Machine Learning and Data Mining: Overview

Héctor Corrada Bravo

University of Maryland, College Park, USA

CMSC643: 2018-08-28

Course Organization

Course Webpage

- http://www.hcbravo.org/dscert-mldm/
- Shortened: http://bit.ly/hcb-mldm

Other Sites

- **ELMS**: Grades, assignments, etc.
- **Piazza**: Discussion, questions, etc.

Links in course webpage

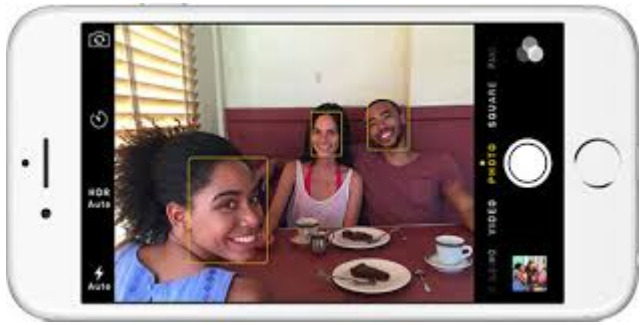
UMD Data Science Certificate

- CSMC 641: Principles of Data Science
- CMSC 642: Big Data Systems
- **CMSC 643: Machine Learning and Data Mining**
- CMSC 644: Algorithms for Data Science

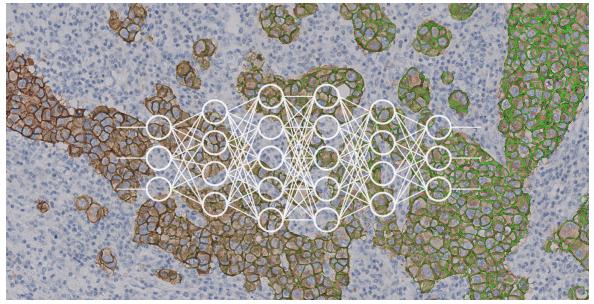
Machine Learning in the modern world

Self-driving cars

Image processing



Medical Diagnosis



Smart Cities

Introduction and Preliminaries

- What is Machine Learning, what is it useful for;
- types of Machine Learning problems and solutions;
- challenges in application of Machine Learning;
- ethical and fair application of Machine Learning;
- deployment of Machine Learning systems

Structure

- Project-based (scikit-learn, Keras)
- Problem sets

- Oral examination final (tech interview)

- Tuesdays 7:00-9:30pm CSIC 2118

Introduction and Overview

What is Machine Learning?

Machine Learning is the practice of creating computational systems that predict the future from the past

What is Machine Learning?

Suppose we want to predict if customer Bob has health insurance.

We can make use of attributes about Bob, e.g.,

- Bob's age,
- Bob's wage,
- Bob's level of education,

and try to **predict** if Bob has health insurance.

What is Machine Learning?

First Attempt: Hand-made ruleset

Create a set of rules that let's us predict if customers have health insurance.

What is Machine Learning?

First Attempt: Hand-made ruleset

Create a set of rules that let's us predict if customers have health insurance.

Example - "customers with only a High School diploma that make less than 25,000 dollars a year will not have insurance".

What is Machine Learning?

First Attempt: Hand-made ruleset

- Takes a lot of expertise to create such a rulebase

What is Machine Learning?

First Attempt: Hand-made ruleset

- Takes a lot of expertise to create such a rulebase
- Takes a lot of work to maintain it too: what if we get new attributes about customers?

What is Machine Learning?

First Attempt: Hand-made ruleset

- Takes a lot of expertise to create such a rulebase
- Takes a lot of work to maintain it too: what if we get new attributes about customers?
- What to do with attributes that are much harder to deal with, e.g., Bob's LinkedIn activity?

What is Machine Learning

Second Attempt: Instance-based Predictions

Gather data about customers, those that have health insurance and those that do not

What is Machine Learning

Second Attempt: Instance-based Predictions

Gather data about customers, those that have health insurance and those that do not

predict based on customers that have the same attribute values

What is Machine Learning

Second Attempt: Instance-based Predictions

Suppose Alice is a past customer. She has

- the same age as Bob,
- the same wage as Bob,
- the same education level as Bob
- and has health insurance.

What is Machine Learning

Second Attempt: Instance-based Predictions

Suppose Alice is a past customer. She has

- the same age as Bob,
- the same wage as Bob,
- the same education level as Bob
- and has health insurance.

Then predict that Bob will also have health insurance.

What is Machine Learning

What if there is no customer Alice, that is, there is no customer that has exactly the same attributes as Bob?

What is Machine Learning

What if there is no customer Alice, that is, there is no customer that has exactly the same attributes as Bob?

Even if we were able to make accurate predictions this way, have we gained any insight about customers and why they may have health insurance?

What is Machine Learning

In Machine Learning we build systems that use existing data, **training data**, to create a model that predicts a particular outcome

What is Machine Learning

Arthur Samuel defined Machine Learning as

> the field of study that gives computers the ability to learn without being explicitly programmed

What is Machine Learning

In our example, the rulebased system is a computational system that is *explicitly programmed*.

What is Machine Learning

In our example, the rulebased system is a computational system that is *explicitly programmed*.

Instead, ML creates systems that learn from examples

What is Machine Learning

In our example, the rulebased system is a computational system that is *explicitly programmed*.

Instead, ML creates systems that learn from examples

- examples: data from previous instances

Data Mining

ML: use probabilistic and statistical principles to build systems that *learn* from examples

Data Mining: use computational approaches to answer complex queries about data

Data Mining

- Summarization: effectively summarize large collections of data
  - e.g., Google PageRank algorithm as summary of World Wide Web
  - e.g., clustering as a way to summarize collections of data into smaller groups
- Feature extraction: find most prominent features of the data
  - e.g., frequent itemsets
  - e.g., similar items

Why use ML?

Why use ML?

- Building rule systems that capture non-trivial prediction problems require substantial expertise

Why use ML?

- Building rule systems that capture non-trivial prediction problems require substantial expertise
- Using attributes like social media activity, is very difficult to encode in explicit programs.

Why use ML?

There is *natural, non-reducible, variation* in the tasks we are trying to model.

Why use ML?

There is *natural, non-reducible, variation* in the tasks we are trying to model.

Even if Bob and Alice are exactly alike according to the attributes we have measured,

Bob may have health insurance while Alice does not due to reasons we are not aware of.

Why use ML?

Capturing this type of natural variation is very difficult in rule-based systems

ML models try to capture it as stochastic behavior.

Why use ML?

Learning systems can help human users gain insights.

Why use ML?

Learning systems can help human users gain insights.

In cases where instances are measured by a large amount of attributes,

Why use ML?

Learning systems can help human users gain insights.

In cases where instances are measured by a large amount of attributes,

ML systems can identify attributes that are important for prediction,

Why use ML?

Learning systems can help human users gain insights.

In cases where instances are measured by a large amount of attributes,

ML systems can identify attributes that are important for prediction,

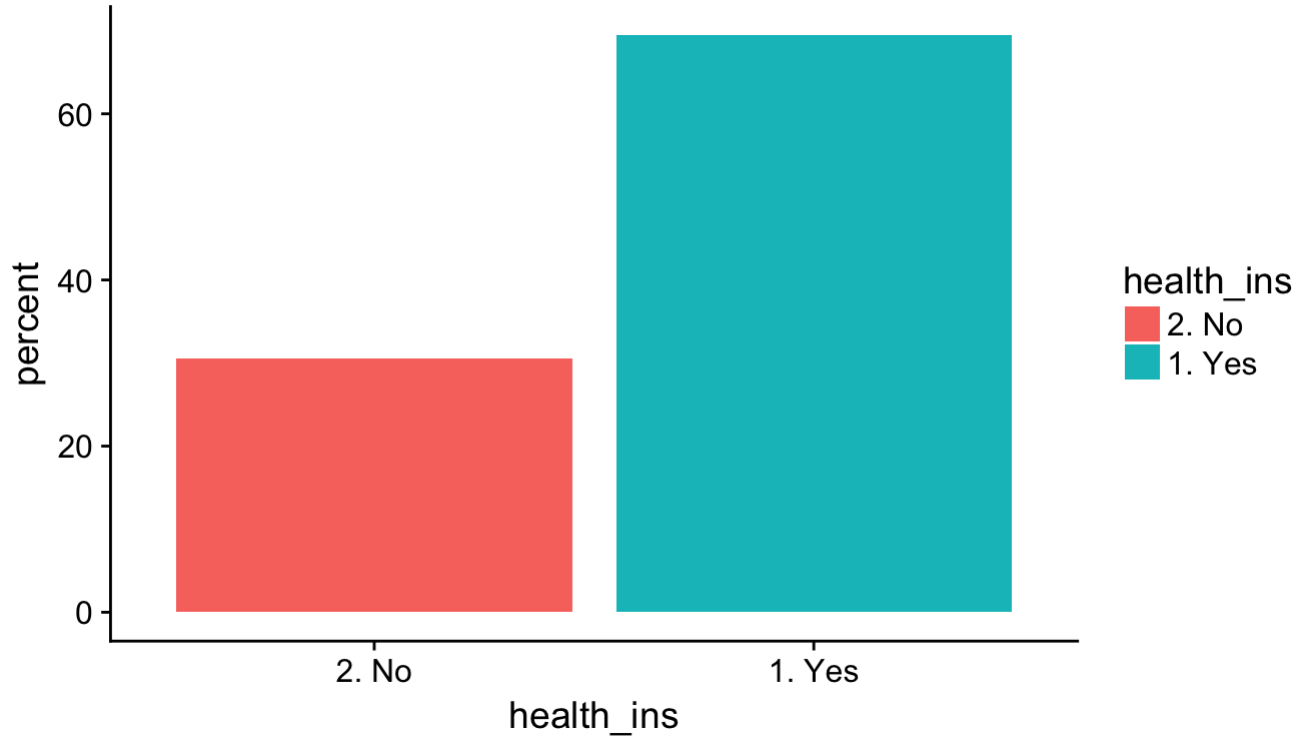
Users can then in turn study further to understand the relationship between these attributes and the outcome of interest.

An illustrative example

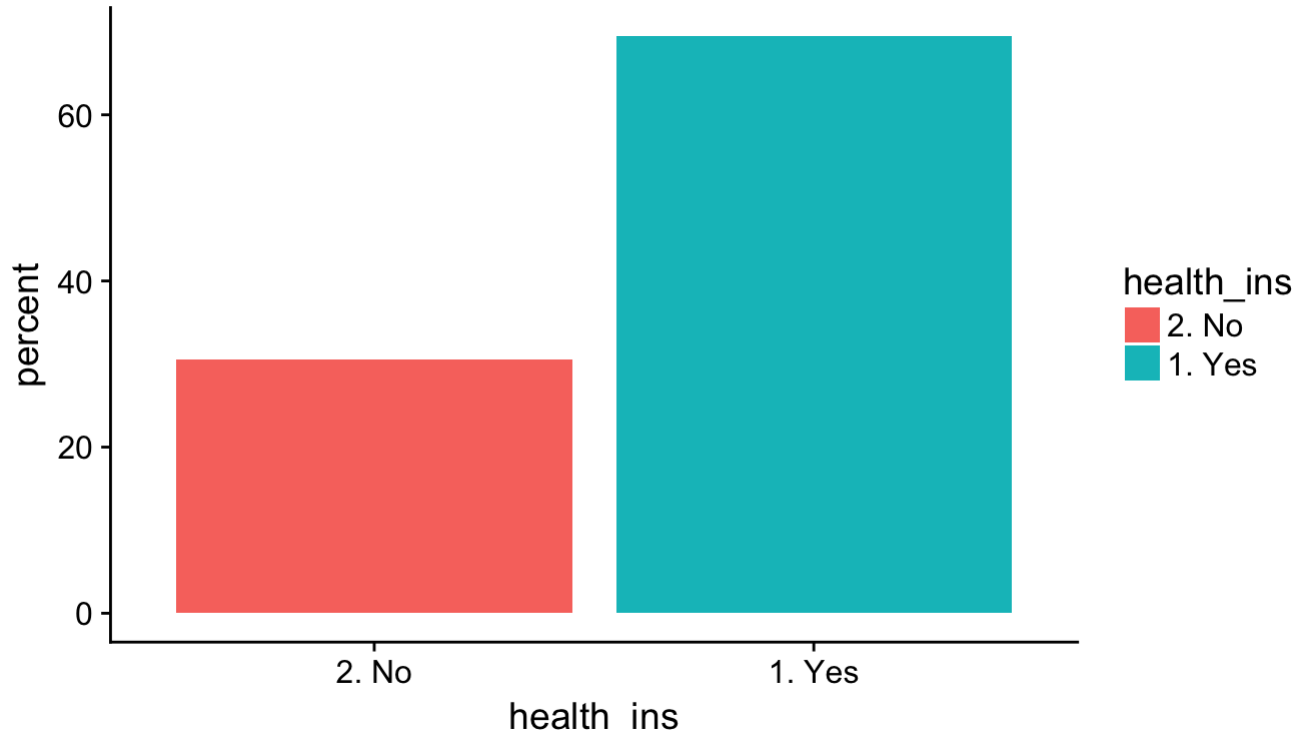
Suppose I have training data from customers and want to model health insurance status

maritl	race	education	jobclass	health_ins
1. Never Married	1. White	1. < HS Grad	1. Industrial	2. No
1. Never Married	1. White	4. College Grad	2. Information	2. No
2. Married	1. White	3. Some College	1. Industrial	1. Yes
2. Married	3. Asian	4. College Grad	2. Information	1. Yes
4. Divorced	1. White	2. HS Grad	2. Information	1. Yes

A first simple rule is to predict health status from the majority of the training instances.



A first simple rule is to predict health status from the majority of the training instances.

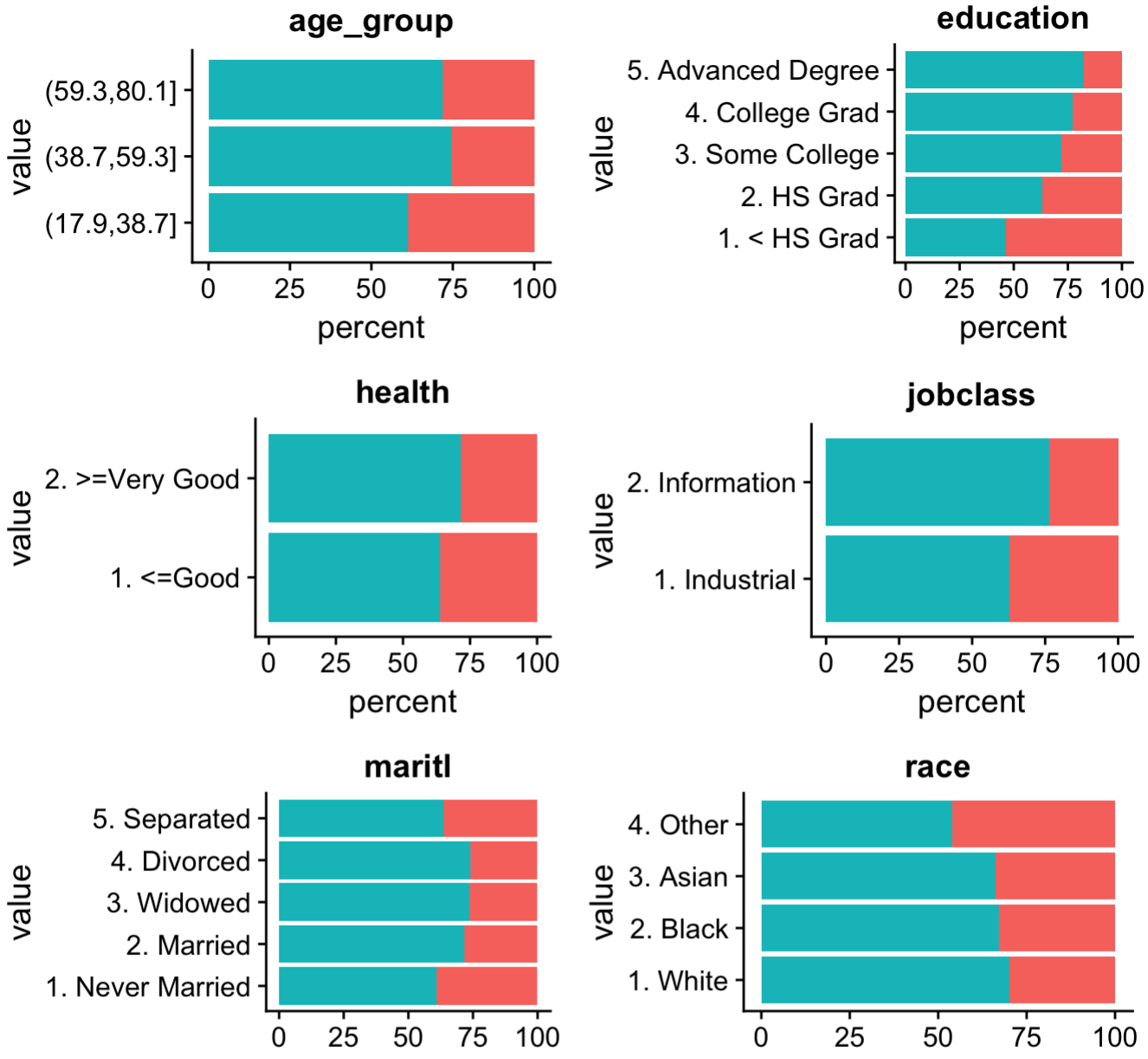


We would be wrong about 30% of new customers (assuming we see the same rate of insurance).

How can we do better?

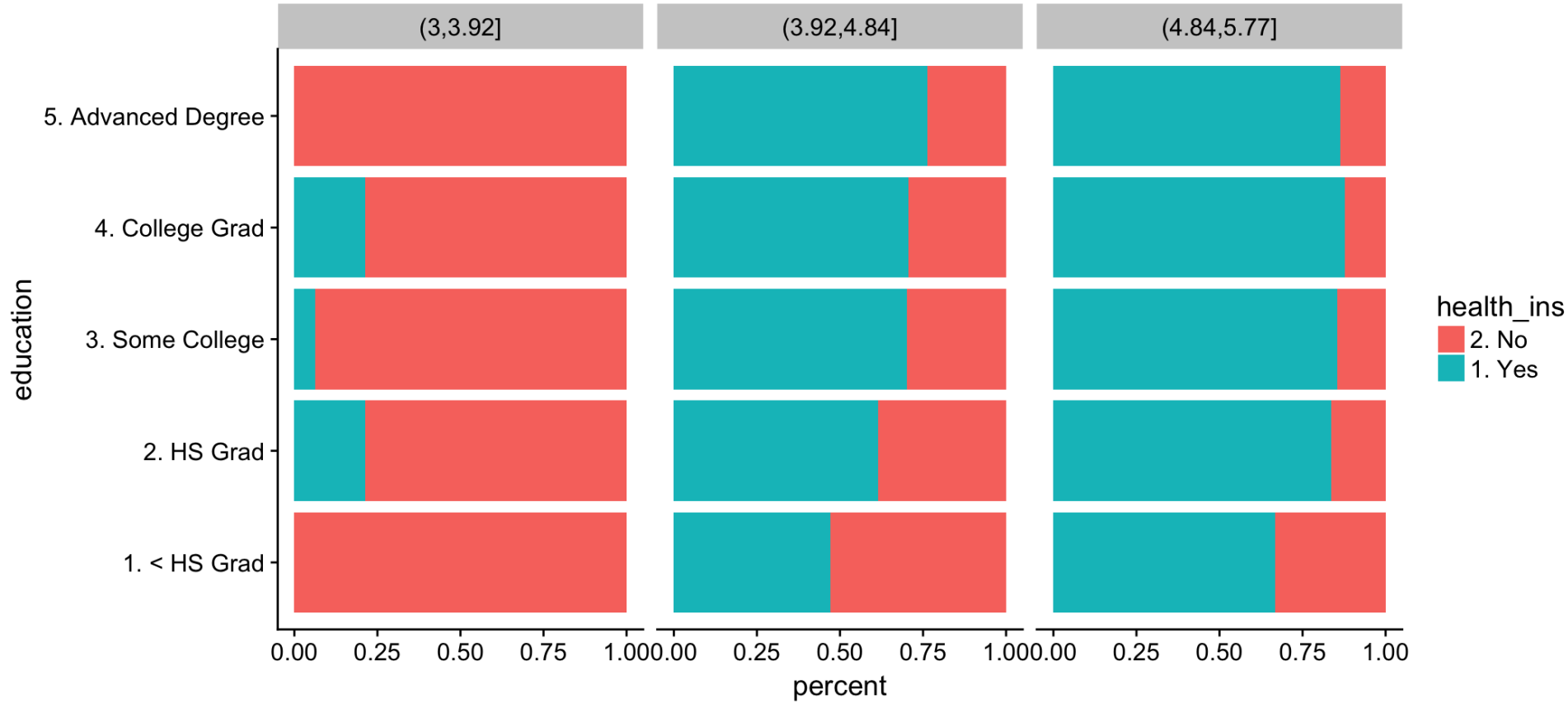
If the rate of insurance changes depending on other attributes, we can adjust our prediction accordingly.

In this case, we see that the rate of insurance changes significantly depending on the other measured attributes.


age_group
(59.3,80.1]
(38.7,59.3]
(17.9,38.7]
value
0
25
50
75
100
percent
education
5. Advanced Degree
4. College Grad
3. Some College
2. HS Grad
1. < HS Grad
value
0
25
50
75
100
percent
health
2. >=Very Good
1. <=Good
value
0
25
50
75
100
percent
jobclass
2. Information
1. Industrial
value
0
25
50
75
100
percent
maritl
5. Separated
4. Divorced
3. Widowed
2. Married
1. Never Married
value
0
25
50
75
100
race
4. Other
3. Asian
2. Black
1. White
value
0
25
50
75
100

Perhaps the insured rate based on education level also depends on customer's wage.

That is, there is an interaction between wage and education level that would affect our prediction of insurance status.



the rate of insurance in the high earners is similar regardless of education level, whereas for mid earners, the insurance rate varies significantly depending on education level.

These interactions could be extended to more than two attributes.

A rule-based system that incorporates this type of interaction, would perhaps require a large number of rules if the number of attributes is very large.

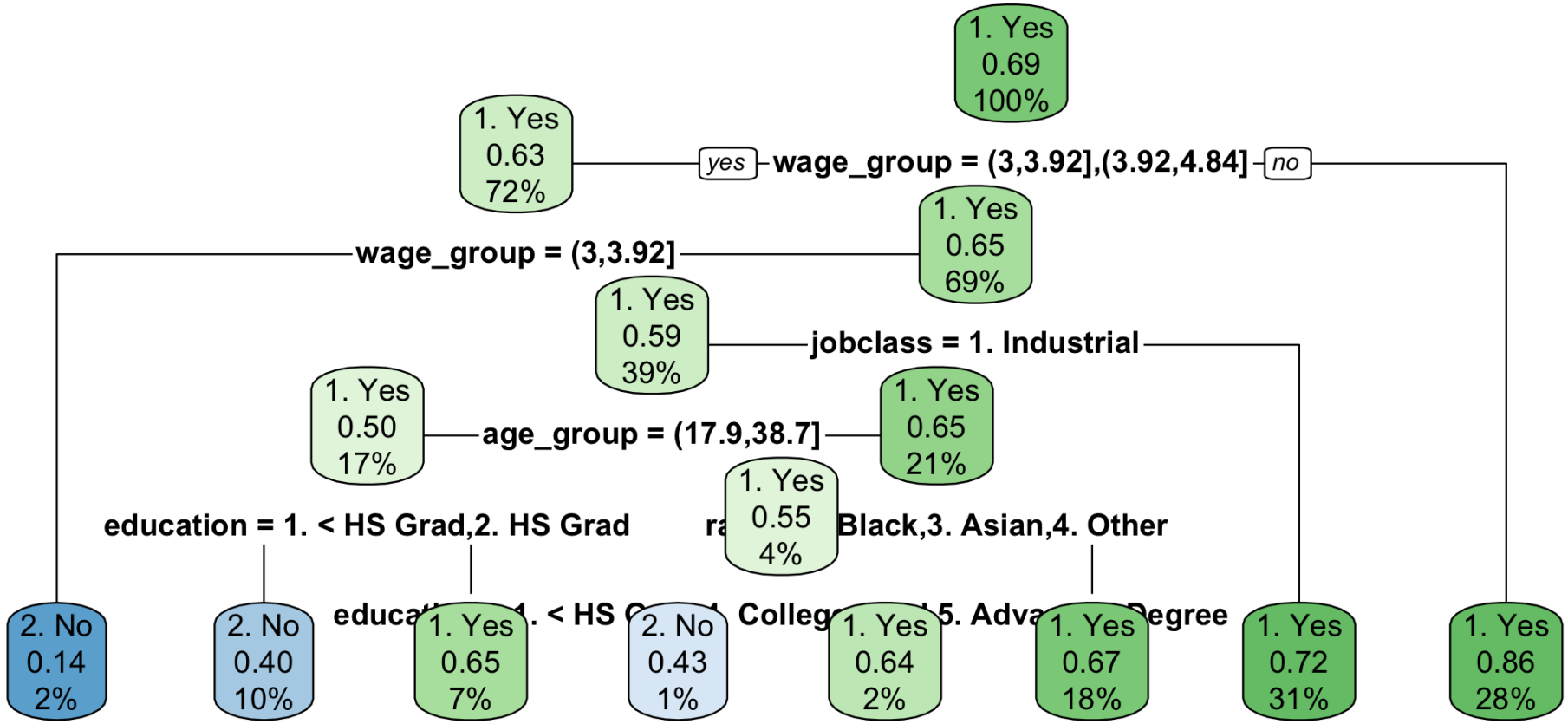
A ML Approach

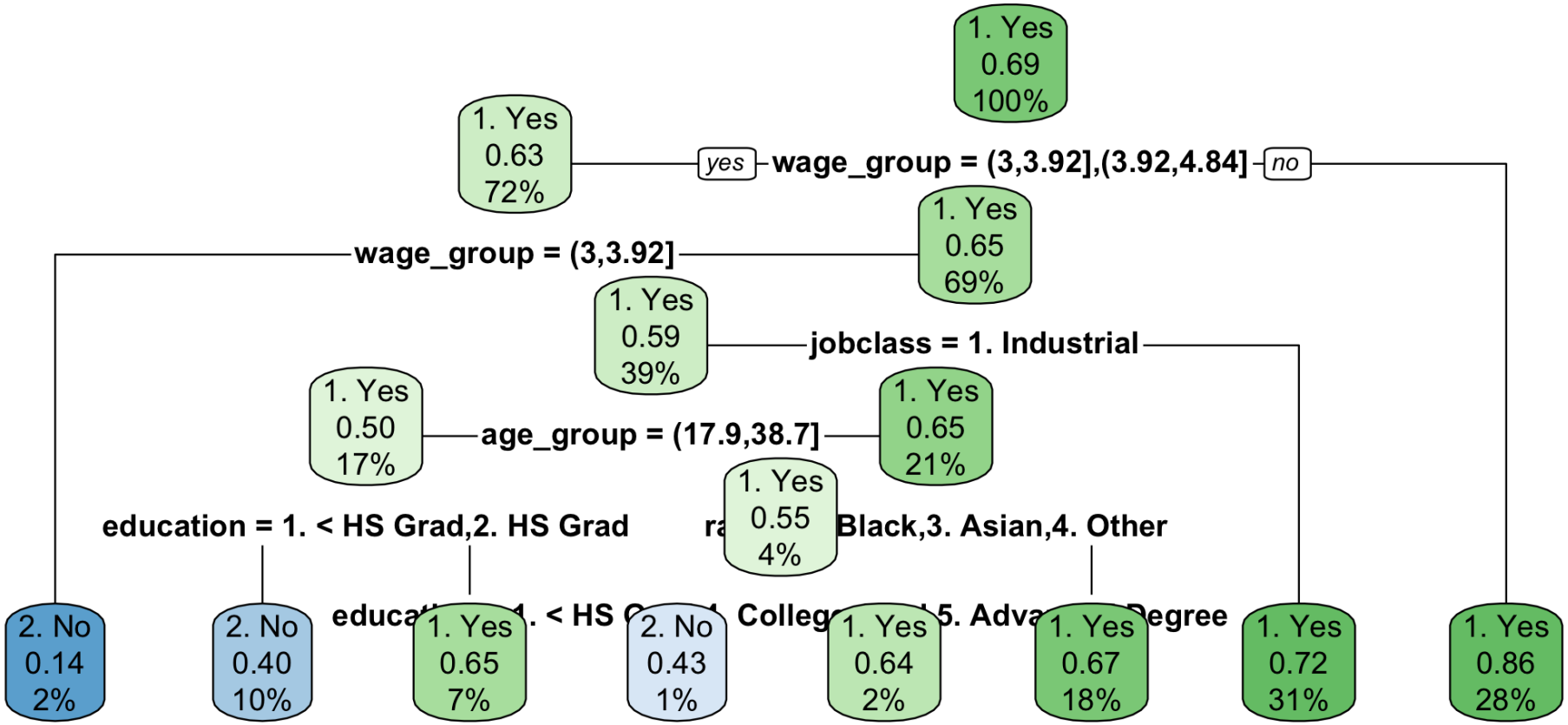
One of the workhorse methods in ML, which we will see again in more detail at later sessions, is the Decision Tree.

A ML Approach

One of the workhorse methods in ML, which we will see again in more detail at later sessions, is the Decision Tree.

It is a highly interpretable model since we can think the Decision Tree algorithm as *learning* a rule-based system consisting of multiple interaction rules over the attributes of a given dataset.


1. Yes
0.69
100%
1. Yes
0.63
72%
yes
wage_group = (3,3.92],(3.92,4.84]
no
wage_group = (3,3.92]
1. Yes
0.65
69%
1. Yes
0.59
39%
jobclass = 1. Industrial
1. Yes
0.50
17%
age_group = (17.9,38.7]
1. Yes
0.65
21%
1. Yes
0.55
4%
education = 1. < HS Grad,2. HS Grad
Black,3. Asian,4. Other
2. No
0.14
2%
2. No
0.40
10%
1. Yes
0.65
7%
2. No
0.43
1%
1. Yes
0.64
2%
1. Yes
0.67
18%
1. Yes
0.72
31%
1. Yes
0.86
28%



The predictions given by this tree would be wrong roughly 27% of the time.

An illustrative example

Now that we have trained this Decision Tree we can deploy it to make predictions on new customers.

An illustrative example

Now that we have trained this Decision Tree we can deploy it to make predictions on new customers.

Do we think our predictions will be incorrect 27% of the time?

An illustrative example

Now that we have trained this Decision Tree we can deploy it to make predictions on new customers.

Do we think our predictions will be incorrect 27% of the time?

We'll see a little later that this may not be quite correct.

Types of Machine Learning Systems

Machine Learning methods can be categorized based on

Types of Machine Learning Systems

Machine Learning methods can be categorized based on

- the nature of the task they are trying to solve,

Types of Machine Learning Systems

Machine Learning methods can be categorized based on

- the nature of the task they are trying to solve,
- the way data is used to train the model,

Types of Machine Learning Systems

Machine Learning methods can be categorized based on

- the nature of the task they are trying to solve,
- the way data is used to train the model,
- and the nature of the learning algorithm itself.

Types of ML Systems

Supervised vs. Unsupervised Learning

Our running example of predicting insurance status is a *supervised* learning problem.

Types of ML Systems

Supervised vs. Unsupervised Learning

Our running example of predicting insurance status is a *supervised* learning problem.

We have attributes over observations of interest, we want to *predict* a specific outcome and our data measures this outcome for all observations in our training data.

Types of ML Systems

Supervised vs. Unsupervised Learning

Our running example of predicting insurance status is a *supervised* learning problem.

We have attributes over observations of interest, we want to *predict* a specific outcome and our data measures this outcome for all observations in our training data.

each instance is *labeled* with the outcome we want to learn.

Types of ML Systems

Supervised vs. Unsupervised Learning

In the insurance case, the outcome is categorical (e.g., Yes/No).

Types of ML Systems

Supervised vs. Unsupervised Learning

In the insurance case, the outcome is categorical (e.g., Yes/No).

In general, the task of predicting a categorical outcome is called *classification*, as we are trying to classify instances into one of multiple classes.

Types of ML Systems

Supervised vs. Unsupervised Learning

In the insurance case, the outcome is categorical (e.g., Yes/No).

In general, the task of predicting a categorical outcome is called *classification*, as we are trying to classify instances into one of multiple classes.

Regression problems are those tasks where the outcome we are trying to predict is numerical.

Types of ML Systems

Supervised vs. Unsupervised Learning

In *unsupervised* learning we are not interested in predicting an outcome, but rather trying to learn some underlying structure, or patterns, from the data.

Types of ML Systems

Supervised vs. Unsupervised Learning

In *unsupervised* learning we are not interested in predicting an outcome, but rather trying to learn some underlying structure, or patterns, from the data.

In this case, instances in our training data are *unlabeled*.

Types of ML Systems

Supervised vs. Unsupervised Learning

The two most common applications in unsupervised learning are:

Clustering:

partition instances into multiple groups of similar instances,

Types of ML Systems

Supervised vs. Unsupervised Learning

The two most common applications in unsupervised learning are:

Clustering:

partition instances into multiple groups of similar instances,

Dimensionality reduction:

represent instances for which we have a large number of attributes using a small number of dimensions.

Method	Goal
K-Means	clustering
Hierarchical clustering	clustering
Expectation Maximization	clustering
Principal Component Analysis	dimensionality reduction
Locally-Linear Embedding	dimensionality reduction
t-distributed Stochastic Neighbor Embedding (tSNE)	dimensionality reduction

Types of ML Systems

Supervised vs. Unsupervised Learning

Another setting is *semi-supervised* learning where we have labels for a portion of our training data.

In this case, we want to use similarity between unlabeled and labeled data to learn a classification function.

Types of ML Systems

Batch vs. Online Learning

In *online learning*, we assume training data for our system arrives in a stream, a small number of training instances at a time.

We want to build a system that continuously updates its prediction function as new training instances arrives.

Types of ML Systems

Batch vs. Online Learning

In contrast, *batch learning* is the case where we have all of the training instances we want to use to build our system at training time.

Batch learning is the most common use case, although online learning is useful for extremely large datasets.

Types of ML Systems

Instance vs. Model-based Learning

Instance-based systems are based on directly looking up instances in a training set that are similar to the instance we want to predict and making a prediction based on the outcomes for those similar instances.

We can think of K-Nearest neighbors as an instance-based method.

Types of ML Systems

Instance vs. Model-based Learning

Model-based methods summarize training instances and learn a function based on that summary to make predictions.

Types of ML Systems

Instance vs. Model-based Learning

Model-based methods summarize training instances and learn a function based on that summary to make predictions.

A rule of thumb:

systems that can make predictions after discarding the training data are *model-based* systems.

Types of ML Systems

Instance vs. Model-based Learning

Model-based methods summarize training instances and learn a function based on that summary to make predictions.

A rule of thumb:

systems that can make predictions after discarding the training data are *model-based* systems.

systems that need to keep the training data to make predictions are *instance-based* methods.

Types of ML Systems

We will spend most of our time in this course covering *batch*, *model-based* learning for *supervised* and *unsupervised* tasks.

Challenges of ML Application

While Machine Learning models can be very powerful tools, their application can pose some problems.

Challenges of ML Application

Insufficient training data

For very complex tasks, think image or voice recognition, or cases where we have many relevant attributes, many training instances are required to succesfully learn useful models.

In cases where there is not enough training data to do so, predictions made by the trained models may not be very good.

Challenges of ML Application

Nonrepresentative training data

If the training data is not representative of the instances the model will make predictions for after deployment, the predictions will not be very good.

Challenges of ML Application

Nonrepresentative training data

If the training data is not representative of the instances the model will make predictions for after deployment, the predictions will not be very good.

For example, suppose in our insurance example, all our training data is for customers under the age of 60, our model will not be able to make good predictions for customers older than 60.

Challenges of ML Application

Poor-quality data

Data that is missing and/or incorrect will make training extremely difficult.

A significant effort is usually spent cleaning data before training ML models.

Challenges of ML Application

Irrelevant data

Including attributes that are irrelevant to the predictions we want to make will also make training more difficult.

While many ML methods are capable of alleviating this problem by ignoring irrelevant attributes, it is a good practice to not include these in the training data.

Challenges of ML Application

In general, the last few problems we've mentioned can be summarized as "garbage in, garbage out",

Challenges of ML Application

In general, the last few problems we've mentioned can be summarized as "garbage in, garbage out",

the better the training data, the better the predictions we will get from our models.

Challenges of ML Application

Overfitting training data

The ultimate goal when applying ML models is to be able to make good predictions for new unobserved data based on training data.

Challenges of ML Application

Overfitting training data

The ultimate goal when applying ML models is to be able to make good predictions for new unobserved data based on training data.

There is a danger however, that our model will learn to predict the training data so well, that it will not be able to make good predictions on new unobserved data.

Challenges of ML Application

Overfitting training data

The ultimate goal when applying ML models is to be able to make good predictions for new unobserved data based on training data.

There is a danger however, that our model will learn to predict the training data so well, that it will not be able to make good predictions on new unobserved data.

This problem is called "over-fitting".

Challenges of ML Application

Overfitting training data

The ultimate goal when applying ML models is to be able to make good predictions for new unobserved data based on training data.

There is a danger however, that our model will learn to predict the training data so well, that it will not be able to make good predictions on new unobserved data.

This problem is called "over-fitting".

We will see later, that most methods provide ways to control over-fitting during training.

Challenges of ML Application

Overfitting

In Decision Trees, we can characterize the complexity of the models we learn by measuring the "depth" of the tree.

Challenges of ML Application

Overfitting

In Decision Trees, we can characterize the complexity of the models we learn by measuring the "depth" of the tree.

Depth is the length of the longest path from root to leaf in the tree.

Challenges of ML Application

Overfitting

In Decision Trees, we can characterize the complexity of the models we learn by measuring the "depth" of the tree.

Depth is the length of the longest path from root to leaf in the tree.

In a decision tree, this is the number of tests used to make a prediction.

Challenges of ML Application

Overfitting

In Decision Trees, we can characterize the complexity of the models we learn by measuring the "depth" of the tree.

Depth is the length of the longest path from root to leaf in the tree.

In a decision tree, this is the number of tests used to make a prediction.

The more tests we make, the more complex the model.

Challenges of ML Application

Overfitting

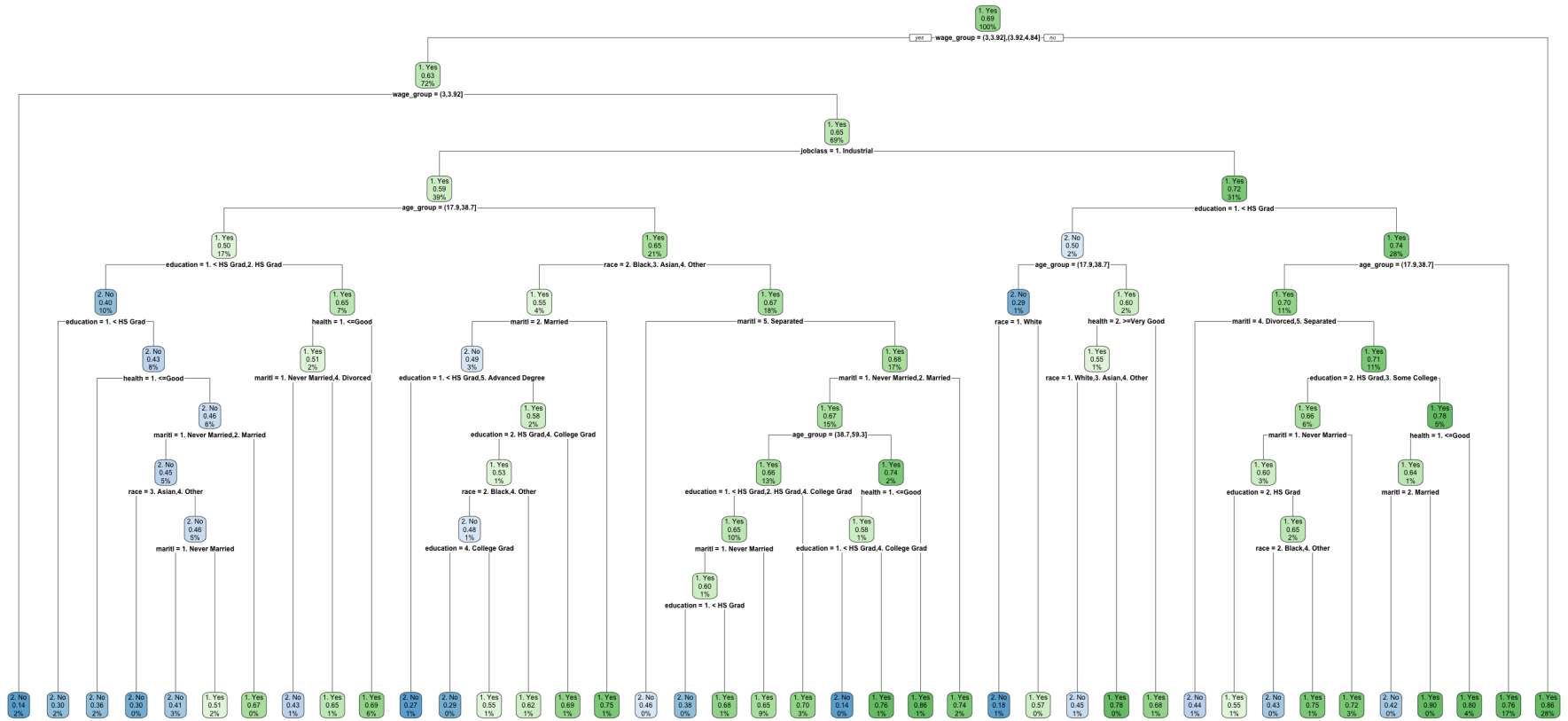
In Decision Trees, we can characterize the complexity of the models we learn by measuring the "depth" of the tree.

Depth is the length of the longest path from root to leaf in the tree.

In a decision tree, this is the number of tests used to make a prediction.

The more tests we make, the more complex the model.

If we make very deep trees we can easily overfit the data.


1. Yes 0.69 100%
yes
wage_group = (3,3.92],(3.92,4.84]
no
1. Yes 0.63 72%
wage_group = (3,3.92]
1. Yes 0.65 69%
jobclass = 1. Industrial
1. Yes 0.59 39%
age_group = (17.9,38.7]
1. Yes 0.72 31%
education = 1. < HS Grad
1. Yes 0.50 17%
education = 1. < HS Grad,2. HS Grad
1. Yes 0.65 21%
race = 2. Black,3. Asian,4. Other
2. No 0.50 2%
age_group = (17.9,38.7]
1. Yes 0.74 28%
age_group = (17.9,38.7]
2. No 0.40 10%
education = 1. < HS Grad
1. Yes 0.65 7%
health = 1. <=Good
1. Yes 0.55 4%
maritl = 2. Married
1. Yes 0.67 18%
maritl = 5. Separated
2. No 0.29 1%
race = 1. White
1. Yes 0.60 2%
health = 2. >=Very Good
1. Yes 0.70 11%
maritl = 4. Divorced,5. Separated
1. Yes 0.71 11%
education = 2. HS Grad,3. Some College
2. No 0.43 8%
health = 1. <=Good
1. Yes 0.51 2%
maritl = 1. Never Married,4. Divorced
2. No 0.49 3%
education = 1. < HS Grad,5. Advanced Degree
1. Yes 0.58 2%
education = 2. HS Grad,4. College Grad
1. Yes 0.68 17%
maritl = 1. Never Married,2. Married
1. Yes 0.55 1%
race = 1. White,3. Asian,4. Other
1. Yes 0.66 6%
maritl = 1. Never Married
1. Yes 0.78 5%
health = 1. <=Good
2. No 0.46 6%
maritl = 1. Never Married,2. Married
1. Yes 0.53 1%
race = 2. Black,4. Other
1. Yes 0.67 15%
age_group = (38.7,59.3]
1. Yes 0.60 3%
education = 2. HS Grad
1. Yes 0.64 1%
maritl = 2. Married
2. No 0.45 5%
race = 3. Asian,4. Other
2. No 0.48 1%
education = 4. College Grad
1. Yes 0.66 13%
education = 1. < HS Grad,2. HS Grad,4. College Grad
1. Yes 0.74 2%
health = 1. <=Good
1. Yes 0.65 2%
race = 2. Black,4. Other
2. No 0.46 5%
maritl = 1. Never Married
1. Yes 0.65 10%
maritl = 1. Never Married
1. Yes 0.58 1%
education = 1. < HS Grad,4. College Grad
1. Yes 0.60 1%
education = 1. < HS Grad
2. No 0.14 2%
2. No 0.30 2%
2. No 0.36 2%
2. No 0.30 0%
2. No 0.41 3%
1. Yes 0.51 2%
1. Yes 0.67 0%
2. No 0.43 1%
1. Yes 0.65 1%
1. Yes 0.69 6%
2. No 0.27 1%
2. No 0.28 0%
1. Yes 0.55 1%
1. Yes 0.62 1%
1. Yes 0.69 1%
1. Yes 0.75 1%
2. No 0.46 0%
2. No 0.38 0%
1. Yes 0.68 1%
1. Yes 0.65 9%
1. Yes 0.70 3%
2. No 0.14 0%
1. Yes 0.76 1%
1. Yes 0.86 1%
1. Yes 0.74 2%
2. No 0.19 1%
1. Yes 0.57 0%
2. No 0.45 1%
1. Yes 0.76 0%
1. Yes 0.68 1%
2. No 0.44 1%
1. Yes 0.55 1%
2. No 0.43 0%
1. Yes 0.75 1%
1. Yes 0.72 3%
2. No 0.42 0%
1. Yes 0.90 0%
1. Yes 0.80 4%
1. Yes 0.76 17%
1. Yes 0.86 28%

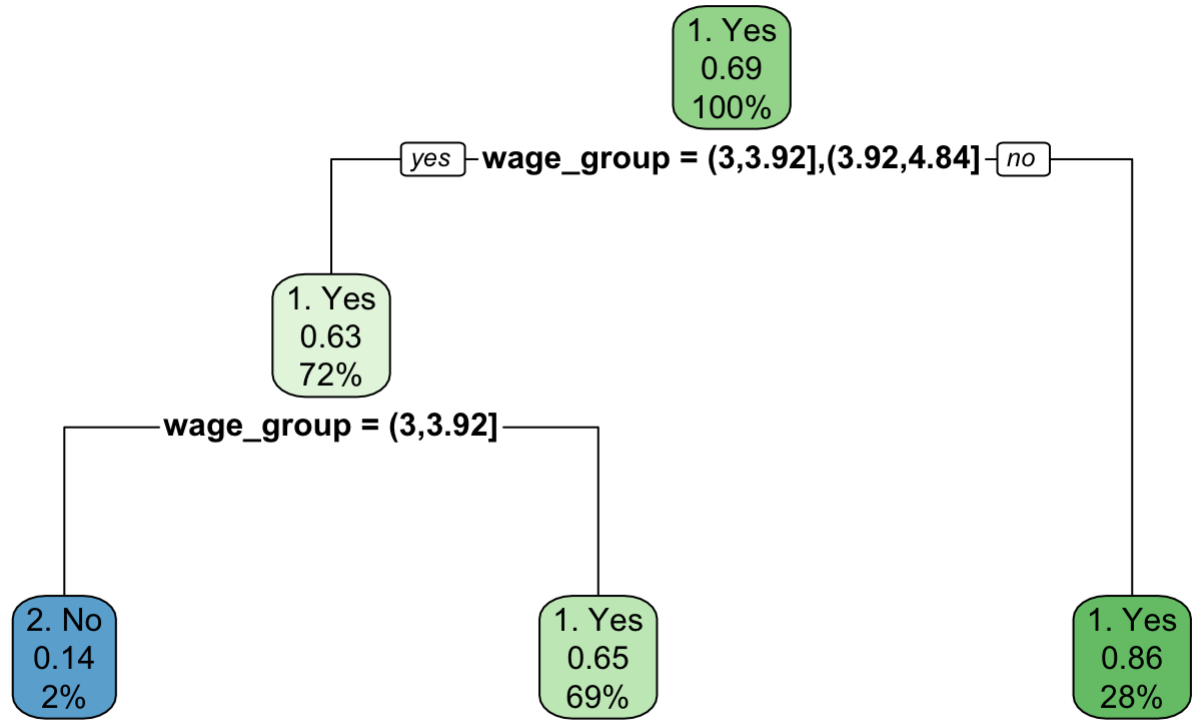
Challenges of ML Application

Underfitting

The opposite problem of course is under-fitting.

In this case, the data analyst is so over-zealous about avoiding over-fitting training data that models fail to learn how to make useful predictions.

In the decision tree case, this would result from building trees that are too shallow.


1. Yes
0.69
100%
yes
wage_group = (3,3.92],(3.92,4.84]
no
1. Yes
0.63
72%
wage_group = (3,3.92]
2. No
0.14
2%
1. Yes
0.65
69%
1. Yes
0.86
28%

Testing and validating Machine Learning systems

For the most part, the goal of applying ML systems is to obtain accurate predictions for future data by building models using past data.

Testing and validating Machine Learning systems

For the most part, the goal of applying ML systems is to obtain accurate predictions for future data by building models using past data.

This leads to the natural questions of

a) how do we quantify prediction accuracy, and b) how can we estimate what that accuracy will be for future data.

Testing and validating Machine Learning systems

For the most part, the goal of applying ML systems is to obtain accurate predictions for future data by building models using past data.

This leads to the natural questions of

a) how do we quantify prediction accuracy, and b) how can we estimate what that accuracy will be for future data.

This introduces two fundamental notions in ML: the *performance metrics* and *performance estimation*.

Testing and validating Machine Learning systems

Performance Metrics

To determine how useful are ML models we are building we first need to specify a metric to quantify the accuracy of predictions.

Testing and validating Machine Learning systems

Performance Metrics

To determine how useful are ML models we are building we first need to specify a metric to quantify the accuracy of predictions.

This decision must be made specifically for the task we are targeting.

Testing and validating Machine Learning systems

Performance Metrics

Consider our insurance case again.

A natural metric is the *error rate*, the rate at which our system makes erroneous predictions.

Testing and validating Machine Learning systems

Performance Metrics

Consider our insurance case again.

A natural metric is the *error rate*, the rate at which our system makes erroneous predictions.

The first tree we built previously made wrong predictions about 27% of the time on the training set.

Testing and validating Machine Learning systems

Performance Metrics

But what if our interest in using our model is a bit more nuanced?

What if we wanted to make sure we are able to correctly identify almost all of our customers with insurance (say 95%) while minimizing *false positives*.

Testing and validating Machine Learning systems

Confusion Matrix

For classification, we use a more precise language to describe classification accuracy:

	True Class +	**True Class -**	**Total**
Predicted Class +	True Positive (TP)	False Positive (FP)	P*
Predicted Class -	False Negative (FN)	True Negative (TN)	N*
Total	P	N	

Testing and validating Machine Learning systems

Performance Metrics

Name	Definition	Synonyms
False Positive Rate (FPR)	FP / N	Type-I error, 1-Specificity
True Positive Rate (TPR)	TP / P	1 - Type-II error, power, sensitivity, **recall**
Positive Predictive Value (PPV)	TP / P*	**precision**, 1-false discovery proportion
Negative Predicitve Value (NPV)	FN / N*	

Testing and validating Machine Learning systems

Performance Metrics

In the insurance case we may want to

- increase **TPR** (recall, make sure we catch all customers with insurance)

Testing and validating Machine Learning systems

Performance Metrics

In the insurance case we may want to

- increase **TPR** (recall, make sure we catch all customers with insurance)
- at the expense of **FPR** (1-Specificity, customers we may incorrectly sell a particular product to because we think they have health insurance).

Testing and validating Machine Learning systems

Performance Metrics

Here is the confusion matrix for our first decision tree and the training data we used to train it:

	2. No	**1. Yes**
2. No	268	152
1. Yes	649	1931

Testing and validating Machine Learning systems

Performance Metrics

Here is the confusion matrix for our first decision tree and the training data we used to train it:

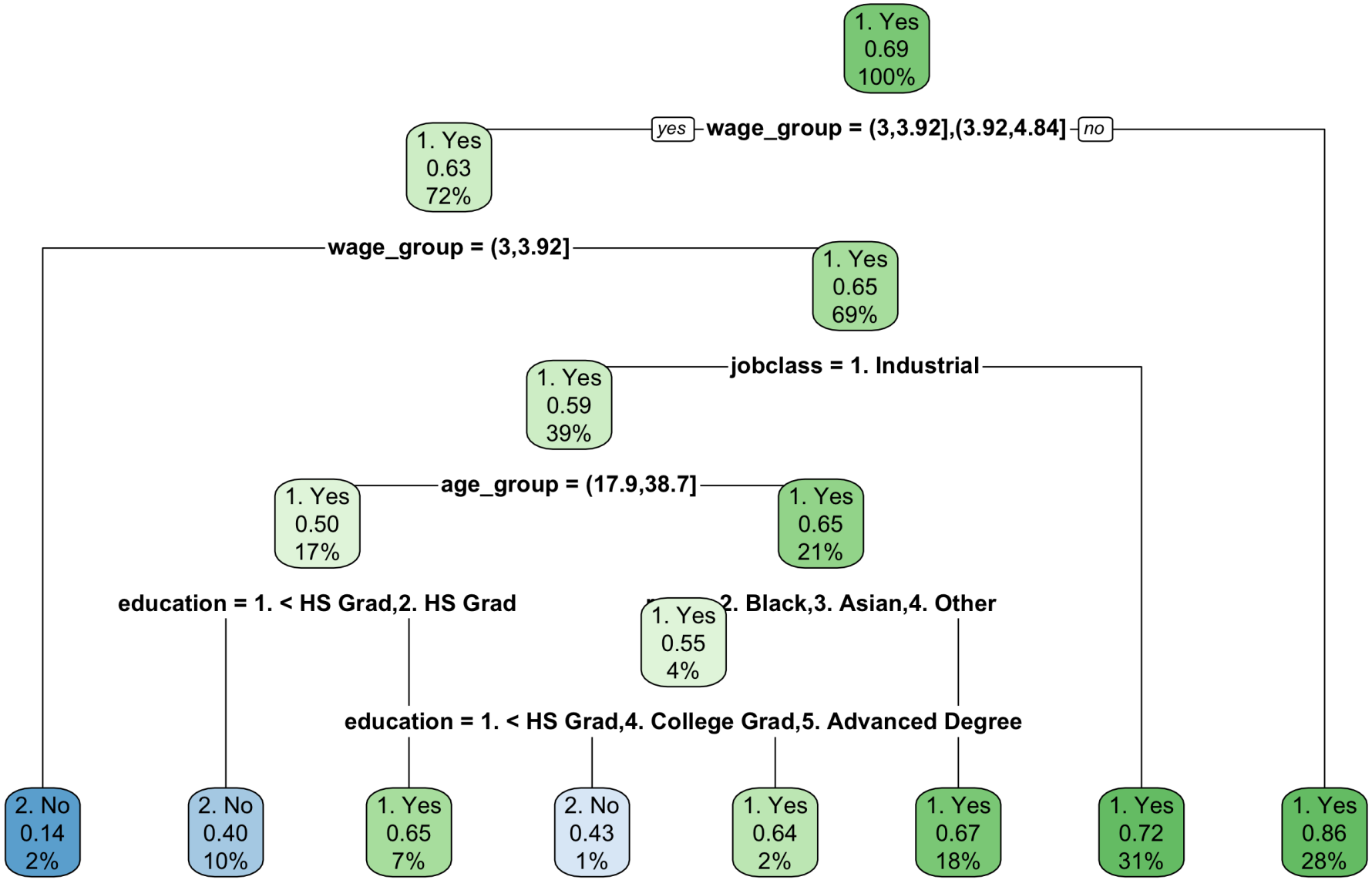
	2. No	1. Yes
2. No	268	152
1. Yes	649	1931

From this matrix you should be able to calculate all metrics defined above. What is this tree's recall (TPR)? It's False Positive Rate?

Testing and validating Machine Learning systems

Performance Metrics

This leads to a natural question: Can we adjust TPR and FPR for our predictors?


1. Yes
0.69
100%
yes
wage_group = (3,3.92],(3.92,4.84]
no
1. Yes
0.63
72%
wage_group = (3,3.92]
1. Yes
0.65
69%
jobclass = 1. Industrial
1. Yes
0.59
39%
age_group = (17.9,38.7]
1. Yes
0.50
17%
1. Yes
0.65
21%
education = 1. < HS Grad,2. HS Grad
2. Black,3. Asian,4. Other
1. Yes
0.55
4%
education = 1. < HS Grad,4. College Grad,5. Advanced Degree
2. No
0.14
2%
2. No
0.40
10%
1. Yes
0.65
7%
2. No
0.43
1%
1. Yes
0.64
2%
1. Yes
0.67
18%
1. Yes
0.72
31%
1. Yes
0.86
28%

Predictions at each leaf are made based on the majority label at that leaf.

Predictions at each leaf are made based on the majority label at that leaf.

So, if a leaf represents customers where a majority of them have insurance, then we would predict customers that are consistent with that leaf to have insurance.

What if we wanted to increase our TPR? How should we change our predictions?

What if we wanted to increase our TPR? How should we change our predictions?

We can do so based on a proportion cutoff instead of a simple majority.

What if we wanted to increase our TPR? How should we change our predictions?

We can do so based on a proportion cutoff instead of a simple majority.

Suppose we wanted to increase TPR, we could predict insurance if at least 30% of the customers represented by that leaf have insurance.

What if we wanted to increase our TPR? How should we change our predictions?

We can do so based on a proportion cutoff instead of a simple majority.

Suppose we wanted to increase TPR, we could predict insurance if at least 30% of the customers represented by that leaf have insurance.

This would of course increase TPR, but might decrease FPR as well.

Testing and validating Machine Learning systems

Performance Metrics

Since we can use different prediction cutoffs to get different TPR and FPR, comparing ML models becomes a bit more challenging.

We need a way of capturing the behavior of models across these different cutoffs.

Testing and validating Machine Learning systems

Performance Metrics

We can do this using a ROC curve. ROC refers to Receiver-Operator Characteristic curves, first used to describe the performance of radar operators in the military.

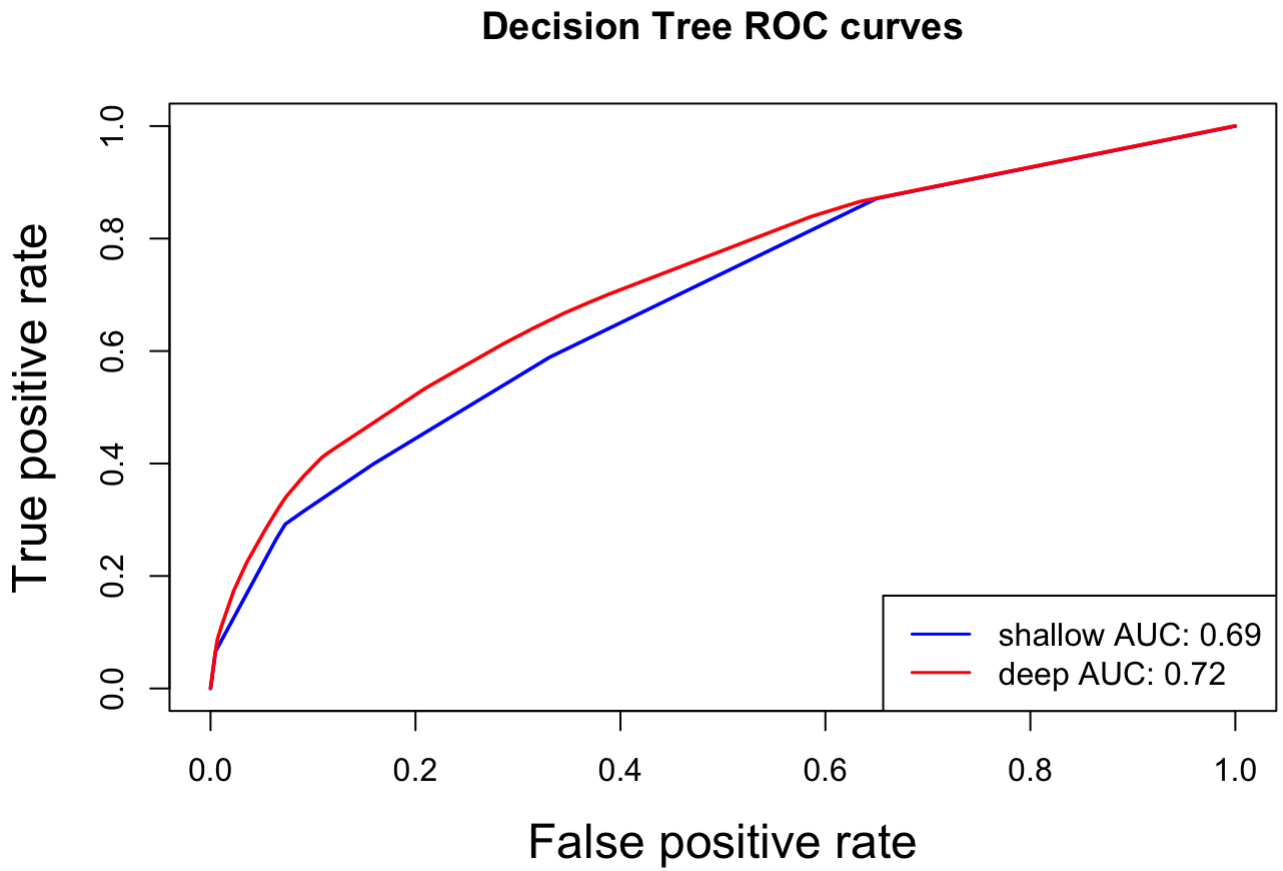
Testing and validating Machine Learning systems

Performance Metrics

We can do this using a ROC curve. ROC refers to Receiver-Operator Characteristic curves, first used to describe the performance of radar operators in the military.

Plots TPR and FPR at different cutoffs comparing the behavior of two predictors across the cutoff range.

Compare behavior of numerically based by computing the *area under the ROC curve*.



Testing and validating Machine Learning systems

Performance Metrics

For regression problems there is usually a bit less variety in performance metrics.

The most common metric is the root mean squared error (RMSE).

Testing and validating Machine Learning systems

Performance Estimation

Now that we established a *performance metric* the next question is how to *estimate performance* of ML systems on future data given the data we have at hand.

Testing and validating Machine Learning systems

Performance Estimation

Measuring performance of an ML system on the training data we used to create the model is not correct for two related reasons:

1) if we choose models based on performance on the training data we would be biased to choose models that *overfit* the training data

Testing and validating Machine Learning systems

Performance Estimation

Measuring performance of an ML system on the training data we used to create the model is not correct for two related reasons:

1) if we choose models based on performance on the training data we would be biased to choose models that *overfit* the training data

2) performance on the training data is not reflective of performance after we deploy the system where we make predictions on new unseen instances.

Testing and validating Machine Learning systems

Performance Estimation

The key here is to have access to *test data* and measure system performance on the test data.

However, it is important to remember to **never look at the test data** while developing or training the ML system.

Testing and validating Machine Learning systems

Performance Estimation

Here are some guidelines on how to use test data:

- **Ideal**: Use a *completely independent* dataset as a test set. This could be data that was measured on a different cohort, or data obtained in a second sampling window over the same cohort. Ideally, you would know what the schema of this data will be during system development but you would not have access to the data.

Testing and validating Machine Learning systems

Performance Estimation

Here are some guidelines on how to use test data:

- **Ideal**: Use a *completely independent* dataset as a test set.
- **Good**: Set aside a portion of your training data (say 20%-50% depending on how much training data you have) as test data, and *don't look at it again* until you have finished developing your system and are ready to measure performance before deployment.

Testing and validating Machine Learning systems

Performance Estimation

Here are some guidelines on how to use test data:

- **Ideal**: Use a *completely independent* dataset as a test set.
- **Good**: Set aside a portion of your training data (say 20%-50%)
- **Minimal Practice**: Use cross-validation to estimate performance.

Testing and validating Machine Learning systems

Cross-validation

1. Partition the training data into 10 groups (for example).
2. Now treat each group of training instances as a test set in turn:

Testing and validating Machine Learning systems

Cross-validation

1. Partition the training data into 10 groups (for example).
2. Now treat each group of training instances as a test set in turn:

- first, train the ML system using instances from all the other groups,

Testing and validating Machine Learning systems

Cross-validation

1. Partition the training data into 10 groups (for example).
2. Now treat each group of training instances as a test set in turn:

- first, train the ML system using instances from all the other groups,
- then, measure performance on set-aside instances.

Testing and validating Machine Learning systems

Cross-validation

1. Partition the training data into 10 groups (for example).
2. Now treat each group of training instances as a test set in turn:

- first, train the ML system using instances from all the other groups,
- then, measure performance on set-aside instances.

This will give you 10 (for example) measures of perforamance to compare the models using standard statistical comparison methods (e.g., statistical hypothesis testing).

Testing and validating Machine Learning systems

Cross-validation

The downside of this approach is that in practice people tend to do this cross-validation exercise multiple times during the development of the ML system which invalidates the maxim of *never looking at the test data*.

Testing and validating Machine Learning systems

Performance Estimation for Model Selection

We saw that many of these models have tuning parameters that we use to control over-fitting.

In the case of the decision tree, the depth of the tree controls over-fitting.

How do we determine which tuning parameters to use in our ML system?

Testing and validating Machine Learning systems

Performance Estimation for Model Selection

The same principle holds, you do not want to use tuning parameters that perform best on the *training data*.

In this case we use *validation data* to determine tuning set parameters to use.

Testing and validating Machine Learning systems

The ideal workflow would be as follows:

- Set aside *test data* that will be use to measure performance after system development is complete. Do not look at this data during system development.

Testing and validating Machine Learning systems

The ideal workflow would be as follows:

- Set aside *test data* that will be use to measure performance after system development is complete. Do not look at this data during system development.

- Use a cross-validation approach on the *training data* to determine which model and tuning parameters to use. In this case, the cross-validation approach is using *validation data* to measure performance.

Testing and validating Machine Learning systems

The ideal workflow would be as follows:

- Train the ML system using the full *training data* for the model and tuning parameters chosen by cross-validation.

Testing and validating Machine Learning systems

The ideal workflow would be as follows:

- Train the ML system using the full *training data* for the model and tuning parameters chosen by cross-validation.
- Once the ML system is trained, use the *test data* to measure and report performance of the system.

Summary

ML Systems that predict outcomes of future instances from past data

Summary

ML Systems that predict outcomes of future instances from past data

Challenges garbage in garbage out, over-fitting and under-fitting

Summary

ML Systems that predict outcomes of future instances from past data

Challenges garbage in garbage out, over-fitting and under-fitting

System Types supervised or unsupervised, batch or online, instance or model-based

Summary

ML Systems that predict outcomes of future instances from past data

Challenges garbage in garbage out, over-fitting and under-fitting

System Types supervised or unsupervised, batch or online, instance or model-based

Testing and validation performance metrics and performance evaluation

Setting up your learning system

We will use Python to experiment with algorithms and datasets in the course. Follow these instructions to set it up

- Download and install Python 3: https://www.python.org/downloads/
- Download and install the Anaconda scientific python distribution: https://www.continuum.io/downloads/

Setting up your learning system

- You can use anaconda to manage your python environment from the terminal or using the *Anaconda Navigator* graphical interface. For the latter, here is a quick guide: https://docs.continuum.io/anaconda/navigator/getting-started
- Install the modules we will be using in class: `jupyter`, `matplotlib`, `numpy`, `pandas`, `scipy`, `scikit-learn`