Classification Metrics

DATA SCIENCE BOOTCAMP

Classification Metrics

- How do I measure the performance of a (binary) classification algorithm?
- How do I compare different algorithms to see which is the best one?
- How do I know if my current performance is “good enough”?

Types of Classification

Can consider 3 levels of responses to a binary classification problem:

1) **Hard classification** - For each test case, answer yes or no as to whether it is "in class" or not.

2) **Ranking classification** - Rank all the test cases from most likely to least likely to be "in class".

3) **Probability estimation** - For each test case, estimate the probability that that the case is "in class".

Observations

- Can think of the three types of classification in order of increased sophistication.
- A probability estimation yields a ranking.
- A ranking yields a hard classification **for each possible threshold**.
- Each kind of classification has different metrics associated with its performance.

Metrics for Classification Algorithms

- Hard Classification
  - Sensitivity (aka Recall)
  - Pos. Pred. Value (aka Precision)
  - Specificity
  - Accuracy
  - Others…
- Ranking Classification
  - AUC (Area Under ROC Curve, aka C-Statistic)
  - Examine Precision-Recall curve
- Probability Estimation
  - Log-likelihood of the test data (test-Deviance)
  - Brier Score
  - AIC, BIC, other penalized likelihood

The Fish / Seaweed Analogy

A good net catches most of the fish, but allows the seaweed to pass through.

We can "cast a wider net" (lower the threshold) to catch more fish, but will probably catch more seaweed in the process.

TP = Fish in the net (good)
FP = Seaweed in the net (bad)
TN = Seaweed not in the net (good)
FN = Fish not in the net (bad)

Metrics in Fish / Seaweed Framework

Sensitivity (aka Recall / TPR):
  What % of the fish did I catch in my net?
  (Denominator: Fish)

PPV/Precision:
  What % of my net is fish (vs. seaweed)?
  (Denominator: Net)

Specificity:
  What % of the seaweed did I leave out of the net?
  (Denominator: Seaweed)

NPV:
  What % of the non-net stuff is seaweed?
  (Denominator: non-Net)

Accuracy:
  What % of stuff is in the right place?
  (Denominator: Everything)

Balanced and Imbalanced Classes

- Some classification problems are **balanced**. (“in-class” and “out-of-class” are each about 50%)
- Many (most?) are **imbalanced**. Some extremely so.
  - Fraud detection (from .01% to .0001%)
  - Disease detection (<1%)
- In balanced problems, there is symmetry (and intuition) that falls apart on imbalanced problems.
- Suggestion: Keep an imbalanced problem in your head as your “canonical” example.
- Take-home: The more imbalanced the problem, the more careful one must be interpreting metrics.
- Main culprits: Accuracy, Specificity, False Positive Rate, and anything derived from them (AUC).

A (Roughly) Balanced Problem

	Fish	Seaweed
Net	80	25
Non-Net	20	75

Sens = TP/(TP+FN) = 80/100 =.8

Spec = TN/(TN + FP) = 75/100 = .75

Acc = (TP+TN)/(TP+TN+FP+FN) = .775

PPV = TP/(TP+FP) = 80/105 =.762

NPV = TN/(TN+FN) = 75/95 = .789

Note: Seaweed = Fish = 100

so: Acc = (Sens+Spec)/2

Net ~= Non-Net (105 vs 95)

so: Acc ~= (NPV + PPV)/2

Imbalanced Problem (100x seaweed)

	Fish	Seaweed
Net	80	2500
Non-Net	20	7500

Sens = TP/(TP+FN) = 80/100 =.8
Spec = TN/(TN + FP) = 7500/10000 = .75
Acc = (TP+TN)/(TP+TN+FP+FN) = 7580/10100 =.7504
PPV = TP/(TP+FP) = 80/2580 =.031
NPV = TN/(TN+FN) = 7500/7520 = .997

Note: Seaweed = 100 x Fish
so: Acc ~= Spec

and: Net ~= 1/3 * Non-Net
so: Acc ~= (3*NPV + PPV)/4

and NPV is inflated!

Imbalanced Problem (Even more seaweed!)

	Fish	Seaweed
Net	80	2500
Non-Net	20	17500

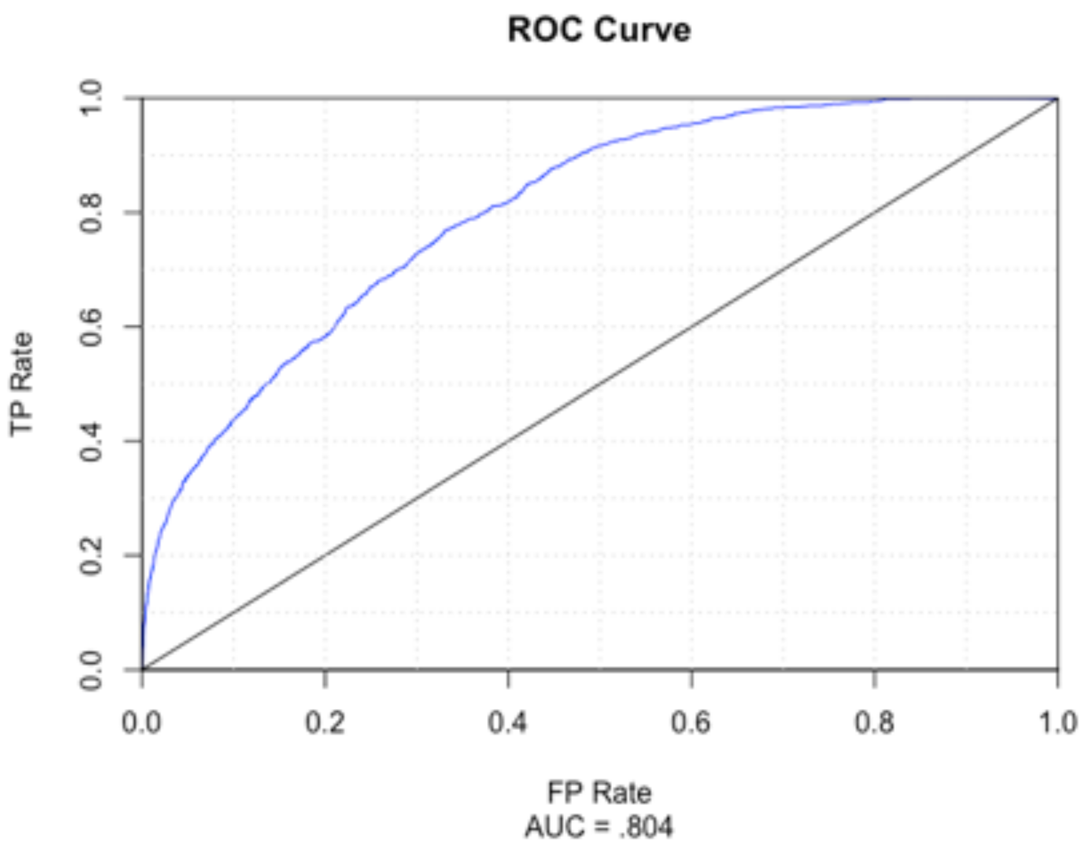
Add “easily filterable” seaweed to our ocean.

Sens = TP/(TP+FN) = 80/100 =.8
Spec = TN/(TN + FP) = 17500/20000 = .875
Acc = (TP+TN)/(TP+TN+FP+FN) = 17580/20100 =.8746
PPV = TP/(TP+FP) = 80/2580 = .031
NPV = TN/(TN+FN) = 17500/17520 = .9989

Specificity, Accuracy and NPV all improved! Why?
Sensitivity and PPV did not.

Metrics for Ranking Classification

ROC Curve - Plot Sensitivity vs FPR (FPR = 1-Spec)
(ROC = Receiver Operating Characteristic)



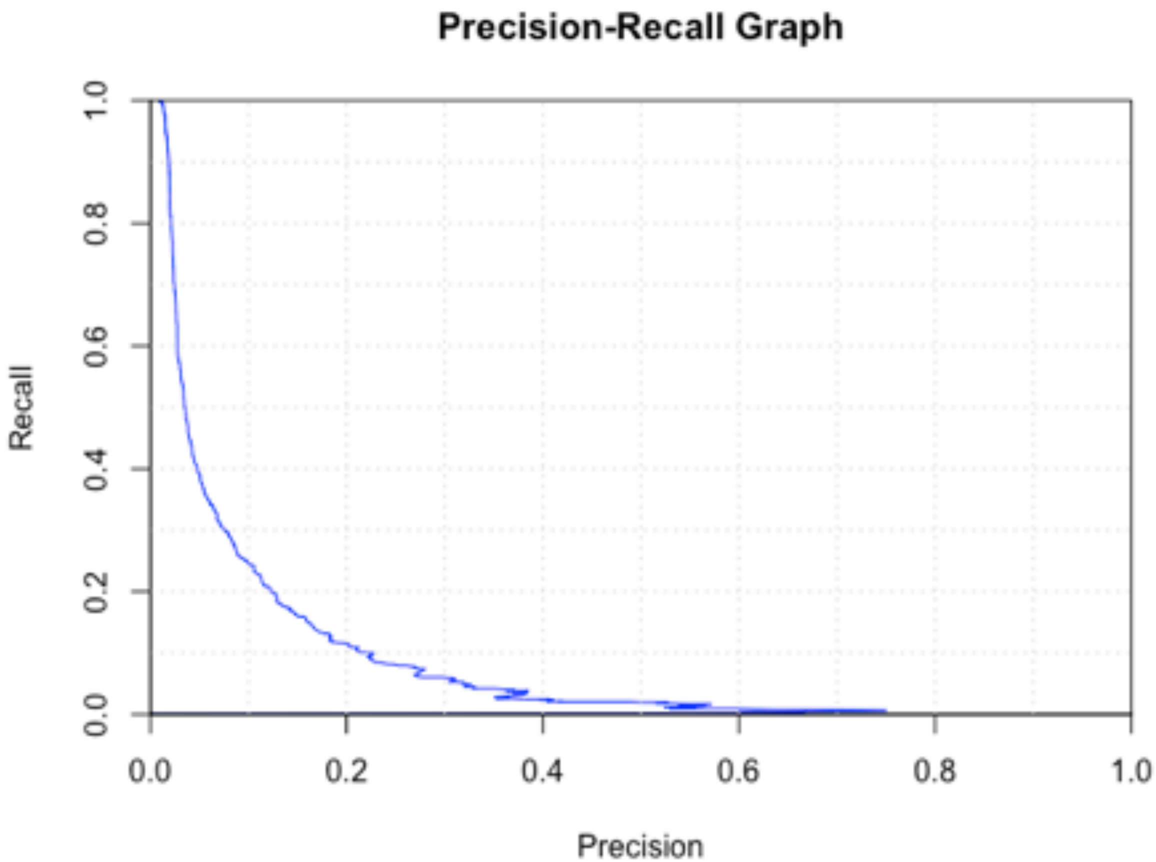
Metrics for Ranking Classification

AUC - Area Under the ROC Curve

- Average specificity across all sensitivity values
- AUC is the probability that a random “in-class” case is ranked higher than a random “out-of-class” case
- Can game AUC (and Spec., Acc.) by adding obvious negatives to population.
- Not a useful measure to compare models unless they are on the exact same test set.
- Evaluates entire range of thresholds, even though you may only operate in a narrow range

Metrics for Ranking Classification

The Precision-Recall graph gives a different perspective



Probability Estimation vs. Ranking

- For some applications, it may be important to estimate an accurate probability of being in-class.
- May be desirable to use a metric that rewards accuracy and calibration as well as stratification.
- Simple idea: evaluate the likelihood of the test data

  $p_1 = .6$, $p_2 = .8$, $p_3 = .9$ and $y_1 = 1$, $y_2 = 0$, $y_3 = 1$

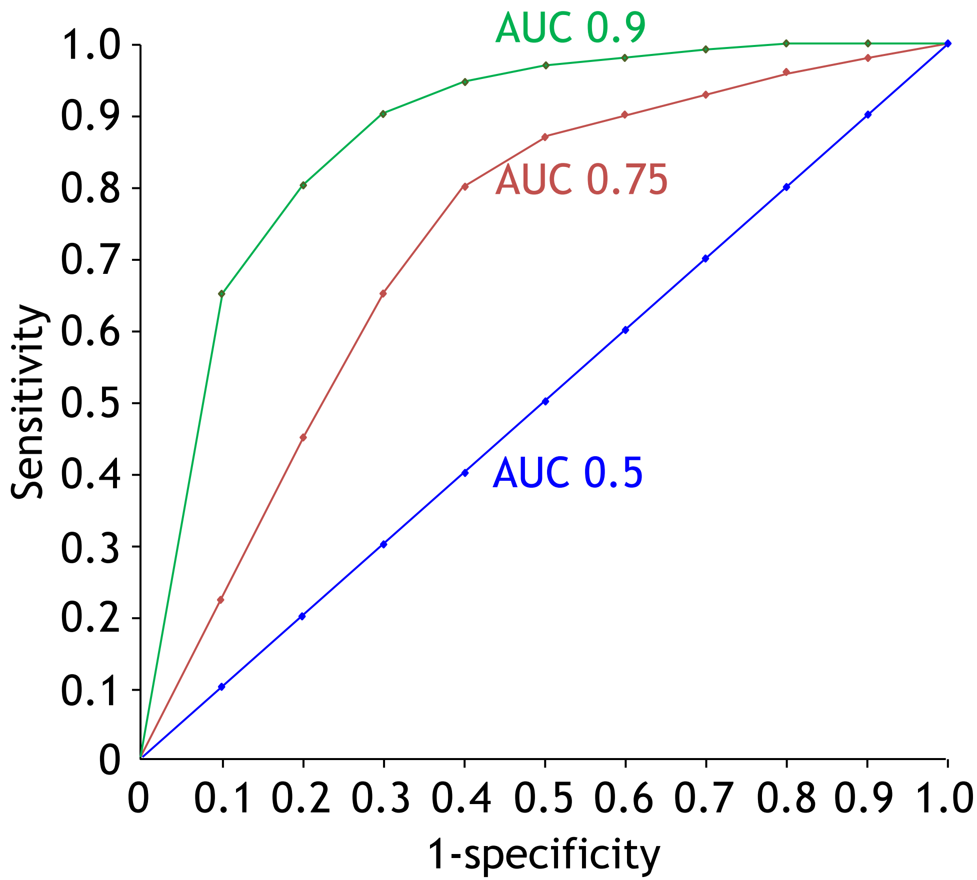
  Likelihood = .6 * (1-.8) * .9
- Test-Deviance (DIC) = -2 * (log-likelihood of test data)(smaller deviance is better)
- Brier Score: Essentially the Mean Squared Error

$$BS = \frac{1}{n}\sum_{i=1}^{n}(\hat{p}_i - y_i)^2$$

Penalized Likelihoods

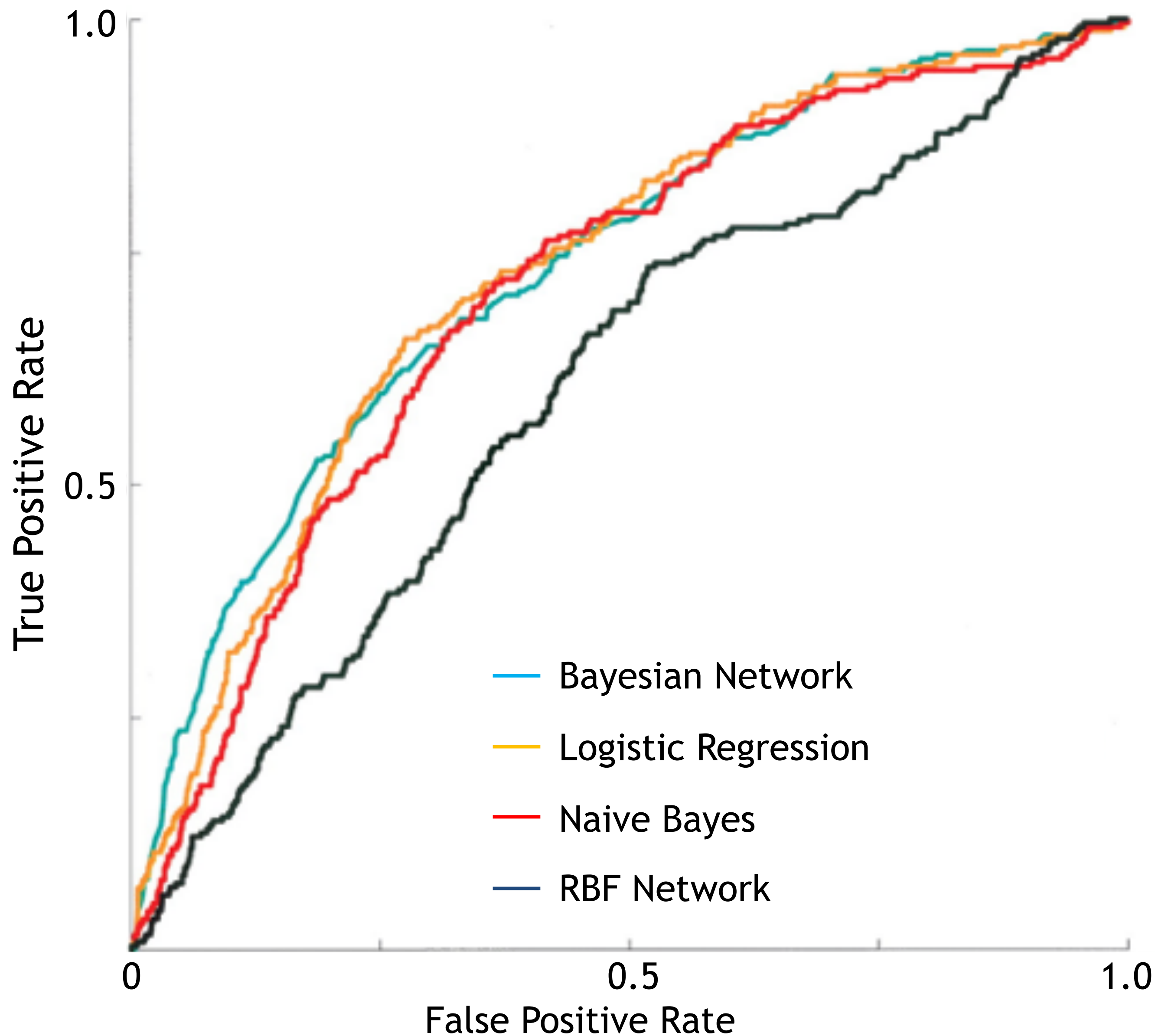
- Typically used to compare models when an independent test set is **not** available
- Idea - adding more parameters (e.g. in logistic regression) will always reduce likelihood of the **training** data.
- Therefore, to compare models we consider the likelihood plus a penalty term for model complexity.
- Attempts to measure whether the gain in likelihood "justifies" the additional parameters.
- AIC = -2(LL) + 2k (where k is number of parameters)
- BIC = -2(LL) + k*(ln n) (more conservative: ln n > 2)

Some Extra Slides Follow



Area under curve (AUC)

An evaluation of a classification algorithm (including all possible thresholds)



1.0
0.5
True Positive Rate
Bayesian Network
Logistic Regression
Naive Bayes
RBF Network
0
0.5
1.0
False Positive Rate

from sklearn.metrics import ……

Classification metrics

See the *Classification metrics* section of the user guide for further details.

metrics.accuracy_score(y_true, y_pred[, …])	Accuracy classification score.
metrics.auc(x, y[, reorder])	Compute Area Under the Curve (AUC) using the trapezoidal rule
metrics.average_precision_score(y_true, y_score)	Compute average precision (AP) from prediction scores
metrics.classification_report(y_true, y_pred)	Build a text report showing the main classification metrics
metrics.confusion_matrix(y_true, y_pred[, …])	Compute confusion matrix to evaluate the accuracy of a classification
metrics.f1_score(y_true, y_pred[, labels, …])	Compute the F1 score, also known as balanced F-score or F-measure
metrics.fbeta_score(y_true, y_pred, beta[, …])	Compute the F-beta score
metrics.hamming_loss(y_true, y_pred[, classes])	Compute the average Hamming loss.
metrics.hinge_loss(y_true, pred_decision[, …])	Average hinge loss (non-regularized)
metrics.jaccard_similarity_score(y_true, y_pred)	Jaccard similarity coefficient score
metrics.log_loss(y_true, y_pred[, eps, …])	Log loss, aka logistic loss or cross-entropy loss.
metrics .matthews_corrcoef(y_true, y_pred)	Compute the Matthews correlation coefficient (MCC) for binary classes
metrics. precision_recall_curve(y_true, …)	Compute precision-recall pairs for different probability thresholds
	Compute precision, recall, F-measure and support for each class
	Compute the precision
	Compute the recall
	Compute Area Under the Curve (AUC) from prediction scores

Always remember,

Fit the model to a training set,

Calculate performance
(accuracy, precision, recall, f1, AUC,
etc.) on a test set
or (better) on a k-fold cross validation
scheme