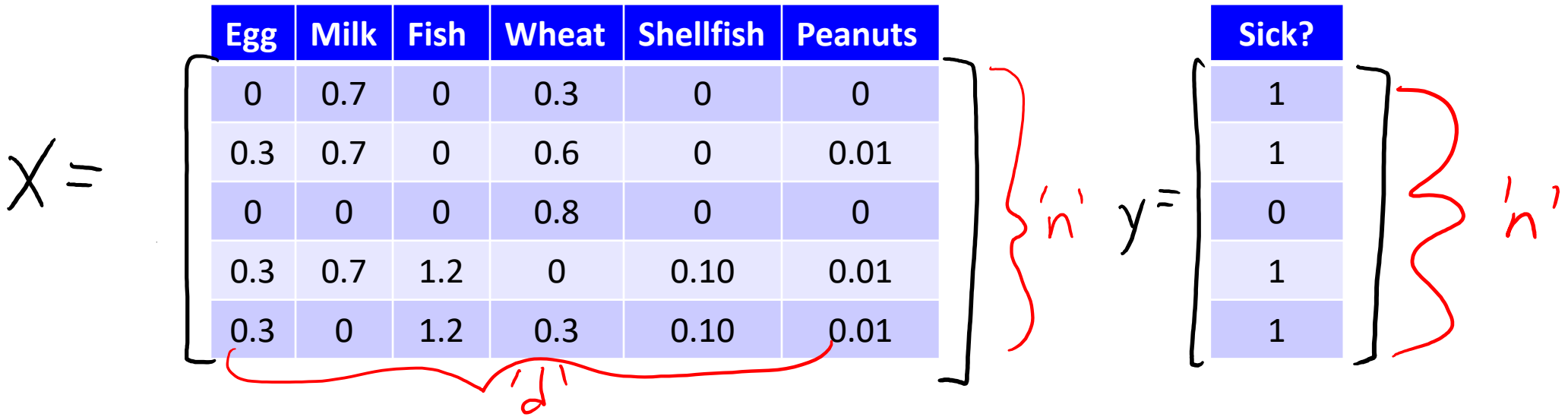
CPSC 340:
Machine Learning and Data Mining

Fundamentals of Learning

Admin

- Assignment 1 is due Monday: finish early!

Last Time: Supervised Learning Notation



- Feature matrix 'X' has rows as examples, columns as features.
  - x_{ij} is feature 'j' for example 'i' (quantity of food 'j' on day 'i').
  - x_i is the list of all features for example 'i' (all the quantities on day 'i').
  - x^j is column 'j' of the matrix (the value of feature 'j' across all examples).
- Label vector 'y' contains the labels of the examples.
  - y_i is the label of example 'i' (1 for "sick", 0 for "not sick").

Supervised Learning Application

- We motivated supervised learning by the “food allergy” example.

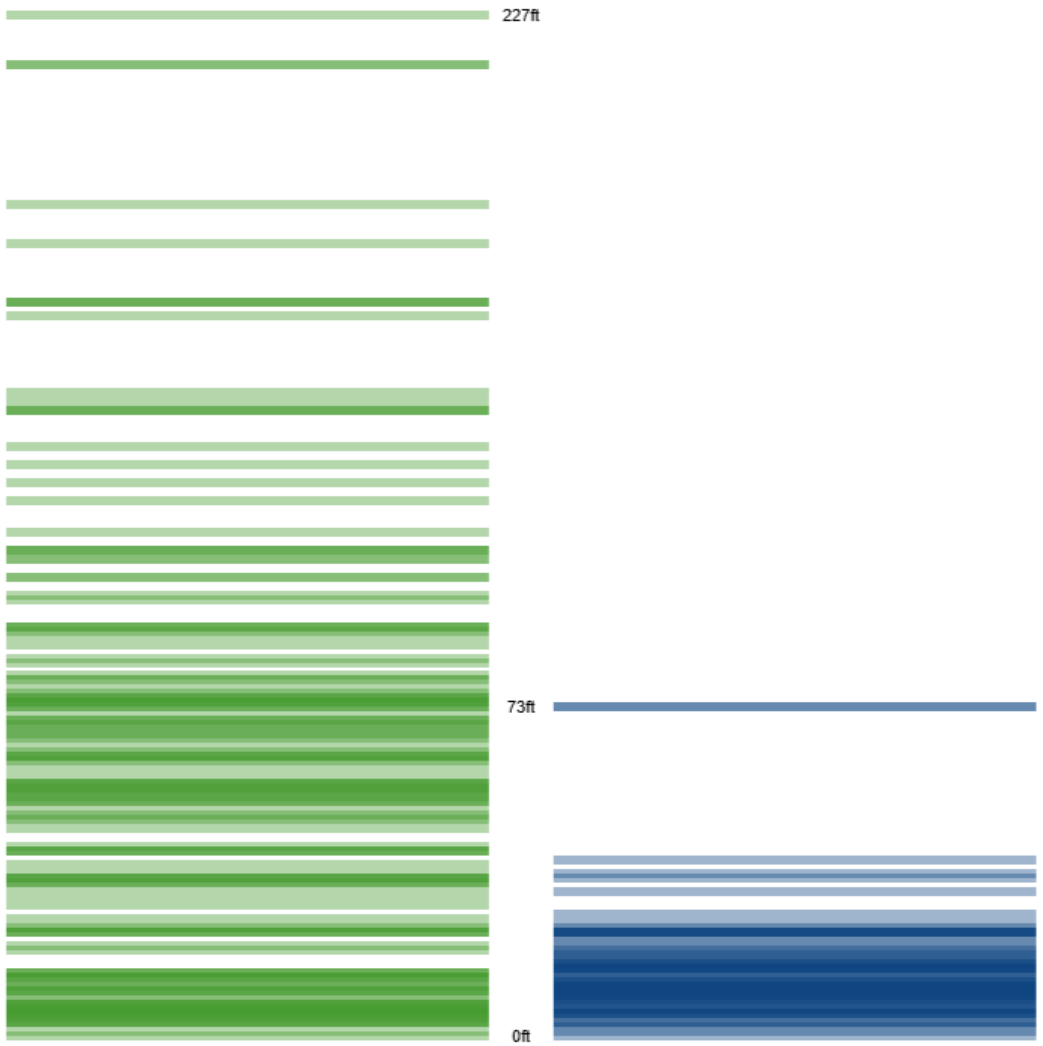
- But we can use supervised learning for any input:output mapping.
  - E-mail spam filtering.
  - Optical character recognition on scanners.
  - Recognizing faces in pictures.
  - Recognizing tumours in medical images.
  - Speech recognition on phones.
  - Your problem in industry/research?

Motivation: Determine Home City

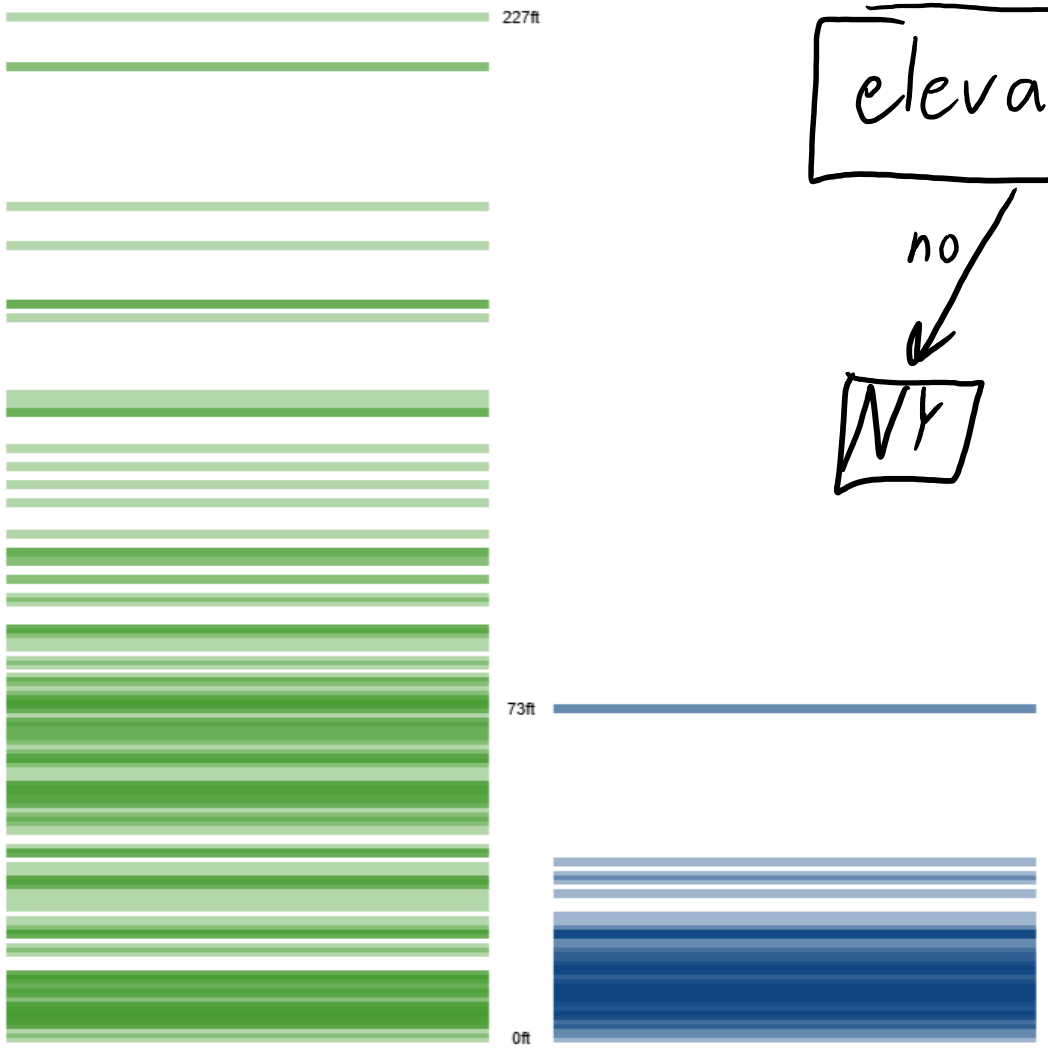
- We are given data from 248 homes.
- For each home/example, we have these features:
  - Elevation.
  - Year.
  - Bathrooms
  - Bedrooms.
  - Price.
  - Square feet.
- Goal is to build a program that predicts SF or NY.

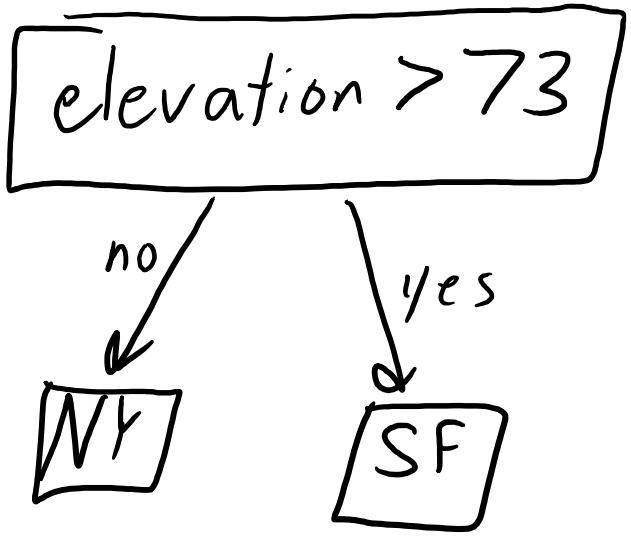
This example and images of it come from:
http://www.r2d3.us/visual-intro-to-machine-learning-part-1

Plotting Elevation

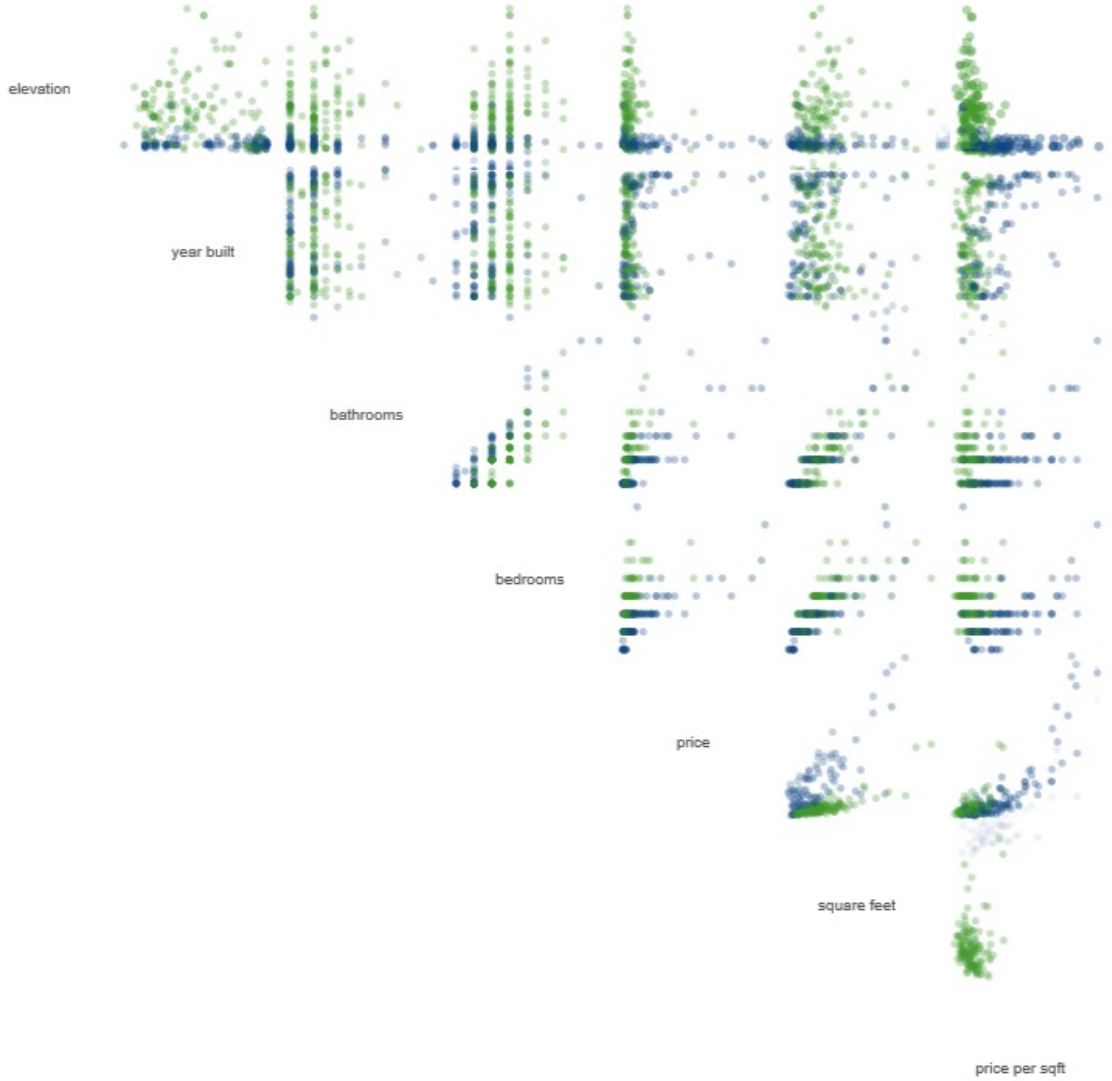


Simple Decision Stump

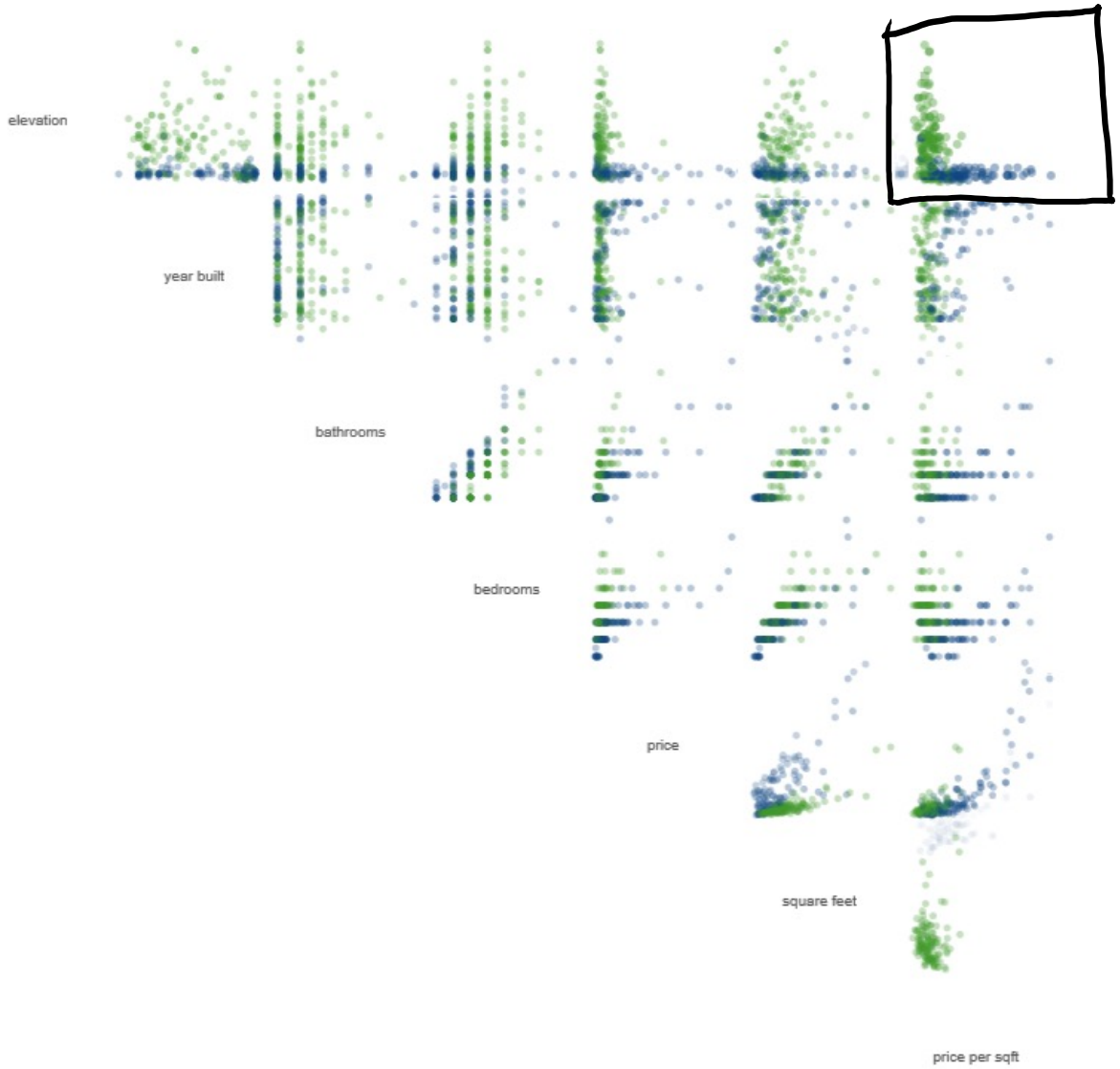




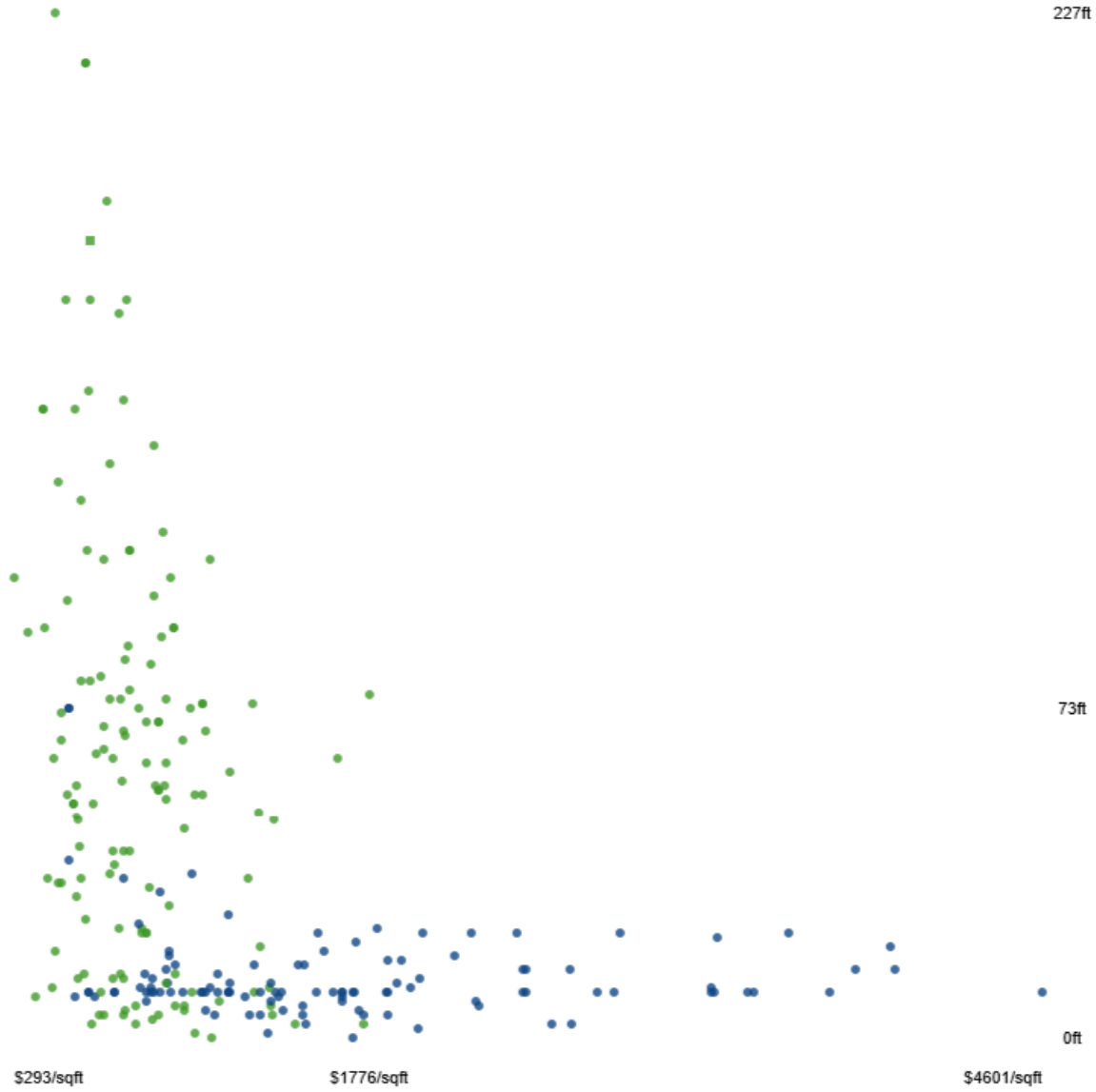
Scatterplot Array



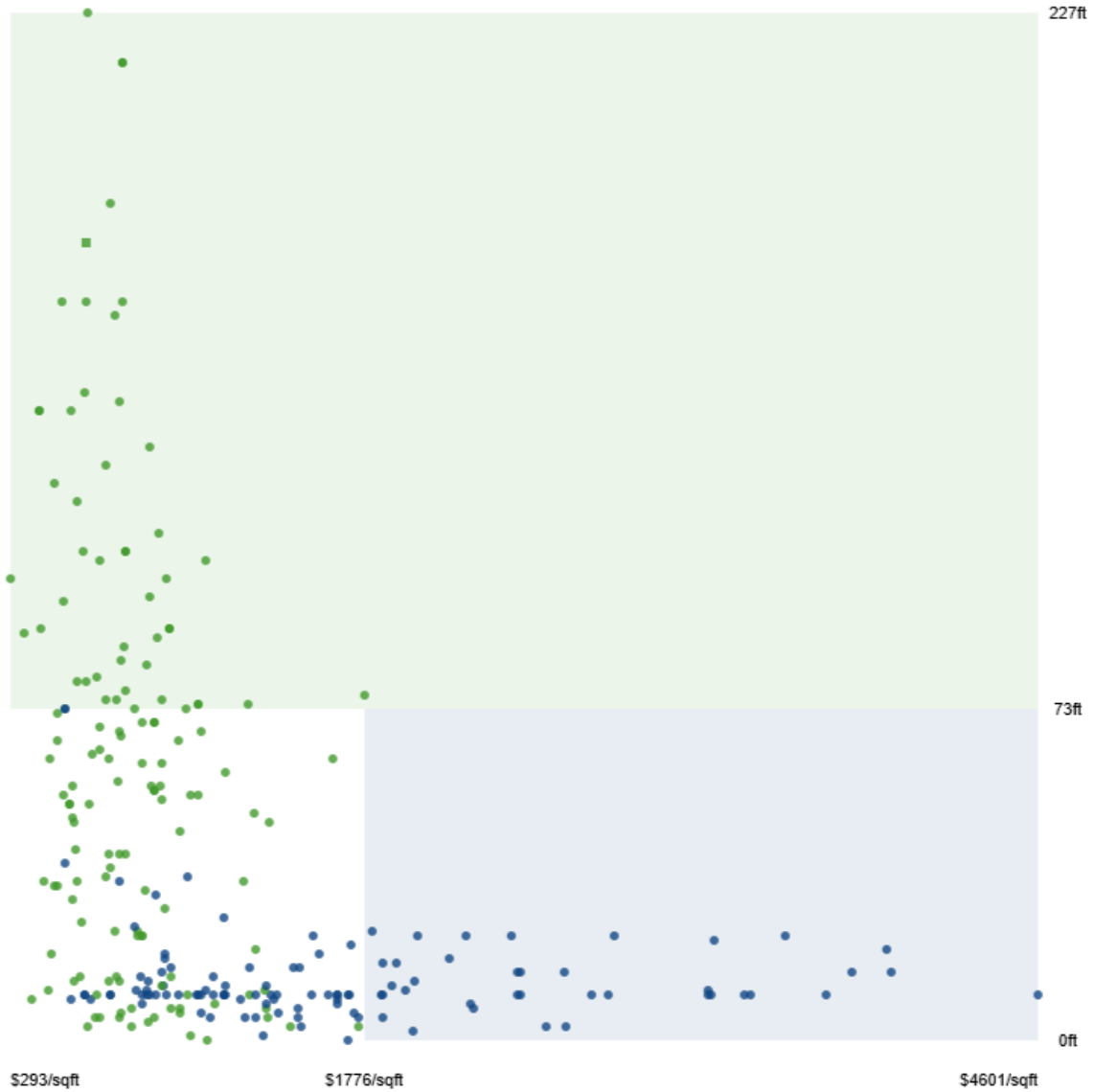
Scatterplot Array



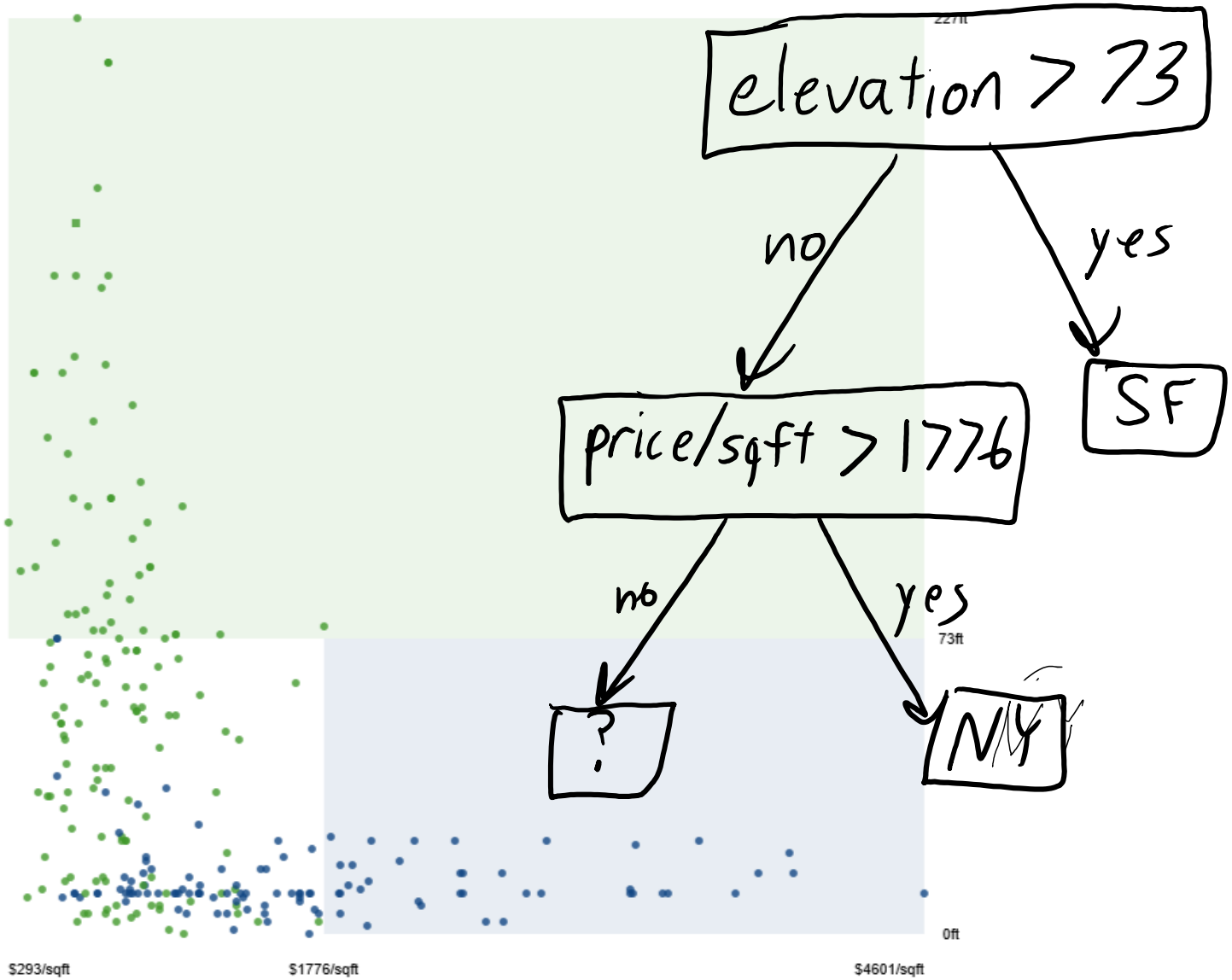
Plotting Elevation and Price/SqFt



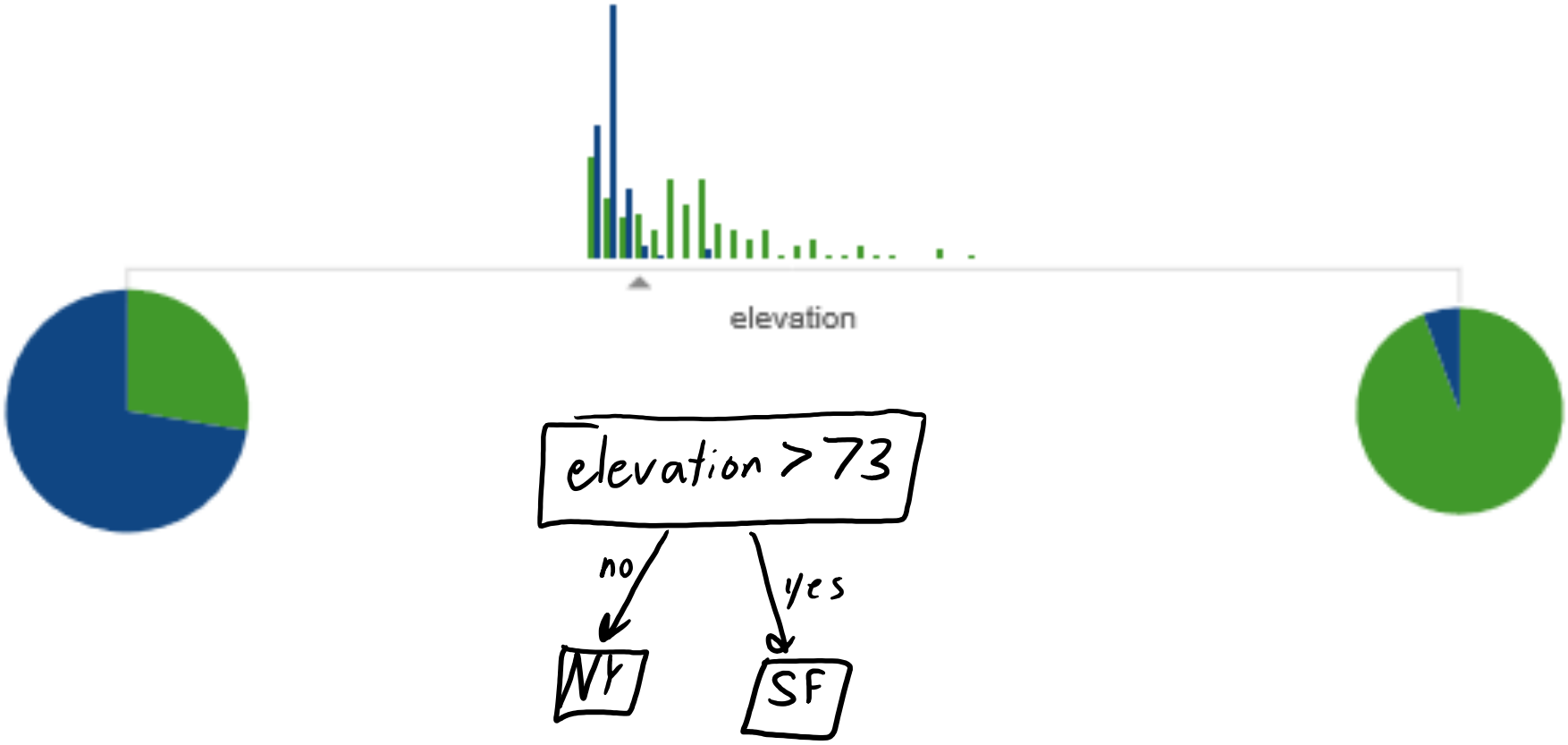
Simple Decision Tree Classification



Simple Decision Tree Classification

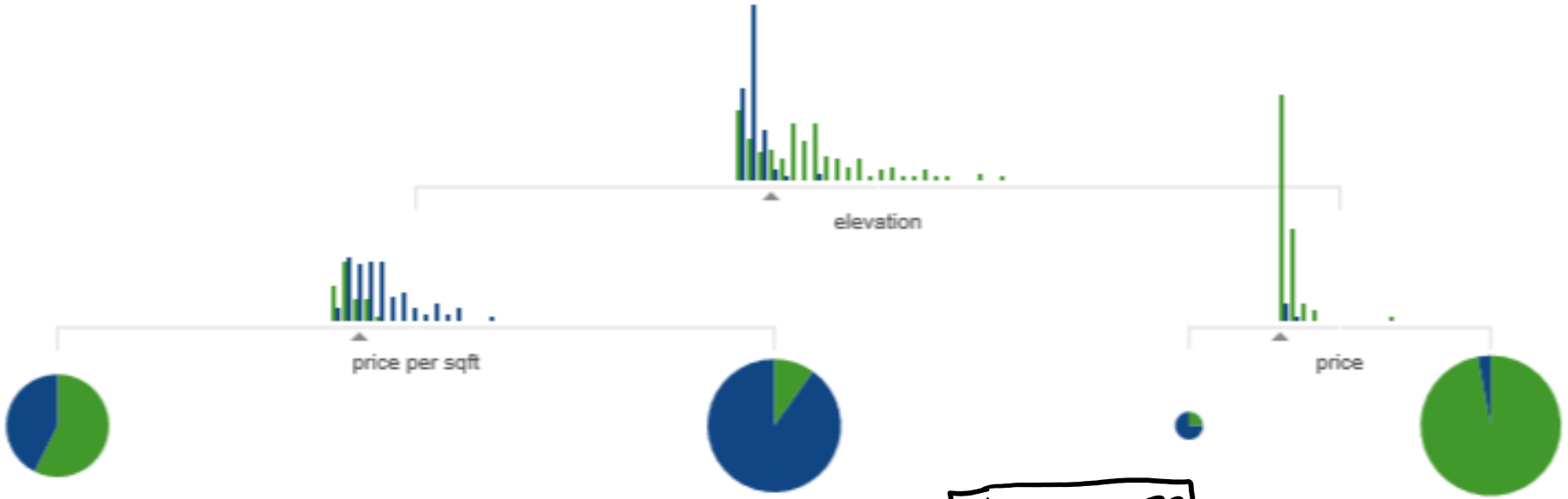


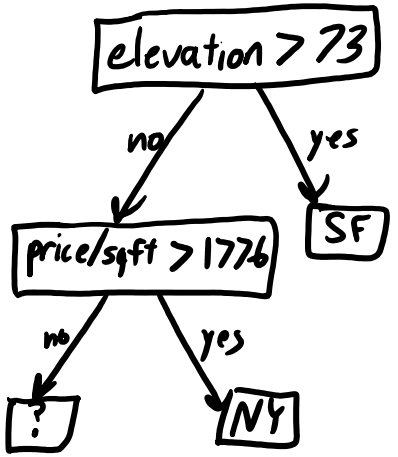
How does the depth affect accuracy?



This is a good start (> 75% accuracy).

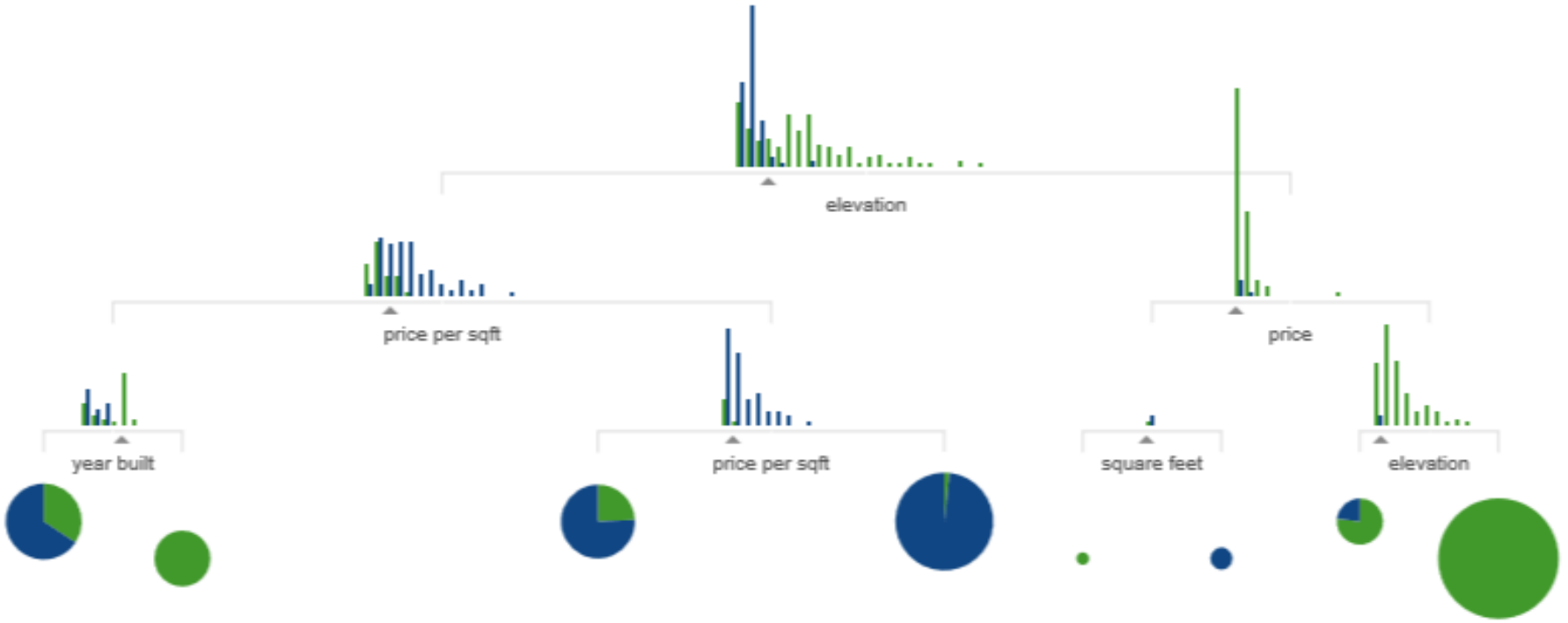
How does the depth affect accuracy?





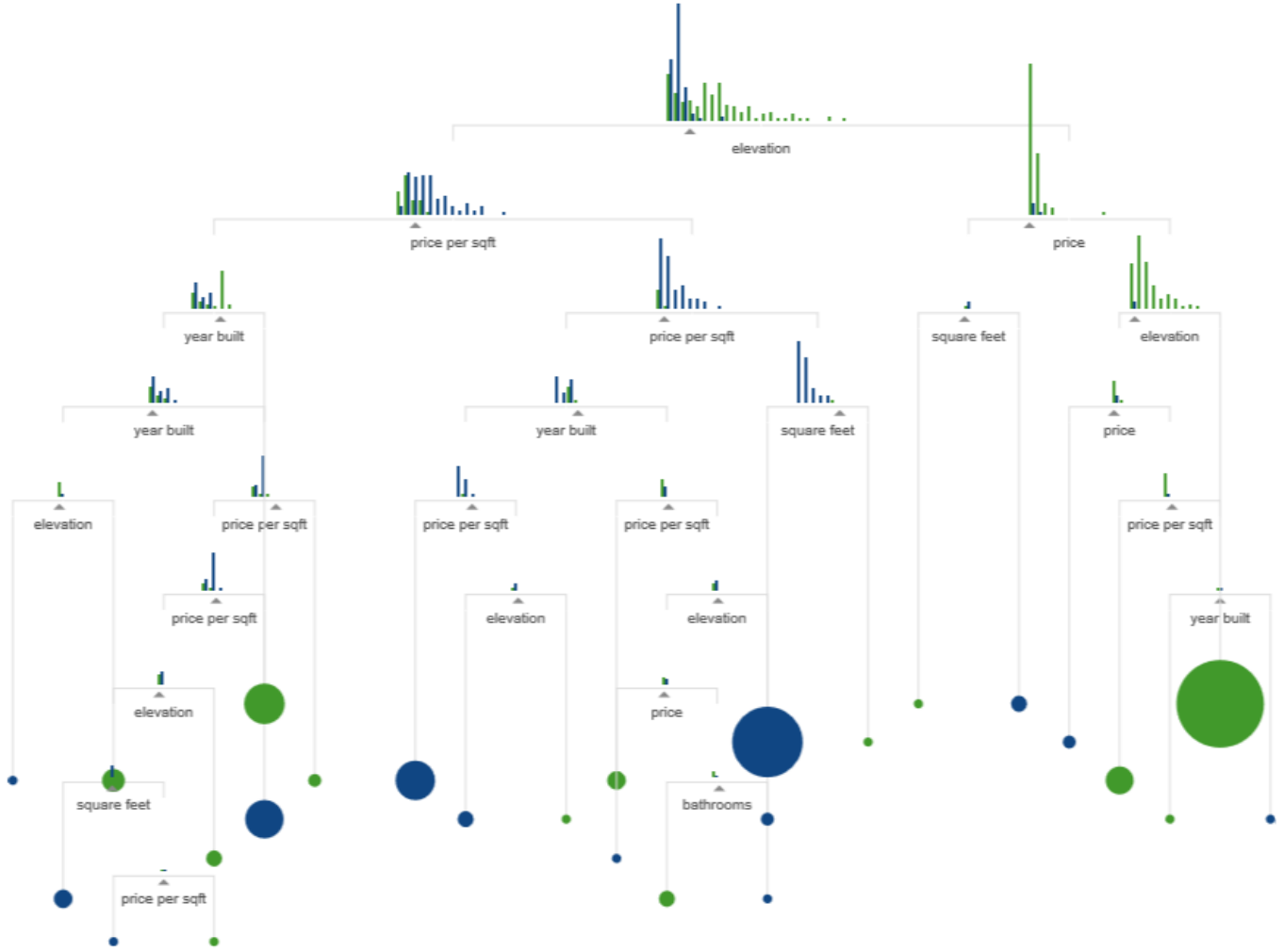
Start splitting the data recursively…

How does the depth affect accuracy?



Accuracy keeps increasing as we add depth.

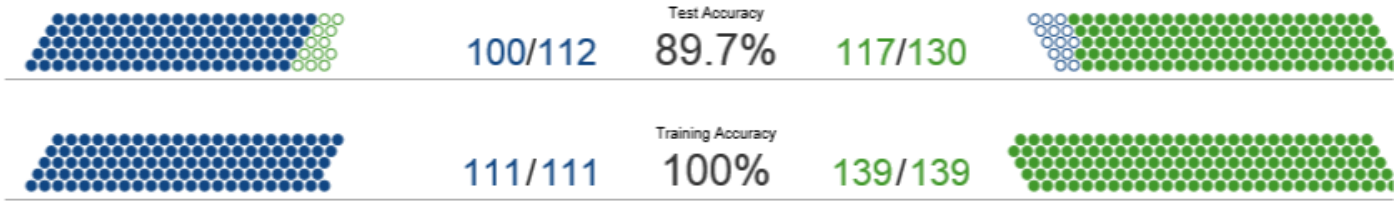
How does the depth affect accuracy?



Eventually, we can perfectly classify all of our data.

Training vs. Testing Error

- With this decision tree, 'training accuracy' is 1.
  - It perfectly labels the data we used to make the tree.
- We are now given features for 217 new homes.
- What is the 'testing accuracy' on the new data?
  - How does it do on data not used to make the tree?



- Overfitting: lower accuracy on new data.
  - Our rules got too specific to our exact training dataset.
  - Some of the "deep" splits only use a few examples (bad "coupon collecting").

Supervised Learning Notation

- Recall: we are given training data where we know labels:

$X =$ ($n \times d$)

Egg	Milk	Fish	Wheat	Shellfish	Peanuts	…
0	0.7	0	0.3	0	0	
0.3	0.7	0	0.6	0	0.01	
0	0	0	0.8	0	0	
0.3	0.7	1.2	0	0.10	0.01	
0.3	0	1.2	0.3	0.10	0.01	

$y =$ ($n \times 1$)

Sick?
1
1
0
1
1

- But there is also testing data we want to label:

There are two testing data → Those Comming From training

Real New

$\tilde{X} =$ ($t \times d$)

Egg	Milk	Fish	Wheat	Shellfish	Peanuts	…
0.5	0	1	0.6	2	1	
0	0.7	0	1	0	0	
3	1	0	0.5	0	0	

$\tilde{y} =$ ($t \times 1$)

Sick?
?
?
?

Supervised Learning Notation

data

- Typical supervised learning steps:
  1. Build model based on training data X and y (training phase).
  2. Model makes predictions $\hat{y}$ on test data $\tilde{X}$ (testing phase).

- Instead of training error, consider test error:
  - Are predictions $\hat{y}$ similar to true unseen labels $\tilde{y}$?

Goal of Machine Learning

- In machine learning:
  - Goal is to do well on the test error!
- Midterm analogy:
  - Training error: how you do on the practice midterm.
  - Test error: how you do on the actual midterm.
  - Goal: do well on actual midterm, not the practice one.
- Memorization vs learning:
  - Can do well on training data by memorizing it.
  - You've only learned if you can do well in new situations.

Key principle: "do not let the test data influence the training"

- We care about is test error, but we need to follow a key principle:
  - THE TEST DATA CANNOT INFLUENCE THE TRAINING PHASE IN ANY WAY.

- We are measuring test error to see how well we do on new data:
  - If test data influences training, test error is not on "new" data.
  - You can start to overfit if you use it during training.

- Note that you can make this mistake unintentionally!

Key principle: "do not let the test data influence the training"

- We care about is test error, but we need to follow a key principle:
  - THE TEST DATA CANNOT INFLUENCE THE TRAINING PHASE IN ANY WAY.

Tom Simonite
June 4, 2015

Why and How Baidu Cheated an Artificial Intelligence Test

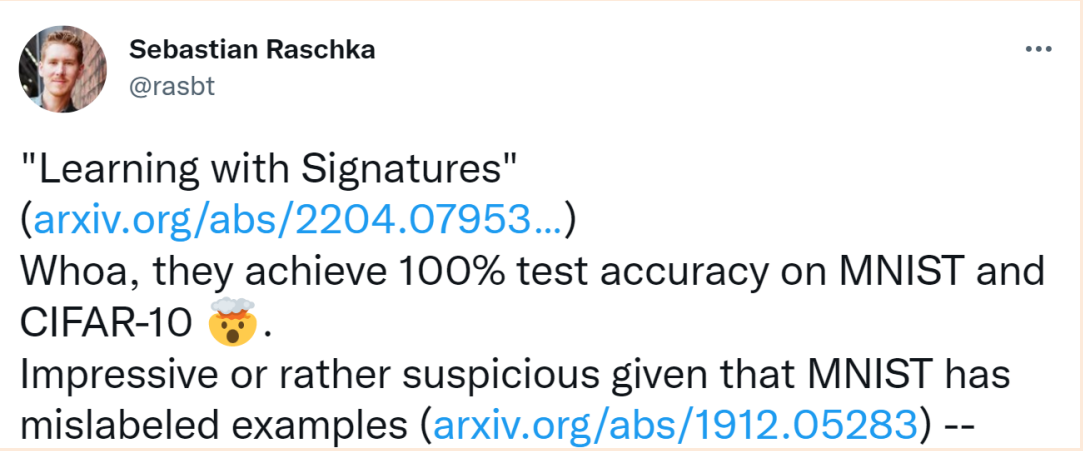
Machine learning gets its first cheating scandal.

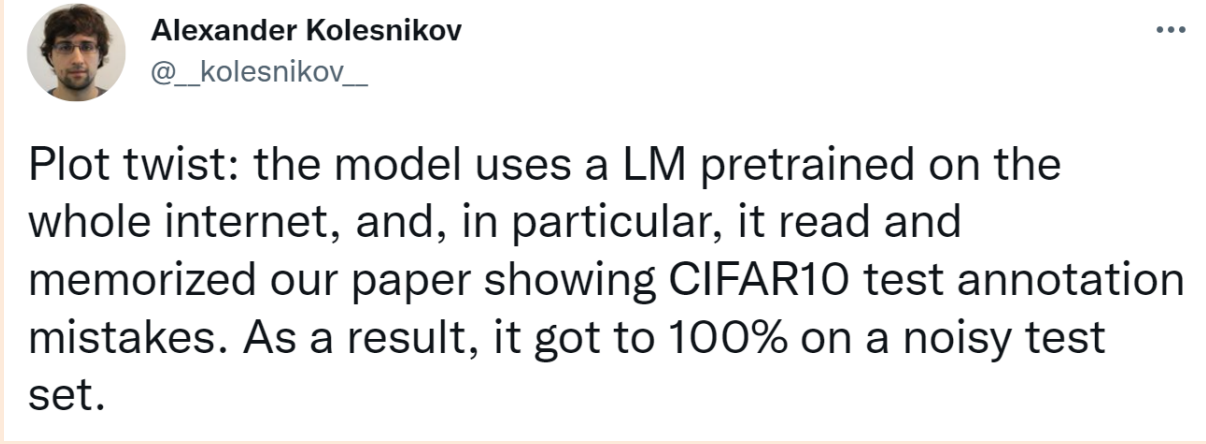
The sport of training software to act intelligently just got its first cheating scandal. Last month Chinese search company Baidu announced that its image recognition software had inched ahead of Google's on a standardized

http://www.technologyreview.com/view/538111/why-and-how-baidu-cheated-an-artificial-intelligence-test/

Key principle: “do not let the test data influence the training”

- We care about is test error, but we need to follow a key principle:
  - THE TEST DATA CANNOT INFLUENCE THE TRAINING PHASE IN ANY WAY.

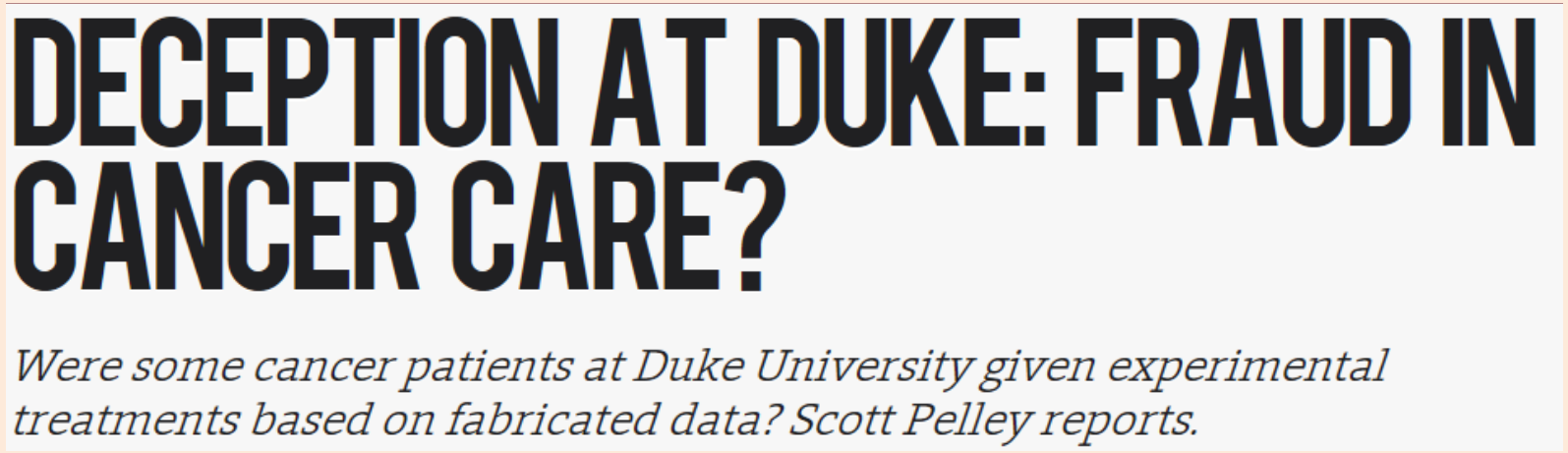




Key principle: "do not let the test data influence the training"

- We care about is test error, but we need to follow a key principle:
  - THE TEST DATA CANNOT INFLUENCE THE TRAINING PHASE IN ANY WAY.

- You also shouldn't change the test set to get the result you want.



  - http://blogs.sciencemag.org/pipeline/archives/2015/01/14/the_dukepotti_scandal_from_the_inside

https://www.cbsnews.com/news/deception-at-duke-fraud-in-cancer-care/

Digression: Key Principle and Hypothesis Testing

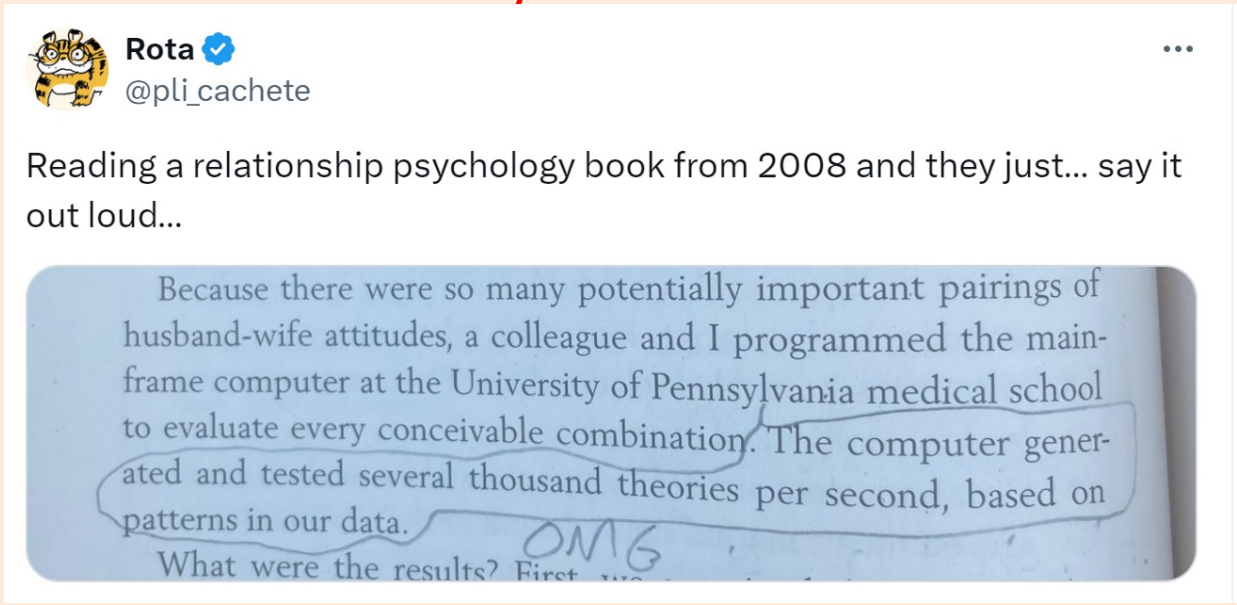
- Key principle frequently violated in hypothesis testing in many fields.
  - Data that you collect can't influence the hypotheses that you test.

- EXTREMELY COMMON and a MAJOR PROBLEM, coming in many forms:
  - Collect more data until you coincidentally get significance level you want.
  - Try different ways to measure performance, choose the one that looks best.
  - Choose a different type of model/hypothesis after looking at the test data.

- If you want to modify your hypotheses, you need to test on new data.
  - Or at least be aware and honest about this issue when reporting results.

Digression: Key Principle and Hypothesis Testing

- Key principle frequently violated in hypothesis testing in many fields:
  - Data that you collect can't influence the hypotheses that you test.



- If you want to modify your hypotheses, you need to test on new data.
  - Or at least be aware and honest about this issue when reporting results.

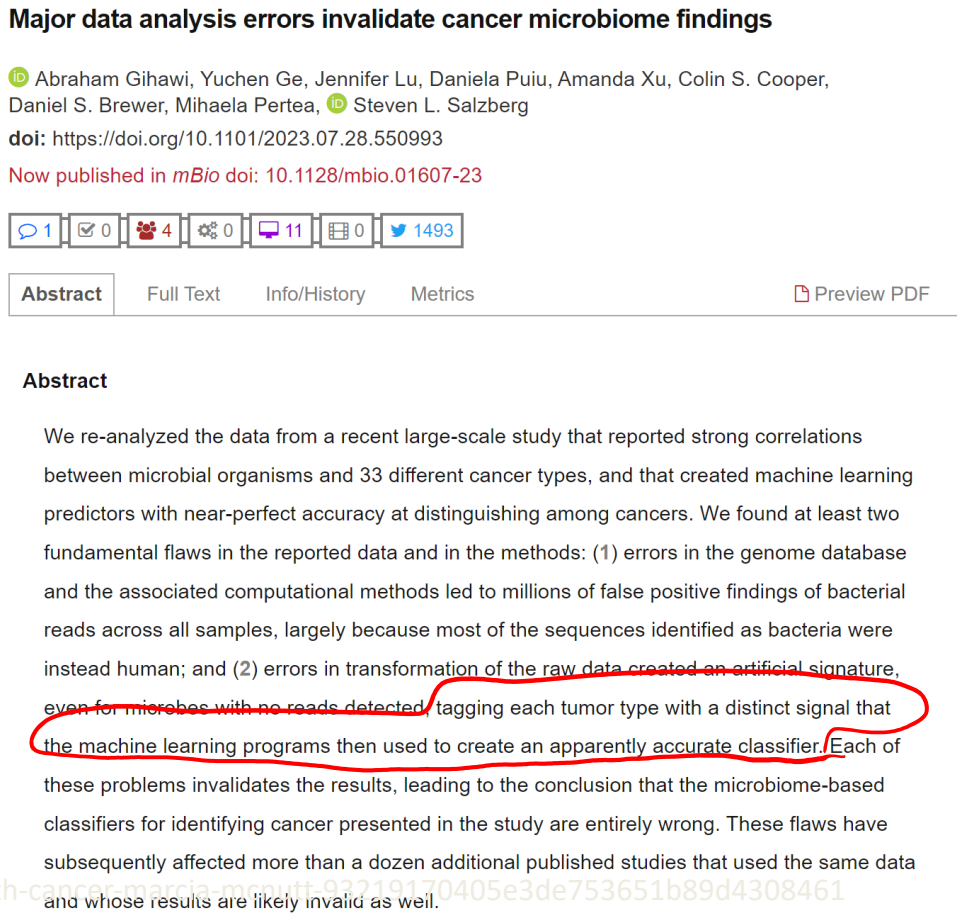
https://apnews.com/article/science-business-health-cancer-marcia-mcnutt-93219170405e3de753651b89d4308461

Digression: Key Principle and Hypothesis Testing

- Related reading:
  - "Replication crisis in Science".
  - "Why Most Published Research Findings are False".
  - "False-Positive Psychology: Undisclosed Flexibility in Data Collection and Analysis Allows Presenting Anything as Significant".
  - "HARKing: Hypothesizing After the Results are Known".
  - "Hack Your Way To Scientific Glory".
  - "Estimating the reproducibility of psychological science"
  - "The ASA's Statement on p-Values: Context, Process, and Purpose"
  - "Psychology's Replication Crisis Has Made The Field Better" (some solutions).
  - "Scientists rise up against statistical significance" (for the opposite direction).
  - "Steampunk Data Science" (figuring out nutrition/vitamins *before* p-values).

"Features Should Not Have Labels Included"

- Related principle: do not make features depending on the labels.
  - Because you will not have labels on new data.



Major data analysis errors invalidate cancer microbiome findings

Abraham Gihawi, Yuchen Ge, Jennifer Lu, Daniela Puiu, Amanda Xu, Colin S. Cooper, Daniel S. Brewer, Mihaela Pertea, Steven L. Salzberg

doi: https://doi.org/10.1101/2023.07.28.550993

Now published in *mBio* doi: 10.1128/mbio.01607-23

1 0 4 0 11 0 1493

Abstract Full Text Info/History Metrics Preview PDF

Abstract

We re-analyzed the data from a recent large-scale study that reported strong correlations between microbial organisms and 33 different cancer types, and that created machine learning predictors with near-perfect accuracy at distinguishing among cancers. We found at least two fundamental flaws in the reported data and in the methods: (1) errors in the genome database and the associated computational methods led to millions of false positive findings of bacterial reads across all samples, largely because most of the sequences identified as bacteria were instead human; and (2) errors in transformation of the raw data created an artificial signature, even for microbes with no reads detected, tagging each tumor type with a distinct signal that the machine learning programs then used to create an apparently accurate classifier. Each of these problems invalidates the results, leading to the conclusion that the microbiome-based classifiers for identifying cancer presented in the study are entirely wrong. These flaws have subsequently affected more than a dozen additional published studies that used the same data and whose results are likely invalid as well.

when things get too good to be true; be suspicious.

https://apnews.com/article/science-business-health-cancer-marcia-mcnutt-93219170405e3de753651b89d4308461

Is Learning Possible?

- Does training error say anything about test error?
  - In general, NO: Test data might have nothing to do with training data.
  - E.g., "adversary" takes training data and flips all labels.

X =

Egg	Milk	Fish
0	0.7	0
0.3	0.7	1
0.3	0	0

y =

Sick?
1
1
0

$\tilde{X}$ =

Egg	Milk	Fish
0	0.7	0
0.3	0.7	1
0.3	0	0

$\tilde{y}$ =

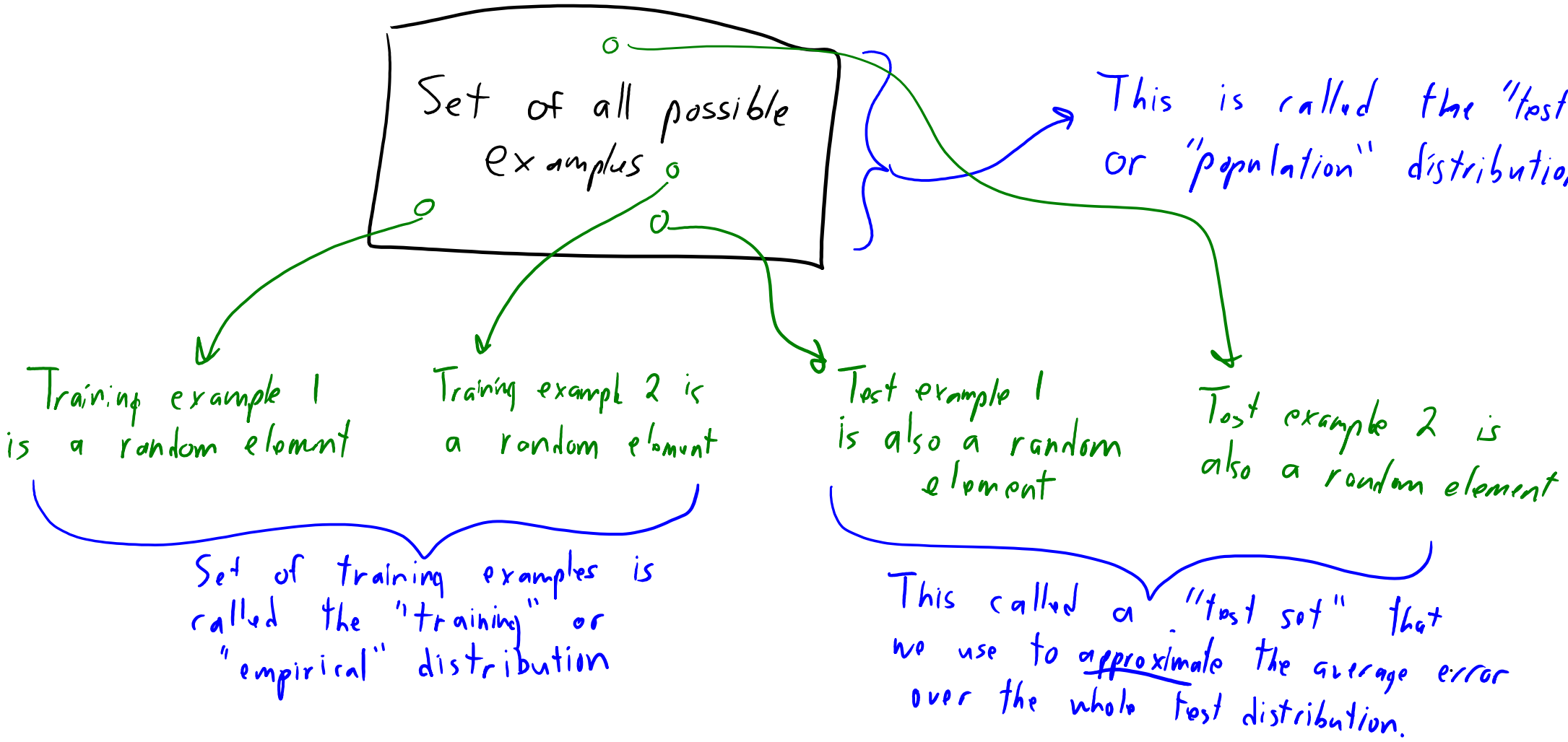
Sick?
0
0
1

- In order to learn, we need assumptions:
  - The training and test data need to be related in some way.
  - Most common assumption: independent and identically distributed (IID).

Identically distributed!

Data-Generating Distribution

- Informally: IID assumption is that data randomly comes from "same place":



IID Assumption

- Training/test data is independent and identically distributed (IID) if:
  - All examples come from the same distribution (identically distributed).
  - The example are sampled independently (order doesn't matter).

Row 1 comes from same distribution as rows 2-3.

Age	Job?	City	Rating	Income
23	Yes	Van	A	22,000.00
23	Yes	Bur	BBB	21,000.00
22	No	Van	CC	0.00
25	Yes	Sur	AAA	57,000.00

Row 4 does not depend on values in rows 1-3.

- Examples in terms of cards:
  - Pick a card, put it back in the deck, re-shuffle, repeat. → IID
  - Pick a card, put it back in the deck, repeat.
  - Pick a card, don't put it back, re-shuffle, repeat. } Not IID

IID Assumption and Food Allergy Example

- Is the food allergy data IID?
  - Do all the examples come from the same distribution?
  - Does the order of the examples matter?
- No!
  - Being sick might depend on what you ate yesterday (not independent).
  - Your eating habits might changed over time (not identically distributed).
- What can we do about this?
  - Just ignore that data isn't IID and hope for the best?
  - For each day, maybe add the features from the previous day?
  - Maybe add time as an extra feature?

IID Assumption and Bad "Medical AI"

- Suppose you want to detect a specific type of cancer.
  - You collect measurements from hospital patients having this cancer.
  - You collect measurements from healthy UBC students.
  - You build a classifier that distinguishes these groups with 100% accuracy.

- Success?

- Classifier might just detect UBC students from hospital patients, and nothing specifically related to the cancer.
  - IID assumption violations are a key cause of failure in ML applications.

IID Assumption and Bad "Medical AI"

- Related: including data from same patient in train and test data.
  - Over-estimates performance:
    - "Test" information used in training.

Voodoo Machine Learning for Clinical Predictions

Sohrab Saeb, Luca Lonini, Arun Jayaraman, David C. Mohr, Konrad P. Kording

doi: https://doi.org/10.1101/059774

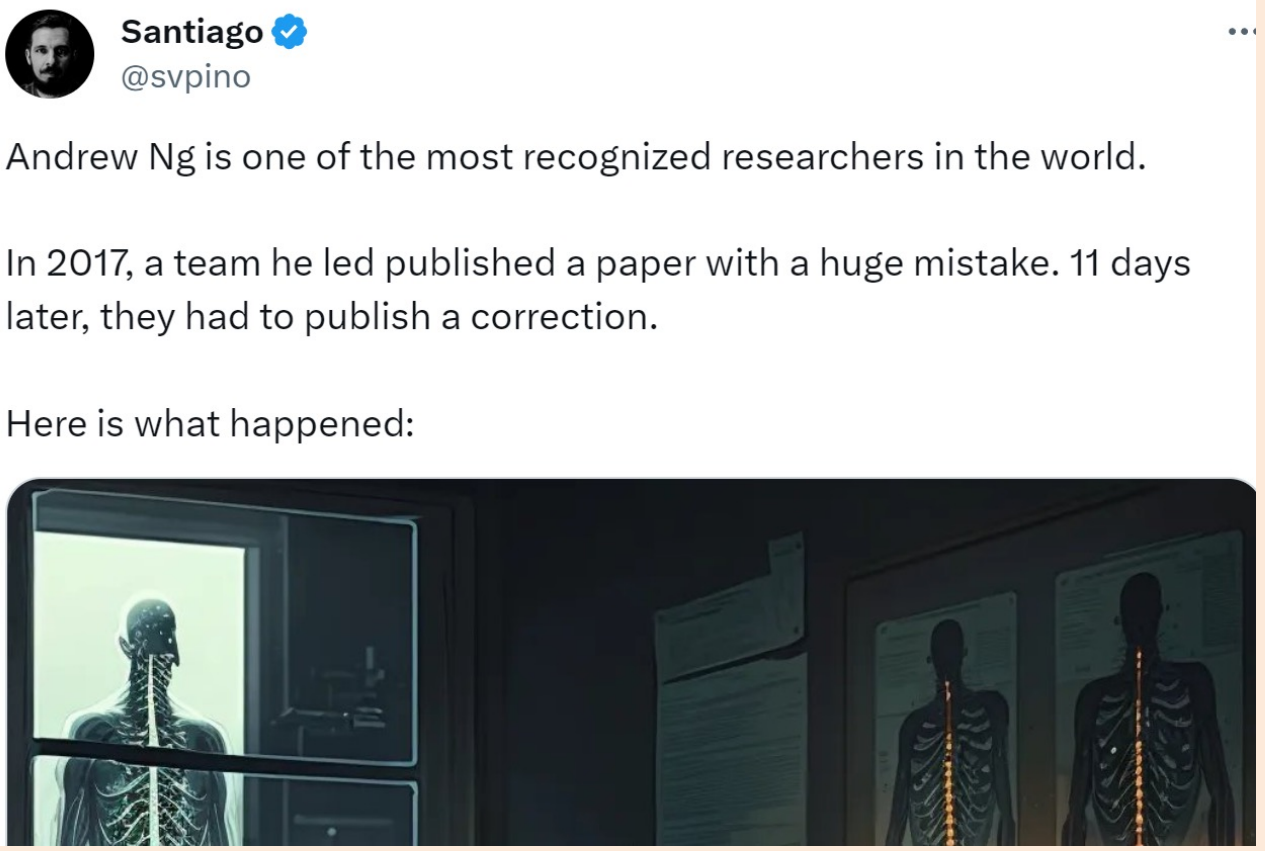
Now published in *GigaScience* doi: 10.1093/gigascience/gix019

8 | 0 | 0 | 0 | 7 | 0 | 150

Abstract | Full Text | Info/History | Metrics | Preview PDF

Abstract

The availability of smartphone and wearable sensor technology is leading to a rapid accumulation of human subject data, and machine learning is emerging as a technique to map that data into clinical predictions. As machine learning algorithms are increasingly used to support clinical decision making, it is important to reliably quantify their prediction accuracy. Cross-validation is the standard approach for evaluating the accuracy of such algorithms; however, several cross-validations methods exist and only some of them are statistically meaningful. Here we compared two popular cross-validation methods: record-wise and subject-wise. Using both a publicly available dataset and a simulation, we found that record-wise cross-validation often massively overestimates the prediction accuracy of the algorithms. We also found that this erroneous method is used by almost half of the retrieved studies that used accelerometers, wearable sensors, or smartphones to predict clinical outcomes. As we move towards an era of machine learning based diagnosis and treatment, using proper methods to evaluate their accuracy is crucial, as erroneous results can mislead both clinicians and data scientists.



https://twitter.com/svpino/status/1651556823428911104

Learning Theory

- Why does the IID assumption make learning possible?
  - Patterns in training examples are likely to be the same in test examples.
- The IID assumption is rarely true:
  - But it is often a good approximation.
  - There are other possible assumptions.
- Also, we're assuming IID across examples but not across features.

- Learning theory explores how training error is related to test error.
- We'll look at a simple example, using this notation:
  - E_{train} is the error on training data.
  - E_{test} is the error on testing data.

Fundamental Trade-Off

- Start with $E_{test} = E_{test}$, then add and subtract E_{train} on the right:

$$E_{test} = (E_{test} - E_{train}) + E_{train}$$

"test error" "generalization gap" E_{gap} "training error"

- How does this help?
  - If E_{gap} is small, then E_{train} is a good approximation to E_{test}.
- What does E_{gap} ("amount of overfitting") depend on?
  - It tends to get smaller as 'n' gets larger.
  - It tends to grow as model get more "complicated".

Training set

Fundamental Trade-Off

- This leads to a fundamental trade-off:
  1. E_{train}: how small you can make the training error.
  vs.
  2. E_{gap}: how close training error is to test error.

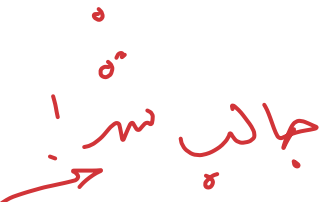
Bias

Variance

$$E_{test} = (E_{test} - E_{train}) + E_{train}$$

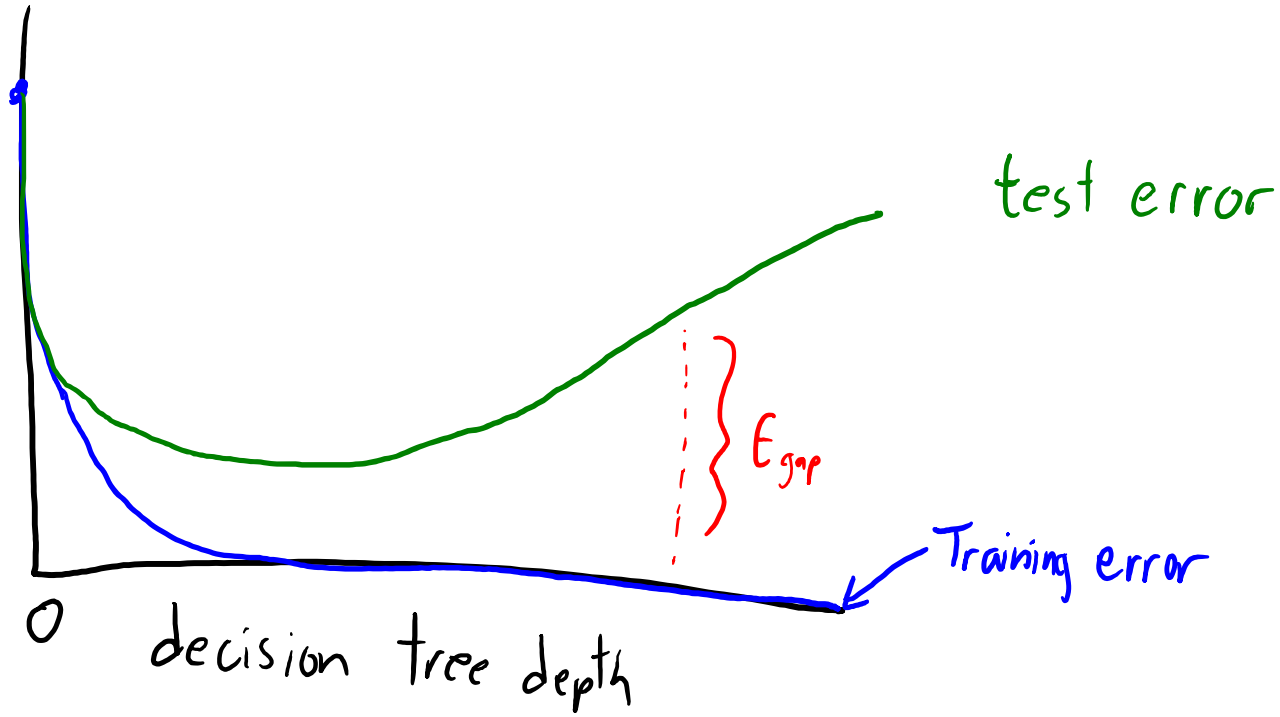
"test error" "generalization gap" "training error"
E_{gap}

- Simple models (like decision stumps):
  - E_{gap} is low (not very sensitive to training set).
  - But E_{train} might be high.
- Complex models (like deep decision trees):
  - E_{train} can be low.
  - But E_{gap} might be high (very sensitive to training set).



Fundamental Trade-Off

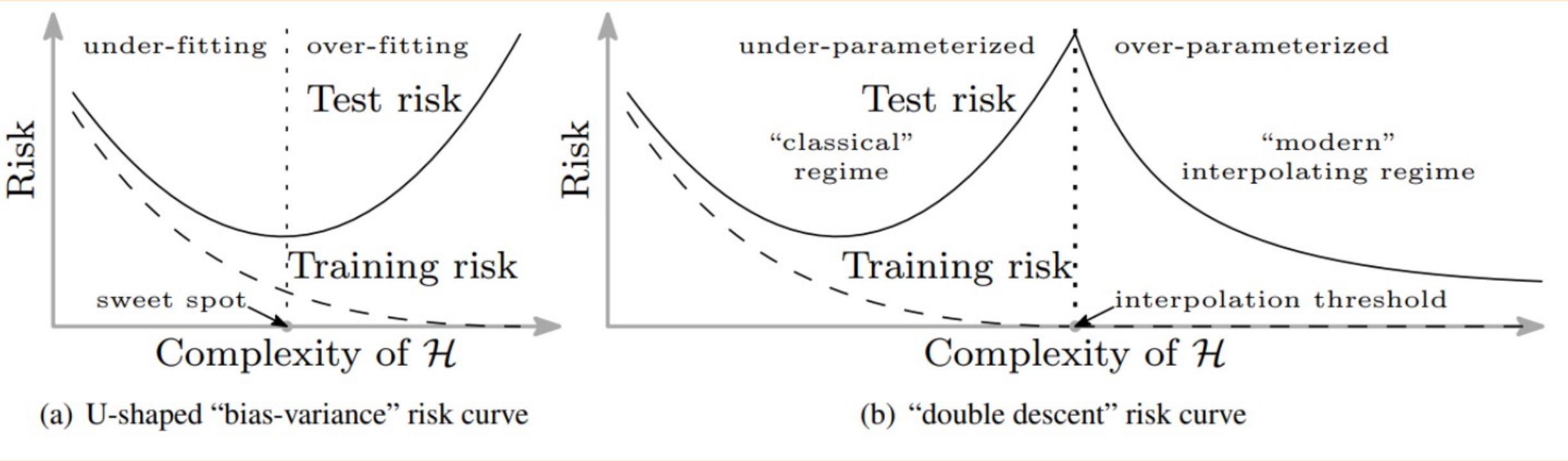
- Training error vs. test error for choosing depth:
  - Training error is high for low depth (underfitting)
  - Training error gets better with depth.
  - Test error initially goes down, but can eventually increase (overfitting).



bonus!

"Double Descent"

- Not always the whole story!



(a) U-shaped "bias-variance" risk curve

(b) "double descent" risk curve

CPSC 532S: Modern Statistical Learning Theory

Instructor: Danica Sutherland - ICICS X563, dsuth@cs.ubc.ca.
Lecture info: Mondays/Wednesdays, 13:30 - 15:00, DMP 101.
Term: 2021-22 W2 (January – April 2022).

Validation Error

- How do we decide decision tree depth?
- We care about test error.
- But we can't look at test data.
- So what do we do?????

- One answer: Use part of the training data to approximate test error.
- Split training examples into training set and validation set:
  - Train model based on the training data.
  - Test model based on the validation data.

Validation Error

$$X = \begin{bmatrix} X_{train} \\ \hline X_{validate} \end{bmatrix} \quad y = \begin{bmatrix} y_{train} \\ \hline y_{validate} \end{bmatrix} \begin{matrix} \} \text{"train"} \\ \} \text{"validation"} \end{matrix}$$

Step 1 is training: $model = train(X_{train}, y_{train})$

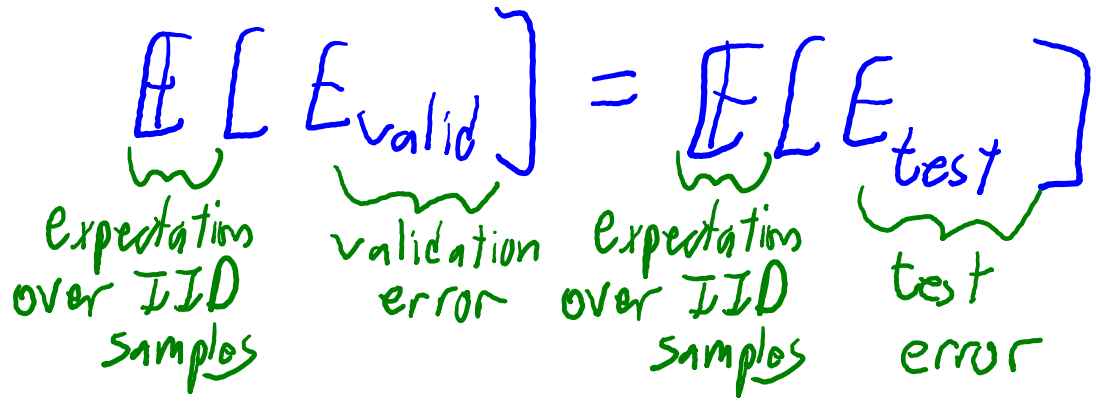
Step 2 is predicting: $\hat{y} = predict(model, X_{validate})$

Step 3 is validating: $error = sum(\hat{y} \neq y_{validate})$

Note: if examples are ordered, split should be random.

Validation Error

- IID data: validation error is unbiased approximation of test error.



- Midterm analogy:
  - You have 2 practice midterms.
  - You hide one midterm, and spend a lot of time working through the other.
  - You then do the other practice term, to see how well you'll do on the test.

- We typically use validation error to choose "hyper-parameters"…

Notation: Parameters and Hyper-Parameters

- The decision tree rule values are called "parameters".
  - Parameters are variables we adjust during training to fit a dataset.
  - We "train" a model by trying to find the best parameters on training data.

- The decision tree depth is a called a "hyper-parameter".
  - Hyper-parameters are variables that are inputs to training.
  - They are not set based on the training data.
  - Often, hyper-parameters control how complex our model is.
    - You can always fit training data better by making the model more complicated.
  - We typically set hyper-parameters using a validation score.

Choosing Hyper-Parameters with Validation Set

- So to choose a good depth “hyper-parameter”, we could:
  - Try a depth-1 decision tree, compute validation error.
  - Try a depth-2 decision tree, compute validation error.
  - Try a depth-3 decision tree, compute validation error.
  - …
  - Try a depth-20 decision tree, compute validation error.
  - Return the depth with the lowest validation error.

- After you choose the hyper-parameter, we usually re-train on the full training set with the chosen hyper-parameter.

Optimization Bias

- Another name for overfitting is “optimization bias”:
  - How biased is an “error” that we optimized over many possibilities ?

- Optimization bias of parameter learning:
  - During learning, we could search over tons of different decision trees.
  - So we can get “lucky” and find one with low training error by chance.
    - “Overfitting of the training error”.

- Optimization bias of hyper-parameter tuning:
  - Here, we might optimize the validation error over 20 values of “depth”.
  - One of the 20 trees might have low validation error by chance.
    - “Overfitting of the validation error”.

Example of Optimization Bias

- Consider a multiple-choice (a,b,c,d) "test" with 10 questions:
  - If you choose answers randomly, expected grade is 25% (no bias).
  - If you fill out two tests randomly and pick the best, expected grade is 33%.
    - Optimization bias of ~8%.
  - If you take the best among 10 random tests, expected grade is ~47%.
  - If you take the best among 100, expected grade is ~62%.
  - If you take the best among 1000, expected grade is ~73%.
  - If you take the best among 10000, expected grade is ~82%.
    - You have so many "chances" that you expect to do well.

↘ and that is why you pick the unused testig

- But on new questions the "random choice" accuracy is still 25%.

Factors Affecting Optimization Bias

set

- If we instead used a 100-question test then:
  - Expected grade from best over 1 randomly-filled test is 25%.
  - Expected grade from best over 2 randomly-filled test is ~27%.
  - Expected grade from best over 10 randomly-filled test is ~32%.
  - Expected grade from best over 100 randomly-filled test is ~36%.
  - Expected grade from best over 1000 randomly-filled test is ~40%.
  - Expected grade from best over 10000 randomly-filled test is ~47%.

- The optimization bias grows with the number of things we try.
  - "Complexity" of the set of models we search over.
- But, optimization bias shrinks fast with number of validation examples.
  - But it's still non-zero and growing if you over-use your validation set!

Overfitting to the Validation Set?

- We can overfit to the validation set (common in practice):
  - Validation error is only an unbiased approximation if you use it once.
  - Once you start optimizing it, you start to overfit to the validation set.

- But validation error usually has lower optimization bias than train error.
  - Might optimize over 20 values of "depth", instead of millions+ of possible trees.
    - Amount of overfitting to validation set is low if we only try 10 things.

- Optimization bias is larger when the validation set is "small":
  - The optimization bias decreases as the number of validation examples increases.

- Remember, our goal is still to do well on the test set (new data), not the validation set (where we already know the labels).

Optimization Bias in Machine Learning Competitions

- It is common to have machine learning "competitions".
  - Some company releases a training set.
    - Many people try many different things to try to develop the "best" model.
  - At the end of the competition, the methods are compared on unseen test data.
    - And a "winner" or "winners" are declared based on the test set performance.

- In some cases, this has led to major new insights on ML methods.
  - Including the rise in popularity of "deep learning" methods we'll see later.

- In most cases, many people submit very-similar methods.
  - Expected "best test error" from 10000 similar submissions is biased!
    - The "best" methods might just be the one that got the most lucky.

Optimization Bias in Machine Learning Benchmarks

Computer Science > Computer Vision and Pattern Recognition

[Submitted on 16 Sep 2021]

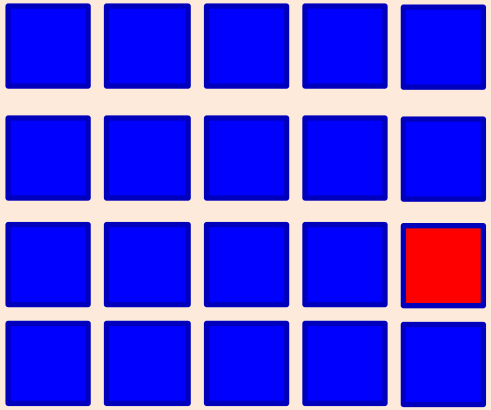
Torch.manual_seed(3407) is all you need: On the influence of random seeds in deep learning architectures for computer vision

David Picard

In this paper I investigate the effect of random seed selection on the accuracy when using popular deep learning architectures for computer vision. I scan a large amount of seeds (up to 10^4) on CIFAR 10 and I also scan fewer seeds on Imagenet using pre-trained models to investigate large scale datasets. The conclusions are that even if the variance is not very large, it is surprisingly easy to find an outlier that performs much better or much worse than the average.

Aside: Optimization Bias leads to Publication Bias

- Suppose that 20 researchers perform the exact same experiment:



- They each test whether their effect is "significant" ($p < 0.05$).
  - 19/20 find that it is not significant.
  - But the 1 group finding it's significant publishes a paper about the effect.

- This is again optimization bias, contributing to publication bias.
  - A contributing factor to many reported effects being wrong.

Summary

- Training error vs. testing error:
  - What we care about in machine learning is the testing error.
- A key principle:
  - The test data cannot influence training the model in any way.
- Independent and identically distributed (IID):
  - One assumption that makes learning possible.
- Fundamental trade-off:
  - Trade-off between getting low training error and having training error close to test error.
- Validation set:
  - We can save part of our training data to approximate test error.
- Hyper-parameters:
  - Parameters that control model complexity, typically set with a validation set.
- Optimization bias: using a validation set too much overfits.

- Next time:
  - We discuss the “best” machine learning method.

More Discussion of Optimization Bias

question ☆ stop following 91 views

Where does the term optimization bias come from

Could you please explain where the name optimization bias comes from, and what does it mean when saying "something" is due to optimization bias.

I learned it is used interchangeably with over-fitting, but they are slightly different. In my opinion, overfitting is a result, a result of using overly complicated models. However, it is confusing when optimization bias used as a reason like "we can't find the true model because of optimization bias".

Mark Schmidt 1 month ago

To me, optimization bias is a very-general abstract concept, and overfitting is a special case of optimization bias.

When we minimize the value of a statistic (like the test error, or a p-value, or whatever) for a particular dataset in terms of some possible "parameters", we expect the value of that statistic to be lower on that particular dataset than it would be on a new dataset. The optimization bias is how much lower we expect it be.

In we define overfitting as the difference between the training error and the test error, then overfitting is the optimization bias due to fitting the parameters on our particular dataset.

"Test Set" vs. "Test Error"

- Formally, the "test error" is the expected error of our model:

$$\mathbb{E}\left[|\hat{y}^i - \tilde{y}^i|\right]$$

  - Here I'm using absolute error between predictions and true labels.
    - But you could use squared error or other losses.
  - The expectation is taken over distribution of test examples.
    - Think of this as the "error with infinite data".
  - We assume that our training examples are drawn IID from this distribution.
    - Otherwise, "training" might not help to reduce "test error".

- Unfortunately, we cannot compute the test error.
  - We don't have access to the distribution over all test examples.

"Test Set" vs. "Test Error"

- We often approximate "test error" with the error on a "test set":

$$\frac{1}{t}\sum_{i=1}^{t} |\hat{y}^i - \tilde{y}^i|$$

  - Here, we are using 't' examples drawn IID from the test distribution.
- Note that "test set error" is not the "test error".
  - The goal is have a low "test error", not "test set error".

- The key principle we discussed in the above context:
  - A "test set" cannot influence the "training" in any way.
  - Otherwise, "test set error" is not an unbiased "test error" approximation.
  - We run the risk of "overfitting" to the "test set".

“test error” vs. “test set error” vs. “validation error”

Chenliang Zhou 8 months ago @Mark
About Q1, wouldn’t the dataset we use to examine our performance be called validation dataset? Mike said that in 340 “testing dataset” refers to those we don’t know.

Lucas Porto 8 months ago I'm now confused about this too. I thought there should be a separate "test set", which you use to measure the performance of your model after training and selection. Selection here meaning hyperparameter tuning with a validation set that not used for training.

Mark Schmidt 8 months ago Unfortunately, there isn't a standard nomenclature for what exactly defines a "test set". But a common convention is this:

1. The "test error" is the expected error over all possible future examples. You can never measure this.

2. We often have a "test set" that we are using to approximate this "test error". So we could say the "test set error" is being used as an approximation of the "test error". If you want this "test set error" to be an unbiased estimate of test error, it should not influence the training in any way. Unfortunately, most people (including your profs) aren't careful about distinguishing "test error" and "test set error".

3. When we tune hyper-parameters, we often use a "validation set" to approximate the "test error". Since we are evaluating the validation error several times, it will have an optimization bias. So it might guide us towards good hyper-parameters (because the bias is typically not that large) but really be used as an unbiased measure of test error.

can't

“test error” vs. “test set error” vs. “validation error”

We are given a huge dataset that we want to make a model from it. We can never know the exact performance of the model for new data that is NOT part of our dataset.

So here is what we can do:

Split the huge dataset into 3 categories:

a) **Training** data: this data is used to train a model
b) **Validation** data: this data is used "**intermediate**" measure the performance of the model we created
c) **Test** data: this data is used as a "**final**" measure of performance of the of the final model that was created

To choose a model do the following:
1. train the model using the **training data**
2. once you have a candidate model, find its performance (i.e validation error) using the **validation data**
3. if you are not satisfied with the performance of the candidate model, find a new model using **training** data and measure its performance using **validation** data. But don't look at your **test** data yet. (i.e do step 1 and 2 again)
4. once you have a model that you are satisfied with (i.e it has low validation error) you can select the model as your **FINAL trained model** meaning that you cannot go back and change the model again.
5. Measure the performance of your **FINAL** model using the **Test** data. You can think of this performance, as the good approximation of the performance of the model on NEW data that the model has never seen.

And the golden rule of ML states that

```
You should NEVER EVER use your test data in order to train a model.
```

If you do so your model will be biased.

Mark Schmidt 8 months ago Great explanation anonymous!

“A visual Introduction to machine learning”

- The “housing prices” example is taken from this website:
  - http://www.r2d3.us/visual-intro-to-machine-learning-part-1
- They also have a “Part 2” here:
  - http://www.r2d3.us/visual-intro-to-machine-learning-part-2
- Part 2 covers similar topics to what we covered in this lecture.

Generalization Gap for Selecting Hyper-Parameters

- From the 2019 EasyMarkit AI Hackathon:
  - "We ended up selecting the hyperparameters that gave us the lowest generalization gap as opposed to the lowest validation error. This was quite a difficult decision for our team since we were only allowed one submission. However, the model with the lowest validation error had a very high generalization gap, which felt too risky, so we went with a model with a slightly higher validation error and much lower generalization gap. When the results were announced, the reported test accuracy was within 0.1% of what our model predicted with the validation set."

- This is the type of reasoning you want to do.
  - A high generalization gap could indicate low validation error by chance.

Typical Supervised Learning Steps (Are Bad?)

- Given data {X,y}, a typical set of supervised learning steps:
  - Data splitting:
    - Split {X,y} into a train set {Xtrain,ytrain} and a validation set {Xvalid,yvalid}.
    - We're going to use the validation set error as an approximation of test error.
  - Tune hyper-parameters (decision tree depth, "regularization", "number of hidden units", etc.):
    - For each candidate value "λ" of the hyper-parameters:
      - Fit a model to the train set {Xtrain,ytrain} using the given hyper-parameters "λ".
      - Evaluate the model on the validation set {Xvalid,yvalid}.
  - Choose the model with the best performance on the validation set.
    - And maybe re-train using hyper-parameter "λ" on the full dataset.

- Can this overfit, even though we used a validation set?
  - Yes, validation set is influencing training. But maybe it's not too bad…

Validation Error, Test Error, and Generalization Gap

- We discuss "fundamental trade-off" with respect to train error.
  - Simple identity relating training set error to test error.
- We have a similar identity for the validation error.
  - If E_{test} is the test error and E_{valid} is the error on the validation set, then:

$$E_{test} = \underbrace{(E_{test} - E_{valid})}_{E_{gap}} + E_{valid}$$

- If E_{gap} is small, then E_{valid} is a good approximation of E_{test}.
  - We can't measure E_{test}, so how do we know if E_{gap} is small?

Bounding E_{gap}

- Let's consider a simple case:
  - Labels y^i are binary, and we try 1 hyper-parameter setting.
  - IID assumption on validation set implies E_{valid} is unbiased: $\mathbf{E}[E_{valid}] = E_{test}$.
- We can bound probability E_{gap} is greater than ε.
  - Assumptions: data is IID (so E_{valid} is unbiased) and loss is in [0,1].
  - By using Hoeffding's inequality:

$$p(|E_{test} - E_{valid}| > \varepsilon) \leq 2\exp(-2\varepsilon^2 t)$$

(E_{gap}; t: number of examples in validation set)

  - Probability that E_{valid} is far from E_{test} goes down exponentially with 't'.
    - This is great: the bigger your validation set, the smaller generalization gap you get.

Bounding E_{gap}

- Let's consider a slightly less-simple case:
  - Labels are binary, and we tried 'k' hyper-parameter values.
  - In this case it's unbiased for each 'k': $E[E_{valid(\lambda)}] = E_{test}$.
  - So for *each* validation error $E_{valid(\lambda)}$ we have:

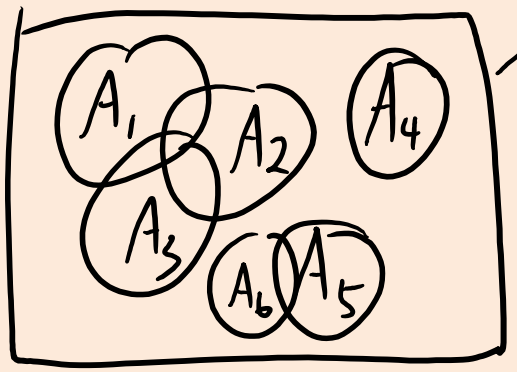
$$p(|E_{test} - E_{valid(\lambda)}| > \varepsilon) \leq 2\exp(-2\varepsilon^2 t)$$

  - But our final validation error is $E_{valid} = \min\{E_{valid(\lambda)}\}$, which is biased.
    - We can't apply Hoeffding because we chose best among 'k' values.
- Fix: bound on probability that **all** $|E_{test} - E_{valid(\lambda)}|$ values are $\leq \varepsilon$.
  - We show it holds for all values of λ, so it must hold for the best value.

Bounding E_{gap}

- The "union bound" for any events $\{A_1, A_2, \ldots, A_k\}$ is that:

$$p(A_1 \cup A_2 \cup \cdots \cup A_K) \leq \sum_{i=1}^{k} p(A_i)$$



Union of areas is less then sum of areas.

- Combining with Hoeffding we can get:

$$p(|E_{test} - \min_{\lambda}\{E_{valid(\lambda)}\}| > \epsilon) \leq p(\text{Exists a } \lambda \text{ where } |E_{test} - E_{valid(\lambda)}| > \epsilon)$$

$$\leq \sum_{\lambda} p(|E_{test} - E_{valid(\lambda)}| > \epsilon)$$

$$\leq \sum_{\lambda} 2\exp(-2\epsilon^2 t)$$

$$= K\, 2\exp(-2\epsilon^2 t)$$

Bounding E_{gap}

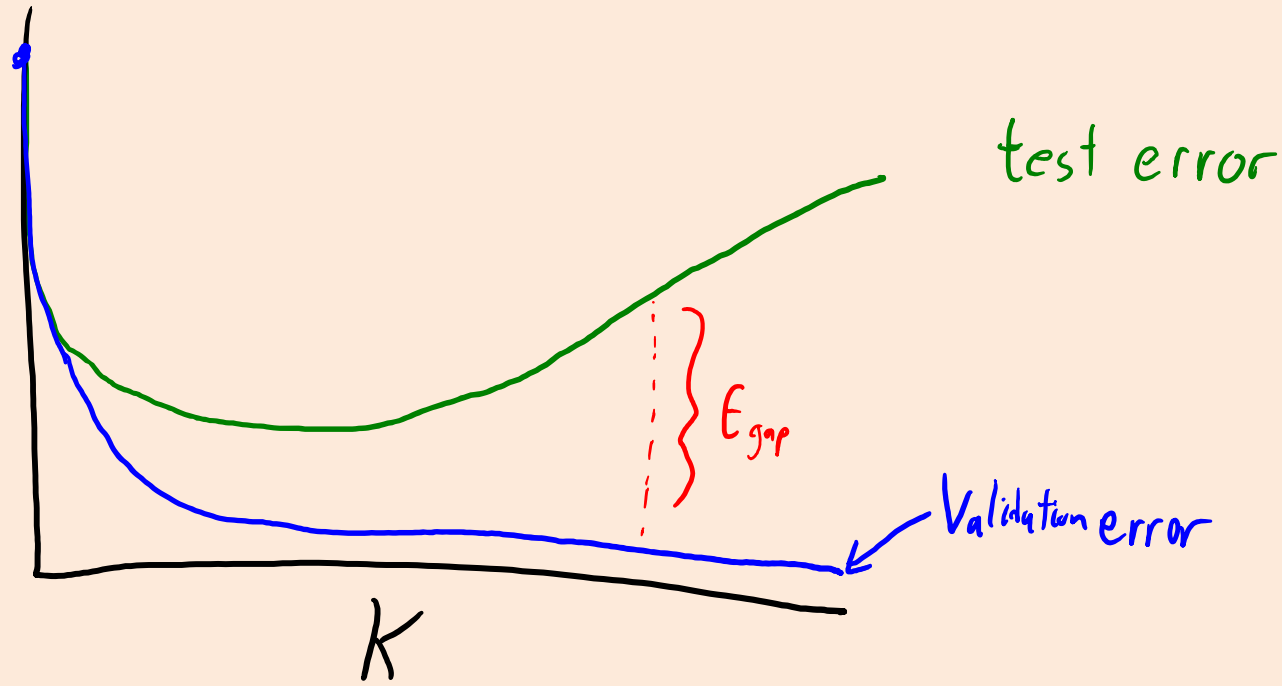
- So if we choose best $E_{valid(\lambda)}$ among 'k' λ values, we have:

$$p(|E_{test} - E_{valid(\lambda)}| > \varepsilon \text{ for any } \lambda) \leq k\, 2\exp(-2\varepsilon^2 t)$$

- So optimizing over 'k' models is ok if we have a large 't'.
  - But if 'k' is too large or 't' is too small the validation error isn't useful.
- Examples:
  - If k=10 and t=1000, probability that $|E_{gap}| > .05$ is less than 0.14.
  - If k=10 and t=10000, probability that $|E_{gap}| > .05$ is less than 10^{-20}.
  - If k=10 and t=1000, probability that $|E_{gap}| > .01$ is less than 2.7 (useless).
  - If k=100 and t=100000, probability that $|E_{gap}| > .01$ is less than 10^{-6}.

Bounding E_{gap}

- Validation error vs. test error for fixed 't'.
  - E_{valid} goes down as we increase 'k', but E_{gap} can go up.
    - Overfitting of validation set.



Discussion

- Bound is usually very loose, but data is probably not fully IID.
  - Similar bounds are possible for cross-validation.

- Similar arguments apply for the E_{gap} of the training error.
  - Value 'k' is the number of hyper-parameters you are optimizing over (even if don't try them all).
  - So 'k' is usually huge: you try out k=O(nd) decision stumps.

- What if we train by optimizing parameters over a continuous space?
  - We're optimizing on continuous space, so k=∞ and the bound is useless.
  - In this case, VC-dimension is one way to replace 'k' (doesn't need union bound).
    - "Simpler" models like decision stumps and linear models will have lower VC-dimension.

- Learning theory keywords if you want to go deeper into this topic:
  - Bias-variance (see bonus slides for details and why this is weird), sample complexity, PAC learning, VC dimension, Rademacher complexity.
  - A gentle place to start is the Learning from Data book.

Refined Fundamental Trade-Off

- Let E_{best} be the irreducible error (lowest possible error for *any* model).
  - For example, irreducible error for predicting coin flips is 0.5.
- Some learning theory results use E_{best} to further decompose E_{test}:

$$E_{test} = (E_{test} - E_{train}) + (E_{train} - E_{best}) + E_{best}$$

$E_{test} - E_{train}$: E_{gap}

$E_{train} - E_{best}$: E_{model}

E_{best}: "noise"

- E_{gap} measures *how sensitive we are to training data*.
- E_{model} measures *if our model is complicated enough to fit data*.
- E_{best} measures how low can **any** model make test error.
  - E_{best} does not depend on what model you choose.

Bias-Variance Decomposition

- You may have seen "bias-variance decomposition" in other classes:
  - Assumes $\tilde{y}_i = \bar{y}_i + \varepsilon$, where ε has mean 0 and variance σ^2.
  - Assumes we have a "learner" that can take 'n' training examples and use these to make predictions $\hat{y}_i$.
- Expected squared test error in this setting is

$$E[(\tilde{y}_i - \hat{y}_i)^2] = E[(\hat{y}_i - \bar{y}_i)]^2 + (E[\hat{y}_i^2] - E[\hat{y}_i]^2) + \sigma^2$$

"test squared error" "bias" "variance" "noise"

  - Where expectations are taken over possible training sets of 'n' examples.
  - Bias is expected error due to having wrong model.
  - Variance is expected error due to sensitivity to the training set.
  - Noise (irreducible error) is the best can hope for given the noise (E_{best}).

Refined Fundamental Trade-Off

- Decision tree with high depth:
  - Very likely to fit data well, so bias is low.
  - But model changes a lot if you change the data, so variance is high.
- Decision tree with low depth:
  - Less likely to fit data well, so bias is high.
  - But model doesn't change much you change data, so variance is low.

- And degree does not affect irreducible error.
  - Irreducible error comes from the best possible model.

Bias-Variance vs. Fundamental Trade-Off

- Both decompositions serve the same purpose:
  - Trying to evaluate how different factors affect test error.

- They both lead to the same 3 conclusions:
  1. Simple models can have high E_{train}/bias, low E_{gap}/variance.
  2. Complex models can have low E_{train}/bias, high E_{gap}/variance.
  3. As you increase 'n', E_{gap}/variance goes down (for fixed complexity).

Refined Fundamental Trade-Off

- Let E_{best} be the irreducible error (lowest possible error for *any* model).
  - For example, irreducible error for predicting coin flips is 0.5.
- Some learning theory results use E_{best} to further decompose E_{test}:

$$E_{test} = (E_{test} - E_{train}) + (E_{train} - E_{best}) + E_{best}$$

E_{gap} E_{model} "noise"

- This is similar to the bias-variance trade-off:
  - You need to trade between having low E_{gap} and having low E_{model}.
  - Powerful models have low E_{model} but can have high E_{gap}.
  - E_{best} does not depend on what model you choose.

Bias-Variance vs. Fundamental Trade-Off

- So why focus on fundamental trade-off and not bias-variance?
  - Simplest viewpoint that gives these 3 conclusions.
  - No assumptions like being restricted to squared error.

  - You can measure E_{train} but not E_{gap} (1 known and 1 unknown).
    - If E_{train} is low and you expect E_{gap} to be low, then you are happy.
      - E.g., you fit a very simple model or you used a huge independent validation set.

  - You can't measure bias, variance, or noise (3 unknowns).
    - If E_{train} is low, bias-variance decomposition doesn't say anything about test error.
      - You only have your training set, not distribution over possible datasets.
      - Doesn't say if high E_{test} is due to bias or variance or noise.

Learning Theory

- Bias-variance decomposition is a bit weird:
  - Considers expectation over *possible training sets*.
- Bias-variance says nothing about your training set.
  - This is different than Hoeffding bounds:
    - Bound the test error based on your actual training set and training/validation error.
- Some keywords if you want to learn about learning theory:
  - Bias-variance decomposition, sample complexity, probably approximately correct (PAC) learning, Vapnik-Chernovenkis (VC) dimension, Rademacher complexity.
- A gentle place to start is the “Learning from Data” book:
  - https://work.caltech.edu/telecourse.html

A Theoretical Answer to "How Much Data?"

- Assume we have a source of IID examples and a fixed class of parametric models.
  - Like "all depth-5 decision trees".
- Under some nasty assumptions, with 'n' training examples it holds that:
  E[test error of best model on training set] – (best test error in class) = O(1/n).

- You rarely know the constant factor, but this gives some guidelines:
  - Adding more data helps more on small datasets than on large datasets.
    - Going from 10 training examples to 20, difference with best possible error gets cut in half.
      - If the best possible error is 15% you might go from 20% to 17.5% (this does **not** mean 20% to 10%).
    - Going from 110 training examples to 120, error only goes down by ~10%.
    - Going from 1M training examples to 1M+10, you won't notice a change.
  - Doubling the data size cuts the error in half:
    - Going from 1M training to 2M training examples, error gets cut in half.
    - If you double the data size and your test error doesn't improve, more data might not help.

Can you test the IID assumption?

- In general, testing the IID assumption is not easy.
  - Usually, you need background knowledge to decide if it's reasonable.

- Some tests do exist, like shuffling the order of data and then measuring if some basic statistics agree.
  - It's reasonable to check if summary statistics of train and test data agree.
    - If not, your trained model may not be so useful.

- Some discussion here:
  - https://stats.stackexchange.com/questions/28715/test-for-iid-sampling

Wrong Decisions under false IID Assumption

There is a different narrative that one can tell about the current era. Consider the following story, which involves humans, computers, data, and life-or-death decisions, but where the focus is something other than intelligence-in-silicon fantasies. When my spouse was pregnant 14 years ago, we had an ultrasound. There was a geneticist in the room, and she pointed out some white spots around the heart of the fetus. "Those are markers for Down syndrome," she noted, "and your risk has now gone up to one in 20." She let us know that we could learn whether the fetus in fact had the genetic modification underlying Down syndrome via an amniocentesis, but amniocentesis was risky—the chance of killing the fetus during the procedure was roughly one in 300. Being a statistician, I was determined to find out where these numbers were coming from. In my research, I discovered that a statistical analysis had been done a decade previously in the UK in which these white spots, which reflect calcium buildup, were indeed established as a predictor of Down syndrome. I also noticed that the imaging machine used in our test had a few hundred more pixels per square inch than the machine used in the UK study. I returned to tell the geneticist that I believed that the white spots were likely false positives, literal white noise.

She said, "Ah, that explains why we started seeing an uptick in Down syndrome diagnoses a few years ago. That's when the new machine arrived."

We didn't do the amniocentesis, and my wife delivered a healthy girl a few months later, but the episode troubled me, particularly after a back-of-the-envelope calculation convinced me that many thousands of people had gotten that diagnosis that same day worldwide, that many of them had opted for amniocentesis, and that a number of babies had died needlessly. The problem that this episode revealed wasn't about my individual medical care; it was about a medical system that measured variables and outcomes in various places and times, conducted statistical analyses, and made use of the results in other situations. The problem had to do not just with data analysis per se, but with what database researchers call *provenance*—broadly, where did data arise, what inferences were drawn from the data, and how relevant are those inferences to the present situation? While a trained human might be able to work all of this out on a case-by-case basis, the issue was that of designing a planetary-scale medical system that could do this without the need for such detailed human oversight.

I'm also a computer scientist, and it occurred to me that the principles needed to build planetary-scale inference-and-decision-making systems of this kind, blending computer science with statistics, and considering human utilities, were nowhere to be found in my education. It occurred to me that the development of such principles—which will be needed not only in the medical domain but also in domains such as commerce, transportation, and education—were at least as important as those of building AI systems that can dazzle us with their game-playing or sensorimotor skills.

https://hdsr.mitpress.mit.edu/pub/wot7mkc1/release/8