 2.1

Difficulty: 2

There are 20 dogs(+) and 10 cats(-). A binary classifier correctly predicts 5 dogs and incorrectly predicts 5 cats. Fill in the following contingency for this binary classifier matrix.

	Predicted +	Predicted -	
Actual +			
Actual -			

Chapter 2.1

Difficulty: 2

Given that:

Total = 100
False Negatives = 10
Precision = 4/5
Recall = 6/7
Can you complete the contingency matrix.

	Predicted +	Predicted -	
Actual +			
Actual -			

Chapter 2.1

Difficulty: 4

Two binary classifiers are used to predicted whether a patent has a life threatening diseases or not. Decide whether Classifier A or B would be better at reducing casualties.

A

10	10	20
40	9940	9980
50	9950	10000

B

13	7	20
87	9813	9980
100	9900	10000

Two contingence matrices.

Tutorials

- https://github.com/mdaugherity/PatternRecognition2018/blob/master/Tutorial%203-1.ipynb

- https://github.com/mdaugherity/PatternRecognition2018/blob/master/Tutorial%203-2.ipynb

Summary

Know the following:

- Accuracy / error rate
- TP, FP, TN, FN in confusion matrix
- Precision
- Recall