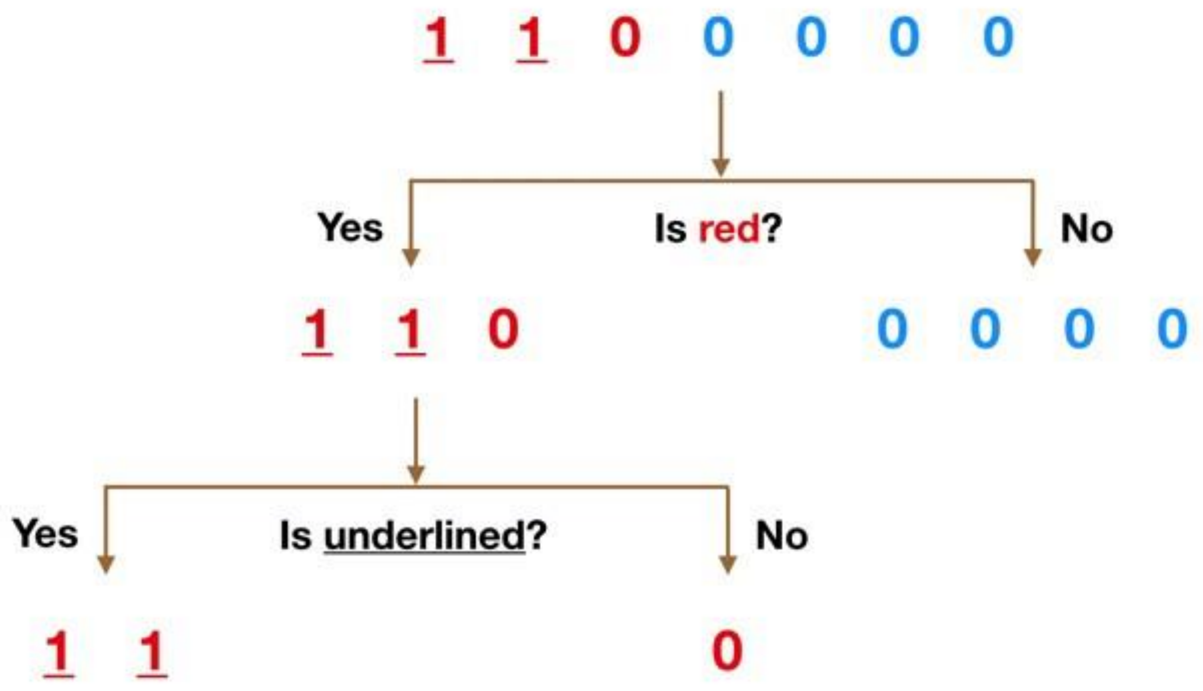
Supervised Machine Learning

Random Forest

Instructor, Nero Chan Zhen Yu

Forward School

Decision Tree



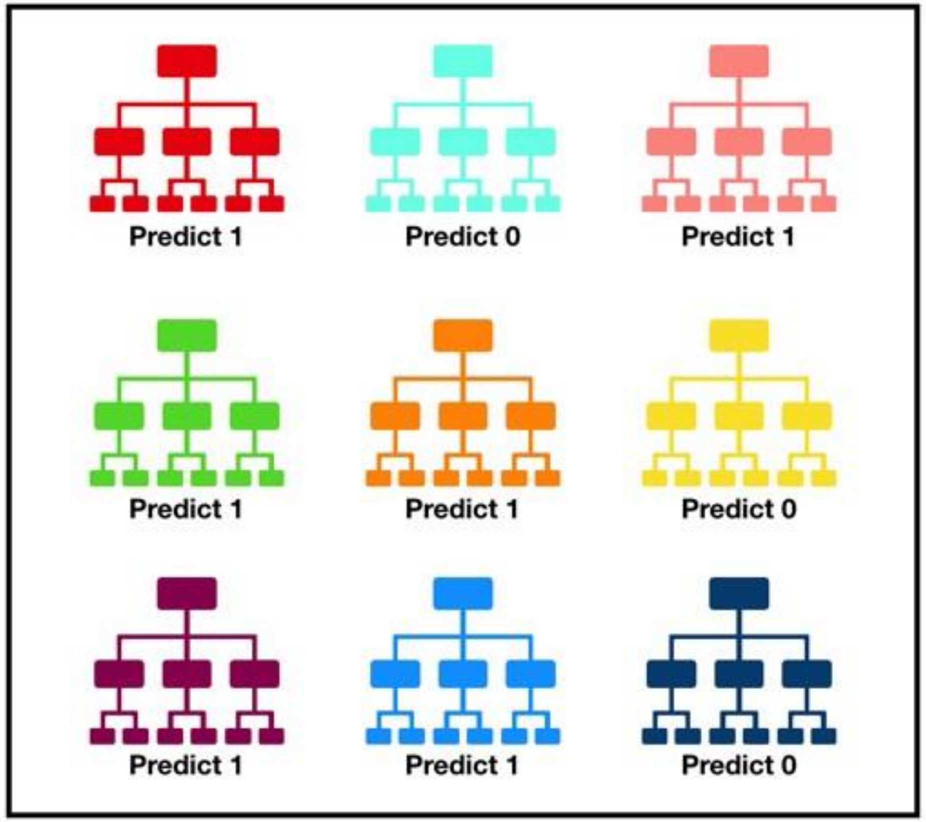
Imagine that our dataset consists of the numbers at the top of the figure to the left.

We have two 1s and five 0s (1s and 0s are our classes) and desire to separate the classes using their features.

The features are color (red vs. blue) and whether the observation is underlined or not. So how can we do this?

Forward School

Random Forest



Tally: Six 1s and Three 0s
Prediction: 1

Random forest, like its name implies, consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction

A large number of relatively uncorrelated models (trees) operating as a committee will outperform any of the individual constituent models.

Forward School

Ensemble Method

- Decision Tree Problem – Decision Trees tend to overfit.

- A group of predictors is called an ensemble; thus, this technique is called Ensemble Learning.
- Train a group of Decision Tree Classifiers, each on a different random subset of the training set.
- To make predictions, obtain the predictions of all individual trees, then predict the class that gets the most votes.
- Such an ensemble of a Decision Trees is called Random Forest.

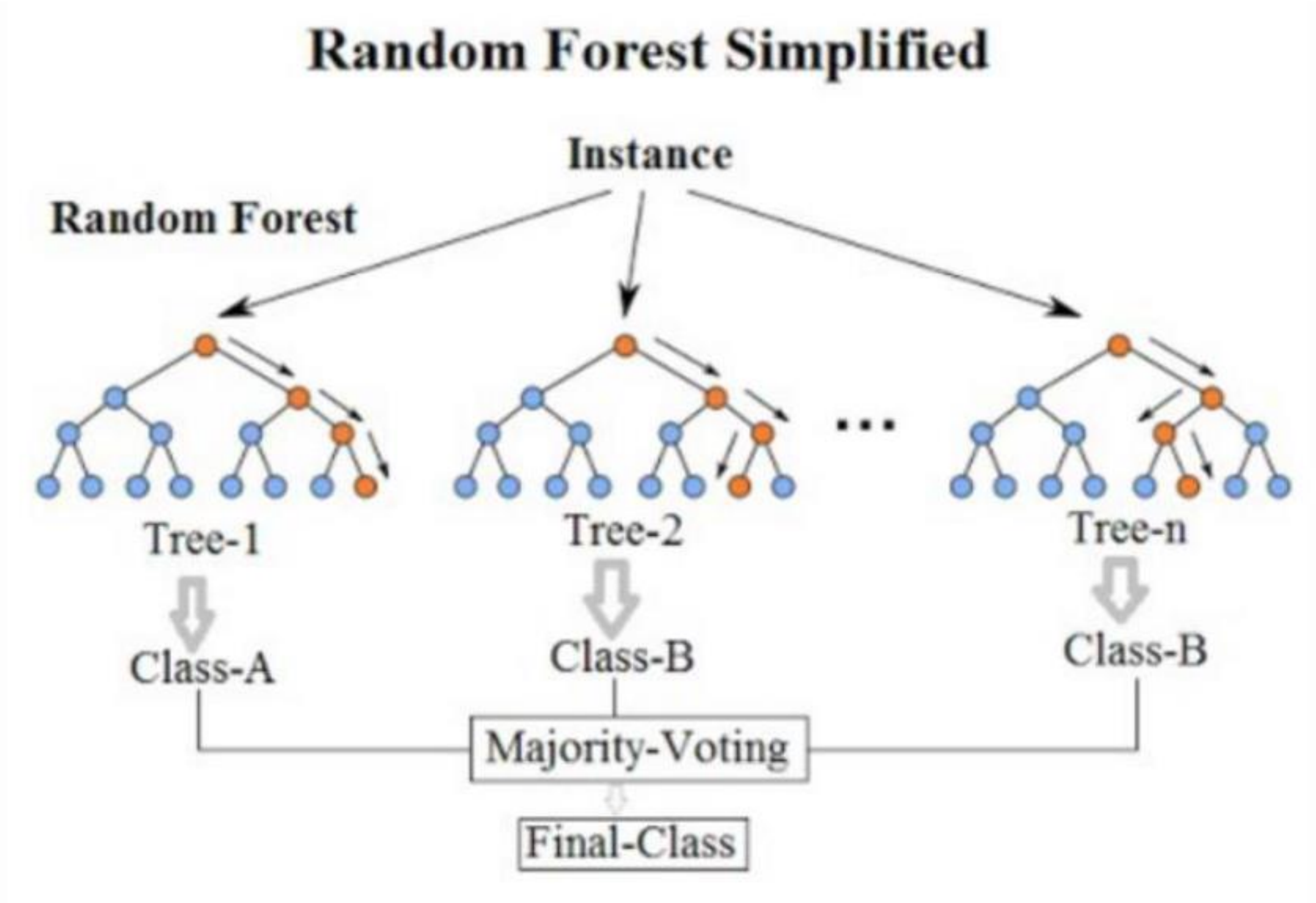
Ensemble Method

- Hard voting classifier - aggregate all predictions and predict the class that gets the most votes.
- Soft voting classifier – Each individual *classifier* provides a probability value that a specific data point belongs to a particular target class. The predictions are weighted by the classifier's importance and summed up. Then the target label with the greatest sum of weighted probabilities wins the vote.

Random Forest

- The reason the decision tree is prone to overfitting when we don’t limit the maximum depth.
- Limit the depth of the tree- reduces variance (good) and increases bias (bad).
- Thus: to combine many decision trees into a single ensemble model - the random forest.

Random Forest



Forward School

An Example of Why Uncorrelated Outcomes are So Great

I use a uniformly distributed random number generator to produce a number.

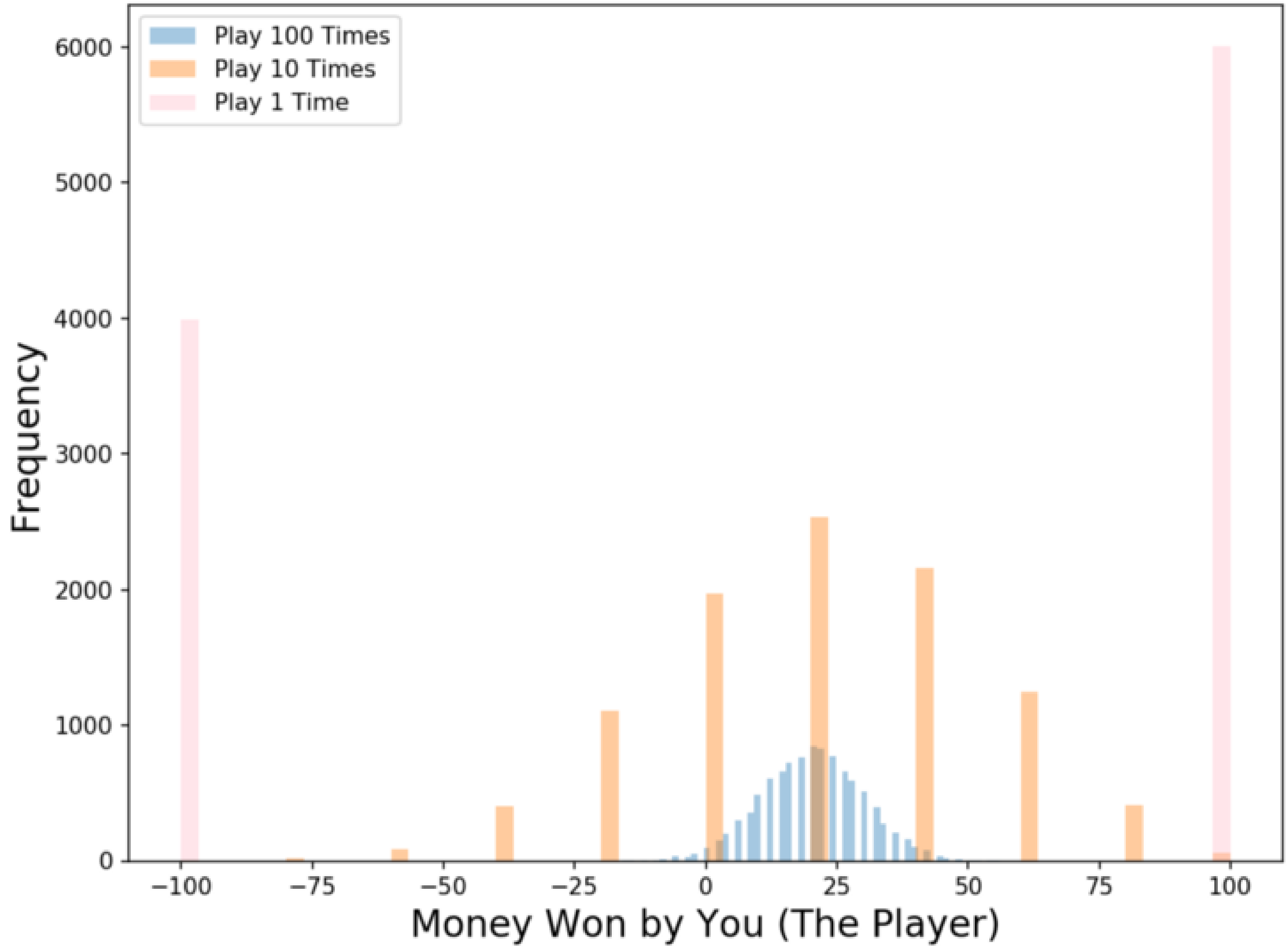
If the number I generate is greater than or equal to 40, you win (so you have a 60% chance of victory) and I pay you some money. If it is below 40, I win and you pay me the same amount.

Now I offer you the the following choices. We can either:

1. **Game 1** — play 100 times, betting RM1 each time.
2. **Game 2**— play 10 times, betting RM10 each time.
3. **Game 3**— play one time, betting RM100.

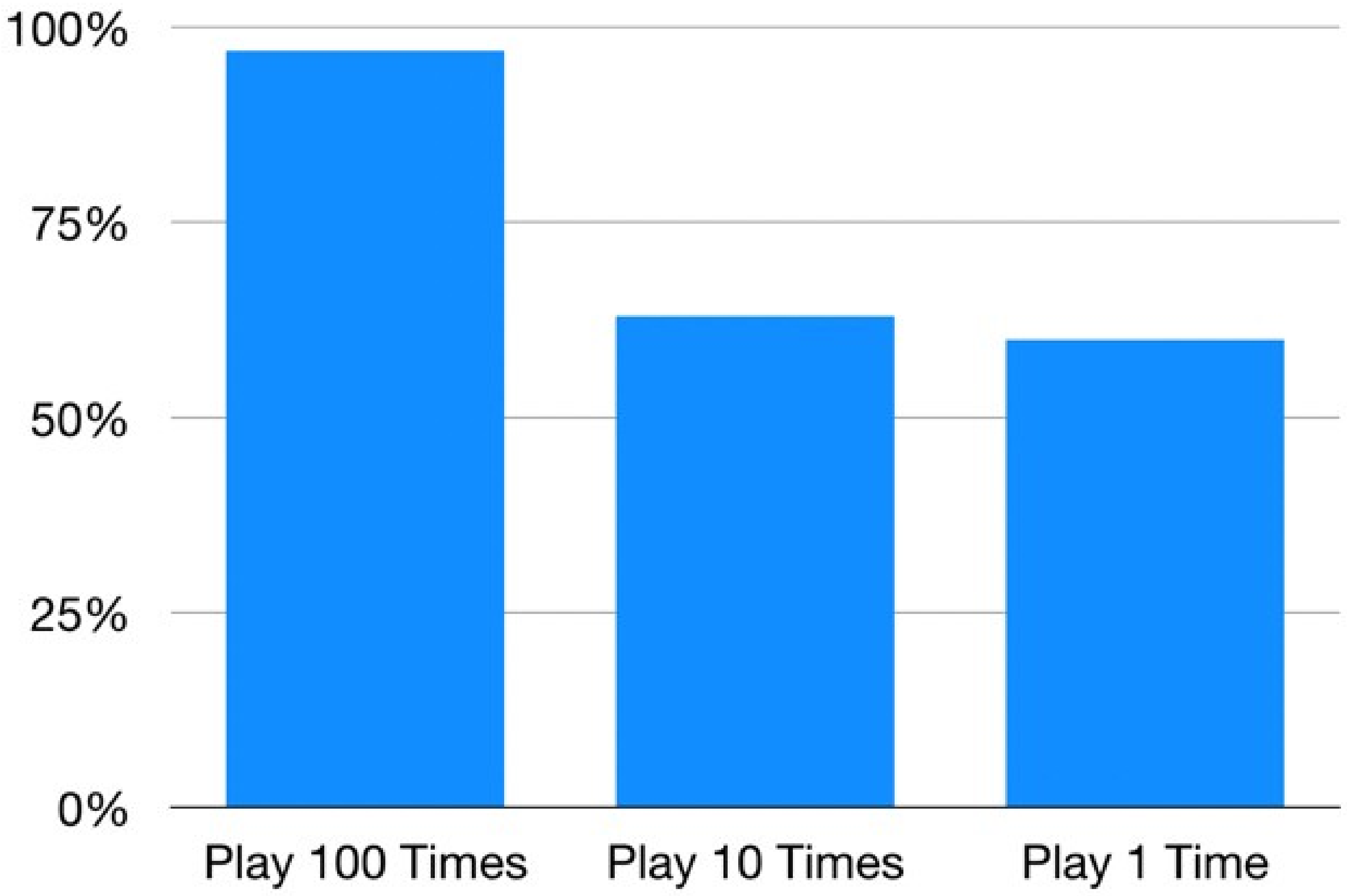
Which would you pick?

Outcome Visualization



Forward School

Probability of making money in each game



Forward School

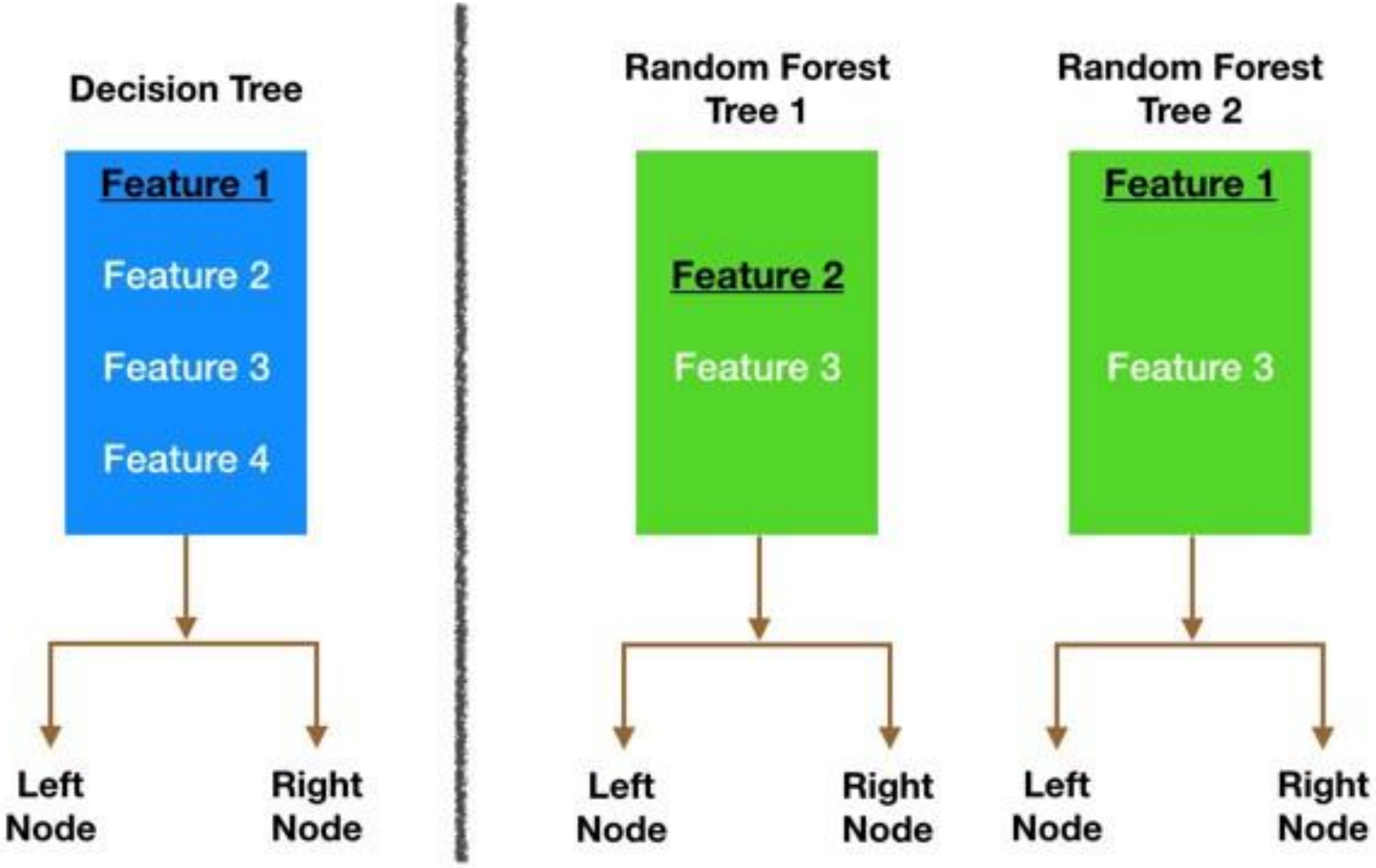
Ensuring that the Models Diversify Each Other

- **Bagging (Bootstrap Aggregation)**

Decisions trees are very sensitive to the data they are trained on — small changes to the training set can result in significantly different tree structures.

- **Feature Randomness**

Ensuring that the Models Diversify Each Other



Forward School

Random Forest Summary

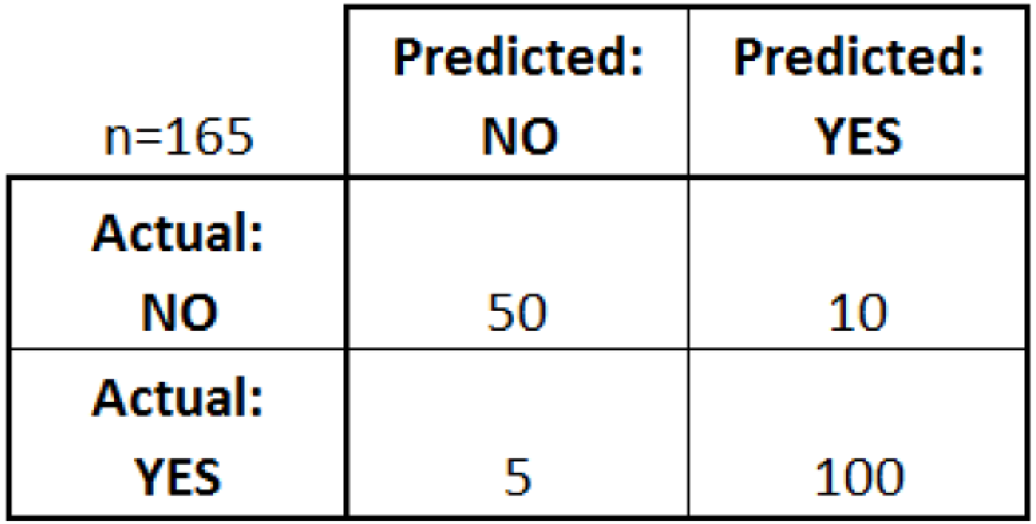
The random forest is a classification algorithm consisting of many decisions trees.

It uses bagging and feature randomness when building each individual tree to try to create an uncorrelated forest of trees *whose prediction by committee is more accurate than that of any individual tree.*

We need features that have at least some predictive power and the trees of the forest and more importantly their predictions need to be uncorrelated

Confusion Matrix

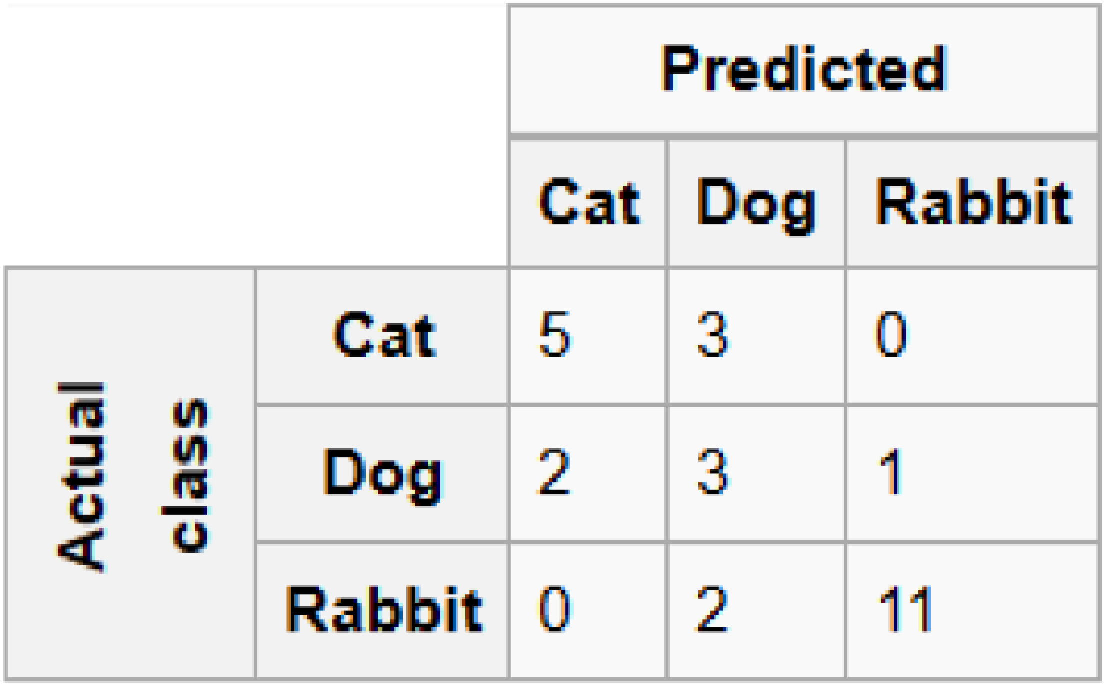
- A confusion matrix is a table that is often used to **describe the performance of a classification model** (or "classifier") on a set of test data for which the true values are known.



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

Confusion Matrix

- Another example of Confusion Matrix



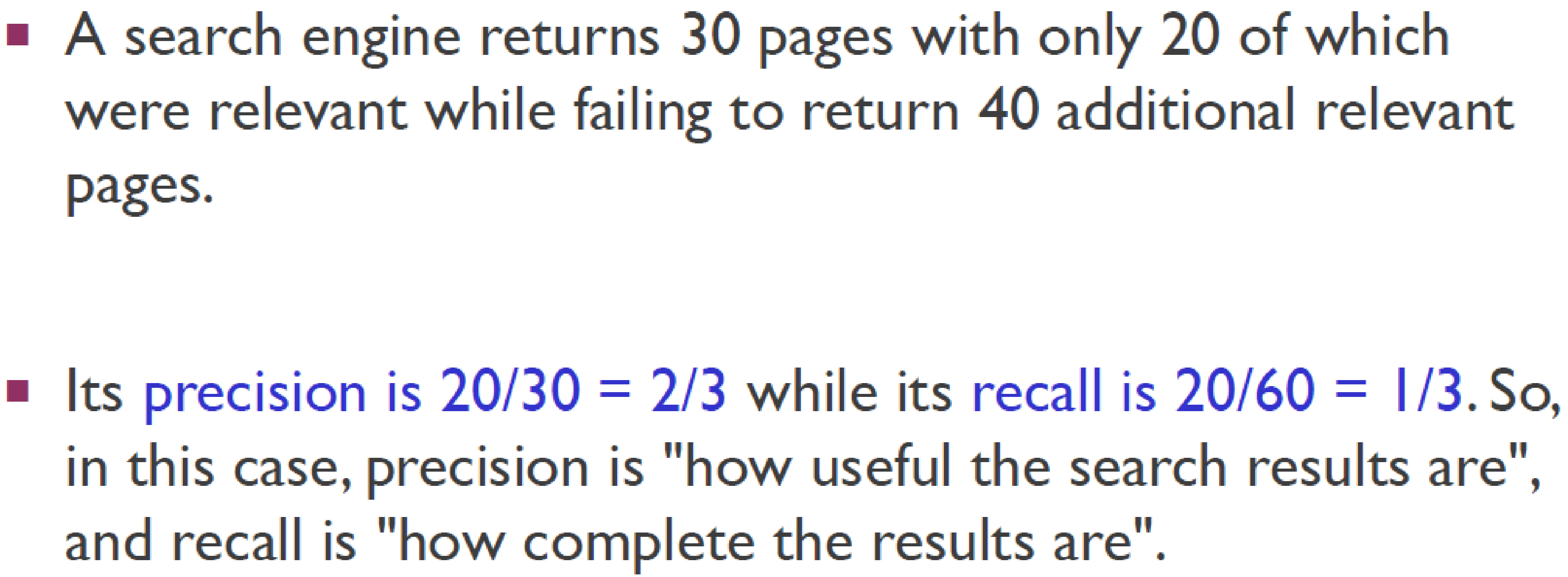
		Predicted		
		Cat	Dog	Rabbit
Actual class	Cat	5	3	0
	Dog	2	3	1
	Rabbit	0	2	11

Accuracy, Precision and Recall

- Suppose a computer program for recognizing dogs in scenes from a video identifies 7 dogs in a scene containing 9 dogs and some cats.

- If 4 of the identifications are correct, but 3 are actually cats, the program's precision is 4/7 while its recall is 4/9.

Accuracy, Precision and Recall



- A search engine returns 30 pages with only 20 of which were relevant while failing to return 40 additional relevant pages.

- Its precision is 20/30 = 2/3 while its recall is 20/60 = 1/3. So, in this case, precision is "how useful the search results are", and recall is "how complete the results are".

Accuracy, Precision and Recall

- Let's now define the most basic terms, which are whole numbers (not rates):

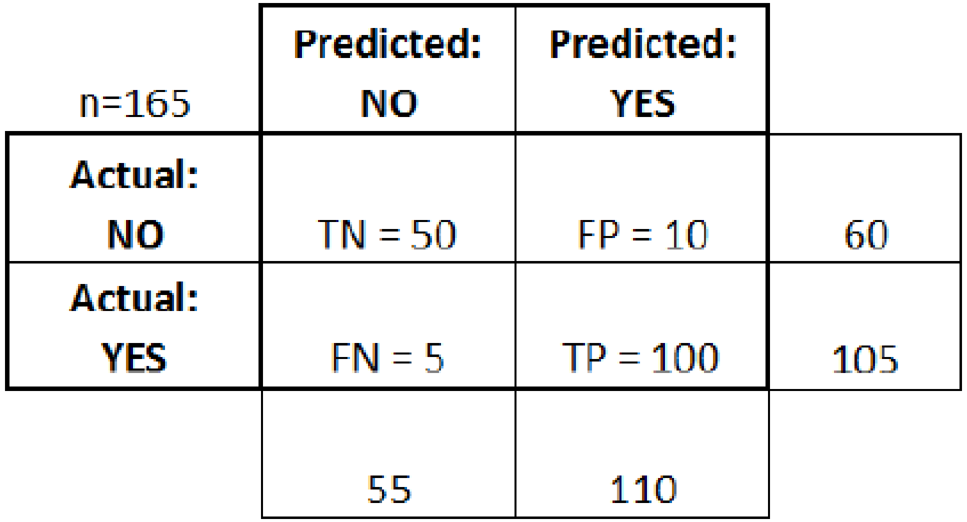
- **true positives (TP):** These are cases in which we predicted yes (they have the disease), and they do have the disease.
- **true negatives (TN):** We predicted no, and they don't have the disease.
- **false positives (FP):** We predicted yes, but they don't actually have the disease. (Also known as a "Type I error.")
- **false negatives (FN):** We predicted no, but they actually do have the disease. (Also known as a "Type II error.")

Accuracy, Precision and Recall

- Accuracy: Overall, how often is the classifier correct?
- (TP+TN)/total = (100+50)/165 = 0.91

- Precision: When it predicts yes, how often is it correc
- TP/predicted yes = 100/110 = 0.91

- Recall: When it's actually yes, how often does it predict yes?
- TP/actual yes = 100/105 = 0.95



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

ROC Curves and Area under the Curve

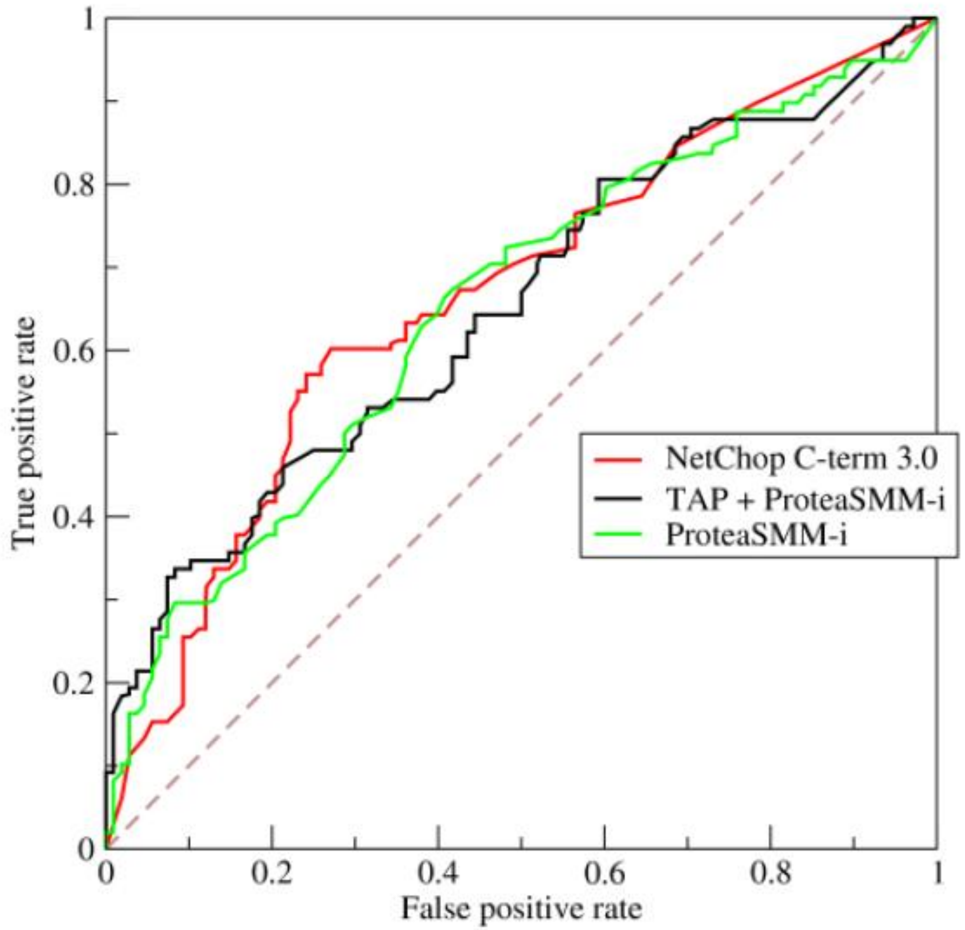
- ROC (Receiver Operating Characteristic) is the most commonly used way to **visualize the performance of a binary classifier**.

- Area Under the Curve (AUC) is (arguably) the best way to summarize its performance in a single number.

ROC Curves and Area under the Curve

TPR=TP/(TP+FN)
FPR=FP/(FP+TN)

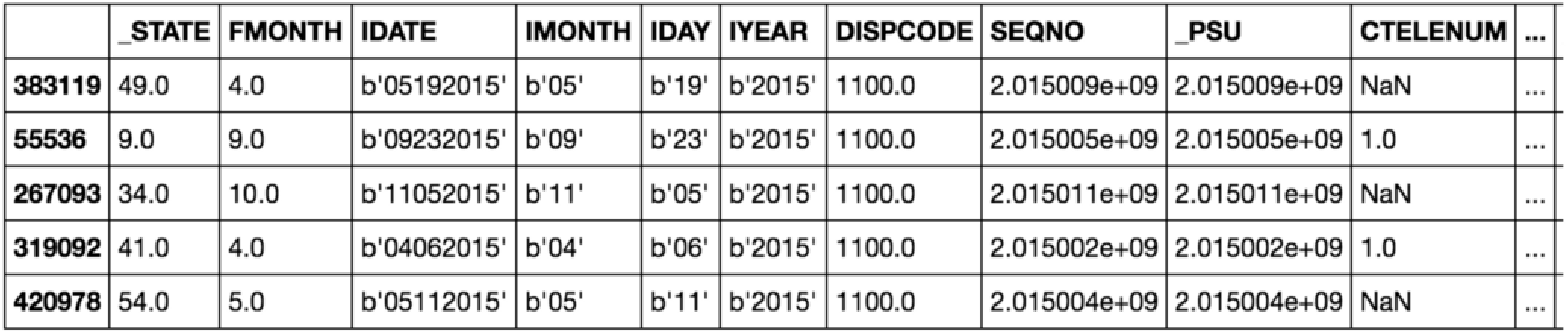
- Example of ROC Curve



Forward School

Random Forest Example

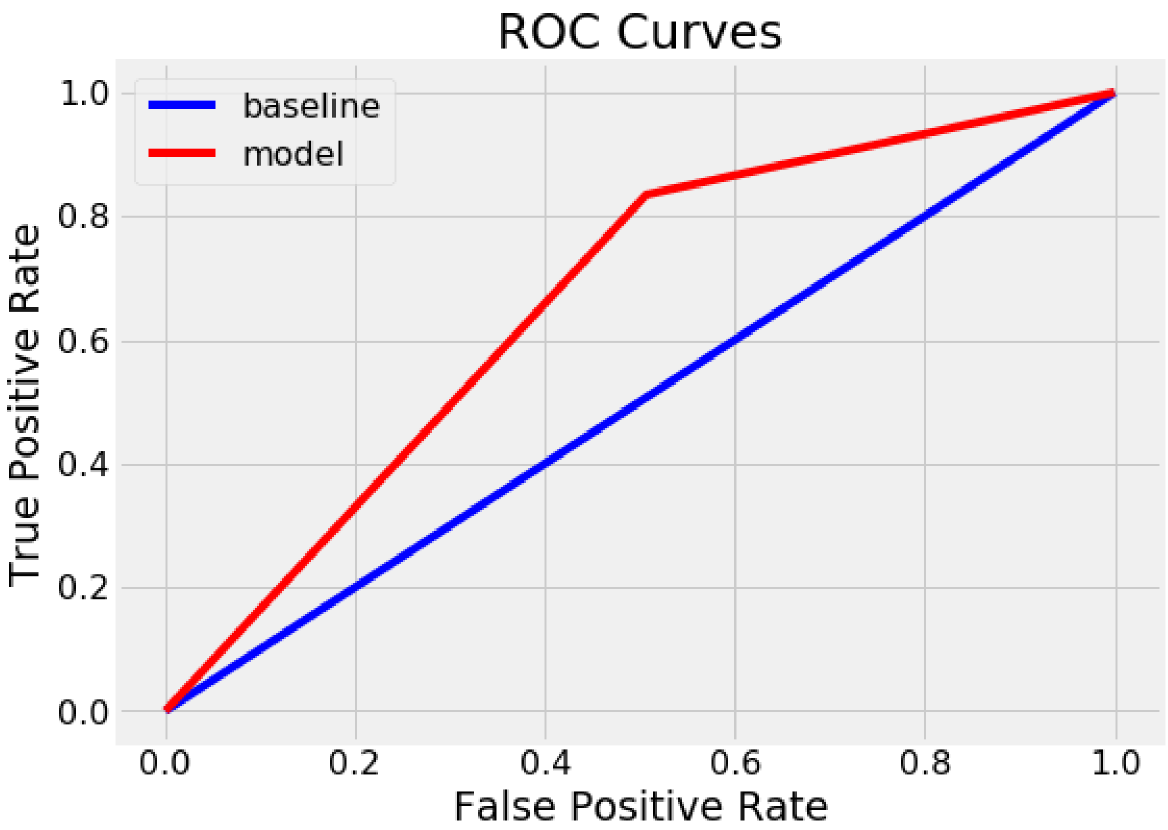
The problem we'll solve is a binary classification task with the goal of predicting an individual's health. The features are socioeconomic and lifestyle characteristics of individuals and the label is "0" for poor health and "1" for good health. This dataset was collected by the Centers for Diseases Control and Prevention.

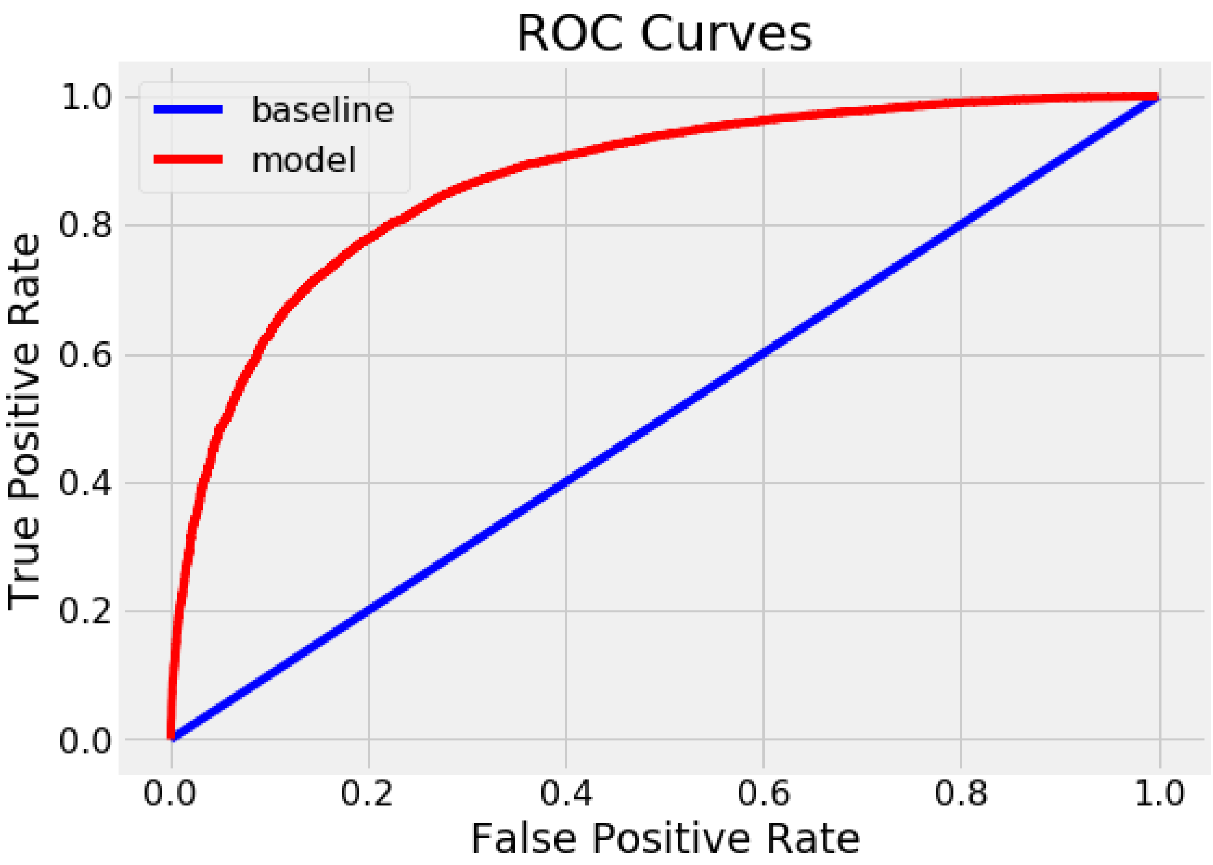


	_STATE	FMONTH	IDATE	IMONTH	IDAY	IYEAR	DISPCODE	SEQNO	_PSU	CTELENUM	...
383119	49.0	4.0	b'05192015'	b'05'	b'19'	b'2015'	1100.0	2.015009e+09	2.015009e+09	NaN	...
55536	9.0	9.0	b'09232015'	b'09'	b'23'	b'2015'	1100.0	2.015005e+09	2.015005e+09	1.0	...
267093	34.0	10.0	b'11052015'	b'11'	b'05'	b'2015'	1100.0	2.015011e+09	2.015011e+09	NaN	...
319092	41.0	4.0	b'04062015'	b'04'	b'06'	b'2015'	1100.0	2.015002e+09	2.015002e+09	1.0	...
420978	54.0	5.0	b'05112015'	b'05'	b'11'	b'2015'	1100.0	2.015004e+09	2.015004e+09	NaN	...

Forward School

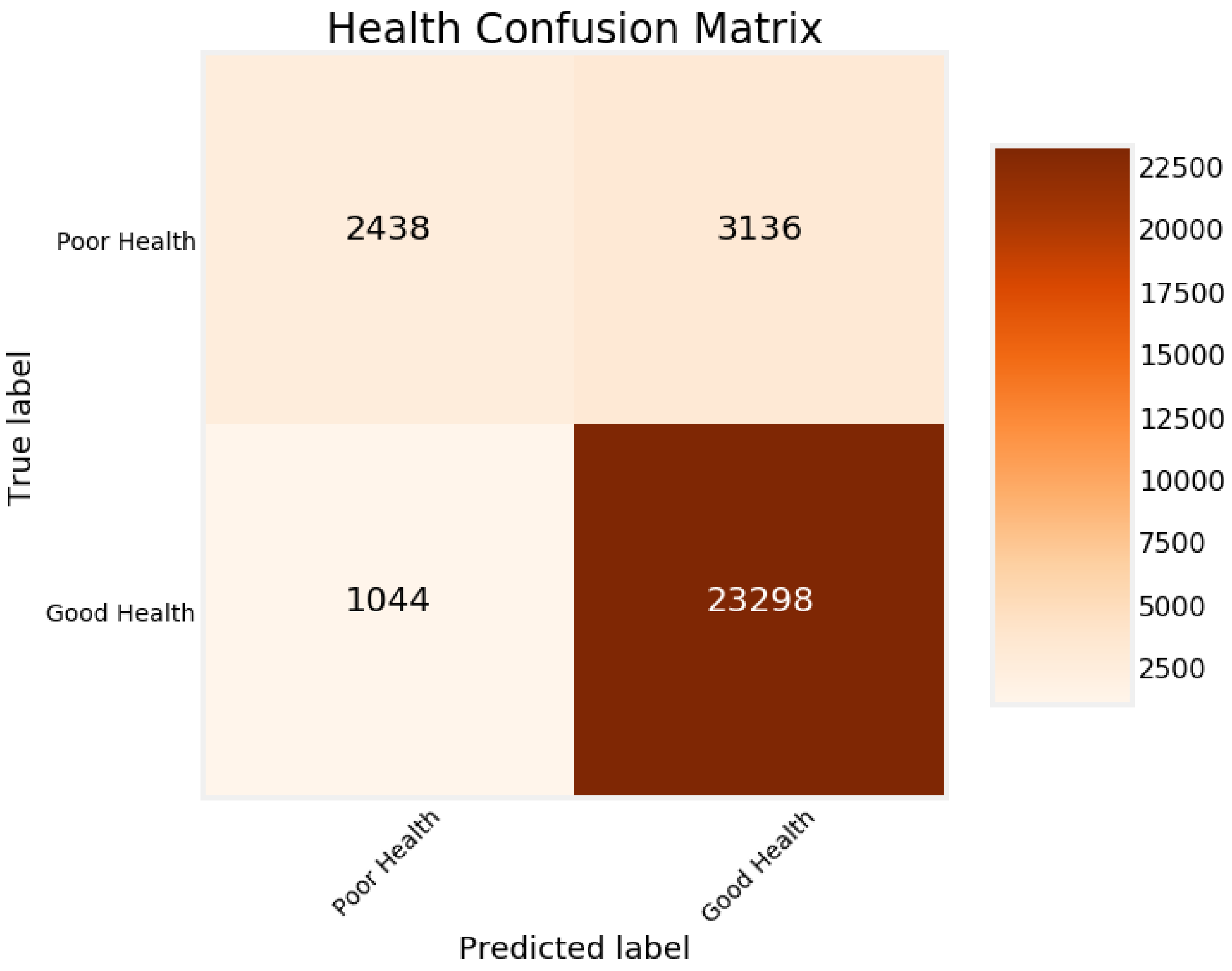
Random Forest Example Results





Forward School

Random Forest Example Results



Feature Importances

- The feature importances in a random forest indicate the sum of the reduction in Gini impurity over all the nodes that are split on that feature. We can use these to try and figure out what predictor variables the random forest considers most important.

- Feature importances can be used for feature engineering by building additional features from the most important.