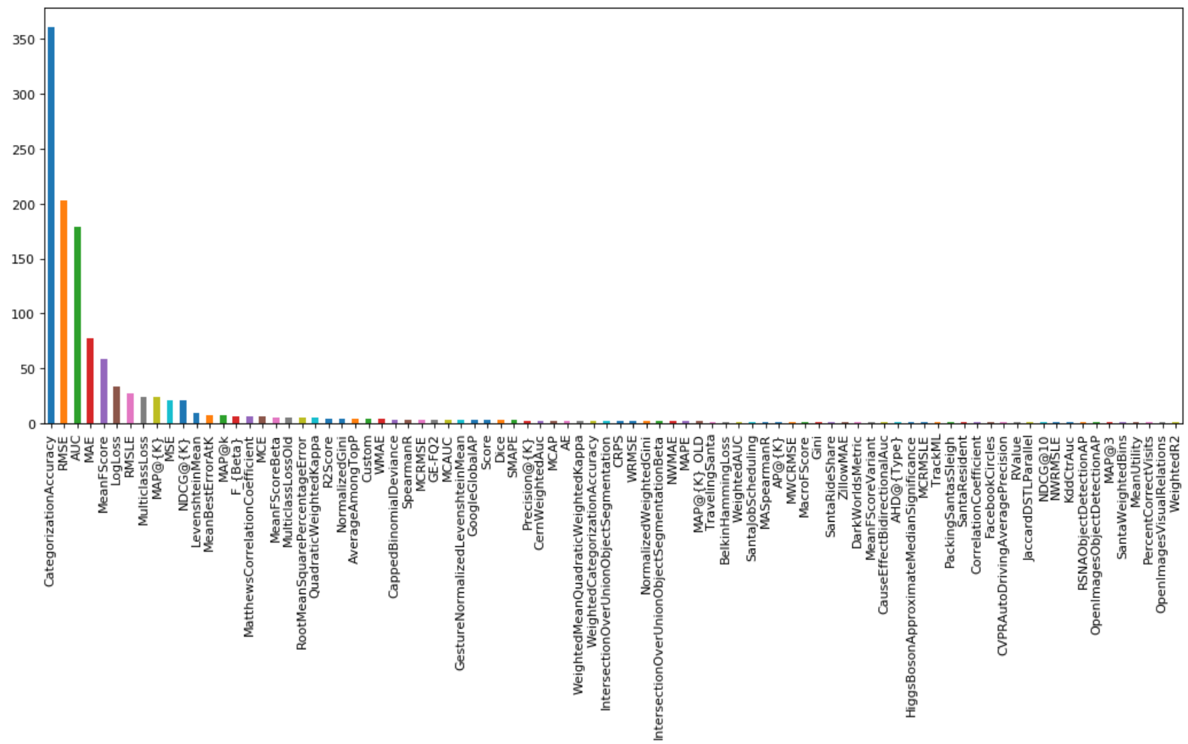
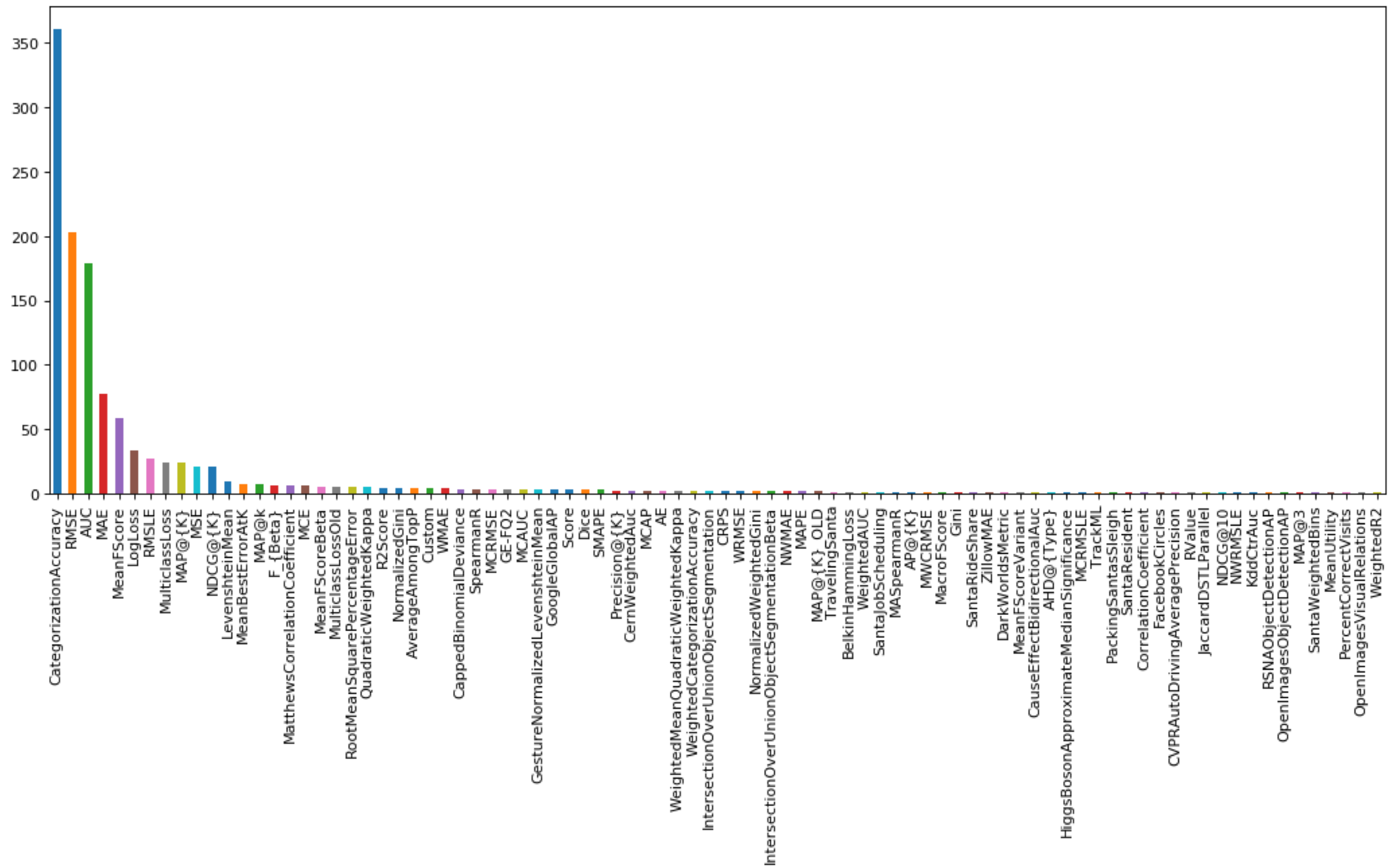


Model Evaluation and Metrics

Jay Urbain, PhD


0
50
100
150
200
250
300
350
CategorizationAccuracy
RMSE
AUC
MAE
MeanFScore
LogLoss
RMSLE
MulticlassLoss
MAP@{K}
MSE
NDCG@{K}
LevenshteinMean
MeanBestErrorAtK
MAP@k
F_{Beta}
MatthewsCorrelationCoefficient
MCE
MeanFScoreBeta
MulticlassLossOld
RootMeanSquarePercentageError
QuadraticWeightedKappa
R2Score
NormalizedGini
AverageAmongTopP
Custom
WMAE
CappedBinomialDeviance
SpearmanR
MCRMSE
GE-FQ2
MCAUC
GestureNormalizedLevenshteinMean
GoogleGlobalAP
Score
Dice
SMAPE
Precision@{K}
CernWeightedAuc
MCAP
AE
WeightedMeanQuadraticWeightedKappa
WeightedCategorizationAccuracy
IntersectionOverUnionObjectSegmentation
CRPS
WRMSE
NormalizedWeightedGini
IntersectionOverUnionObjectSegmentationBeta
NWMAE
MAPE
MAP@{K}_OLD
TravelingSanta
BelkinHammingLoss
WeightedAUC
SantaJobScheduling
MASpearmanR
AP@{K}
MWCRMSE
MacroFScore
Gini
SantaRideShare
ZillowMAE
DarkWorldsMetric
MeanFScoreVariant
CauseEffectBidirectionalAuc
AHD@{Type}
HiggsBosonApproximateMedianSignificance
MCRMSLE
TrackML
PackingSantasSleigh
SantaResident
CorrelationCoefficient
FacebookCircles
CVPRAutoDrivingAveragePrecision
RValue
JaccardDSTLParallel
NDCG@10
NWRMSLE
KddCtrAuc
RSNAObjectDetectionAP
OpenImagesObjectDetectionAP
MAP@3
SantaWeightedBins
MeanUtility
PercentCorrectVisits
OpenImagesVisualRelations
WeightedR2

Topics

- Review: Regression
    - Assessing the accuracy of model coefficients
    - RMSE – Root Mean Squared Error
- Classification
    - Confusion matrix
    - ROC Curve

Review: Assessing the accuracy of model coefficients

Linear regression with residual term. Residual represents what we can't explain with our model.

$$Y = \beta_0 + \beta_1 X + \epsilon.$$

RSS measures the amount of variability that is left unexplained after performing the regression

$$\mathrm{RSS} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2.$$

TSS (Total sum of squares) measures the total variance when measuring the response *y*.

$$\mathrm{TSS} = \sum (y_i - \bar{y})^2$$

R^2 amount of variance explained by our model

$$R^2 = \frac{\mathrm{TSS} - \mathrm{RSS}}{\mathrm{TSS}} = 1 - \frac{\mathrm{RSS}}{\mathrm{TSS}}$$

The *RSE* is an estimate of the standard deviation of ε. It is basically the average amount that the response will deviate from the true regression line.

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

Mean square error.

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n} (Y_i - \hat{Y}_i)^2.$$

*Root mean square error.

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

Root Mean Squared Error

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

- Most popular metric for regression.
- Used for regression problems
- Square root of the mean of the squared errors
- Easily interpretable (in units of "y")
- "Punishes" larger errors
- Other: *absolute error*

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

Confusion Matrix: table to describe the performance of a classifier

n=165	**Predicted: NO**	**Predicted: YES**
Actual: NO	50	10
Actual: YES	5	100

Example: Test for presence of disease
NO = negative test = False = 0
YES = positive test = True = 1

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

Confusion Matrix: table to describe the performance of a classifier

n=165	Predicted: NO	Predicted: YES	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

Basic Terminology:
- True Positives (TP)
- True Negatives (TN)
- False Positives (FP) - Type I Error
- False Negatives (FN) - Type II Error

Accuracy:
- Overall, how often is it **correct**?
- (TP + TN) / total = 150/165 = 0.91

Misclassification Rate (Error Rate):
- Overall, how often is it **wrong**?
- (FP + FN) / total = 15/165 = 0.09

Confusion Matrix: table to describe the performance of a classifier

n=165	Predicted: NO	Predicted: YES	
Actual: NO	TN = 50	FP = 10	60
Actual: YES	FN = 5	TP = 100	105
	55	110	

Sensitivity, **True Positive Rate**, Recall:

- When actual value is **positive**, how often is prediction **correct**?
- TPR = TP / T = 100/105 = 0.95
- “True Positive Rate” or “Recall”

False Positive Rate, Fall-out:

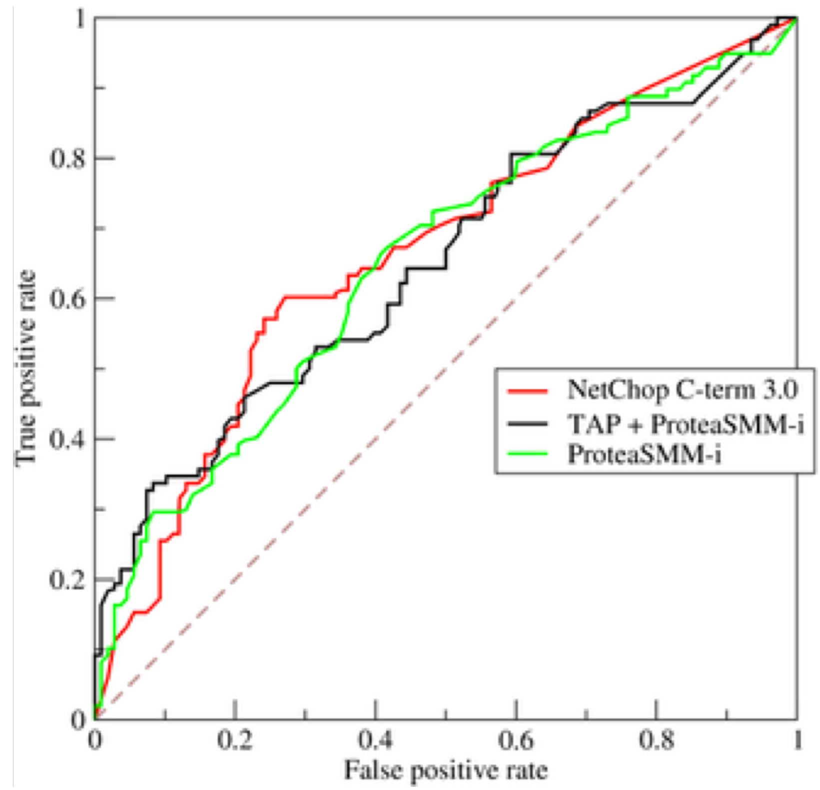
- When actual value is **negative**, how often is prediction **wrong**?
- FPR = (FP / F) = 10/60 = 0.17

Specificity, True Negative Rate:

- When actual value is **negative**, how often is prediction **correct**?
- TNR = (TN / F) = 50/60 = 0.83

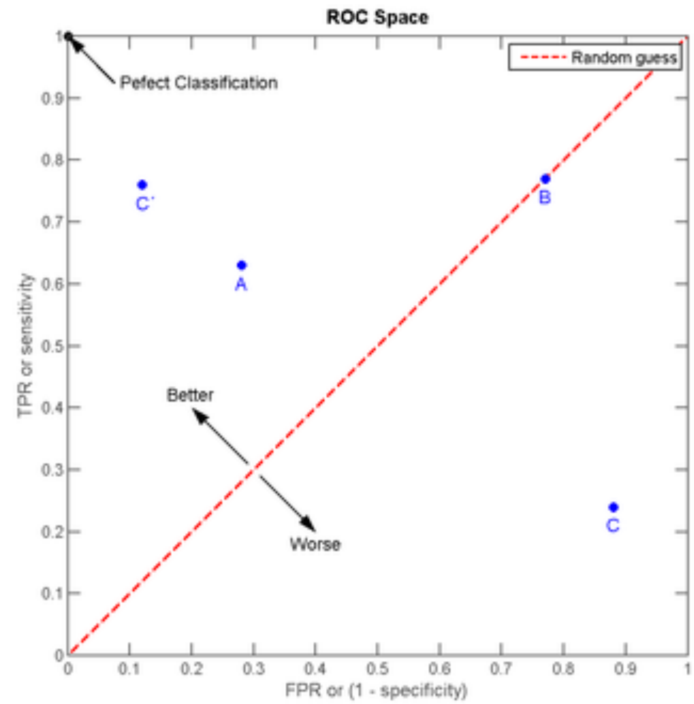
Receiver operating characteristic (ROC) Curve

- The 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.



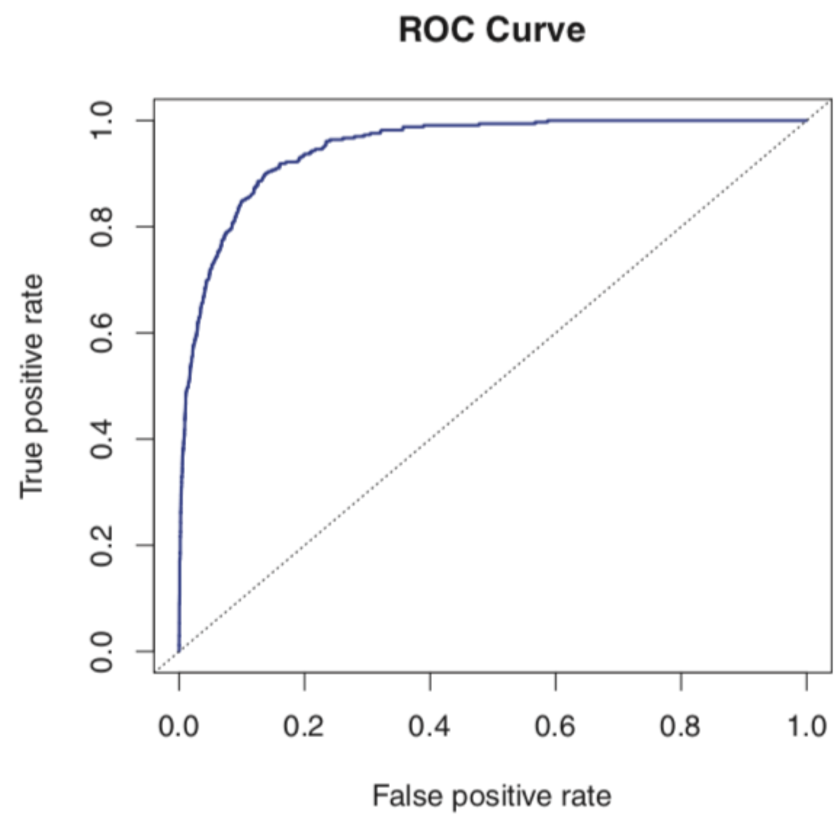
Receiver operating characteristic (ROC) Curve

- To draw a ROC curve, only the true positive rate (TPR) and false positive rate (FPR) are needed (as functions of some classifier).
- A ROC space is defined by FPR and TPR as *x* and *y* axes respectively, which depicts relative trade-offs between true positive (benefits) and false positive (costs).
- The best possible prediction method would yield a point in the upper left corner or coordinate (0,1) of the ROC space, representing 100% sensitivity (no false negatives) and 100% specificity (no false positives).
- The (0,1) point is also called a *perfect classification*.

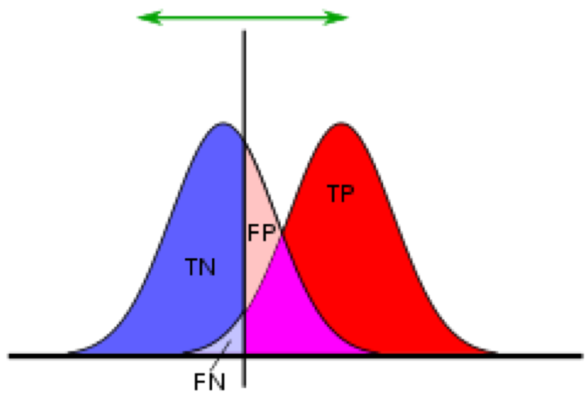


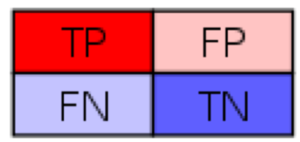
ROC Curves

- The overall performance of a classifier summarized over all possible thresholds, is given by the area under the ROC curve (AUC).
- An ideal ROC curve will hug the top left corner, so the larger the AUC the better the classifier.
- A binary classifier performing not better than chance would have an AUC of 0.5

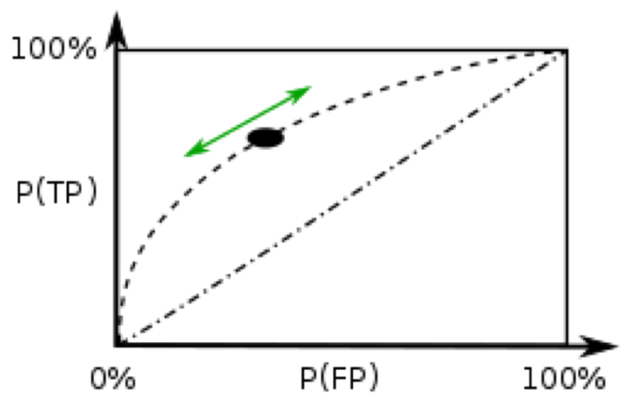


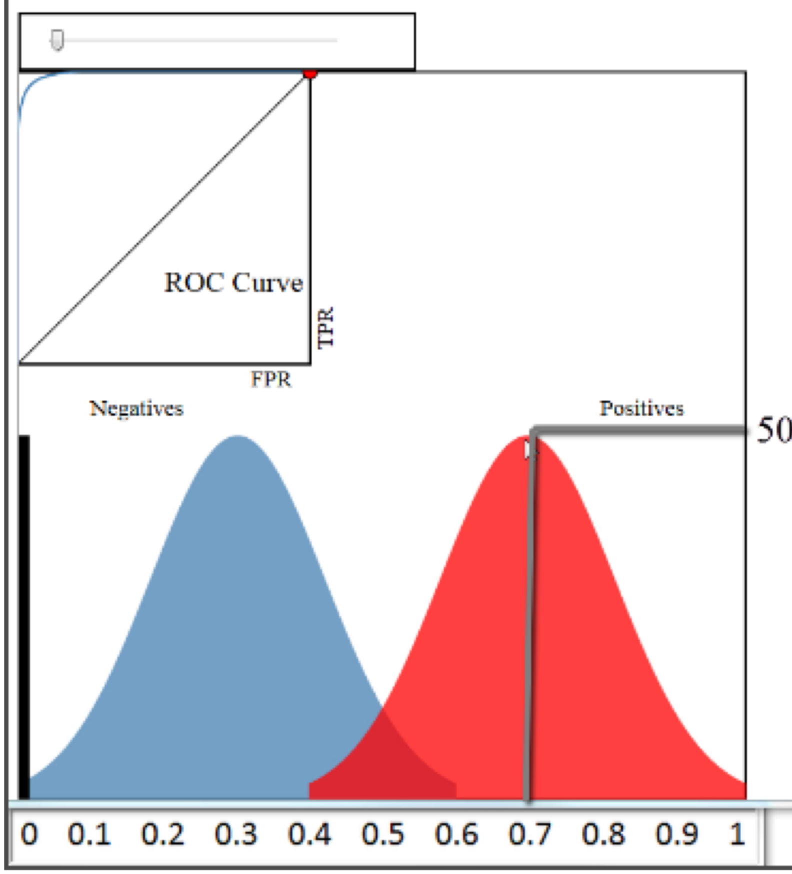
ROC Space





By adjusting the threshold for positive classification you can move to different positions on the ROC curve.





Remember:

blue pixel = "not admitted"
red pixel = "admitted"

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

ROC Curve

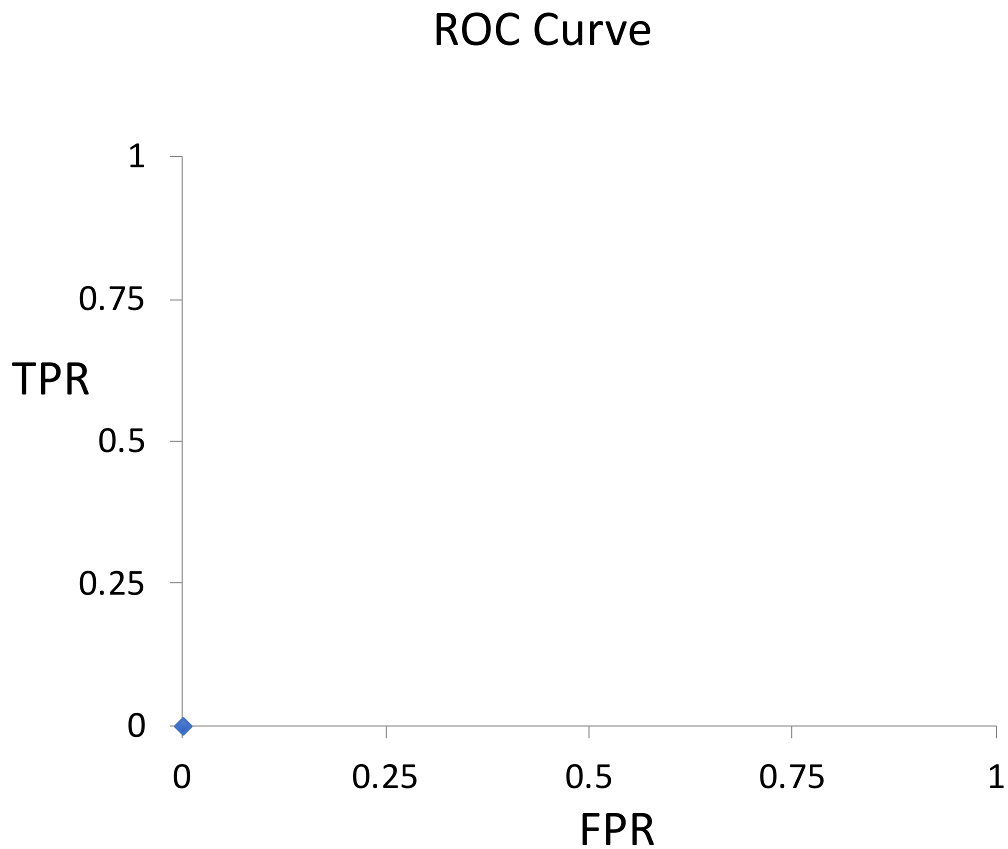
Email Number	Score	True Label
5	0.99	Spam
8	0.82	Spam
2	0.60	Spam
1	0.60	Ham
7	0.48	Spam
3	0.22	Ham
4	0.10	Ham
6	0.02	Ham

Every email is assigned a “spam” score (probability) by our classification algorithm. To actually make our predictions (classifications), we choose a threshold for classifying an example as spam.

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

ROC Curve

ID	Classifier Score	True Label	TPR	FPR
5	0.99	T		
8	0.82	T		
2	0.72	F		
1	0.68	T		
7	0.51	F		
3	0.48	T		
4	0.30	F		
6	0.02	F		



ROC Curve

Email Number	Score	True Label
5	0.99	Spam
8	0.82	Spam
2	0.60	Spam
1	0.60	Ham
7	0.48	Spam
3	0.22	Ham
4	0.10	Ham
6	0.02	Ham

<u>TPR</u>: When actual value is **spam**, how often is prediction **correct**?

<u>FPR</u>: When actual value is **ham**, how often is prediction **wrong**?

Cutoff	TPR (y)	FPR (x)	Cutoff	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

ROC Curve

Email Number	Score	True Label
5	0.99	Spam
8	0.98	Spam
2	0.97	Spam
1	0.97	Ham
7	0.96	Spam
3	0.95	Ham
4	0.94	Ham
6	0.93	Ham

Q: Would the ROC Curve (and AUC) change if the **scores** changed, but the **ordering** remained the same?

A: No. The ROC Curve is only sensitive to **rank ordering** and does not require **calibrated scores**.

notebooks/multinomial_classification_solution.ipynb

ROC Curves

```
from sklearn.metrics import roc_curve

fpr, tpr, thresholds = roc_curve(y_train_5, y_scores)
```

```
def plot_roc_curve(fpr, tpr, label=None):
    plt.plot(fpr, tpr, linewidth=2, label=label)
    plt.plot([0, 1], [0, 1], 'k--')
    plt.axis([0, 1, 0, 1])
    plt.xlabel('False Positive Rate', fontsize=16)
    plt.ylabel('True Positive Rate', fontsize=16)
    plt.title("ROC")

plt.figure(figsize=(8, 6))
plot_roc_curve(fpr, tpr)
# save_fig("roc_curve_plot")
plt.show()
```