Metrics

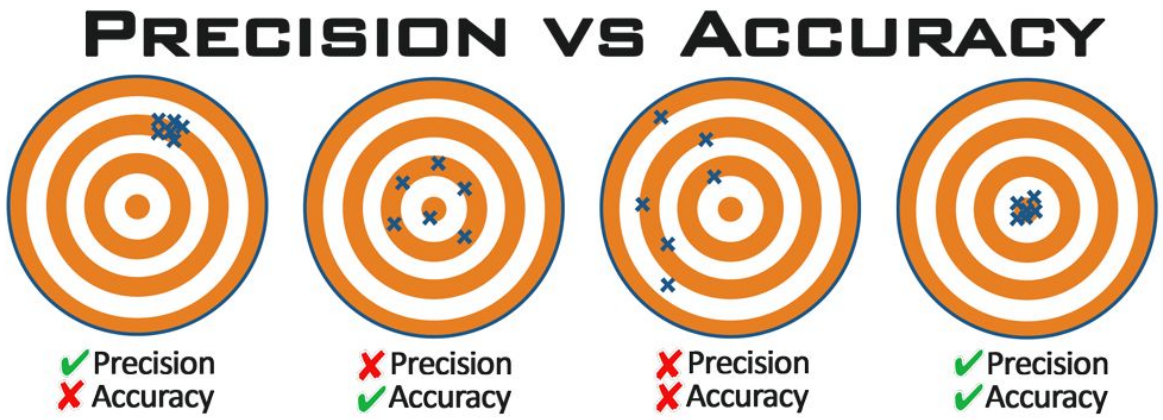
PhD. Msc. David C. Baldears S.
PhD(s). Msc. Diego Lopez Bernal

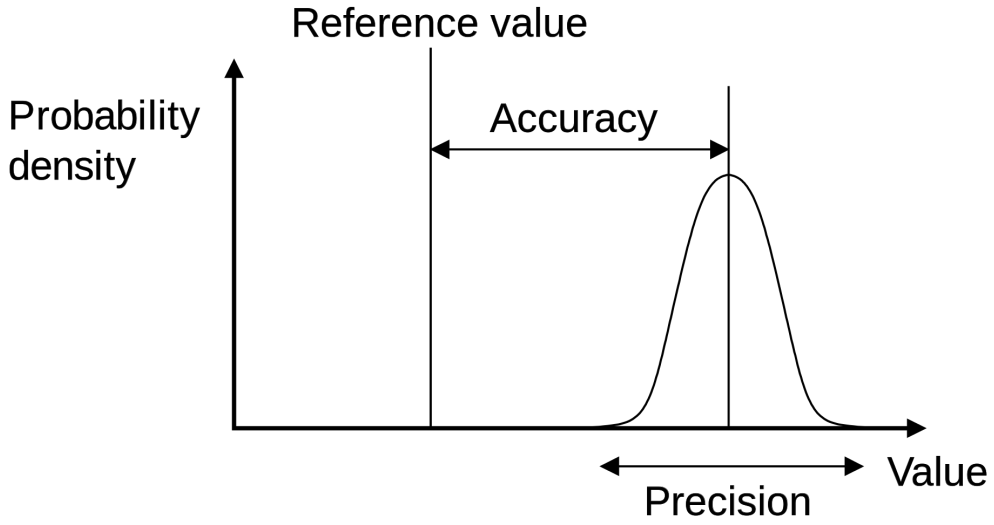
TC3007C

Accuracy

Accuracy is how close or far off a given set of measurements (observations or readings) are to their true value

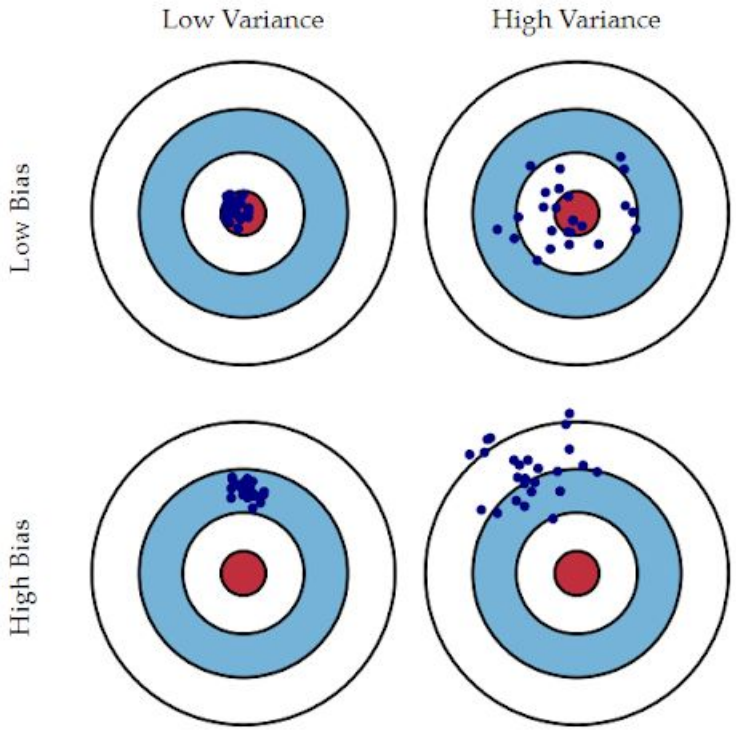
Precision is how close or dispersed the measurements are to each other





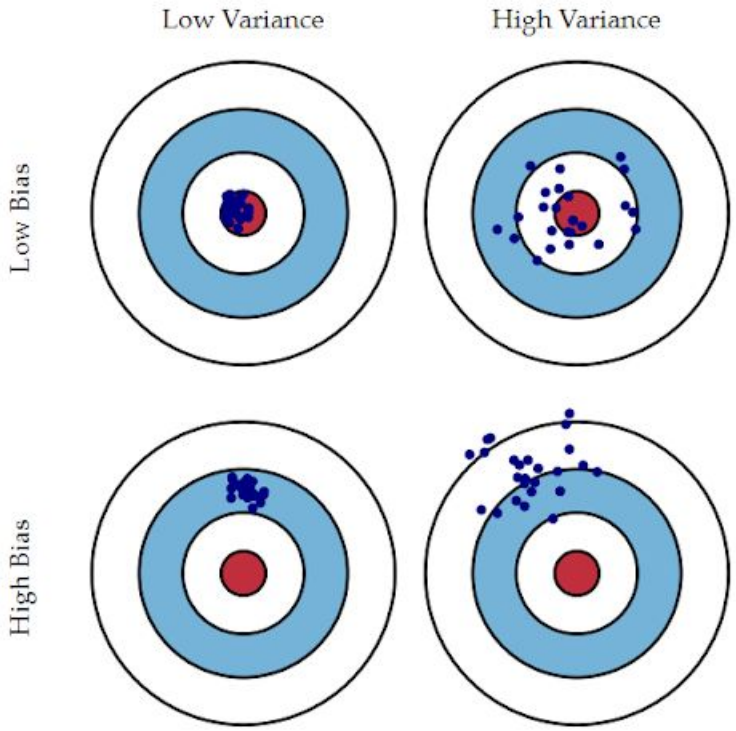
Bias and Variance

- The performance of an algorithm can also be defined with the bias and variance.
- The bias–variance dilemma is the conflict in trying to simultaneously minimize these two sources of error that prevent supervised learning algorithms from generalizing beyond their training set:

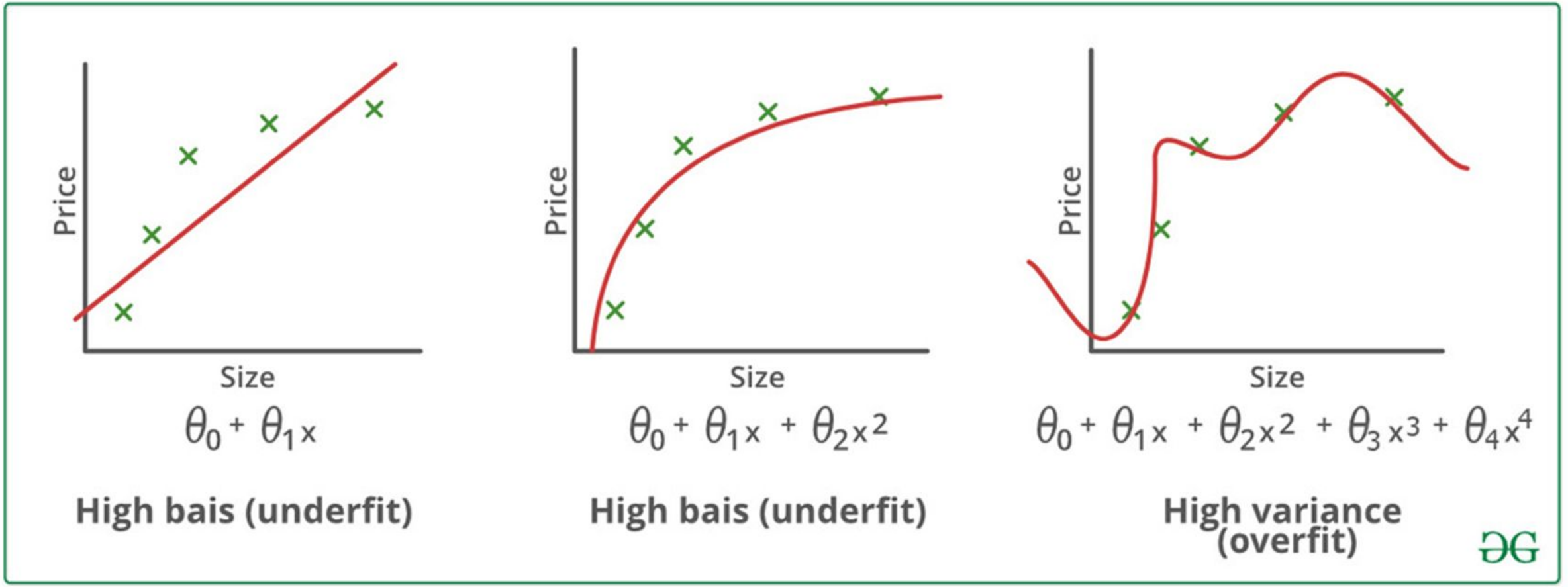


Bias and Variance

- The bias error is an error from erroneous assumptions in the learning algorithm. High bias can cause an algorithm to miss the relevant relations between features and target outputs (underfitting).
- The variance is an error from sensitivity to small fluctuations in the training set. High variance may result from an algorithm modeling the random noise in the training data (overfitting).



Bias and Variance



Accuracy and Error

Accuracy:

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$$

Error:

$$\text{Error} = \frac{\text{Number of Incorrect Predictions}}{\text{Total Number of Predictions}}$$

Then $\text{Accuracy} = 1 - \text{Error}$

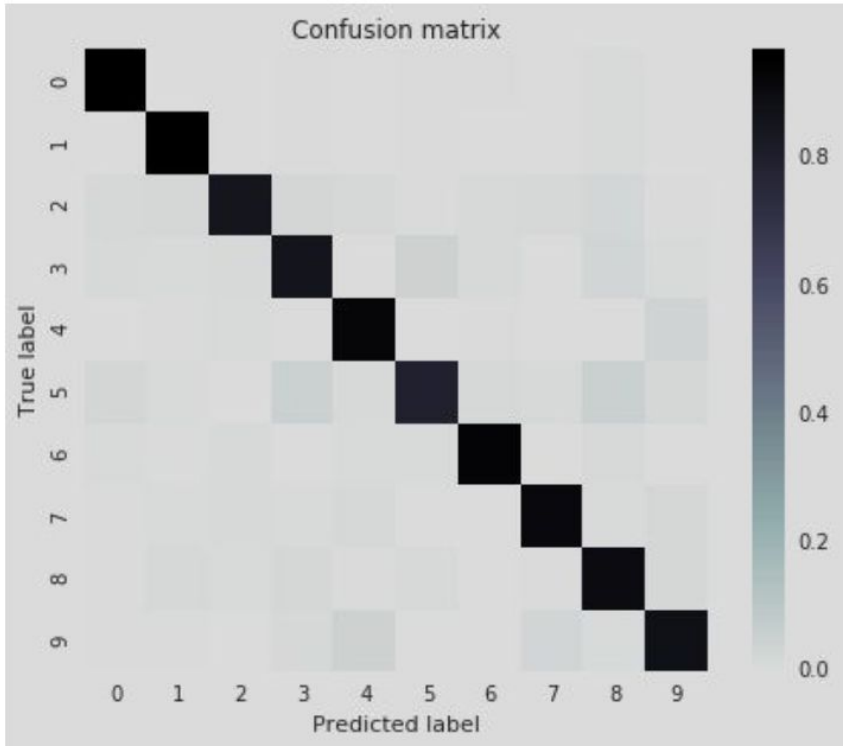
Confusion Matrix

- Accuracy can be misleading especially if the classes are unbalanced.
- The Confusion Matrix is a table that displays the performance of an algorithm where the true values are known and can be compared against the classifier correct and incorrect values.
- For example confusion matrix Two Classes

		Actual		
		Class1	Class 2	Total
Predicted	Class 1	True Positive (T_p)	False Positive (F_p)	Predicted Positive ($\hat{p}$)
	Class 2	False Negative (F_n)	True Negative (T_n)	Predicted Negative ($\hat{n}$)
	Total	Positive (p)	Negative (n)	# Examples (N_e)

Confusion Matrix

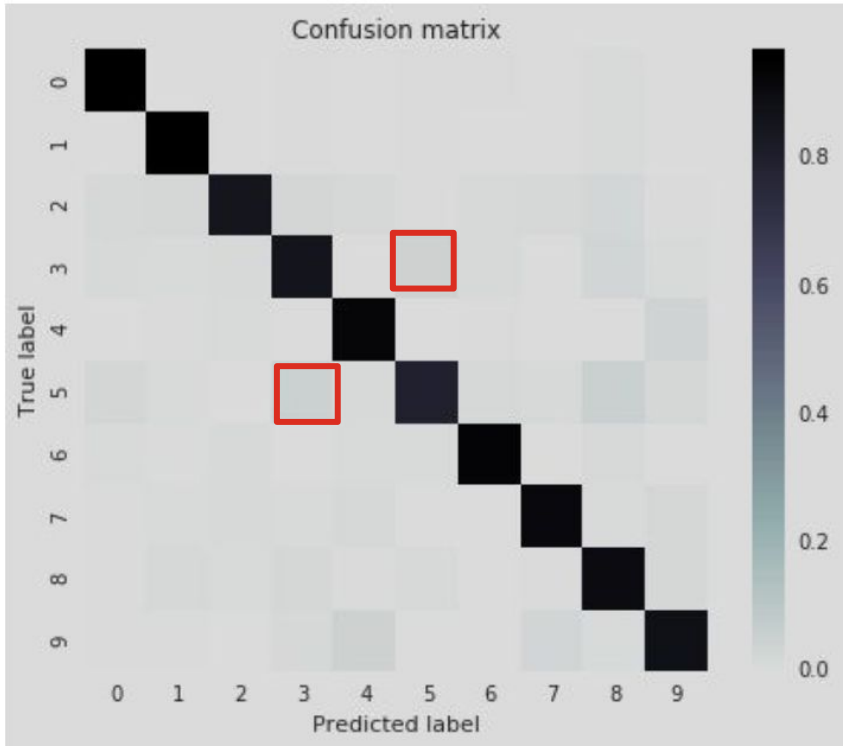
- The confusion matrix is a square matrix of size I×I associated with each class it shows the predicted and actual values, where I is the number of different classes.



For example the confusion matrix for the task of digit recognition (so possible labels are 0,1, ..., 9). The bar on the right shows the mapping from color to the probability. The high values along the diagonal indicate a strong model.

Confusion Matrix

- The confusion matrix is a square matrix of size I×I associated with each class it shows the predicted and actual values, where I is the number of different classes.

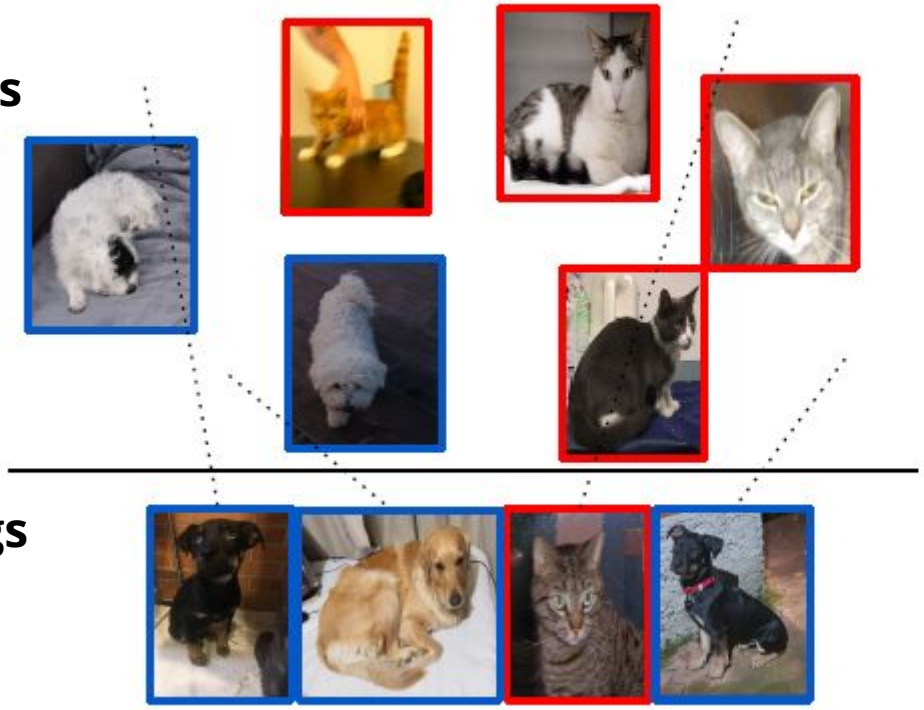


Two regions outlined in red illustrate 3 and 5 being confused.

Example

Cat / Dog Classifier: In this example the classifier correctly detect 3 dogs (Tp) and 4 cats (Tn); further, it falsely detects 1 cat as dog (Fp) and 2 dogs as cats (Fn).

Class 2: Cats



Class 1: Dogs

	Actual		
	Class1	Class 2	Total
Predicted Class 1	$T_p = 3$	$F_p = 1$	$\hat{p} = 4$
Predicted Class 2	$F_n = 2$	$T_n = 4$	$\hat{n} = 6$
Total	$p = 5$	$n = 5$	$N_e = 10$

Accuracy and Error

Accuracy and Error can be calculated from those values:

$$Acc = \frac{(T_p + T_n)}{(p+n)} = \frac{(T_p + T_n)}{(T_p + T_n + F_p + F_n)}$$

$$Error = \frac{(F_p + F_n)}{(p+n)} = \frac{(F_p + F_n)}{(T_p + T_n + F_p + F_n)}$$

Example:

$$Acc = \frac{(3+4)}{(3+4+1+2)} = 0.7$$

$$Error = \frac{(1+2)}{(3+4+1+2)} = 0.3$$

In porcentages: 70% Accurate with 30% Error

Statistical Metrics

- TPR (sensitivity or recall) which measures the probability to detect a condition when it is present, in other words perceive a condition.
- FNR (Miss Rate) is the probability to not detect a condition when the it is present, in other words, the probability of missing a condition.
- FPR(Fall-out) which measures the probability to detect when a condition when the condition is not present, in other words, the probability of a false alarm.
- TNR(or specificity) which measures the ability to not detect a condition when the condition is not present, in other words, rejecting a false condition.
- PPV(or precision) is a proportion of positives that correspond to a present condition.
- NPV is a proportion of negatives that correspond to the absence condition.

		True Condition		
		present	absent	
Predicted Condition	positive	True positive (T_p)	False positive (F_p)	Positive Predictive Value(PPV)
	negative	False negative (F_n)	True negative (T_n)	Negative Predictive Value(NPV)
		True Positive Rate(TPR)	False Positive Rate(FPR)	Accuracy (ACC)
		False Negative Rate(FNR)	True Negative Rate(TNR)	

$$ACC = (T_p + T_n)/(T_p + T_n + F_n + F_p)$$
$$TPR(Sensitivity) = T_p/(T_p + F_n)$$
$$FNR(MissRate) = F_n/(T_p + F_n)$$
$$FPR(Fall - out) = F_p/(T_n + F_p)$$
$$TNR(Specificity) = T_n/(T_n + F_p)$$
$$PPV(Precision) = T_p/(T_p + F_p)$$
$$NPV = T_n/(T_n + F_n)$$

Importance

- It is important to check all the values otherwise the analysis could be skewed.
- For instance, just using the accuracy can give a good idea how good the classifier is, yet, the result could be biased to the value that has the most elements.
  - For example, if a data set has 90% negative and 10% positively labeled observations, a classifier that labels everything as negative would be useless, and yet it would have 90% accuracy.
  - In the same problem it would have 0% precision since it focuses only on the cases classified as positive and there are none (T_p/(T_p+F_p)).
  - Similarly, Recall consider everything that was classified as positive and note the similarities to everything the was meant to be labeled as positive T_p/(T_p+F_n) which in this case is also 0%.

F1-score

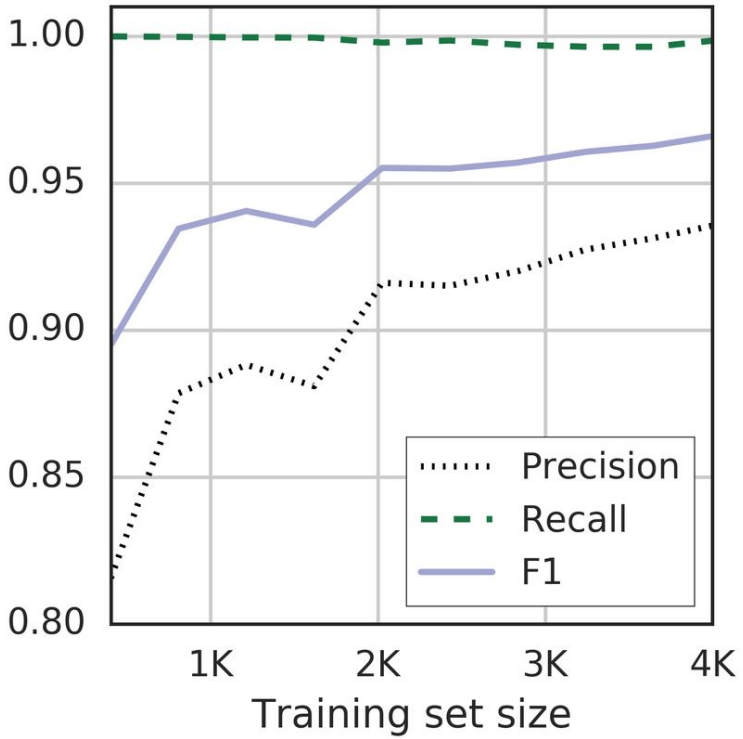
- Another metric is the F_1 Score that is a weighted average of Precision and Recall.
- This score takes both false positives and false negatives equally into account.

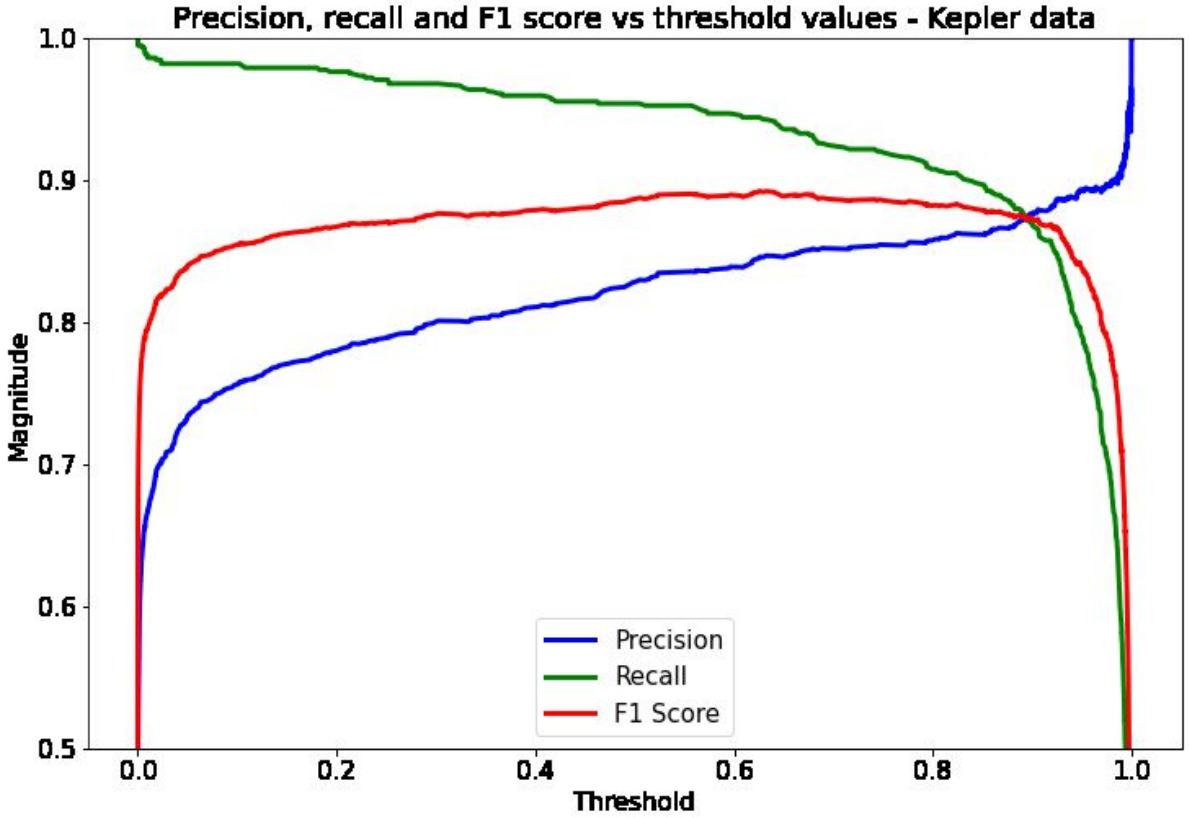
$$F_1 Score = 2 * \frac{Recall * Precision}{Recall + Precision}$$

- This measurement uses the harmonic mean since it penalize extreme values.
- Hence, F_1 score would be a better metric used on imbalance classification problems where it is required to have a high precision and high recall.
- For example, if the method has high precision of 1.0 and a low recall of 0.0 it would have a F_1 of zero.
- It is important to notice that these metrics can be easily expanded to a matrix of size n x n with n as the total number of classes.
    - In each row it is stated how many of that class was predicted as the correct class or other.

F1-score

$$F_1 Score = 2 * \frac{Recall * Precision}{Recall + Precision}$$





Example (cont)

Following the example of the values would be:

$$TPR = \frac{T_p}{T_p + F_n} = \frac{3}{3+2} = 0.6,\ FNR = \frac{F_n}{T_p + F_n} = \frac{2}{3+2} = 0.4,$$
$$FPR = \frac{F_p}{T_n + F_p} = \frac{1}{4+1} = 0.2,\ TNR = \frac{T_n}{T_n + F_p} = \frac{4}{4+1} = 0.8,$$
$$PPV = \frac{T_p}{T_p + F_p} = \frac{3}{3+1} = 0.75,\ NPV = \frac{T_n}{T_n + F_n} = \frac{4}{4+2} = 0.666$$

In the example from before the F_1 Score has a value of:

$$F_1 Score = 2 * \frac{Recall * Precision}{Recall + Precision} = 2 * \frac{0.6 * 0.75}{0.6 + 0.75} = \overline{0.666}$$

Precision vs Recall Trade-Offs

A system with high precision might let through some spam but is also very unlikely to miss any good emails (Tp)/(Tp+Fp).

A system with high recall might leave out some good emails, but it is very unlikely to let spam through Tp/(Tp+Fn).

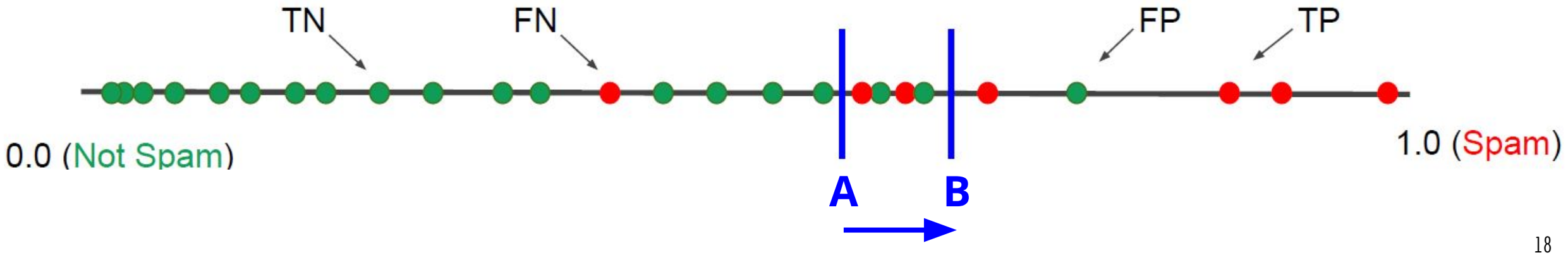
The trick is to balance them.
What kind of error is more problematic? Depends a lot on the application.

Examples?

Selecting a Threshold

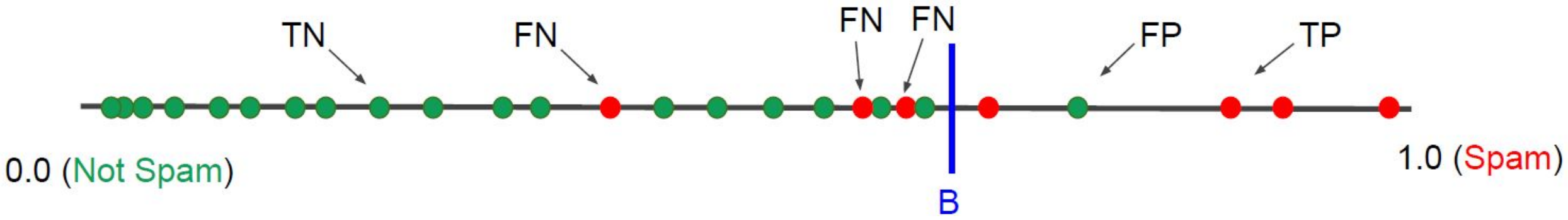
- True label shown by color green or red
- Prediction from model shown as location on the line

- What happens to precision and recall with a larger decision threshold (B vs A)?
  - Prec = TP / (TP + FP)
  - Recall = TP / (TP + FN)



Selecting a Threshold

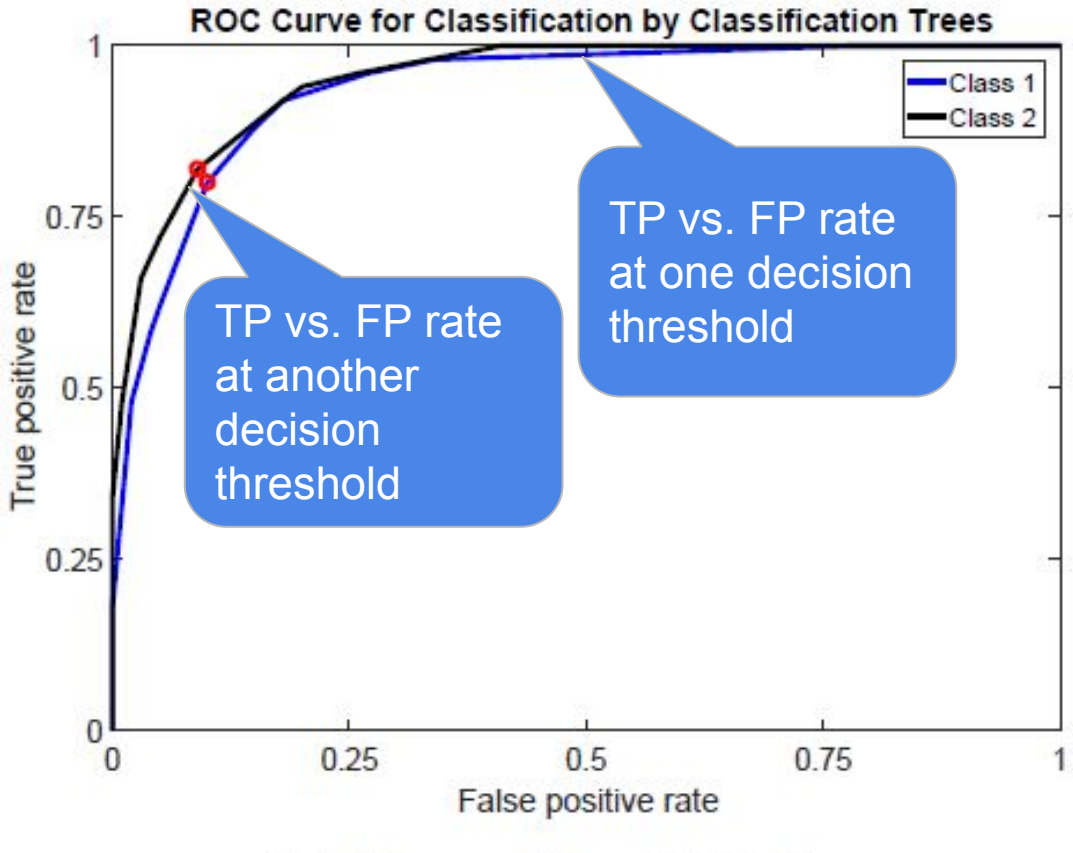
- What happens to precision and recall with a larger decision threshold (B vs A)?
  - Prec = TP / (TP + FP)
  - Recall = TP / (TP + FN)



RoC Curves and Kappa Value

- Although, accuracy is the first measure to assess a classifier efficiency, alone is typically not enough to decide on how good is a classifier.
  - Since it becomes a problematic metric as it does not compensate for success due to mere chance.
  - For example two class classifier with 50% accuracy
- Hence, the use of two measurements that take randomness into account:
  - The Receiver Operating Characteristic(ROC curves)
  - The Kappa Coefficient.

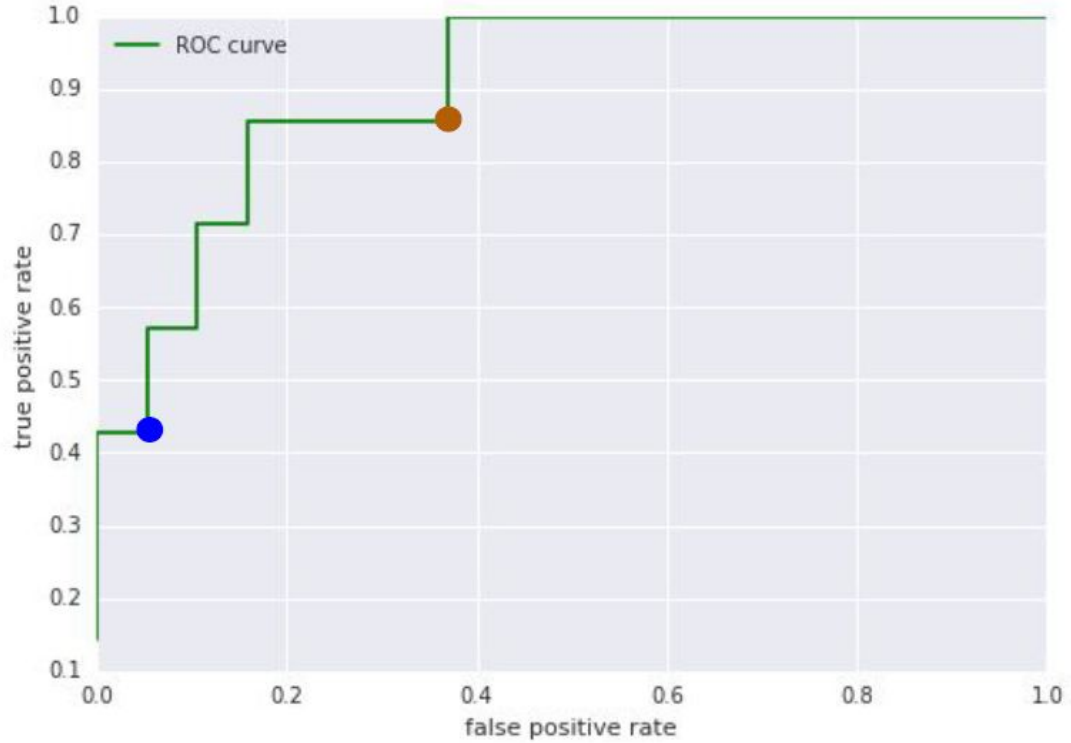
RoC curves

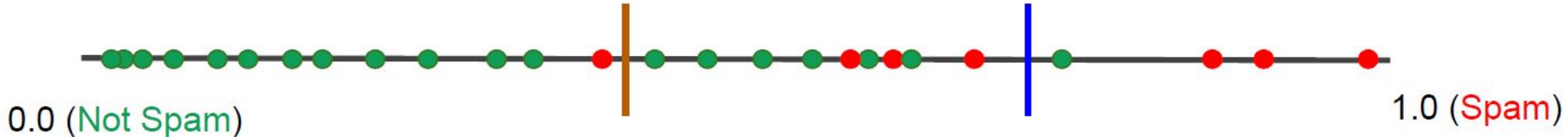


TP Rate = TP / (TP + FN)

FP Rate = FP / (FP + TN)

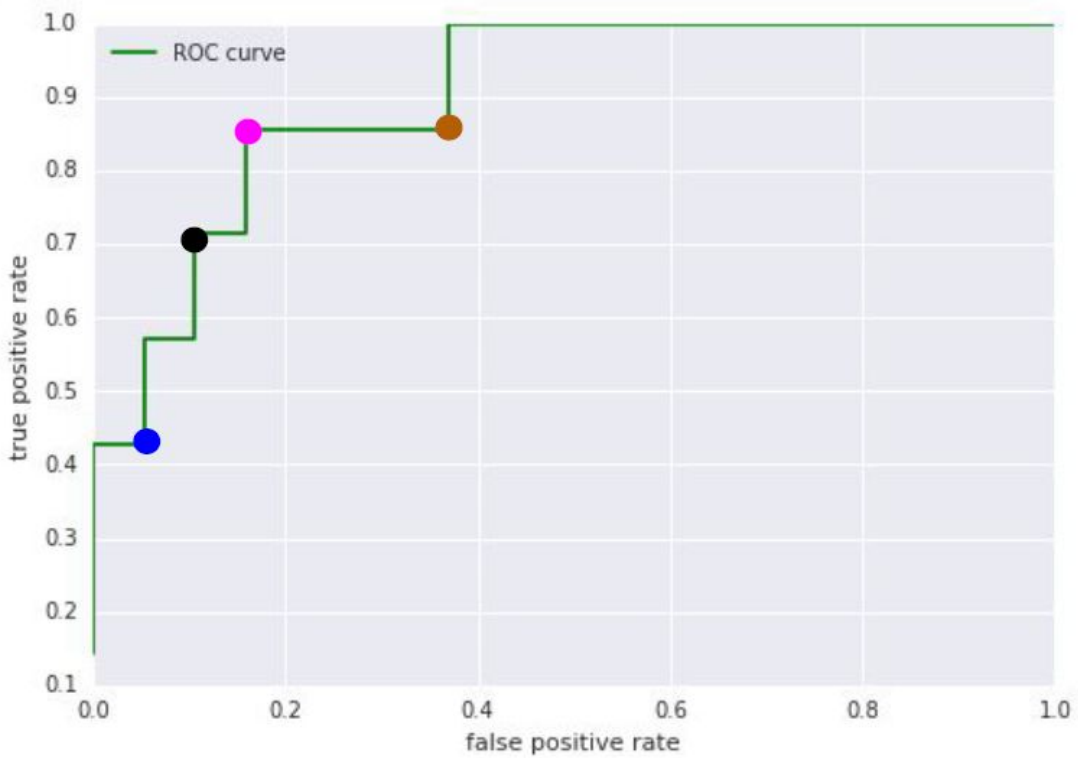
Sample ROC Curve

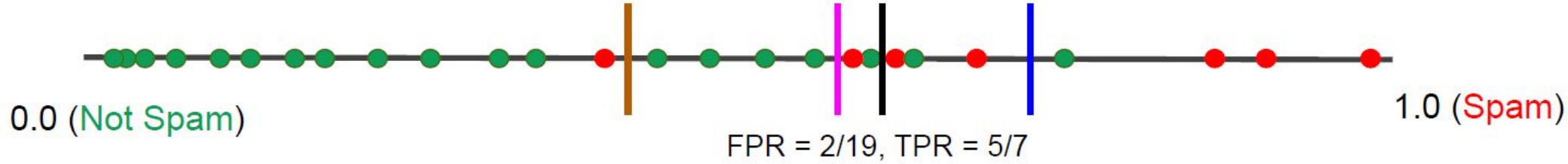




FPR = 7/19, TPR = 6/7 **FPR = 1/19, TPR = 3/7**

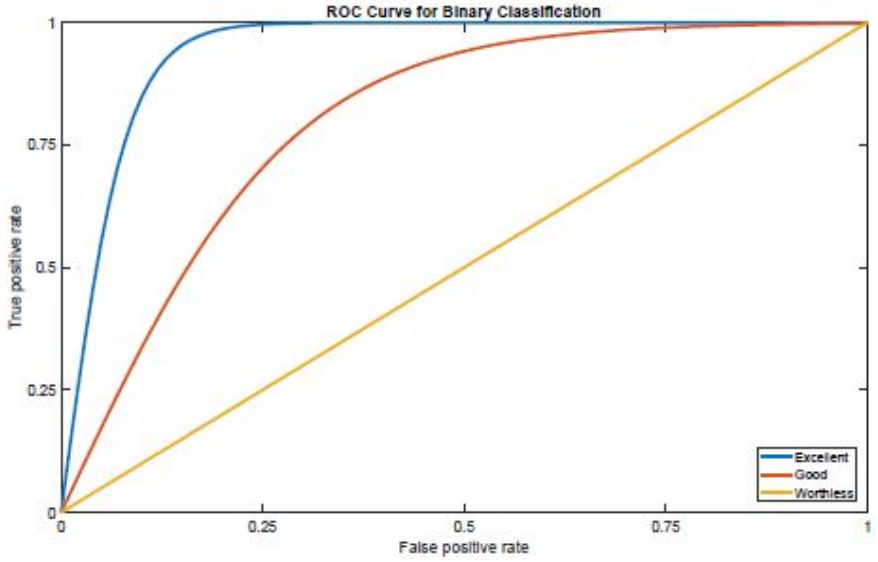
Sample ROC Curve

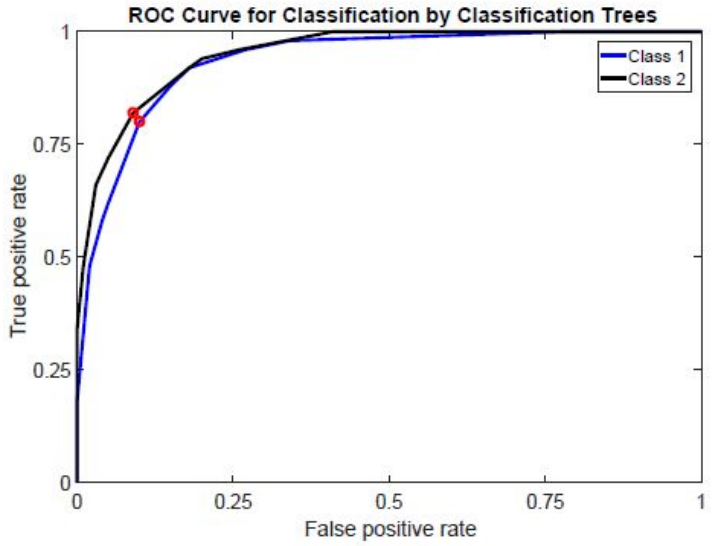




RoC curves

- ROC curves are a very powerful tools to measure classifiers' accuracy in binary-class problems.
- They are two-dimensional graphs which plots sensitivity vs Fall-out (or True Positive Rate against False Positive Rate)
- The closer to the upper left corner the better



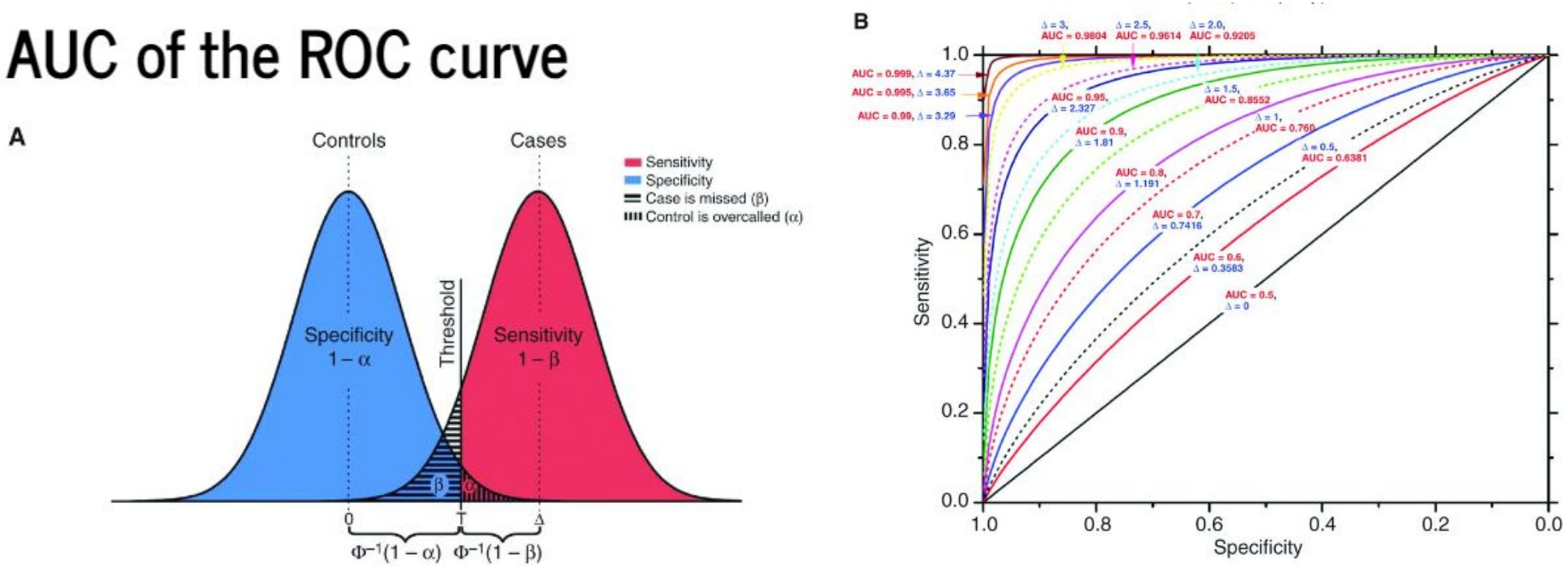


(b) Example of ROC

RoC curves AUC

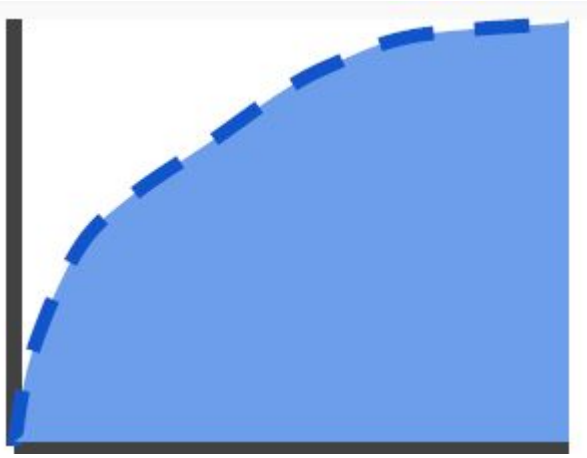
Normally when comparing between classifiers it is desire to have a single number as a measure of the classifier performance.

Hence, the area under the ROC curve AUC.



Evaluation Metrics: AUC

- Interpretation:
  - If we pick a random positive and a random negative,
- what's the probability my model scores them in the correct relative order?
- Intuition: gives an aggregate measure of performance aggregated across all possible classification thresholds



Kappa Coefficient

Despite ROC curves have been generalized for a multi-class setting its practical usability has not been shown yet.

Instead, Cohen's Kappa Coefficient is preferred for its simplicity and multi-class setting.

The Kappa coefficient is calculated as:

$$\kappa = \frac{p_o - p_e}{1 - p_e}$$

where p_0 is the total agreement probability, or the accuracy, and p_e is the agreement probability which is due to chance.

Special Note

- Cohen's Kappa always ranges between 0 and 1, with 0 indicating no agreement between the two raters and 1 indicating perfect agreement between the two raters.
- The following table summarizes how to interpret different values for Cohen's Kappa:
  - Poor agreement = Less than 0.20
  - Fair agreement = 0.20 to 0.40
  - Moderate agreement = 0.40 to 0.60
  - Good agreement = 0.60 to 0.80
  - Very good agreement = 0.80 to 1.00

Example (cont)

	Actual		
	Class1	Class 2	Total
Predicted Class 1	$T_p = 3$	$F_p = 1$	$\hat{p} = 4$
Predicted Class 2	$F_n = 2$	$T_n = 4$	$\hat{n} = 6$
Total	$p = 5$	$n = 5$	$N_e = 10$

- The first step is to calculate the probability that both the raters are in perfect agreement:
  - Observed Agreement, po = (TP + TN) / N = (3+4)/10 = 0.7
- The second step calculate the expected probability that both the raters are in agreement by chance.
  - This is calculated by adding up the multiplied expected probability that both the raters are in agreement that the classes are positive, and the multiplied the expected probability that both the raters are in agreement that the classes are negative.
  - pe = [{Pe(rater 1 says Yes) / N}* {Pe(rater 2 says Yes) / N} + {Pe(rater 1 says no) / N} * {Pe(rater 2 says no) / N}]
  - pe = [(3+1)/10*(3+2)/10 + (2+4)/10*(1+4)/10] = [0.4*0.5 + 0.6*0.5]=0.5
- The Third step calculate Cohen's Kappa:
  - Kappa score = (Po – Pe) / (1 – Pe) = (0.7 - 0.5)/(1 - 0.5) = 0.4
- They would be on "Fair agreement"

Example Cohen's Kappa

Two museum curators are asked to rate 70 paintings on whether they're good to be hung in a exhibit.

		Rater 2	
		Yes	No
Rater 1	Yes	25	10
	No	15	20

- Step 1: Calculate relative agreement (p_o) between raters.
  - p_o = (Both said Yes + Both said No) / (Total Ratings) = (25 + 20) / (70) = 0.6429
- Step 2: Calculate the hypothetical probability of chance agreement (pe) .
  - The probability of both said "Yes" divided by the total number of responses multiplied, added to the probability of both said "No" multiplied
  - p("Yes") = ((25+10)/70) * ((25+15)/70) = 0.285714; p("No") = ((15+20)/70) * ((10+20)/70) = 0.214285
  - pe = 0.285714 + 0.214285 = 0.5
- Step 3: Calculate Cohen's Kappa
  - k = (po – pe) / (1 – pe) = (0.6429 – 0.5) / (1 – 0.5) = 0.2857

- Based on the table from earlier, the two raters only had a "fair" level of agreement.

In General

In general, p_0 and p_c can be calculated as:

$$p_0 = \sum_{i=1}^{I} p(x_{ii}); \quad p_c = \sum_{i=1}^{I} p(x_{i}.)p(x_{.}i),$$

where i is the number of classes, p(x_.i) as the columns marginal probabilities, p(x_i.) as the rows marginal probabilities and p(x_{ii}) is the successful hit probabilities on the main diagonal of the confusion matrix.

Example

Actual (Rater 1) / Predicted (Rater 2)	Cat	Dog	Monkey	Total
Cat	15	10	5	30
Dog	10	20	20	50
Monkey	20	10	10	40
Total	45	40	35	120

- The confusion matrix representing a image classification model which classifies image into three different classes such as cat, dog and monkey.
- Lets calculate the Kappa score for the above confusion matrix representing multiclass classification model:
  - Po = (15 + 20 + 10) / 120 = 45/120 = 0.375
  - Pe = 45/120*30/120 + 40/120*50/120 + 35/120*40/120
    = 0.375*0.25 + 0.33*0.42 + 0.292*0.33 = 0.09375 + 0.1386 + 0.09636 = 0.204
  - K = (Po – Pe) / (1 – Pe) = (0.375 – 0.204) / (1 – 0.204) = 0.171 / 0.796 = 0.215

Python

```
from sklearn.metrics import cohen_kappa_score

#define array of ratings for both raters
rater1 = [0, 1, 1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0]
rater2 = [0, 0, 1, 1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 1, 0]

#calculate Cohen's Kappa
cohen_kappa_score(rater1, rater2)
```