

DATA SCIENCE

CLASS 8: NAÏVE BAYES, DECISION TREES, AND CLASSIFICATION MODEL EVALUATION

- I. BAYES' THEOREM AND NAÏVE BAYES CLASSIFICATION**
- II. CLASSIFICATION VIA DECISION TREES**
- III. CLASSIFICATION MODEL EVALUATION**

I. PROBABILITY AND BAYES' THEROM

- Naïve Bayes is a popular **simple classification technique** used in many applications, most notably textual analysis and medical diagnosis.
- Why do people use it?
 - With proper preprocessing of data, it is **competitive with more sophisticated classification techniques**.
 - It is **easy to debug and understand**.
 - It is robust to highly imbalanced datasets.
 - It is robust to highly dimensional data without having to transform it.
 - It is easily scaleable across large amounts of data.
- Naïve Bayes derives its form from Bayes' theorem: $P(A|B) = P(B|A) * P(A) / P(B)$
 - How many of you are familiar with Bayes' theorem?

- Recall Bayes' theorem: $P(A|B) = P(B|A) * P(A) / P(B)$
- Bayes' theorem lets you calculate the probability of an event (A) happening given some sort of signal (B). The catch here is that you can incorporate the reliability of that signal to understand whether or not it is helpful in determining whether an event is likely to occur.
- Let's use a handy example. Say you are heading to Seattle. The weather forecaster says it is likely to rain tomorrow. However, he is only 80% correct on average. Let's also say that it rains in Seattle 60% of the time, and the weatherman reports that it's raining roughly 75% of the time. What's the probability that it will actually rain?
- The way to think about this is that you want to incorporate the positive signal (the weather forecaster's prediction) to your overall probability. So, even if there's a 60% chance on average of rain, the forecaster's report changes your overall understanding of the actual probability of the event tomorrow.

- Again, recall Bayes' theorem: $P(A|B) = P(B|A) * P(A) / P(B)$
- Let's compute our probabilities.
 - Remember, we are looking to understand the probability that it will rain (A) given that the weatherman is predicting rain (B).
 - The first thing we need to do is understand the probability of the weatherman reporting rain (B) when it actually rains (A), or $P(B|A)$.
 - Remember, the weatherman is right 80% of the time. So, this is .8.
 - Next, we need to calculate the probability that it rains (A). This is 0.6.
 - Last, we need to calculate the probability that the weatherman reports that it rains $P(B)$. In this example, I've given this is 65%.
- So, $P(A|B) = .8 * .6 / .65 = .73$
- So, essentially, the weatherman's report of rain bumped up your belief in rain by roughly 13%.

- Oftentimes, you do not know outright how often the signal occurs $P(B)$. In these cases, you'll need to calculate it yourself.
 - Recall again our example of the weatherman. You can think of the probability of the weatherman reporting rain as the union of when he's correctly reporting rain, and when he's misreporting rain.
 - In other words, you're looking at the probability of a true positive $P(B|A)$ times the likelihood of the event occurring $P(A)$ plus the probability of a false positive $P(B|A^2)$ times the likelihood of the alternate event occurring $P(A^2)$.
 - So, in this example, we know the likelihood of rain (75%), the likelihood of it not raining (25%), and that he's 80% correct on average. So, there's a certain probability that he'll misreport that it will rains, and we need to take that into account.
- So, the probability that the weatherman is reporting that it's raining is:
 - $P(B) = (\text{probability he's reporting it's raining, when it's raining}) + (\text{probability he's reporting its raining, when it's not raining})$
 - $P(B) = (.8 * .75) + (.2 * .25) = .65.$

- Naïve Bayes is just an extension of Bayes' theorem.
- Suppose we have a dataset with features x_1, \dots, x_n and a class label C . What can we say about classification using Bayes' theorem?

$$P(\text{class } C \mid \{x_i\}) = \frac{P(\{x_i\} \mid \text{class } C) \cdot P(\text{class } C)}{P(\{x_i\})}$$

- Bayes' theorem can help us to determine the probability of a record belonging to a class, *given* the data we observe.

WHAT DOES THE NAÏVE BAYES FORMULA MEAN?

9


- Let's break down the Naïve Bayes formula piece by piece.
- This term is the likelihood function. It represents the probability of observing features $\{x_i\}$ given that that record belongs to class C.



$$P(\text{class } C \mid \{x_i\}) = \frac{P(\{x_i\} \mid \text{class } C) \cdot P(\text{class } C)}{P(\{x_i\})}$$

- You can calculate it by figuring out the percent of the class C you observe that has the features you are interested in.

- This term is the prior probability of C, or your '**prior**'. It represents the probability of a record belonging to class C before the new data is taken into account.


$$P(\text{class } C \mid \{x_i\}) = \frac{P(\{x_i\} \mid \text{class } C) \cdot P(\text{class } C)}{P(\{x_i\})}$$


- You can calculate it by figuring out the percent of all the data that has class C.

$$P(\text{class } C \mid \{x_i\}) = \frac{P(\{x_i\} \mid \text{class } C) \cdot P(\text{class } C)}{P(\{x_i\})}$$




- This term is the normalization constant. It doesn't depend on C , and is generally ignored until the end of the computation.
- You can calculate it by looking at the probability of the distinct set of feature levels of the data point are in the overall dataset.

- This term is known as your **posterior**. It's the calculated probability that the record belongs to class C once the data is taken into account.


$$P(\text{class } C \mid \{x_i\}) = \frac{P(\{x_i\} \mid \text{class } C) \cdot P(\text{class } C)}{P(\{x_i\})}$$

- The goal of any Bayesian computation is to find (“learn”) the posterior distribution of a particular feature. This constitutes the training phase of the model.

- In Bayesian inference, we update our beliefs about the distribution of C using the data (“evidence”) at our disposal.


$$P(\text{class } C \mid \{x_i\}) = \frac{P(\{x_i\} \mid \text{class } C) \cdot P(\text{class } C)}{P(\{x_i\})}$$

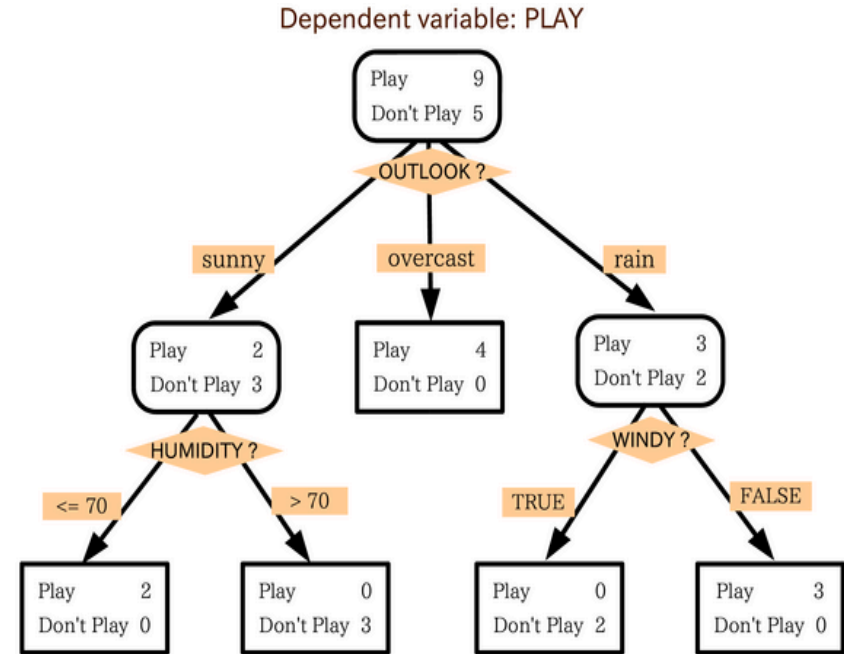
- Then we can use the posterior for prediction.

- 'Full' Bayes is often intractable due to the difficulty of calculating the normalization constant. It forces us to observe every possible combination of features to make a reasonable estimate, $P(\{x_i\}|C) = P(\{x_1, x_2, \dots, x_n\}|C)$.
- However, if we assume that the features x_i are conditionally independent from each other, we can make the likelihood function tractable as the final likelihood is the product of the probabilities of each feature given a particular class, times the likelihood of the class itself.
 - $P(\{x_i\}|C) = P(x_1|C) * P(x_2|C) * \dots * P(x_n|C) * P(C)$

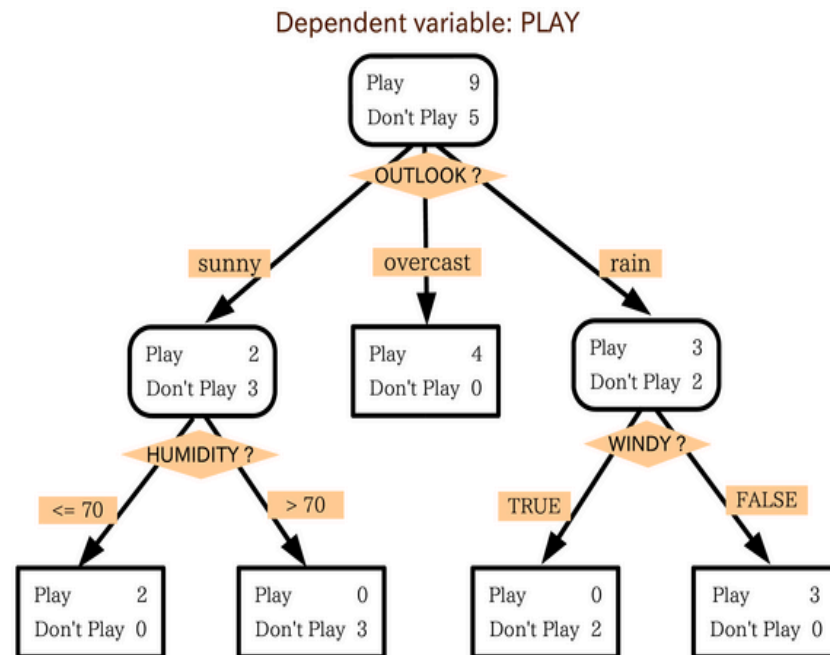
- Recall again that with our independence assumption, the likelihood of an observation being part of a class is the product of the probabilities of each feature given a particular class, times the likelihood of the class itself.
 - $P(\{x_i\}|C) = P(x_1|C) * P(x_2|C) * \dots * P(x_n|C) * P(C)$
- Example: say you're classifying whether an observation is for a rainy day or not. Rainy days occur about 10% of the time. You have two independent variables, the presence of schoolchildren in a park, and the average speed of traffic in a nearby street. It is raining 5% of the time when kids are in the park, and it rains about 30% of the time when the traffic speed is 55 mph. Assume the inverse conditions hold true as well. Say that in the observation, the kids are not in the park, but the traffic speed is 55 mph.
 - $P(\text{Rain}|\text{data}) = 95\% * 30\% * 10\% = 2.85\%$.
 - $P(\text{Not Rain}|\text{data}) = 5\% * 70\% * 90\% = 3.15\%$
- So, I would say it is likely not raining!

II. DECISION TREES

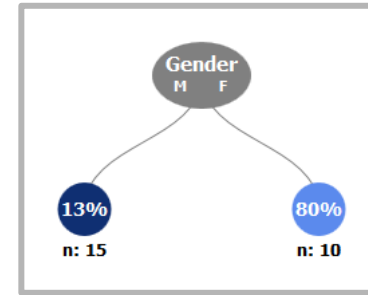
- A decision tree is a way to structure by using heuristics to split it into discreet groups.
- At each split in the tree, the data is divided by a test condition (e.g., is the person male or female?)
- The terminal nodes of the tree show the frequency of the output metric for the members in the most finely-divided group (for classification), or the average value of the output metric in the group (for regression).



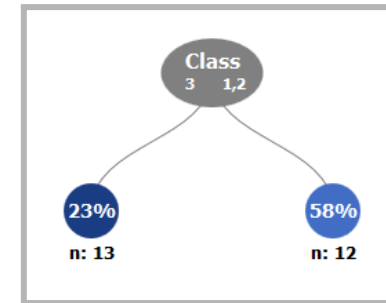
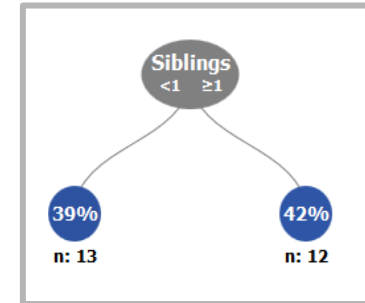
- Decision trees on their own aren't usually all that predictive. However:
 - Decision trees deal with high-dimensional and nonlinear data well;
 - They are highly interpretable; and
 - They are the underpinning of more sophisticated (and some of the most accurate) machine learning techniques, such as random forests and boosted trees.



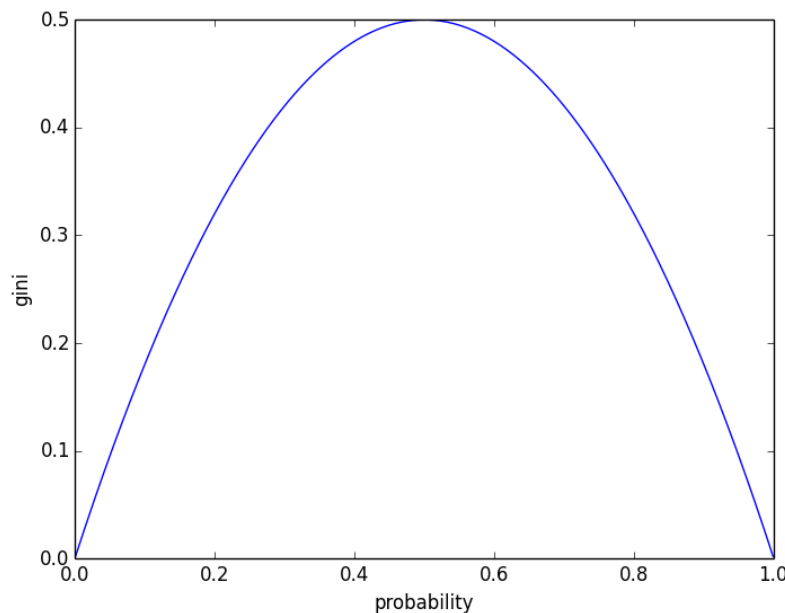
- Different variables and split options are evaluated to determine which split will provide the greatest separation between classes.
- For classification, splits are chosen by measuring the class purity before and after the split. Purity is calculated by the Gini index.
 - For n classes, $Gini = 1 - p_1 - \dots - p_n$ where $p_{i..n}$ is the percent frequency of class n in your overall dataset.



Which split would you choose?



- A Gini index of 0.5 indicates equal representation between both classes, and a Gini index of 0 indicates perfect separation between classes (perfect purity).
- The split with the lowest Gini impurity gets 'chosen' at each step of the tree's construction.
- Nodes typically stop being created once additional splits do not improve Gini purity beyond the level it is already at (say, 0.01).



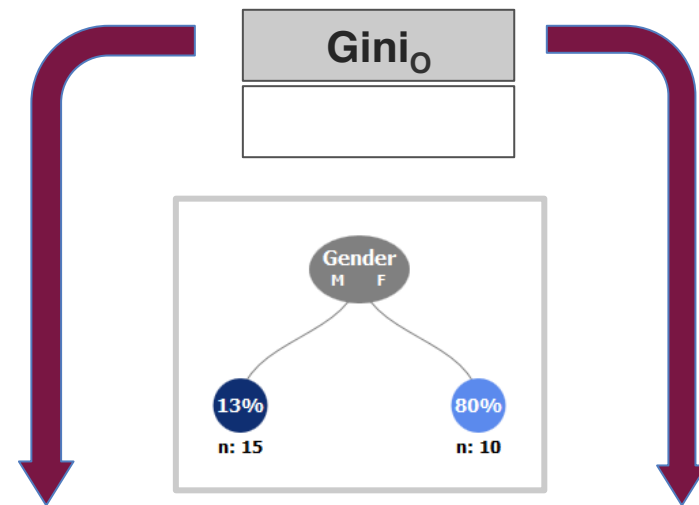
1. Calculate the purity of the data
2. Select a candidate split
3. Calculate the purity of the data after the split
4. Repeat for all variables
5. Choose the variable with the greatest decrease in purity and add it to the tree
6. Repeat for each split until some stop criteria is met

Now, let's move on to a real-world example of a decision tree in action.

- Say you are calculating whether or not a passenger on the Titanic survived the ship's sinking.
- Before your first split, the distribution of the response feature is to the right.
- The Gini coefficient for the data is:
 - $1 - (10/25)^2 - (15/25)^2 = 0.48$

Before Split	All
Survived	10
Died	15

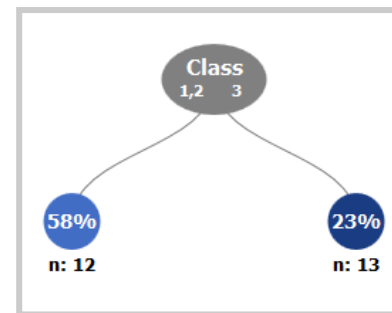
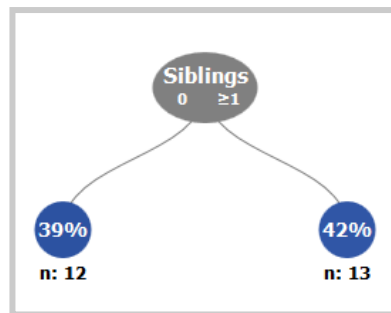
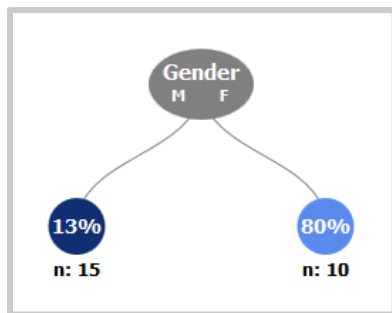
- Let's do your first split (by gender).
- Among men, roughly 86% of passengers died, while among women, only 10% died. Recall that in the overall dataset, the death rate was 75%.
 - Gini in the male group is
 $1 - (2/15)^2 - (13/15)^2 = 0.23$
 - Gini in the female group is
 $1 - (8/10)^2 - (2/10)^2 = 0.32$
- Overall average group Gini is
 $0.23 * (15/25) + 0.32 * (10/25) = 0.27$
- So, Gini went down, and the split achieved its aim!**



Gender	M
Survived	2
Died	13

Gender	F
Survived	8
Died	2

- How does the Gini coefficient compare for the Siblings and class variables?



Gender	M	F
Survived	2	8
Died	13	2
Gini _C	0.27	

Siblings	0	≥1
Survived	5	5
Died	7	8
Gini _C	0.48	

Class	1,2	3
Survived	7	3
Died	5	10
Gini _C	0.42	

- As Gender creates the biggest decrease in impurity, we select it for the first tree split!

III. EVALUATING CLASSIFICATION MODEL ACCURACY

- In last week's class, you learned a few ways to evaluate predictive models for continuous data, with root mean squared error (RMSE) being the most popular and informative.
- For classification models, so far we have only looked at sheer accuracy.
- But, what happens if your data is imbalanced?
 - Ex: when detecting disease prevalence, only 1% of patients have a disease.
 - However, a model that predicts that no one has a disease would be accurate 99% of the time, but in no way would be informative or useful to our needs!

- We typically look at a few metrics to incorporate model meaningfulness for imbalanced data and differences in relative importance of detecting different levels in the data:
 - Confusion Matrices;
 - Receiver Operating Characteristic (ROC) curves; and
 - Precision-Recall curves.

- Confusion matrices show the overall performance of a classifier in its ability to accurately predict n classes inside a test set of data.
- To your right is an example of a matrix for detecting the presence of the disease.
 - The horizontal labels show the actual outcomes in the test set.
 - The vertical labels show the predicted outcomes for the same data in the set.

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

- Confusion matrices show a lot of data.
 - True Positives (TP)
 - True Negatives (TN)
 - False Positives (FP)
 - False Negatives (FN)
- Overall accuracy is $(TP + TN) / \text{Total} = (100 + 50) / 165 = 0.91$
- Error rate is $(FP + FN) / \text{Total} = (10 + 5) / 165 = 0.09$

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

- The false positive rate is $FP / (FP + TN) = 10 / (10 + 50) = 17\%$
- **Sensitivity** or 'recall', one of the most commonly used metrics, records the percent of correct classifications made when the actual value is positive.
 - It's calculated by: $TP / (TP + FN) = 100/105 = 0.95$
- **Specificity** records the percent of correct classifications when the actual value is negative.
 - $TN / (TN + FP) = 50 / (10 + 50) = 0.83$

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

- In the real world, you want to look at specificity and sensitivity a lot.
 - If you want a balanced classifier that gets both classes right most of the time, looking at sensitivity and specificity will help you avoid the pitfalls explained before from looking at sheer accuracy.
 - The F1 score, which is the *harmonic mean* of sensitivity and specificity gives you the combined accuracy of both metrics.
 - It is calculated by $2 * ((\text{precision} * \text{recall}) / (\text{precision} + \text{recall}))$
- However, using both metrics to compare models can be tricky.
 - Ex: is one model 'better' than another if it has 1% more sensitivity but 2% less specificity?
- So, to determine the relative predictiveness of both models, we look at a measure called Cohen's Kappa.

- Cohen's Kappa is a measure of the information gain of your model against random selection.
- For example, a completely naïve classification model would just classify each observation in the class with the highest frequency, as the highest frequency would have the highest probability of being selected.
- Practically, your model—and the data it's incorporating—doesn't have any use or purpose if you can do no better than the naïve method described above.
- So, we calculate Cohen's Kappa by:

$$\kappa = \frac{\text{Pr}(a) - \text{Pr}(e)}{1 - \text{Pr}(e)},$$

- Where $\text{Pr}(a)$ your model's accuracy, and $\text{Pr}(e)$ is the probability of chance agreement (i.e. random selection).

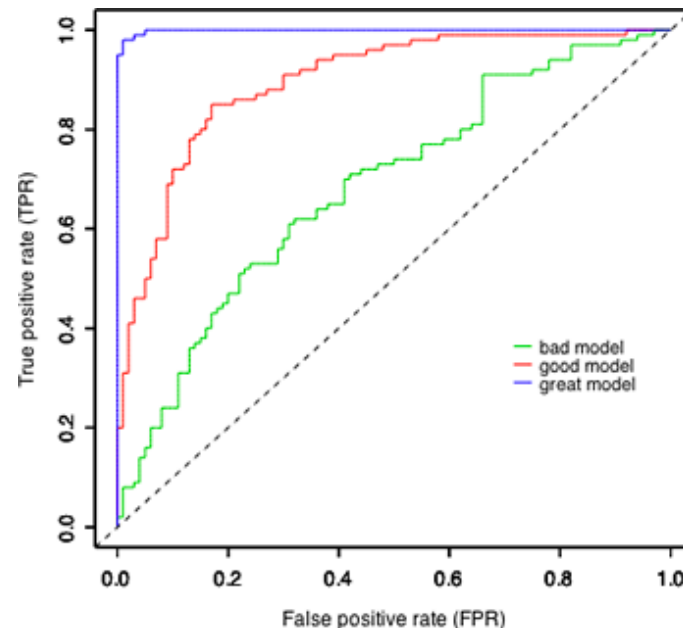
- Cohen's Kappa can vary beyond -1 (complete information loss) to 1 (complete information gain); however, the 'level' of Kappa seen as appropriate depends on the domain.
- Here's the general rules of thumb of how to interpret Kappa:
 - 0-0.20: slight
 - 0.21-0.40: fair
 - 0.41-0.60: moderate
 - 0.61-0.80: substantial
 - 0.81-1: almost perfect
- However, in some of my models, a Kappa of .14 is very meaningful!

- Recall that for most classification models (logistic regression, Naïve Bayes, decision trees), we use we use probabilities to guide which category to classify an observation under.
- But, how do we decide what probability cutoff to classify an observation on one way or another? What happens if the cost of misclassifying one category is far worse than another (for example, in cancer detection)?
- The answer: we use a receiver-operating characteristic curve.

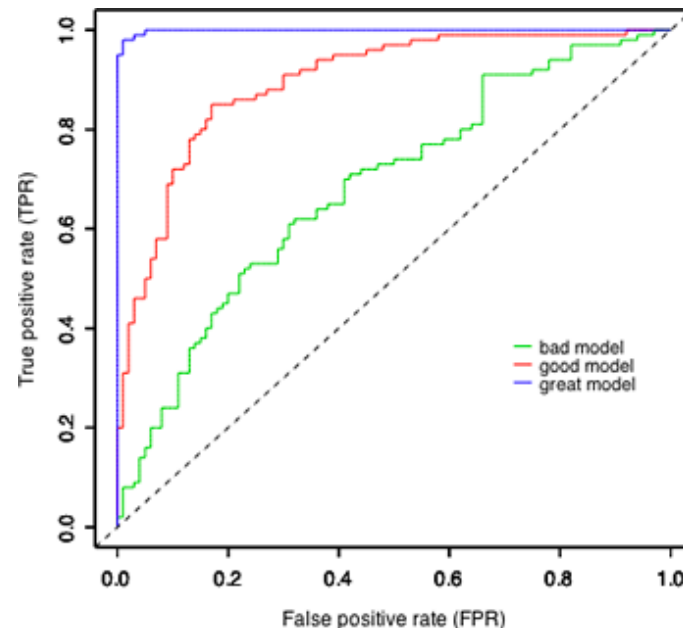
RECEIVER-OPERATING CHARACTERISTIC CURVES

35

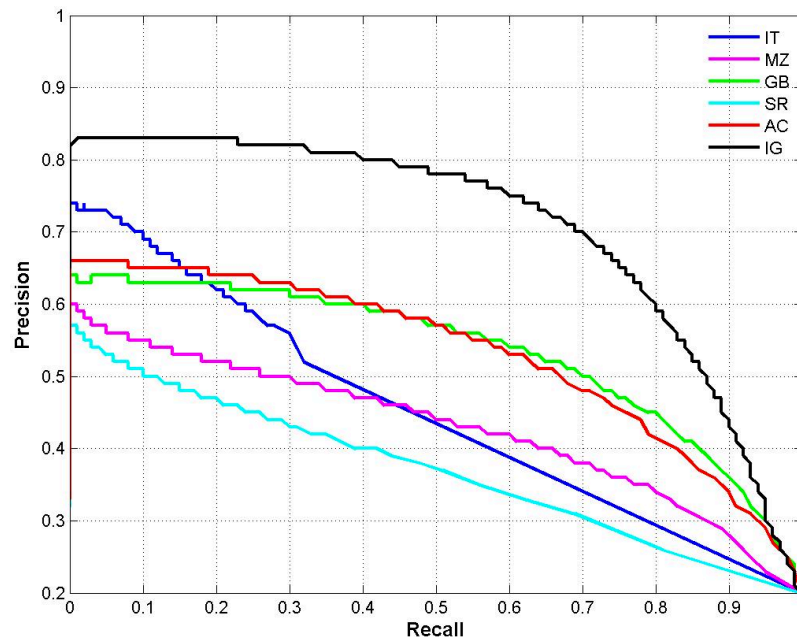
- A receiver-operating characteristic (ROC) curve charts the relationship between the amount of correctly identified observations and the amount of observations that receive false positives at different probability cutoffs in your dataset.
- TPR: when the actual person is identified to have cancer, how often is the prediction **correct**?
- FPR: when the person does not have cancer, how often is the prediction **wrong**?



- You can compare the predictive accuracy of different models using the ROC curve by calculating their area under the curve (AUC).
- An AUC of less than .5 (the dotted line to your right) shows that your probabilities are totally misaligned and there is no additional predictive gain of one class over the other by adjusting the cutoff.
- Here's the general guidelines of model accuracy when looking at AUC:
 - .90-1 = excellent ; .80-.90 = good
 - .70-.80 = fair ; .60-.70 = poor
 - .50-.60 = fail



- A precision-recall curve is related to, but different than, a ROC curve.
- Precision-recall curves are used when you have highly imbalanced datasets, especially when correct prediction of minority classes is paramount.
- Precision-recall curves chart the tradeoff at each probability cutoff between precision - $TP / (TP + FP)$, or accuracy of each class prediction) and recall (sensitivity), $TP / (TP + FN)$, which is your accuracy of predicting the outcomes themselves.



- Compute a Naïve Bayes model of Hall of Fame induction using the same features we used for our K-nearest neighbors homework.
- Compute k-fold cross validation, and score the model's accuracy using a confusion matrix and an ROC curve.
- Create a tree-based model using the same features of Naive Bayes
- Compare its accuracy metrics with the Naïve Bayes model.
- Tune the model's parameters to optimize ROC area under the curve.
- Compare it with our existing tree-based model.

LINEAR REGRESSION

QUESTIONS?