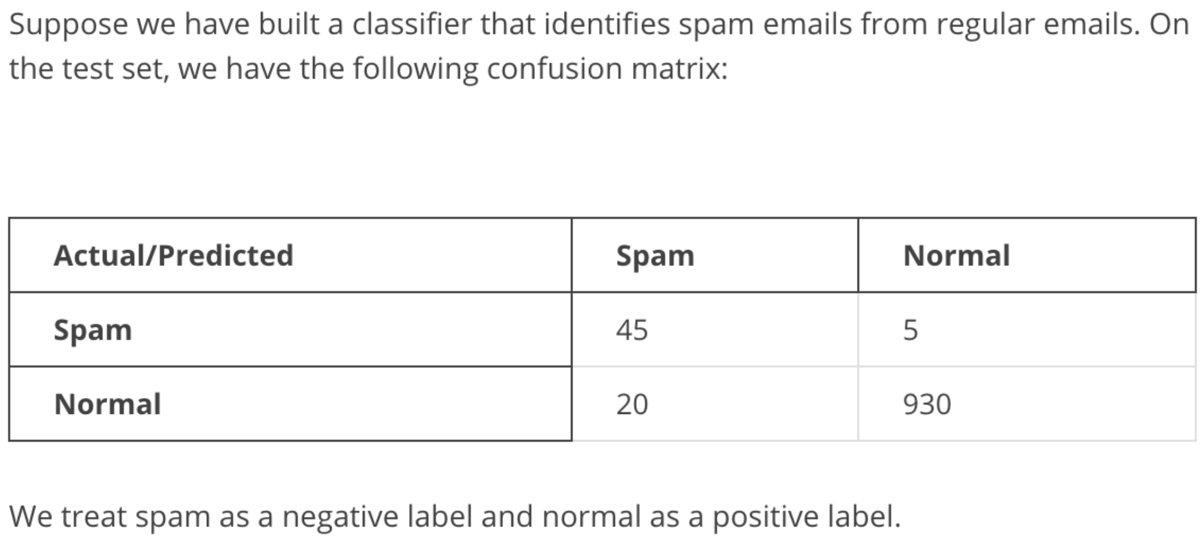
CS412 Office Hours

April 15, 2019

Programming Assignment: Decision Tree

1. You can use either binary split or full split, both will pass the test. Full split is simpler to implement.
2. The displayed score is not equal to prediction accuracy!
   a. Naive implementation should achieve ~50% accuracy and get a score of 90/100.
3. If your decision tree is overfitting, try to prune your tree using a validation set
   a. Some simple ways to prune the tree include stop splitting when the number of datapoints at the node is smaller than k or stop growing the tree when the number of levels exceeds L.
4. If your decision tree is running too slow
   a. Are you copying the dataset around?
   b. Are you iterating through the entire dataset when you only need a partition?
   c. Are you using libraries that are slow? (For example pandas)
   d. It’s ok if your code is not optimal. We will grade for output, not efficiency.

Proper Metrics for Imbalanced Classes



Suppose we have built a classifier that identifies spam emails from regular emails. On the test set, we have the following confusion matrix:

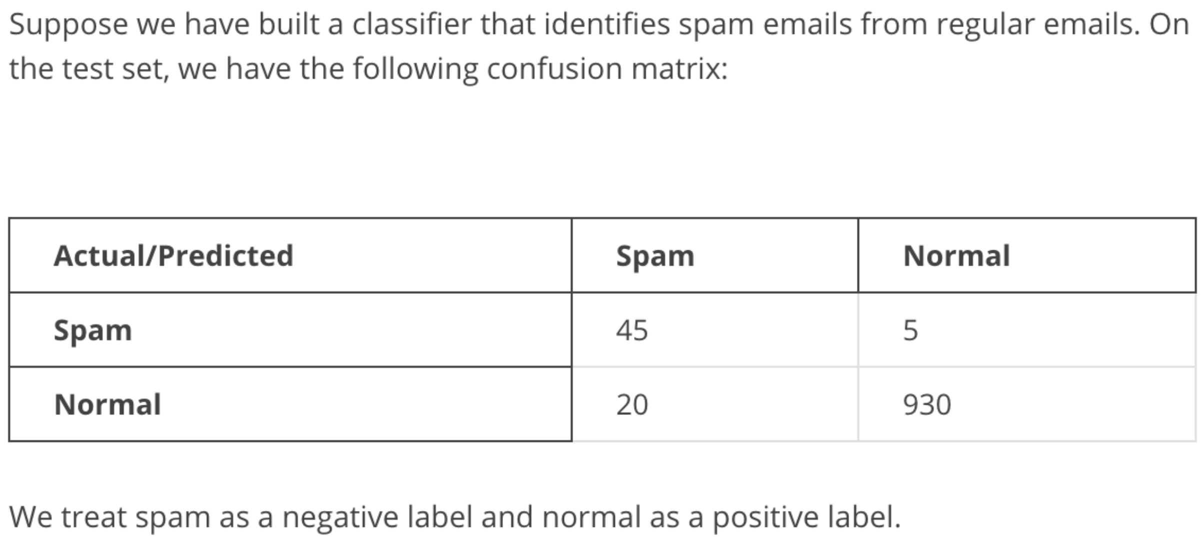
Actual/Predicted	Spam	Normal
Spam	45	5
Normal	20	930

We treat spam as a negative label and normal as a positive label.

When we evaluate the spam classifier above, which of the following metrics are suitable?

Accuracy; Specificity; Sensitivity; Precision; Precision on the negative label.

Proper Metrics for Imbalanced Classes



Suppose we have built a classifier that identifies spam emails from regular emails. On the test set, we have the following confusion matrix:

Actual/Predicted	**Spam**	**Normal**
Spam	45	5
Normal	20	930

We treat spam as a negative label and normal as a positive label.

When we evaluate the spam classifier above, which of the following metrics are suitable?

Accuracy; **Specificity**; Sensitivity; Precision; **Precision on the negative label**.

Not Interested in Dominating Labels

In this question, the positive label (i.e. normal emails) is dominating the dataset, while our task is to identify the negative labels (i.e. spams). Therefore, the positive label is not interesting, and measures depending on the positives are not interesting.

Accuracy = (TP + TN) / All	<- Depends on positives
Specificity = TN / N	**<- Not depends on positives**
Sensitivity = TP / P	<- Depends on positives
Precision = TP / (TP + FP)	<- Depends on positives
Precision on the negative label = TN / (TN + FN)	**<- Not depends on positives**

Why?

Because a dumb model may appear to give very high performance if evaluated with metrics depending on the positive label!

Suppose we use a dumb model “predicting every email as normal (positive)”.

This spam filter is not doing anything useful. What are the metrics for this model?

We have 950 positives and 50 negatives. The dumb model predicts 1000 positives.

Accuracy = (TP + TN) / All = (950 + 0) / 1000 = 0.95 -> The model is good -> Wrong

Specificity = TN / N = 0 / 50 = 0 -> The model is dumb -> Correct

Sensitivity = TP / P = 950 / 950 = 1 -> The model is perfect -> Wrong

Precision = TP / (TP + FP) = 950 / (950 + 50) = 0.95 -> The model is good -> Wrong

Precision on the negative label = TN / (TN + FN) = 0 / (0 + 50) = 0

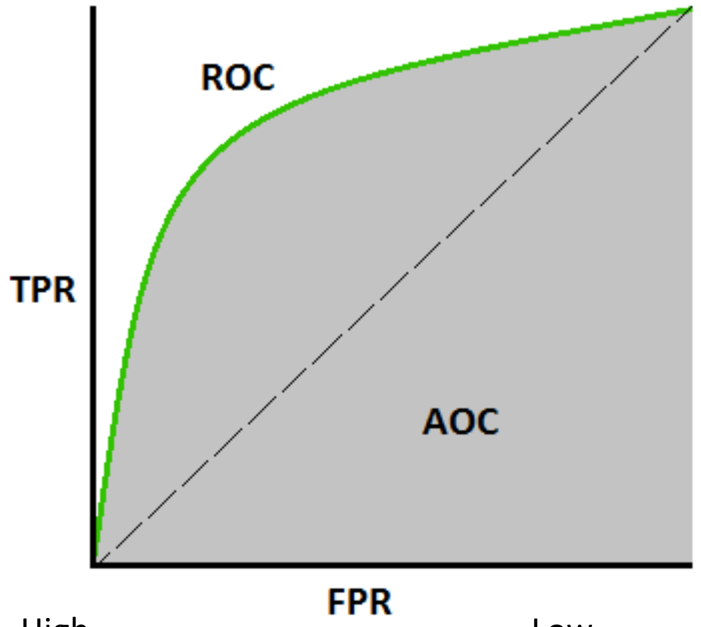
-> The model is dumb -> Correct

Take-Home Message

Before using a metric, think twice about whether it is interesting for the specific problem and dataset.

- ❑ Is the metric truly reflecting how well the intended task is performed?
- ❑ Will a dumb model be evaluated as dumb under this metric?

ROC Curve



High confidence → Low confidence

Points on the ROC curve are obtained by changing the classification threshold.

Higher confidence threshold ⇒ less false positives, less true positives

Lower confidence threshold ⇒ more false positives, more true positives

The ROC curve cannot be read from a single confusion matrix.

One ROC curve can only be used to describe binary classification. For multi-class classification, can plot one VS one or one VS all ROC curves.