Errors in classification

		Condition (as determined by "Gold standard")			
	Total population	Condition positive	Condition negative	Prevalence = $\frac{\Sigma\ \text{Condition positive}}{\Sigma\ \text{Total population}}$	
Test outcome	Test outcome positive	**True positive**	**False positive** (Type I error)	Positive predictive value (PPV, Precision) = $\frac{\Sigma\ \text{True positive}}{\Sigma\ \text{Test outcome positive}}$	False discovery rate (FDR) = $\frac{\Sigma\ \text{False positive}}{\Sigma\ \text{Test outcome positive}}$
	Test outcome negative	**False negative** (Type II error)	**True negative**	False omission rate (FOR) = $\frac{\Sigma\ \text{False negative}}{\Sigma\ \text{Test outcome negative}}$	Negative predictive value (NPV) = $\frac{\Sigma\ \text{True negative}}{\Sigma\ \text{Test outcome negative}}$
	Positive likelihood ratio (**LR+**) = TPR/FPR	True positive rate (TPR, Sensitivity, Recall) = $\frac{\Sigma\ \text{True positive}}{\Sigma\ \text{Condition positive}}$	False positive rate (FPR, Fall-out) = $\frac{\Sigma\ \text{False positive}}{\Sigma\ \text{Condition negative}}$	Accuracy (ACC) = $\frac{\Sigma\ \text{True positive} + \Sigma\ \text{True negative}}{\Sigma\ \text{Total population}}$	
	Negative likelihood ratio (**LR–**) = FNR/TNR	False negative rate (FNR) = $\frac{\Sigma\ \text{False negative}}{\Sigma\ \text{Condition positive}}$	True negative rate (TNR, Specificity, SPC) = $\frac{\Sigma\ \text{True negative}}{\Sigma\ \text{Condition negative}}$		
	Diagnostic odds ratio (**DOR**) = LR+/LR–				

%54 Democrats, %46 republicans

Classify using their votes

%54 Democrats, %46 republicans

Classify using their votes

Model performance:
“How many times did I get it right?”

%54 Democrats, %46 republicans

Classify using their votes

Model performance:
"How many times did I get it right?"

Accuracy: % correct prediction of all predictions

%54 Democrats, %46 republicans

Classify using their votes

Model performance:
“How many times did I get it right?”

Accuracy: % correct prediction of all predictions

95% accuracy: Good job!

%1 have leukemia, %99 are healthy

Classify using health records and tests

%1 have leukemia, %99 are healthy

Classify using health records and tests

"How many times did I get it right?"

Accuracy: % correct prediction of all predictions

%1 have leukemia, %99 are healthy

Classify using health records and tests

"How many times did I get it right?"
Accuracy: % correct prediction of all predictions

Imagine the stupidest predictor:
"Always guess healthy"

%1 have leukemia, %99 are healthy

Classify using health records and tests

"How many times did I get it right?"
Accuracy: % correct prediction of all predictions

Imagine the stupidest predictor:
"Always guess healthy"

What will the accuracy be?

%1 have leukemia, %99 are healthy

Classify using health records and tests

"How many times did I get it right?"
Accuracy: % correct prediction of all predictions

Imagine the stupidest predictor:
"Always guess healthy"

What will the accuracy be?
It will be right 99% of the time!
You won't catch any sick people. Useless.

Confusion Matrix

	P' (Predicted)	n' (Predicted)
P (Actual)	True Positive	False Negative
n (Actual)	False Positive	True Negative

Confusion Matrix

	Spam (Predicted)	Non-Spam (Predicted)
Spam (Actual)	27	6
Non-Spam (Actual)	10	57

Recall (Sensitivity) = TP / (TP + FN)

Precision = TP / (TP+FP)

Specificity = TN / (TN + FP)

Accuracy = (TP + TN) / (TP + TN + FP + FN)

Confusion Matrix

	Spam (Predicted)	Non-Spam (Predicted)
Spam (Actual)	27	6
Non-Spam (Actual)	10	57

Recall (Sensitivity) = TP / (TP + FN) = .82

Precision = TP / (TP+FP) = .73

Specificity = TN / (TN + FP) = . 85

Accuracy = (TP + TN) / (TP + TN + FP + FN) = .84

Precision and recall

Precision and recall

Precision: Out of all cases I predicted as positive, how many times was I right?

Precision and recall

Precision: Out of all cases I <u>predicted as positive,</u>
how many times was I right?

(% times I was right when I told somebody they had leukemia)

Precision and recall

Precision: Out of all cases I predicted as positive,
how many times was I right?
(% times I was right when I told somebody they had leukemia)

Recall: Out of all the (few) positive cases,
how many did I find

Precision and recall

Precision: Out of all cases I <u>predicted as positive,</u>
how many times was I right?
(% times I was right when I told somebody they had leukemia)

Recall: Out of all the (few) positive cases,
how many did I find
(% of actual leukemia patients I could catch with my classifier)

Confusion Matrix

	Spam (Predicted)	Non-Spam (Predicted)
Spam (Actual)	0	10
Non-Spam (Actual)	0	990

Recall (Sensitivity) = TP / (TP + FN) = 0/10 = 0

Precision = TP / (TP+FP) = 0/0 → undefined!

Specificity = TN / (TN + FP) = 100%

Accuracy = (TP + TN) / (TP + TN + FP + FN) = 99%

Confusion Matrix

	P' (Predicted)	n' (Predicted)
P (Actual)	True Positive	False Negative
n (Actual)	False Positive	True Negative

Precision = TP / (TP + FP)

Recall = TP / (TP + FN)

Confusion Matrix

	P' (Predicted)	n' (Predicted)
P (Actual)	True Positive	False Negative
n (Actual)	False Positive	True Negative

Focusing on a single class (positive: the one with small prevalence) in skewed cases

Precision = TP / (TP + FP)

Recall = TP / (TP + FN)

Confusion Matrix

	P' (Predicted)	n' (Predicted)
P (Actual)	True Positive	False Negative
n (Actual)	False Positive	True Negative

Focusing on a single class (positive: the one with small prevalence) in skewed cases

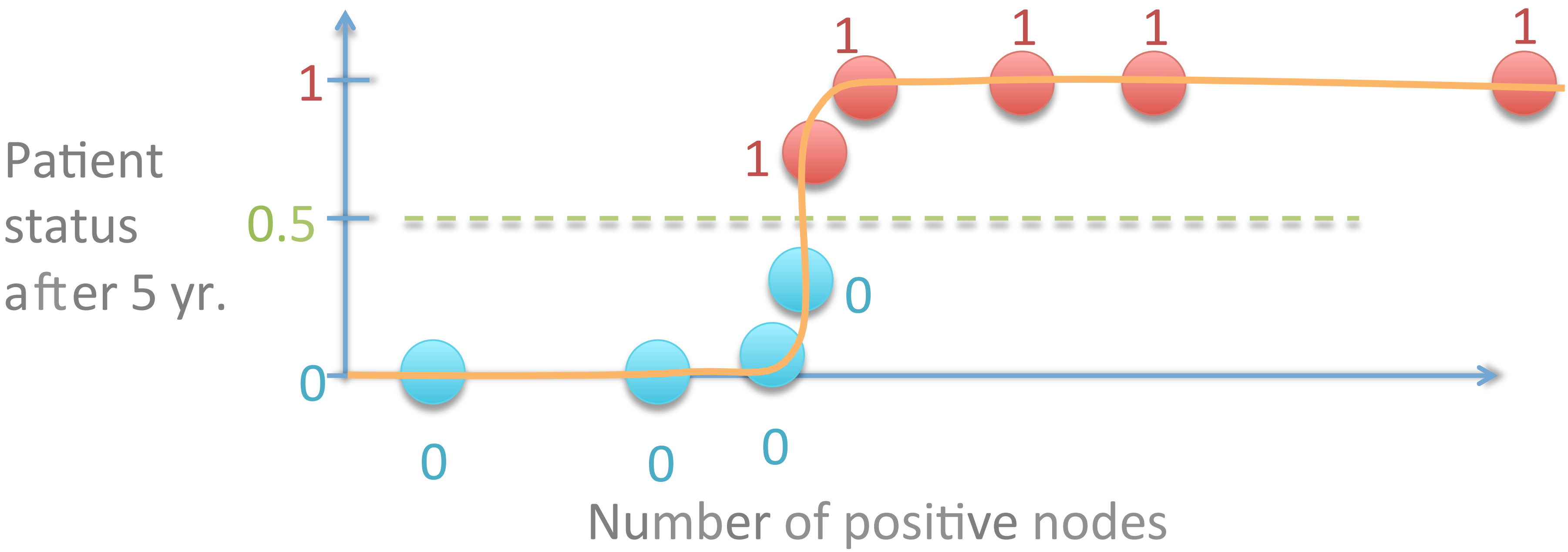
Precision = TP / (TP + FP)

Recall = TP / (TP + FN)

F1 = Their harmonic mean

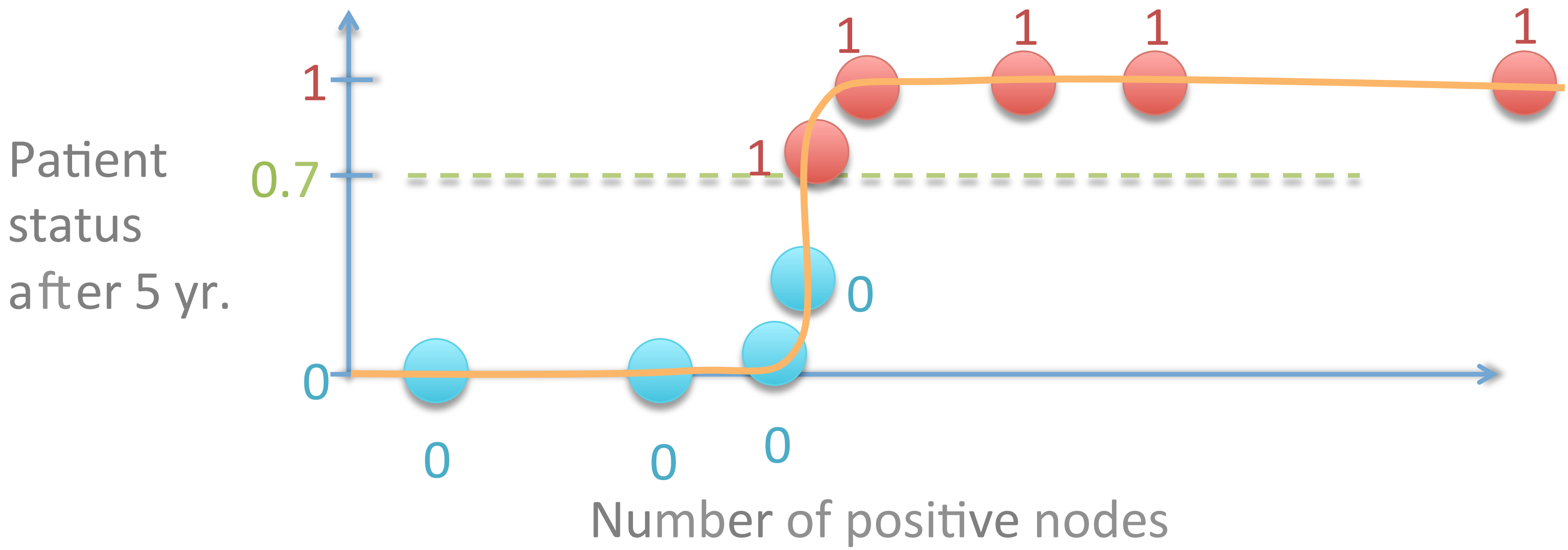
$$F1 = 2 * \frac{precision * recall}{precision + recall}$$

Logistic regression

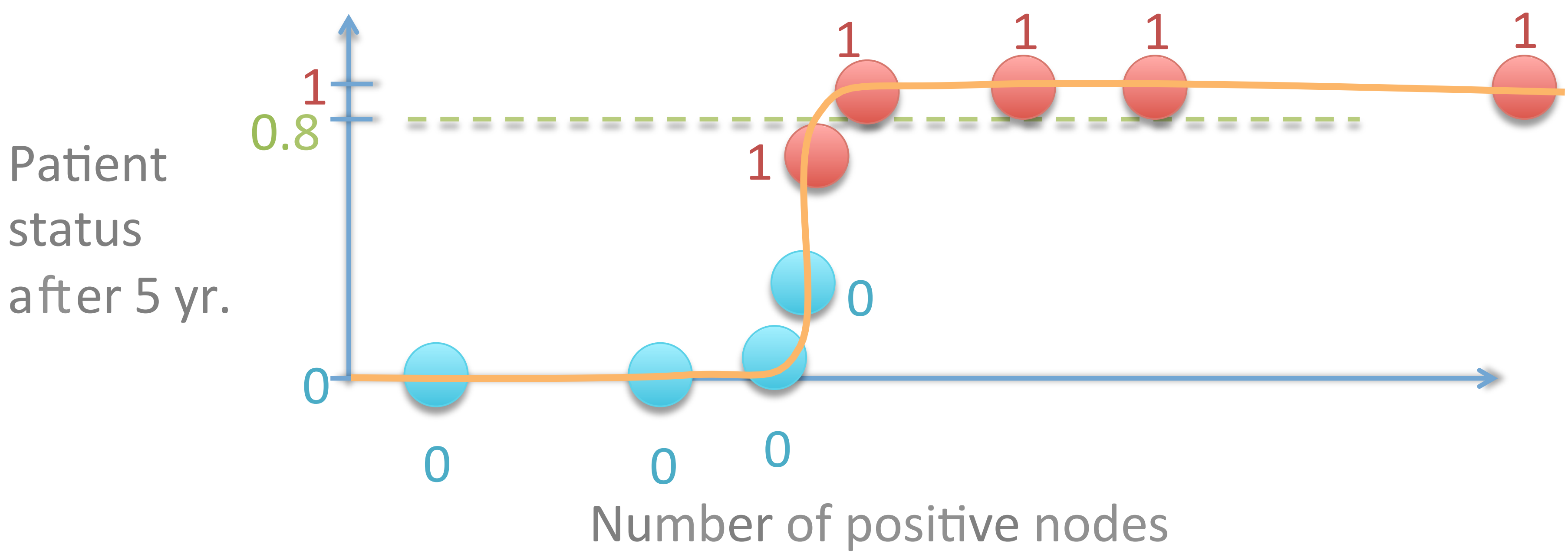


$$y_\beta(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x + \varepsilon)}}$$

Logistic regression



Logistic regression

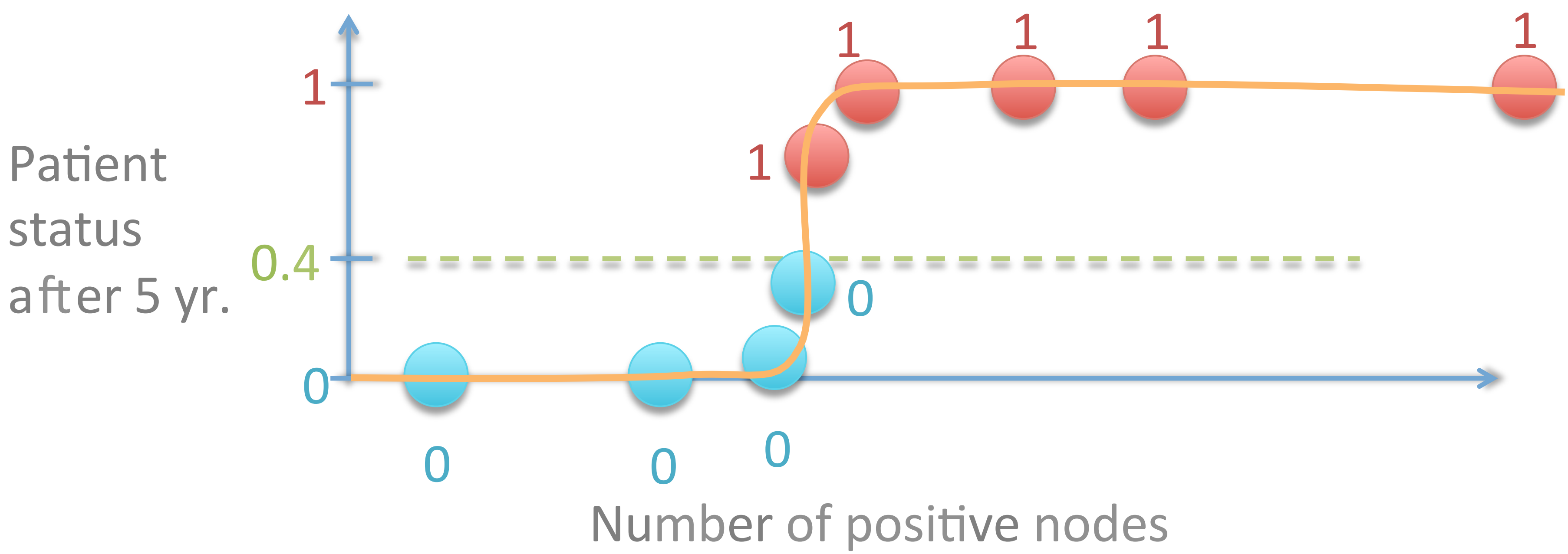


Higher threshold: More sure about positives

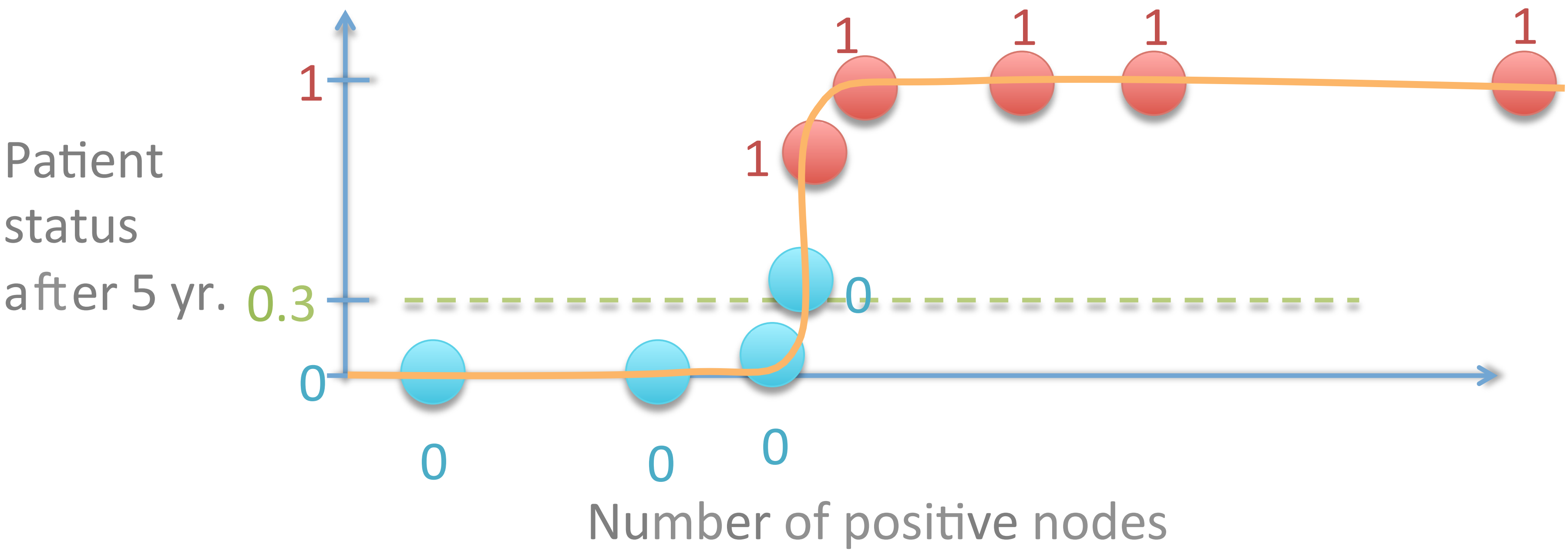
lower recall, higher precision

lower True Positive Rate, lower False Positive Rate

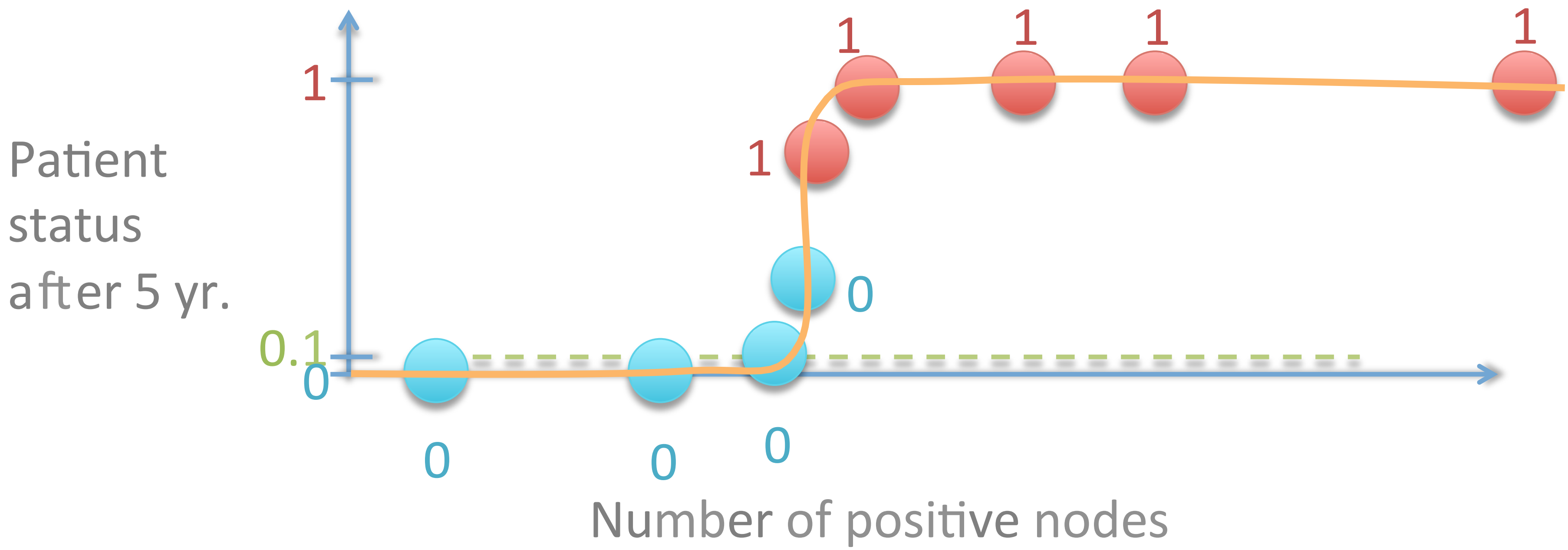
Logistic regression



Logistic regression



Logistic regression



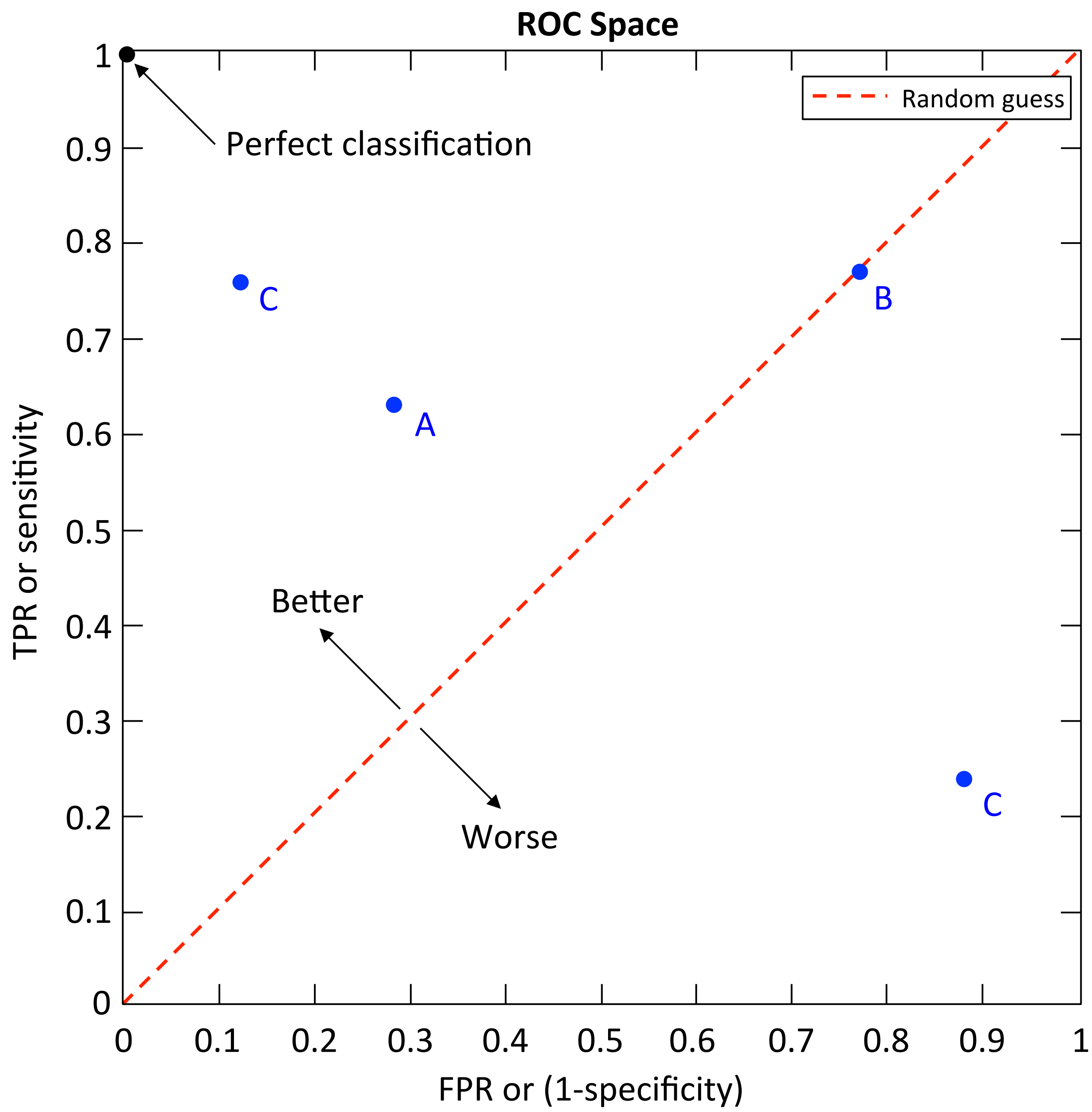
Lower threshold: Better at catching positives

higher recall, less precision

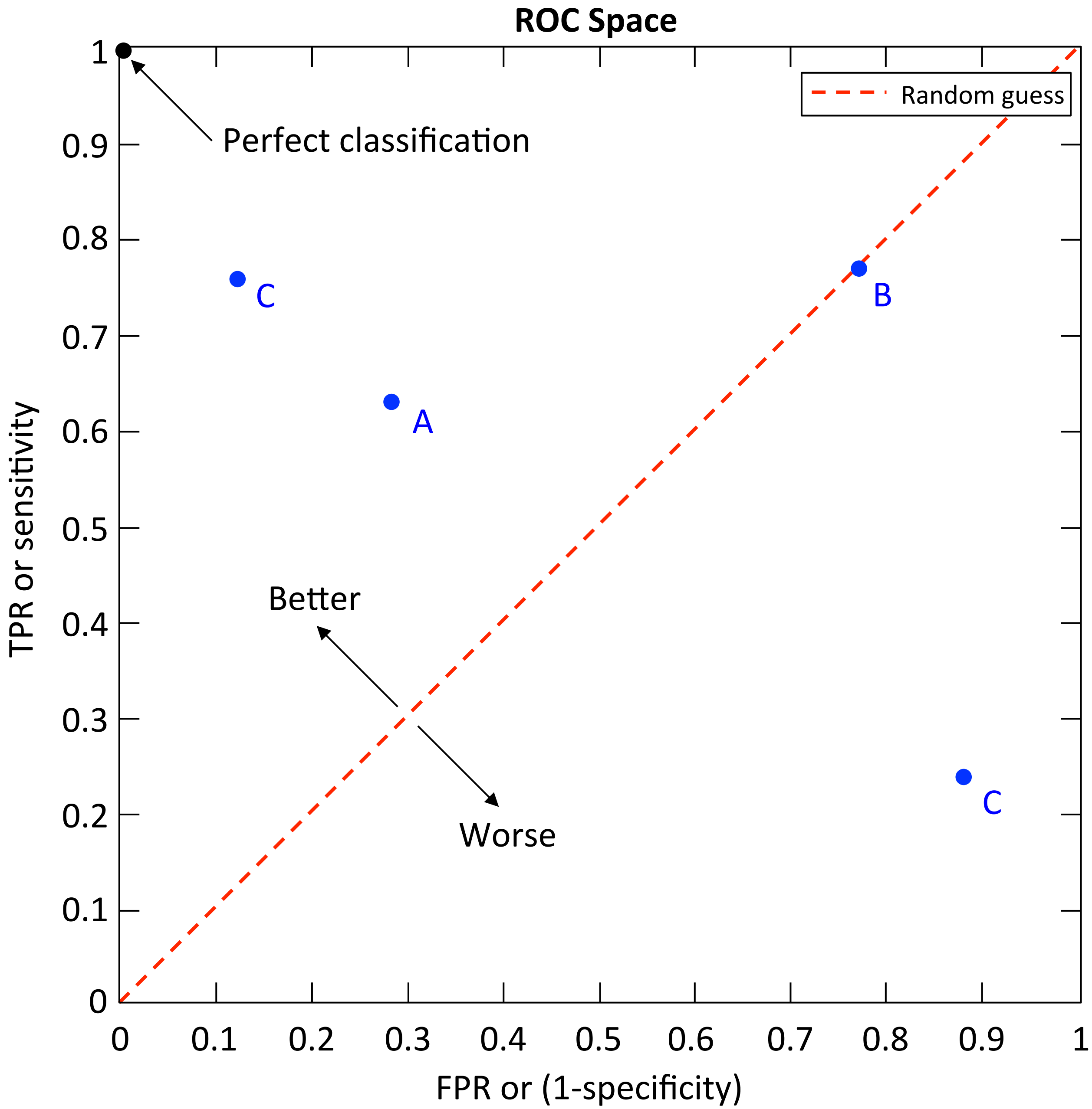
higher True Positive Rate, higher False Positive Rate

Each threshold is a different model

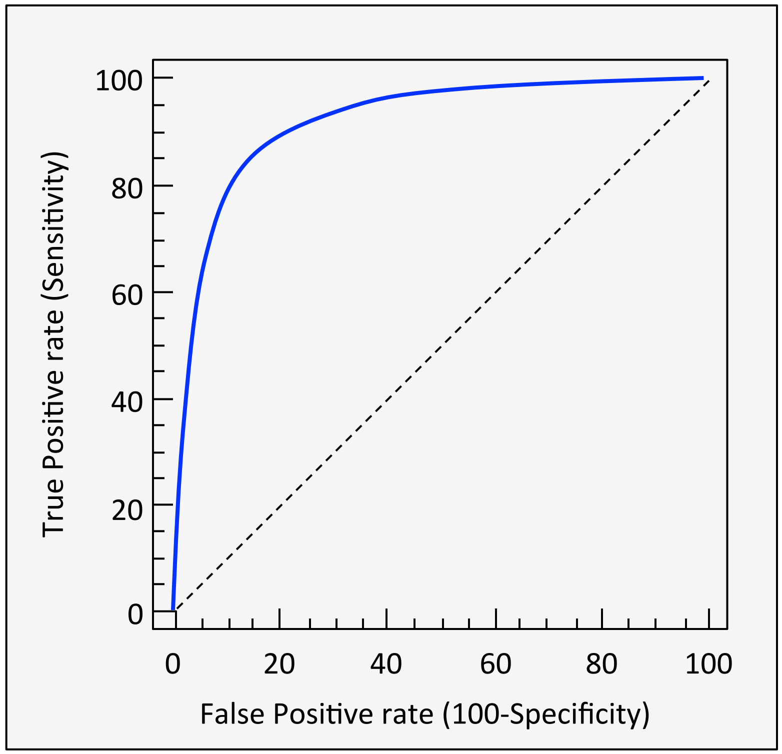
Plot their True Positive Rate & False Positive Rate



ROC Space
Random guess
Perfect classification
C
B
A
Better
Worse
C
TPR or sensitivity
FPR or (1-specificity)
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1



Receiver Operating Characteristic

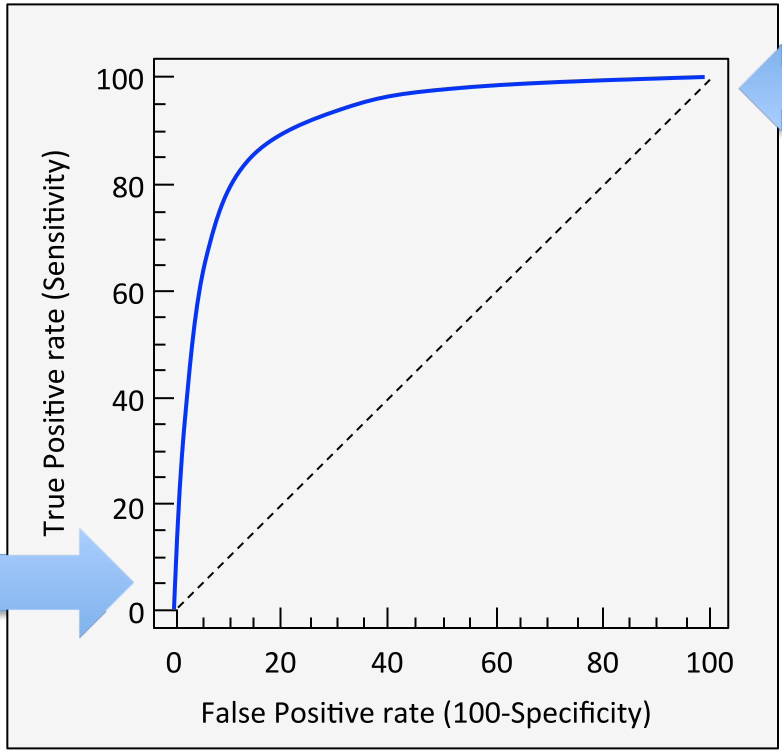


100
80
60
40
20
0
True Positive rate (Sensitivity)
0
20
40
60
80
100
False Positive rate (100-Specificity)

Lower threshold: Better at catching positives

higher recall, less precision

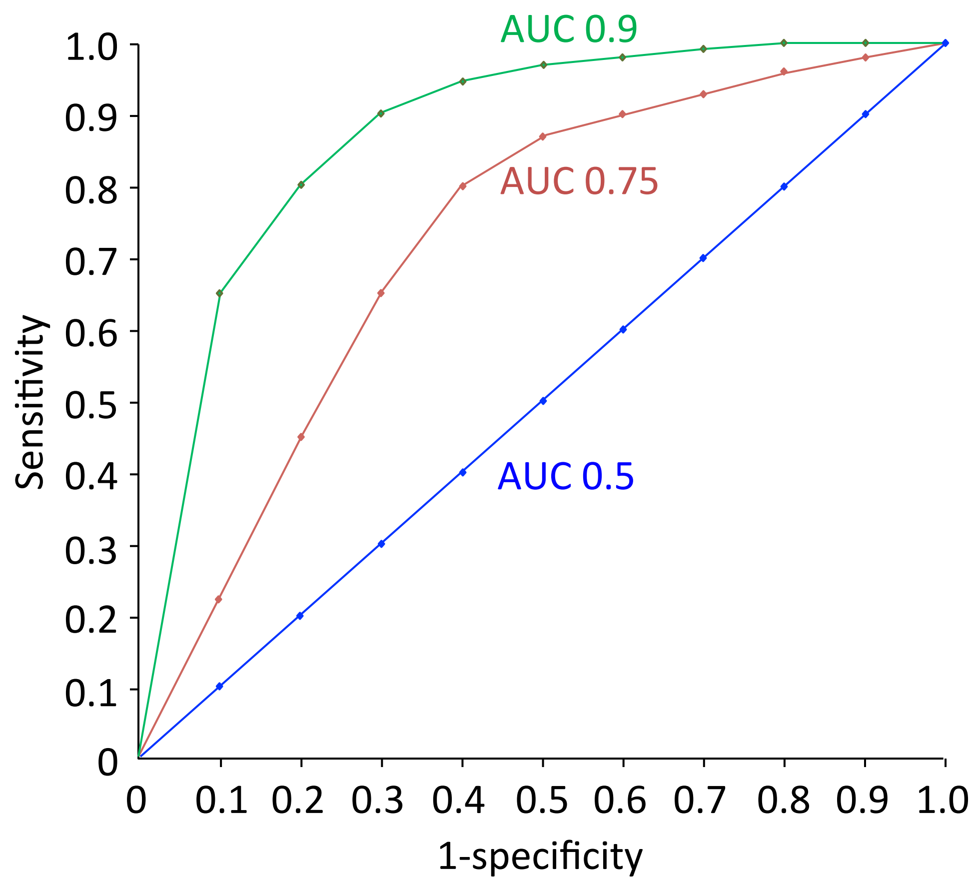
higher True Positive Rate, higher False Positive Rate



Higher threshold: More sure about positives

lower recall, higher precision

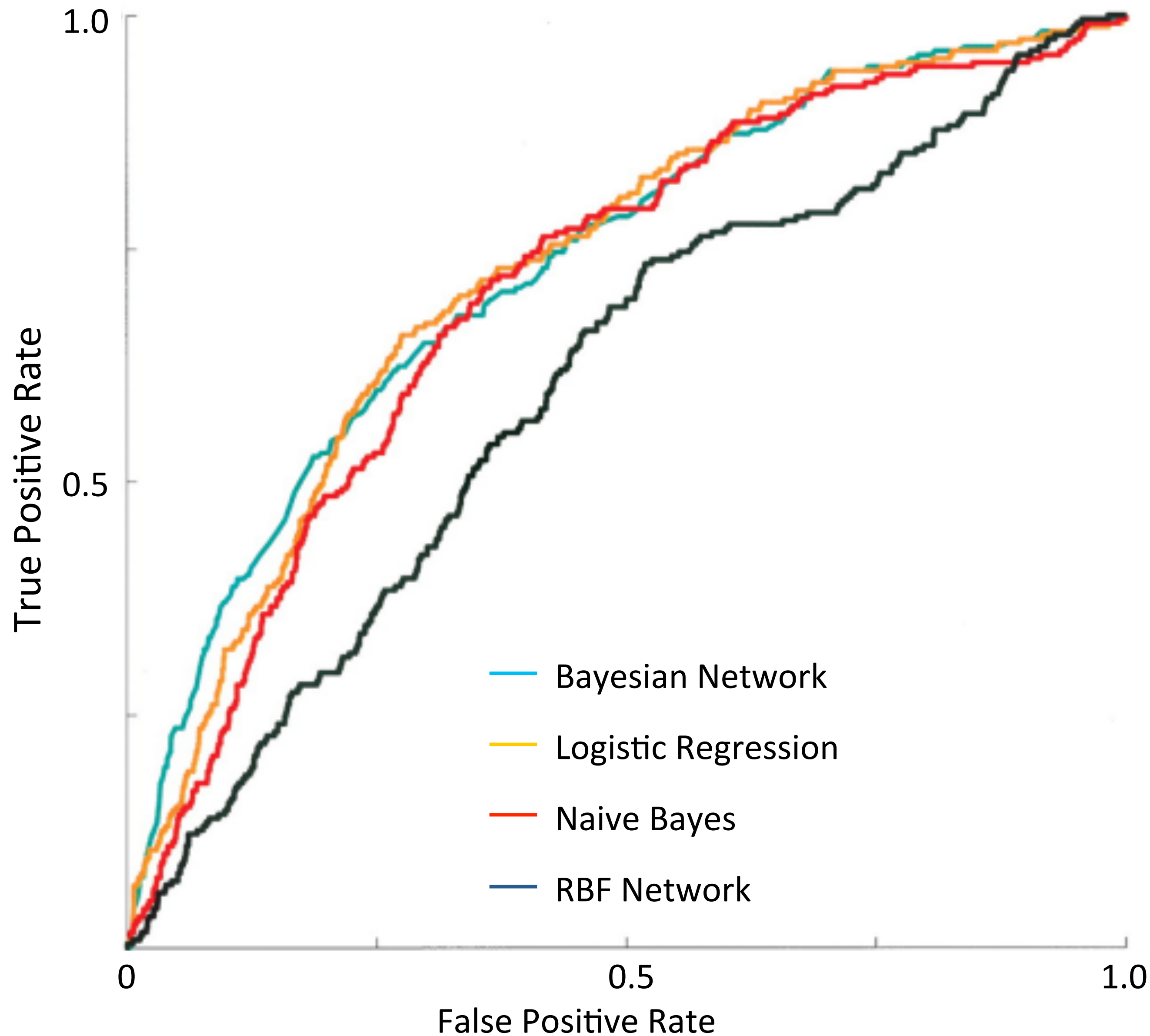
lower True Positive Rate, lower False Positive Rate



Area under curve (AUC)

An evaluation of a classification algorithm

(including all possible thresholds)



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

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.fl_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
metrics.precision_recall_fscore_support(...)	Compute precision, recall, F-measure and support for each class
metrics.precision_score(y_true, y_pred[, ...])	Compute the precision
metrics.recall_score(y_true, y_pred[, ...])	Compute the recall
metrics.roc_auc_score(y_true, y_score[, ...])	Compute Area Under the Curve (AUC) from prediction scores
metrics.roc_curve(y_true, y_score[, ...])	Compute Receiver operating characteristic (ROC)
metrics.zero_one_ioss(y_true, y_pred[, ...])	Zero-one classification loss.

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