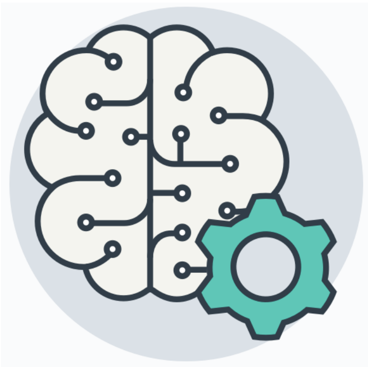
Aprendizado de Máquina

Métricas para Classificação



Prof. Regis Pires Magalhães
regismagalhaes@ufc.br - http://bit.ly/ufcregis

Métricas para classificação

		Actual Class y	
		Positive	Negative
$h_\theta(x)$ **Predicted outcome**	Predicted positive outcome	**True positive** (TP)	**False positive** (FP)
	Predicted negative outcome	**False negative** (FN)	**True negative** (TN)

Métricas para classificação

<table>
<tr><td colspan="2"></td><td colspan="2">True condition</td><td colspan="3"></td></tr>
<tr><td></td><td>Total population</td><td>Condition positive</td><td>Condition negative</td><td>Prevalence $= \frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$</td><td colspan="2">Accuracy (ACC) = $\frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$</td></tr>
<tr><td rowspan="2">Predicted condition</td><td>Predicted condition positive</td><td>True positive,
Power</td><td>False positive,
Type I error</td><td>Positive predictive value (PPV), Precision = $\frac{\Sigma \text{ True positive}}{\Sigma \text{ Predicted condition positive}}$</td><td colspan="2">False discovery rate (FDR) = $\frac{\Sigma \text{ False positive}}{\Sigma \text{ Predicted condition positive}}$</td></tr>
<tr><td>Predicted condition negative</td><td>False negative,
Type II error</td><td>True negative</td><td>False omission rate (FOR) = $\frac{\Sigma \text{ False negative}}{\Sigma \text{ Predicted condition negative}}$</td><td colspan="2">Negative predictive value (NPV) = $\frac{\Sigma \text{ True negative}}{\Sigma \text{ Predicted condition negative}}$</td></tr>
<tr><td colspan="2"></td><td>True positive rate (TPR), Recall, Sensitivity,
probability of detection
$= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$</td><td>False positive rate (FPR), Fall-out,
probability of false alarm
$= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$</td><td>Positive likelihood ratio (LR+)
$= \frac{\text{TPR}}{\text{FPR}}$</td><td rowspan="2">Diagnostic odds ratio (DOR) = $\frac{\text{LR+}}{\text{LR−}}$</td><td rowspan="2">$F_1$ score = $\frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$</td></tr>
<tr><td colspan="2"></td><td>False negative rate (FNR),
Miss rate $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$</td><td>True negative rate (TNR), Specificity (SPC)
$= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$</td><td>Negative likelihood ratio (LR−)
$= \frac{\text{FNR}}{\text{TNR}}$</td></tr>
</table>

Métricas para classificação

Table of error types		Null hypothesis (H_0) is	
		True	**False**
Decision About Null Hypothesis (H_0)	**Fail to reject**	Correct inference (True Positive)	Type II error (False Negative)
	Reject	Type I error (False Positive)	Correct inference (True Negative)

- A type I error occurs when the null hypothesis (H_0) is true, but is rejected.
  - Hypothesis: "Adding water to toothpaste protects against cavities."
  - Null hypothesis (H_0): "Adding water does not make toothpaste more effective in fighting cavities."
- A type II error occurs when the null hypothesis is false, but erroneously fails to be rejected.
  - Hypothesis: "Adding fluoride to toothpaste protects against cavities."
  - Null hypothesis (H0): "Adding fluoride to toothpaste has no effect on cavities."

Métricas para classificação

Measure	Formula
ACC	(TP + TN) / (TP + TN + FN + FP)
ERR	(FP + FN) / (TP + TN + FN + FP)
SN, TPR, REC	TP / (TP + FN)
SP	TN / (TN + FP)
FPR	FP / (TN + FP)
PREC, PPV	TP / (TP + FP)
MCC	$(TP * TN—FP * FN) / ((TP + FP)(TP + FN)(TN + FP)(TN + FN))^{1/2}$
$F_{0.5}$	1.5 * PREC * REC / (0.25 * PREC + REC)
F_1	2 * PREC * REC / (PREC + REC)
F_2	5 * PREC * REC / (4 * PREC + REC)

ACC: accuracy; ERR: error rate; SN: sensitivity; TPR: true positive rate; REC: recall; SP: specificity; FPR: false positive rate; PREC: precision; PPV: positive predictive value; MCC: Matthews correlation coefficient; F: F score; TP: true positives; TN: true negatives; FP: false positives; FN: false negatives

doi:10.1371/journal.pone.0118432.t001

Confusion Matrix

		Actual class		
		Cat	Dog	Rabbit
Predicted class	Cat	5	2	0
	Dog	3	3	2
	Rabbit	0	1	11

Confusion Matrix

		Actual class	
		Cat	**Non-cat**
Predicted class	**Cat**	5 True Positives	2 False Positives
	Non-cat	3 False Negatives	17 True Negatives

Confusion Matrix

		Actual class	
		Cat	Non-cat
Predicted class	Cat	5 True Positives	2 False Positives
	Non-cat	3 False Negatives	17 True Negatives

- Accuracy (Acurácia): Overall, how often is the classifier correct?
  - (TP+TN)/total = (5+17)/27
- Precision (Precisão): When it predicts yes, how often is it correct?
  - TP/predicted yes = 5/7
- Recall (Revocação) or True Positive Rate: When it's actually yes, how often does it predict yes?
  - TP/actual yes = 5/8
  - also known as "Sensitivity" or "Recall”
- F Score: This is a weighted average of the true positive rate (recall) and precision.

F-measure

The traditional F-measure or balanced F-score (F_1 score) is the harmonic mean of precision and recall:

$$F_1 = \frac{2}{\frac{1}{\mathrm{recall}} + \frac{1}{\mathrm{precision}}} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

$$F_\beta = (1 + \beta^2) \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{(\beta^2 \cdot \mathrm{precision}) + \mathrm{recall}}$$

F_2 measure weighs recall higher than precision (by placing more emphasis on false negatives).

$F_{0.5}$ measure weighs recall lower than precision (by attenuating the influence of false negatives).

Confusion Matrix

```python
from sklearn import metrics

y = ['cat', 'cat', 'cat', 'cat', 'cat', 'cat', 'cat', 'cat',
     'dog', 'dog', 'dog', 'dog', 'dog', 'dog',
     'rabbit', 'rabbit', 'rabbit', 'rabbit', 'rabbit', 'rabbit', 'rabbit', 'rabbit',
     'rabbit', 'rabbit', 'rabbit', 'rabbit', 'rabbit']

y_pred = ['cat', 'cat', 'cat', 'cat', 'cat', 'dog', 'dog', 'dog',
     'cat', 'cat', 'dog', 'dog', 'dog', 'rabbit',
     'dog', 'dog', 'rabbit', 'rabbit', 'rabbit', 'rabbit', 'rabbit', 'rabbit',
     'rabbit', 'rabbit', 'rabbit', 'rabbit', 'rabbit']

cm = metrics.confusion_matrix(y, y_pred, labels=['cat', 'dog', 'rabbit'])
print(cm)
```

Predicted class

Actual class

```
       c   d   r
c   [[ 5   3   0]
d    [ 2   3   1]
r    [ 0   2  11]]
```

		Actual class		
		Cat	Dog	Rabbit
Predicted class	Cat	5	2	0
	Dog	3	3	2
	Rabbit	0	1	11

Accuracy

$$
\begin{array}{ccc}
 & & \text{Predicted class} \\
 & & \begin{array}{ccc} c & d & r \end{array} \\
\text{Actual class} & \begin{array}{c} c \\ d \\ r \end{array} & \begin{bmatrix} 5 & 3 & 0 \\ 2 & 3 & 1 \\ 0 & 2 & 11 \end{bmatrix}
\end{array}
$$

Accuracy = (TP + TN) / Total

True Positive (TP) + True Negative (TN) = 19

Total = 27

Accuracy = 19 / 27 = 0.7037037037

```
metrics.accuracy_score(y, y_pred)
```

```
0.7037037037037037
```

```
accuracy = np.sum(np.diagonal(cm)) / np.sum(cm)
print(accuracy)
```

```
0.703703703704
```

Precision / Positive Predictive Value

$$
\begin{array}{cc}
 & \text{Predicted class} \\
 & \begin{array}{ccc} c & d & r \end{array} \\
\text{Actual class} \begin{array}{c} c \\ d \\ r \end{array} & \begin{bmatrix} 5 & 3 & 0 \\ 2 & 3 & 1 \\ 0 & 2 & 11 \end{bmatrix}
\end{array}
$$

Precision = TP / (TP + FP)

$TP_{cat} = 5$

$TP_{cat} + FP_{cat} = 5 + 2 = 7$

$Precision_{cat} = 5 / 7 = 0.7142857143$

```
metrics.precision_score(y, y_pred, average=None)
```

```
[ 0.71428571,  0.375     ,  0.91666667]
```

```
precision = cm[0,0] / np.sum(cm[:,0])
print(precision)
```

```
0.714285714286
```

Recall / True Positive Rate / Sensitivity

$$
\begin{array}{cc|ccc}
 & & \multicolumn{3}{c}{\text{Predicted class}} \\
 & & c & d & r \\
\hline
\text{Actual class} & c & [[\ 5 & 3 & 0] \\
 & d & [\ 2 & 3 & 1] \\
 & r & [\ 0 & 2 & 11]]
\end{array}
$$

Revocação

Recall = TP / (TP + FN)

$TP_{cat} = 5$

$TP_{cat} + FN_{cat} = 5 + 3 = 8$

$\mathbf{Recall_{cat} = 5 / 8 = 0.625}$

```
metrics.recall_score(y, y_pred, average=None)
```

```
[ 0.625      ,  0.5        ,  0.84615385]
```

```
recall = cm[0,0] / np.sum(cm[0,:])
print(recall)
```

```
0.625
```

Classification report

$$
\begin{array}{cc|ccc|c}
 & & & \text{Predicted class} & & \\
 & & c & d & r & s \\
\hline
\text{Actual class} & c & [[\ 5+ & 3+ & 0] & =8 \\
 & d & [\ 2+ & 3+ & 1] & =6 \\
 & r & [\ 0+ & 2+ & 11]] & =13
\end{array}
$$

$Precision_{avg} =$ (0.71 * 8 + 0.38 * 6 + 0.92 * 13) / 27 = 0.74

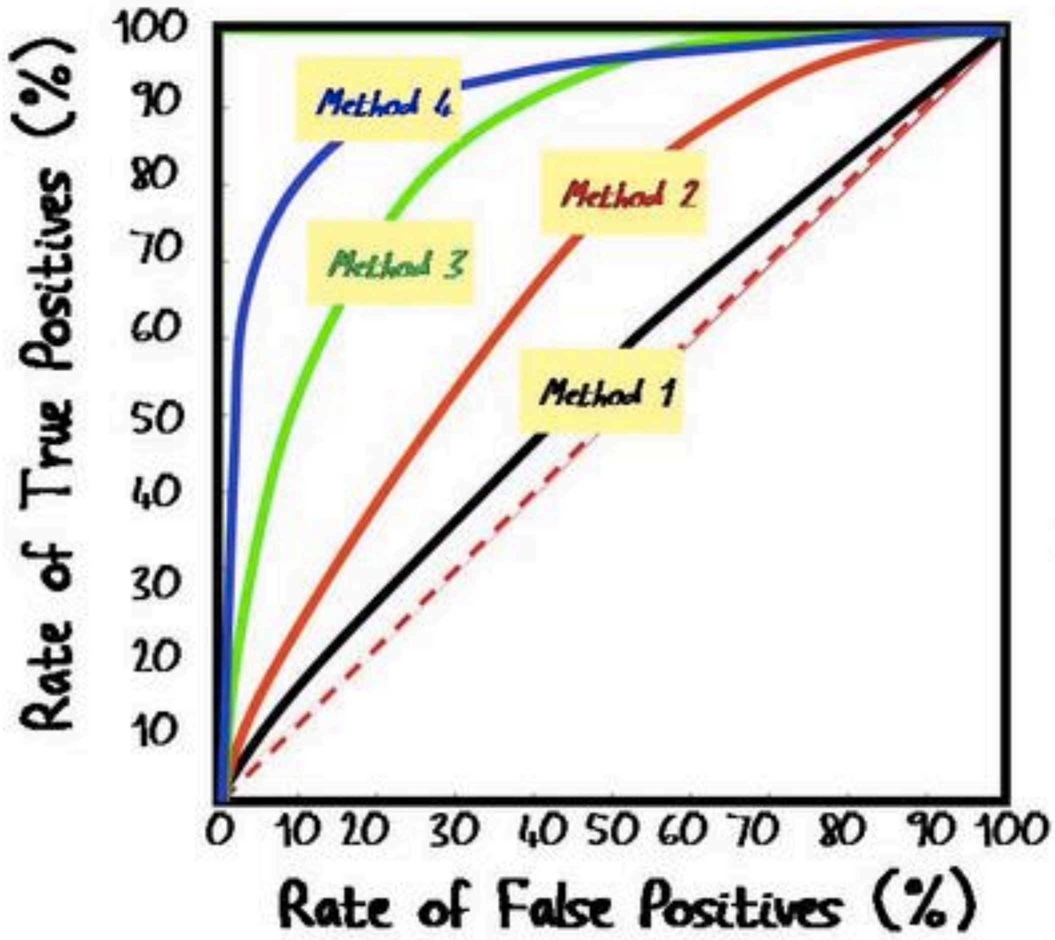
```
metrics.classification_report(y, y_pred)
```

```
             precision    recall  f1-score   support

        cat       0.71      0.62      0.67         8
        dog       0.38      0.50      0.43         6
     rabbit       0.92      0.85      0.88        13

avg / total       0.74      0.70      0.72        27
```

ROC CURVE EXAMPLES

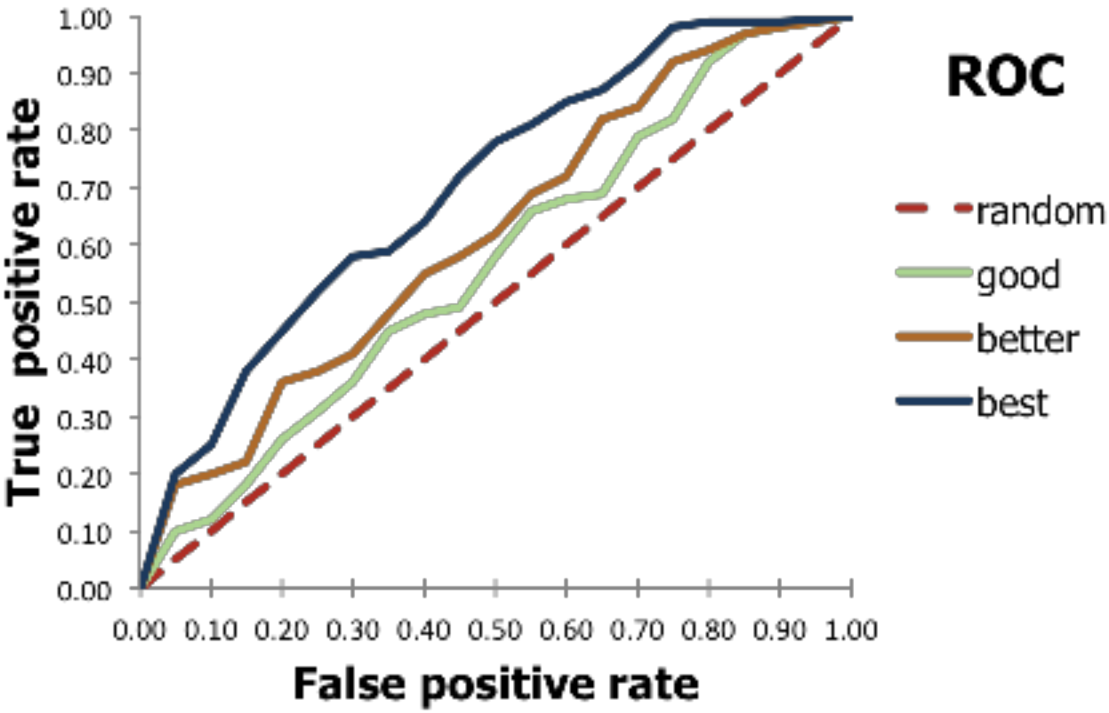


- The best classification has the largest area under the curve.

ROC - Receiver Operating Characteristic

Curva ROC

way to visualize the performance of a binary classifier.

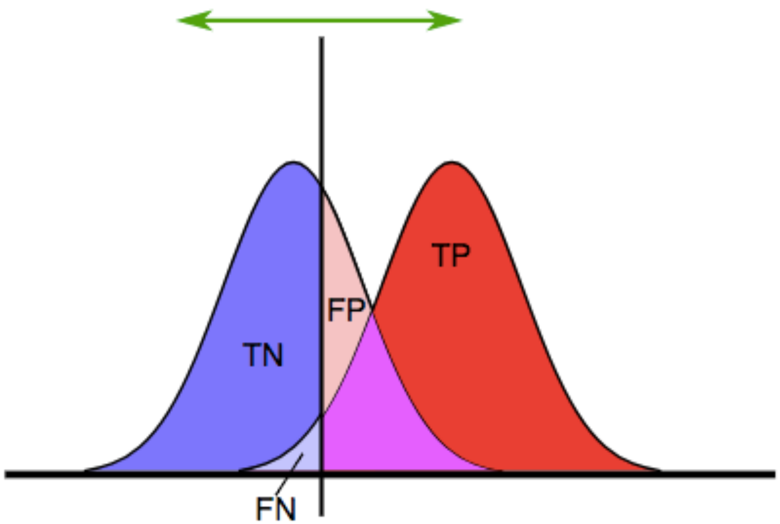


ROC - Receiver Operating Characteristic

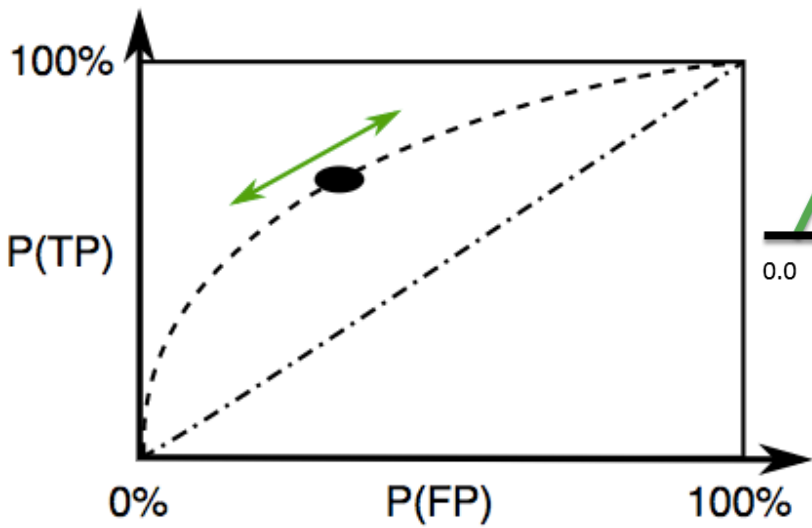
Curva ROC

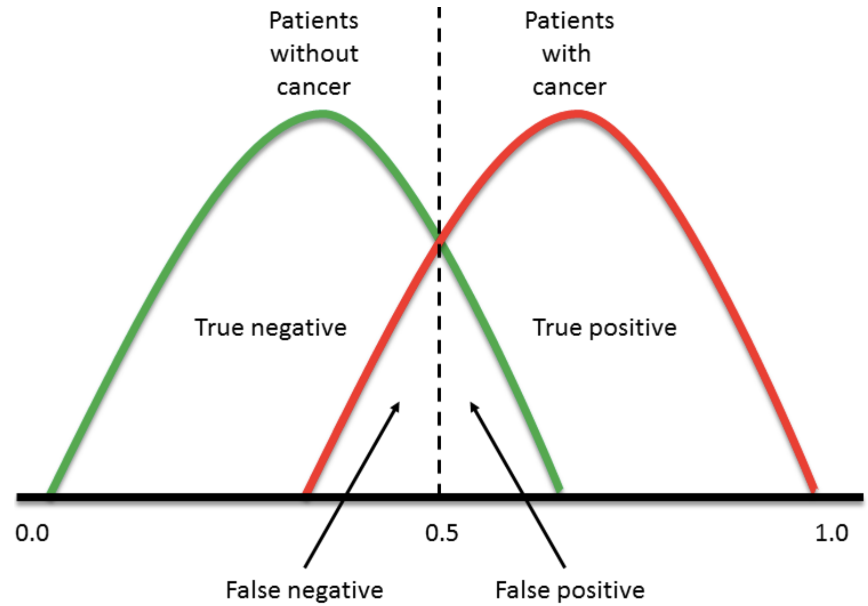
- Way to visualize the performance of a binary classifier.
- Commonly used graph that summarizes the performance of a classifier over all possible thresholds.
- It is generated by plotting the True Positive Rate (y-axis) against the False Positive Rate (x-axis) as you vary the threshold for assigning observations to a given class.

Curva ROC



TP	FP
FN	TN

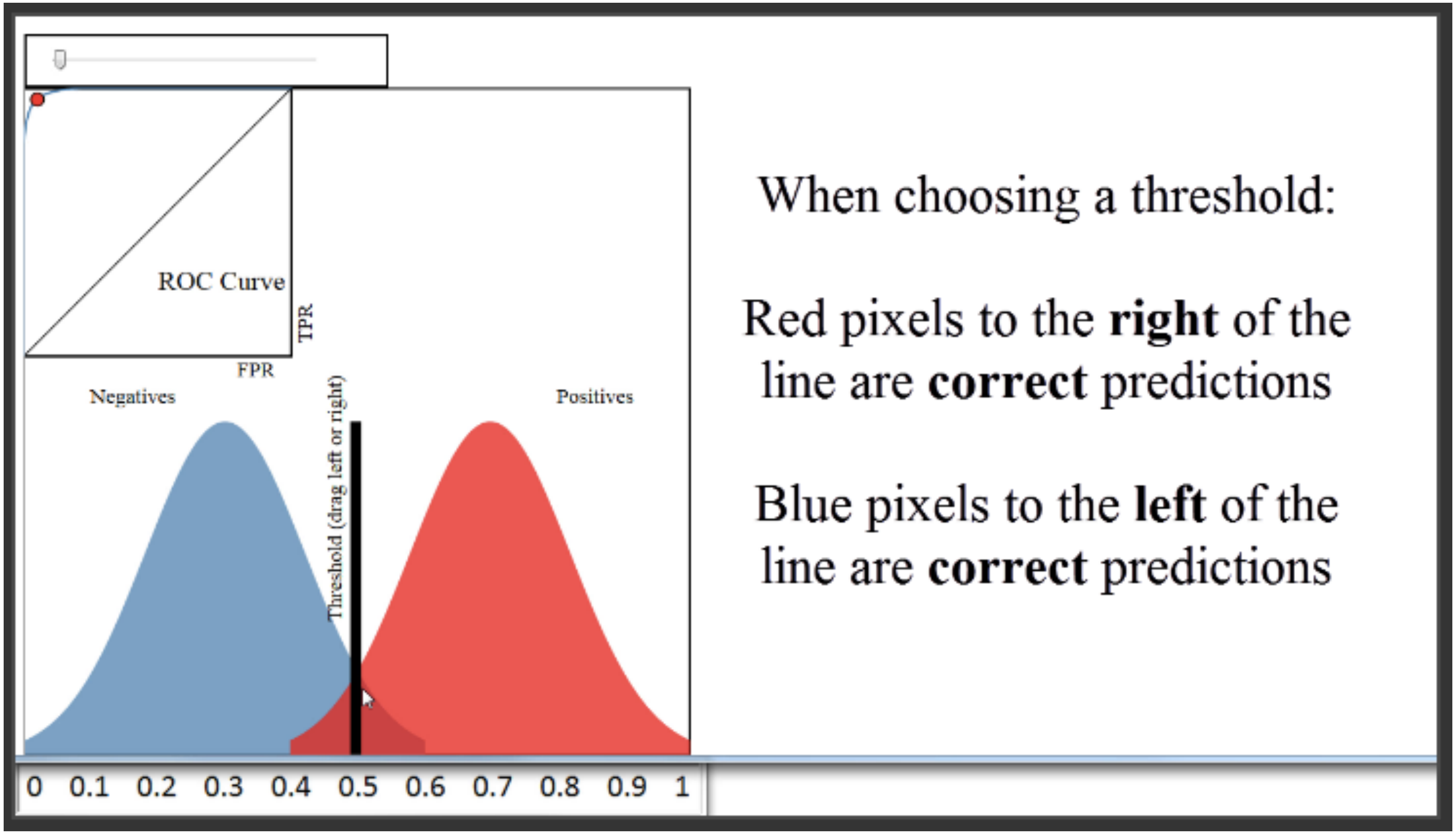




Understanding ROC curves

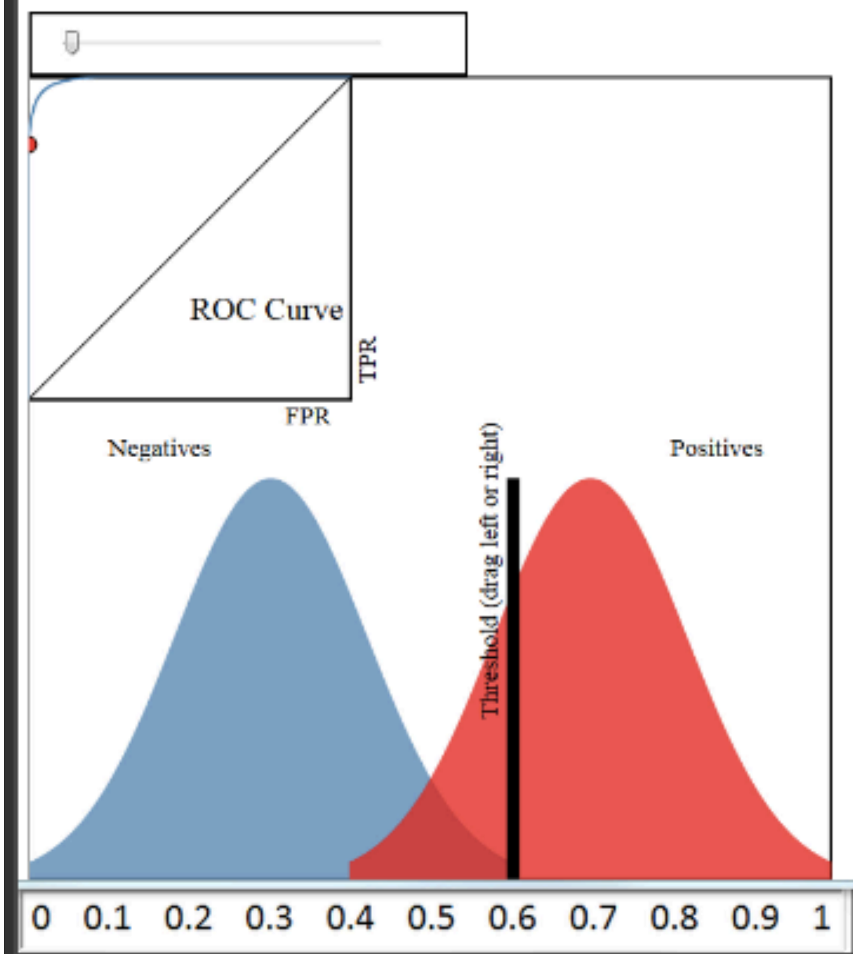
http://www.navan.name/roc/

Threshold = 0.5 → classify everything above 0.5 as admitted and everything below 0.5 as not admitted, which is what most classification methods will do by default.



Understanding ROC curves

http://www.navan.name/roc/

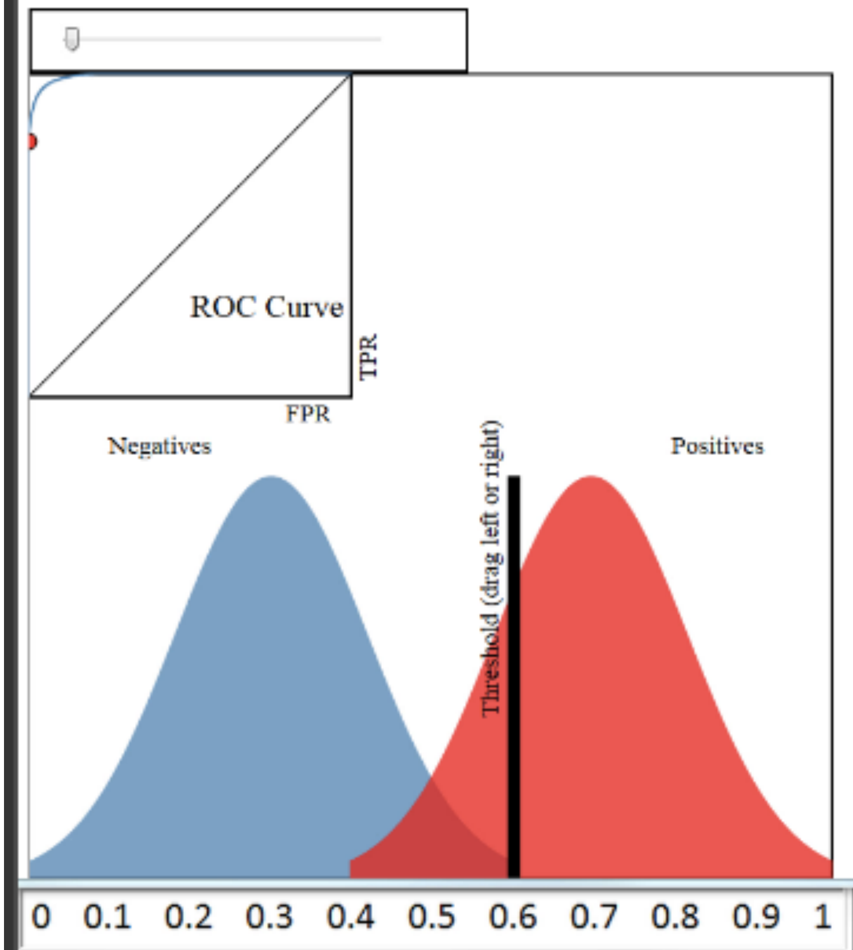


This diagram is **not** representative of most **real-world** problems:

1. Most problems **don't** have balanced classes
2. Predicted probabilites are unlikely to have a "smooth" distribution

Understanding ROC curves

http://www.navan.name/roc/



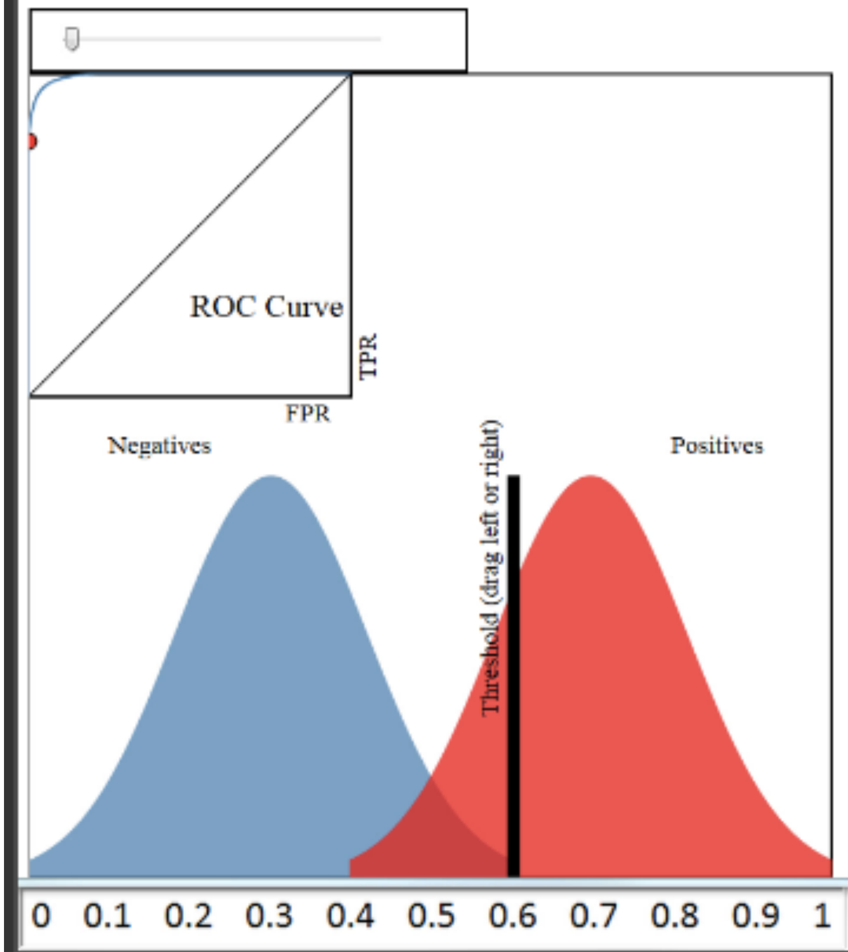
Important Note #1:

ROC curves are useful even if your **predicted probabilities** are **not** "properly calibrated"

AUC is a useful metric even when your classes are **highly unbalanced**

Understanding ROC curves

http://www.navan.name/roc/

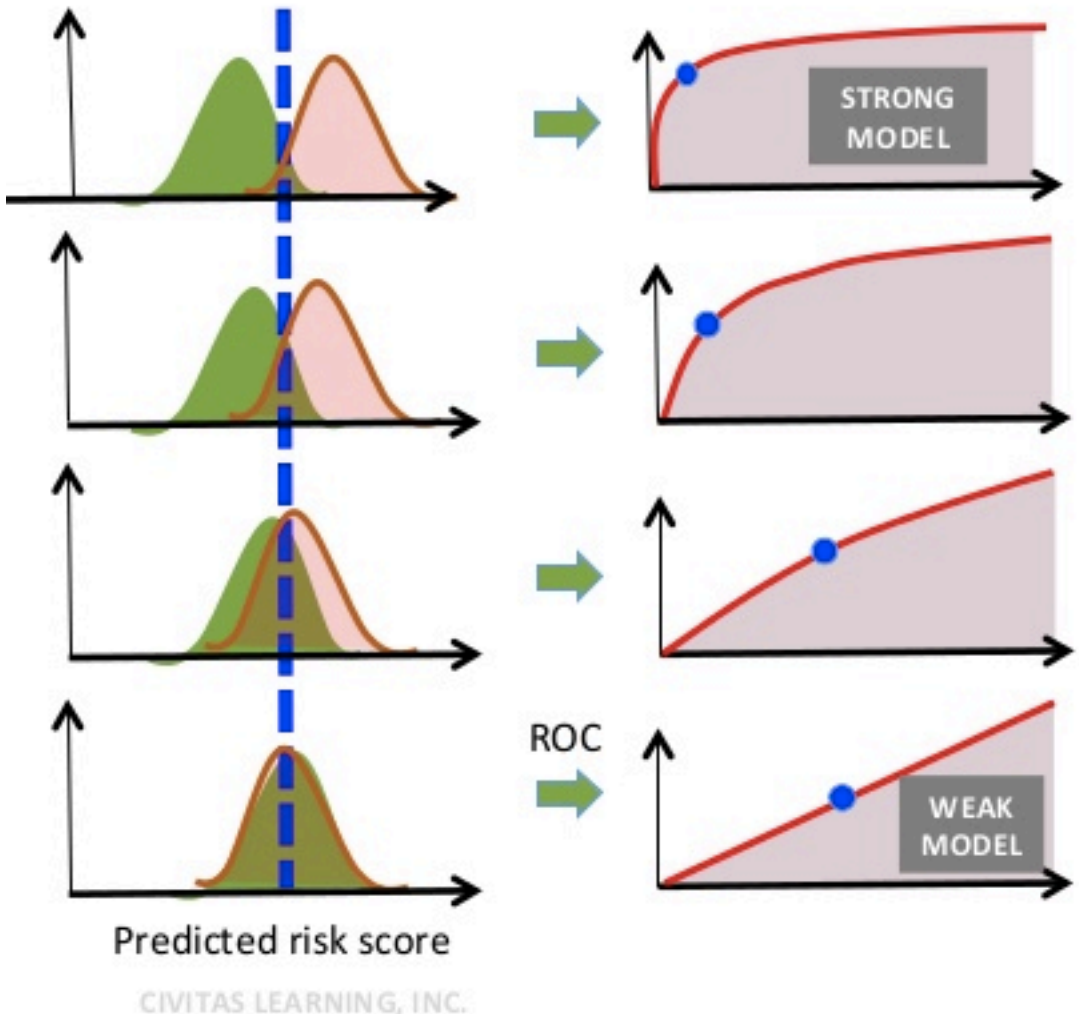


Important Note #2:

ROC curves can be extended to problems with **three or more** classes:

class 1 vs classes 2 & 3
class 2 vs classes 1 & 3
class 3 vs classes 1 & 2

Model Performance

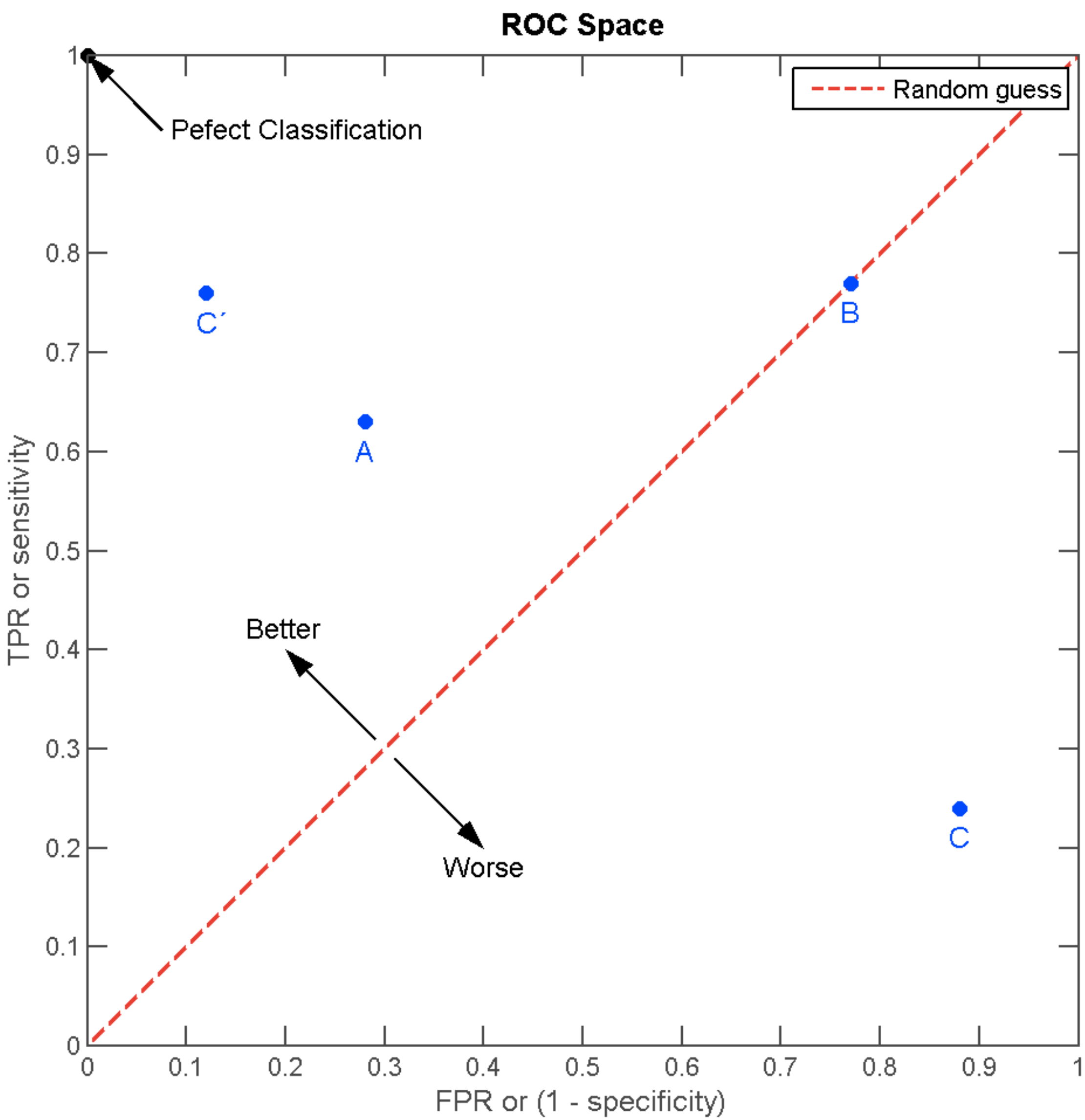


Overlap is a measure of the model's ability to separate between success and failure.

With a strong model you can be confident of assigning a particular score to an outcome category.

With a weaker model, there is a large amount of overlap, so a particular score could mean that an outcome can be either good or bad with equal probability.

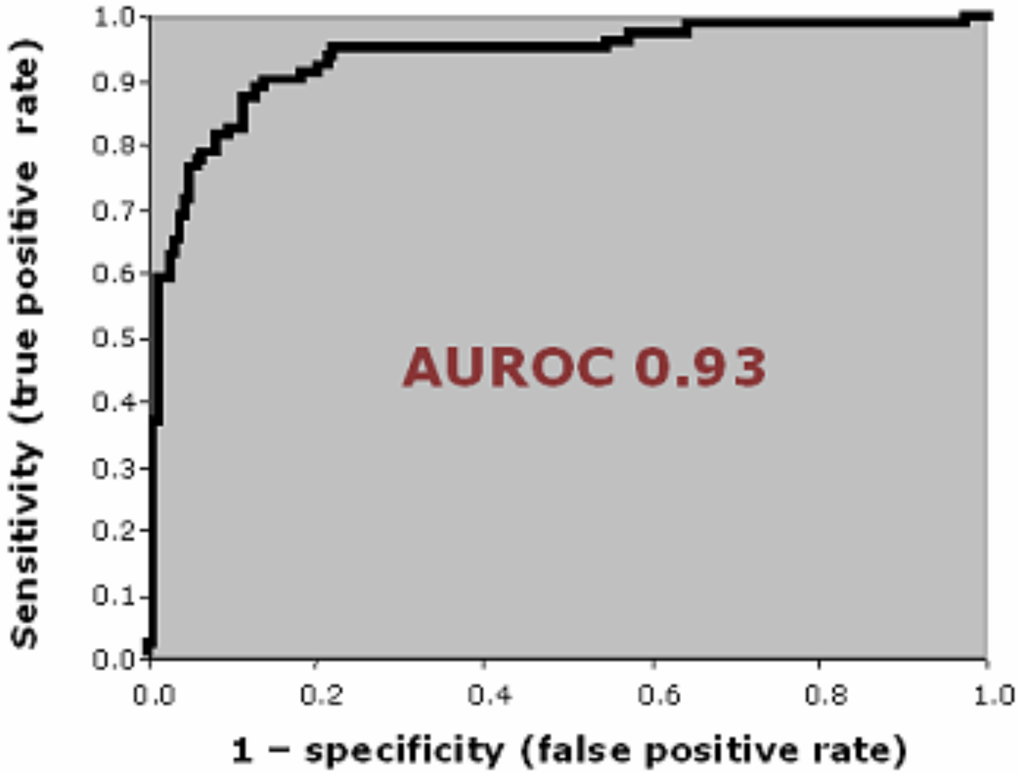
Curva ROC



A			B		
TP=63	FN=37	100	TP=77	FN=23	100
FP=28	TN=72	100	FP=77	TN=23	100
91	109	200	154	46	200
TPR = 0.63			TPR = 0.77		
FPR = 0.28			FPR = 0.77		
PPV = 0.69			PPV = 0.50		
F1 = 0.66			F1 = 0.61		
ACC = 0.68			ACC = 0.50		

C			C′		
TP=24	FN=76	100	TP=76	FN=24	100
FP=88	TN=12	100	FP=12	TN=88	100
112	88	200	88	112	200
TPR = 0.24			TPR = 0.76		
FPR = 0.88			FPR = 0.12		
PPV = 0.21			PPV = 0.86		
F1 = 0.22			F1 = 0.81		
ACC = 0.18			ACC = 0.82		

AU ROC/AUC



AUC - Area Under the Curve

Problem with accuracy

- Problem when the cost of misclassification of the minor class samples are very high.
- If we deal with a rare but fatal disease, the cost of **failing to diagnose the disease of a sick person** (FN) is much higher than the cost of sending a healthy person to more tests (FP).

False Positive (FP)

- False Positive (FP) = False Alarm
  - A pregnancy test is positive, when in fact you aren't pregnant.
  - A cancer screening test comes back positive, but you don't have the disease.
  - A prenatal test comes back positive for Down's Syndrome, when your fetus does not have the disorder.
  - Virus software on your computer incorrectly identifies a harmless program as a malicious one.

False Negative (FN)

- A pregnancy test may come back negative even though you are in fact pregnant.
- A test for cancer might come back negative, when in reality you actually have the disease.
- Quality control: a defective item passes through the cracks.
- Software testing: a test designed to catch something has failed.
- Justice System: a guilty suspect is found "Not Guilty" and allowed to walk free.
- **Problems:**
  - false sense of security.
  - potentially dangerous situations may be missed.

Log Loss

- Logarithmic Loss or **Log Loss**, works by **penalising the false classifications**.
- It works well for multi-class classification.
- When working with Log Loss, the classifier must assign probability to each class for all the samples.

$$LogarithmicLoss = \frac{-1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} * \log(p_{ij})$$

  - N samples
  - M classes
  - y_{ij} indicates whether sample i belongs to class j or not
  - p_{ij} indicates the probability of sample i belonging to class j

Log Loss

$$LogarithmicLoss = \frac{-1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{ij} * \log(p_{ij})$$

- In order to calculate Log Loss the classifier must assign a probability to each class rather than simply yielding the most likely class.
- Log Loss has no upper bound and it exists on the range [0, ∞).
- Log Loss nearer to 0 indicates higher accuracy.
- If the Log Loss is away from 0 then it indicates lower accuracy.
- In general, minimising Log Loss gives greater accuracy for the classifier.

Log loss, aka logistic loss or cross-entropy loss

```
sklearn.metrics.log_loss(y_true, y_pred, eps=1e-15,
normalize=True, sample_weight=None, labels=None)
```

- loss function used in (multinomial) logistic regression and extensions of it

Obrigado!

Dúvidas, comentários, sugestões?

Regis Pires Magalhães

regismagalhaes@ufc.br

UFC