CPSC 340: Machine Learning and Data Mining

Learning Theory

Fall 2016

Admin

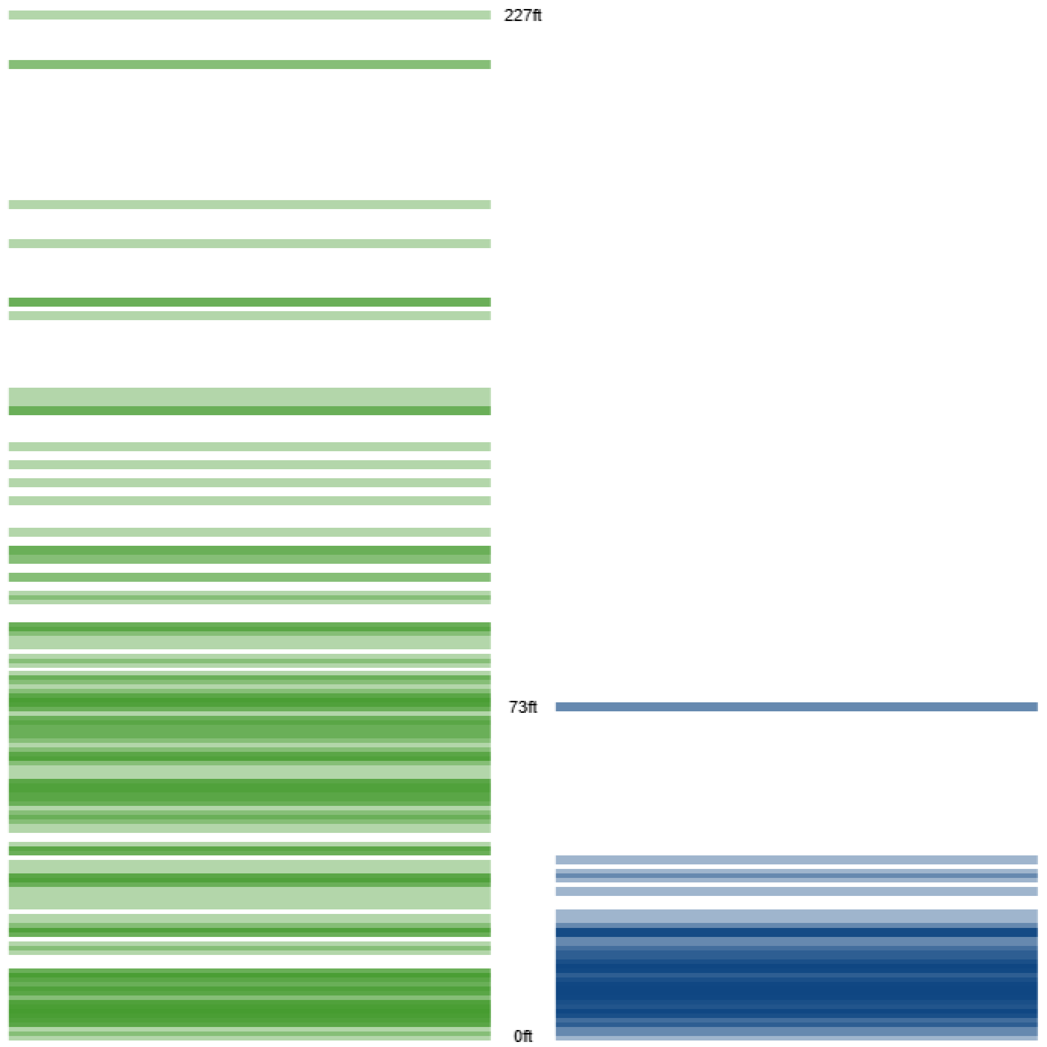
- Assignment 1 is out, due September 23rd.
  - Setup your CS undergrad account ASAP to use Handin:
    - https://www.cs.ubc.ca/getacct
    - Instructions for handin will be posted to Piazza.
  - Try to do the assignment this week, BEFORE add/drop deadline.
    - The material will be getting much harder and the workload much higher.

Motivation: Determine Home City

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

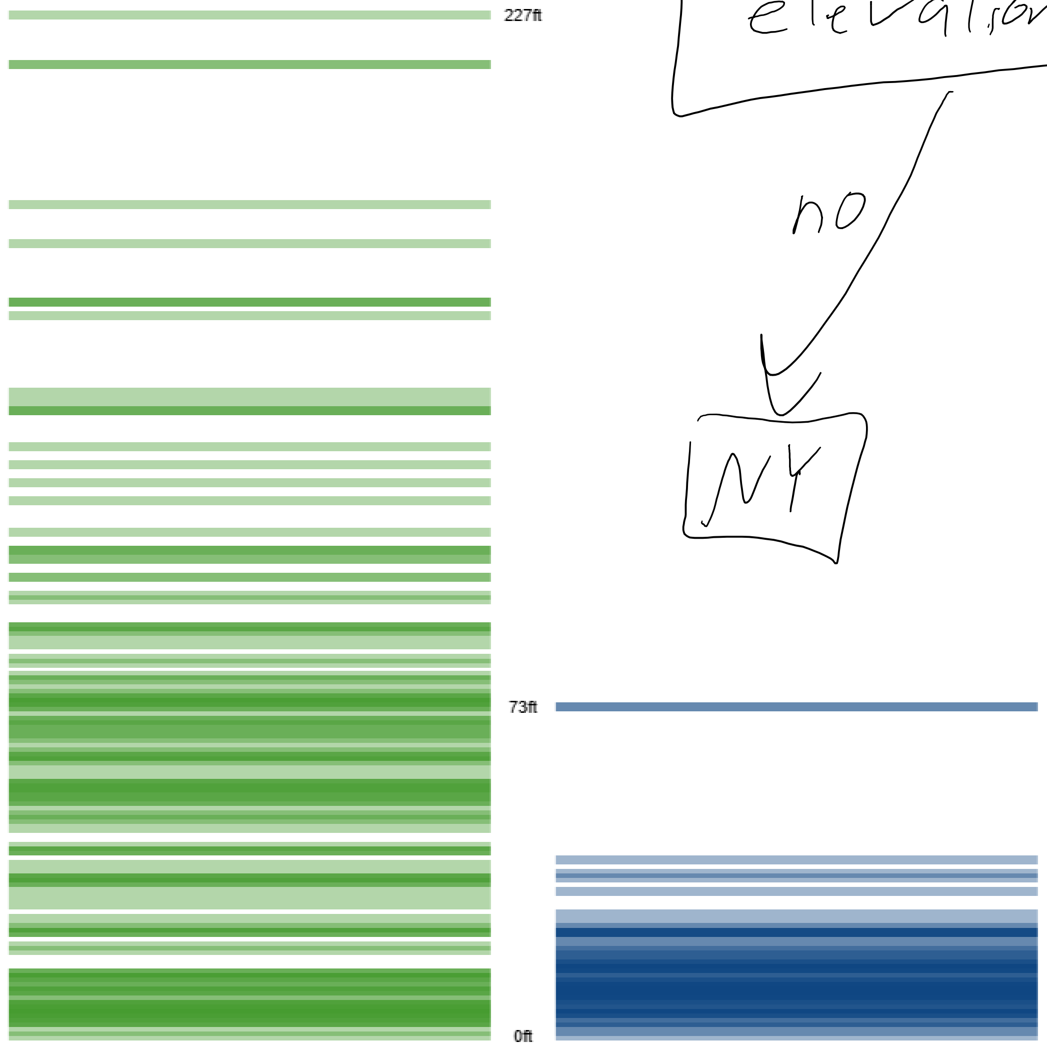
This example and images of it come from:
http://www.r2d3.us/visual-intro-to-machine-learning-part-1

Plotting Elevation

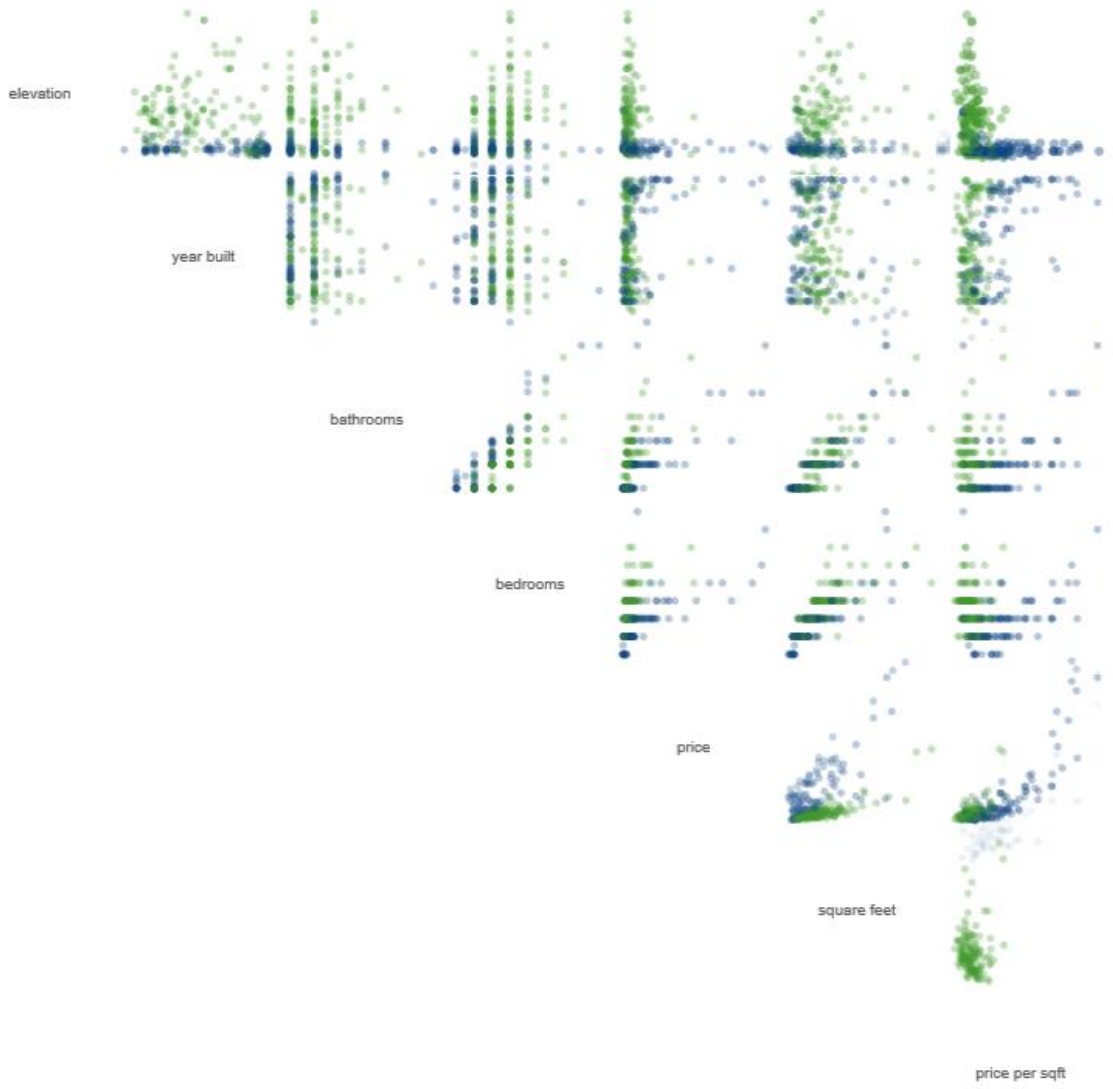


Simple Decision Stump

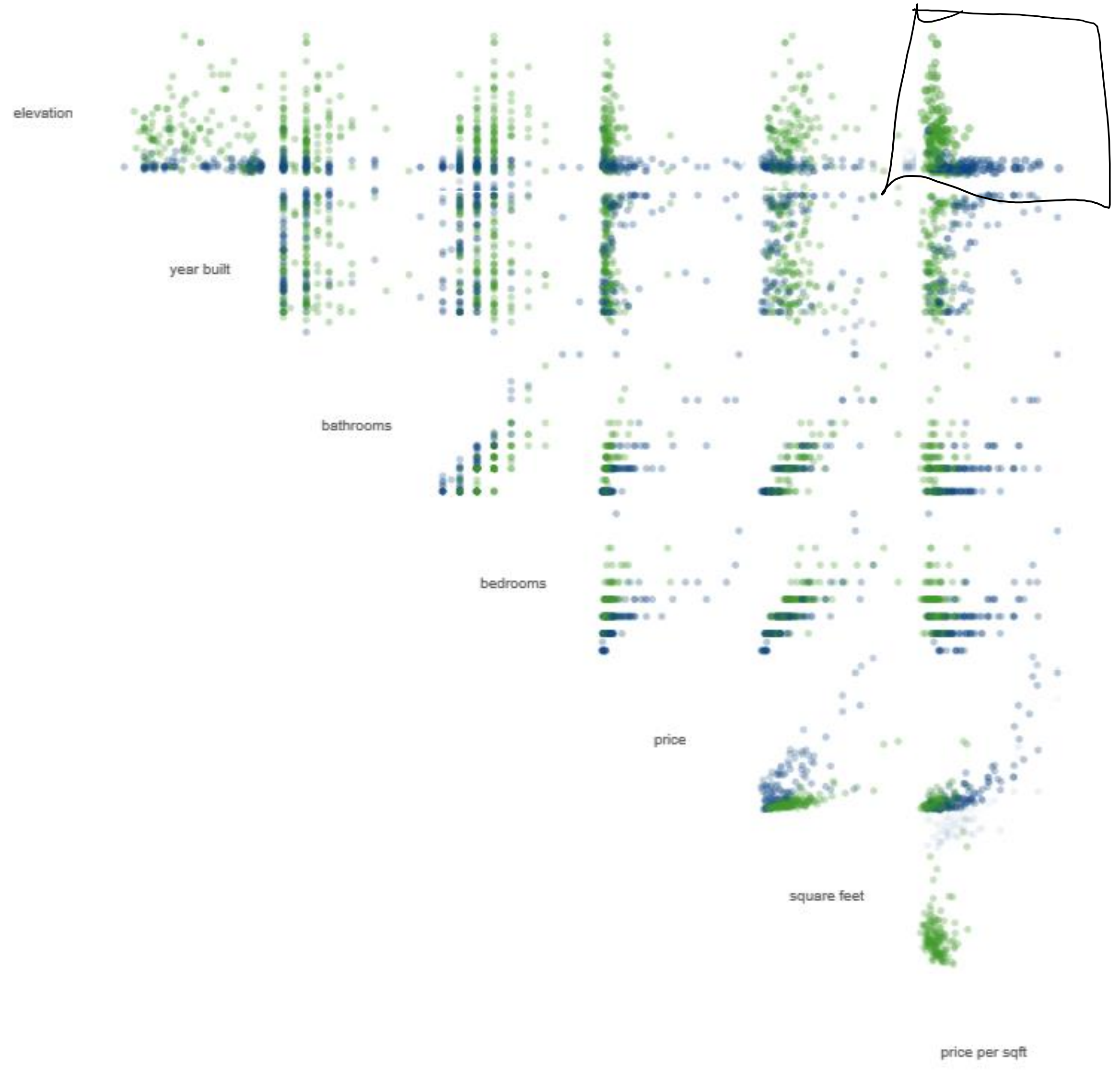
$$\begin{tikzcd}
 & \boxed{\text{elevation} > 73} \arrow[dl, "\text{no}"'] \arrow[dr, "\text{yes}"] & \\
\boxed{\text{NY}} & & \boxed{\text{SF}}
\end{tikzcd}$$



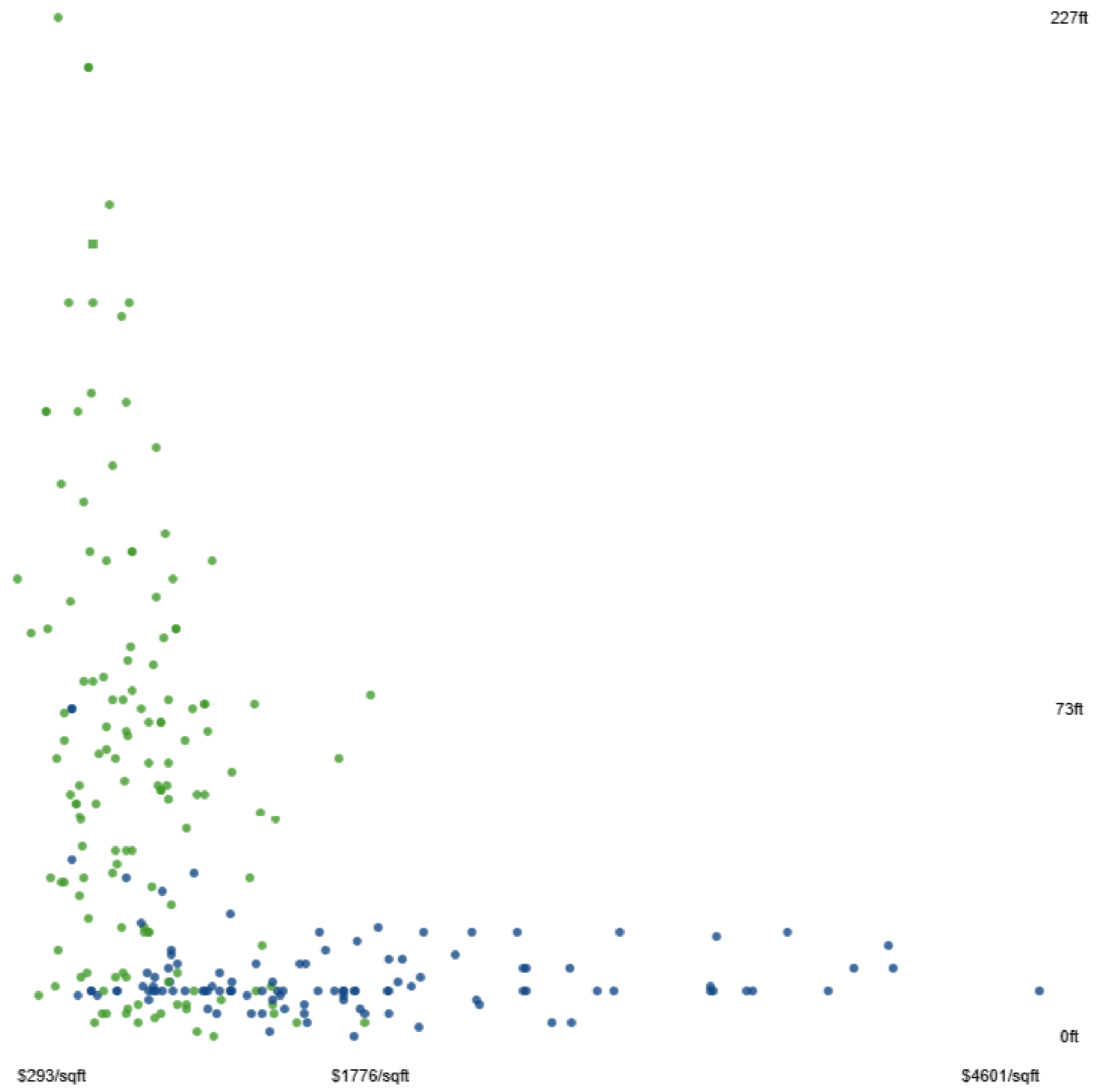
Scatterplot Array



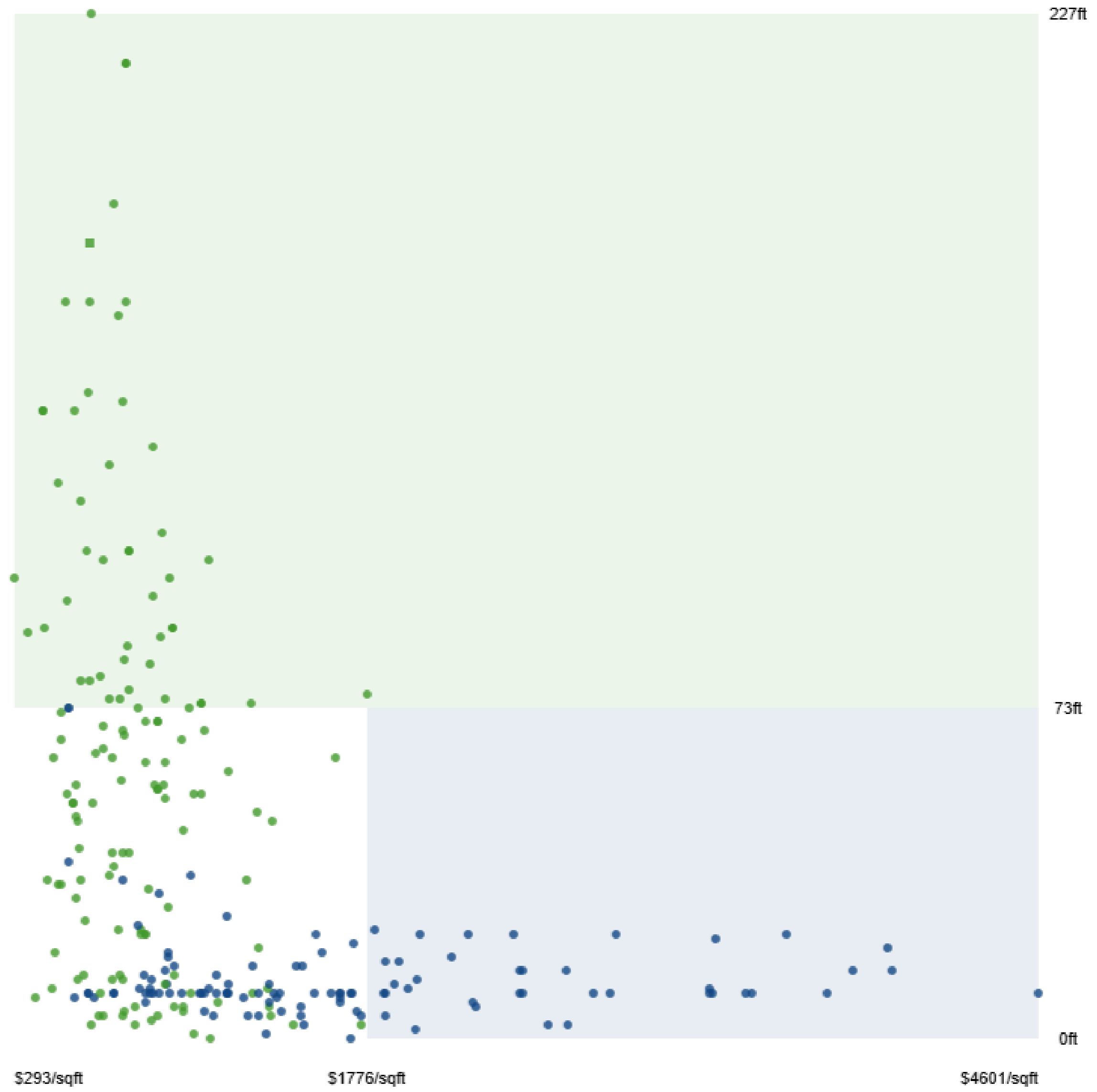
Scatterplot Array



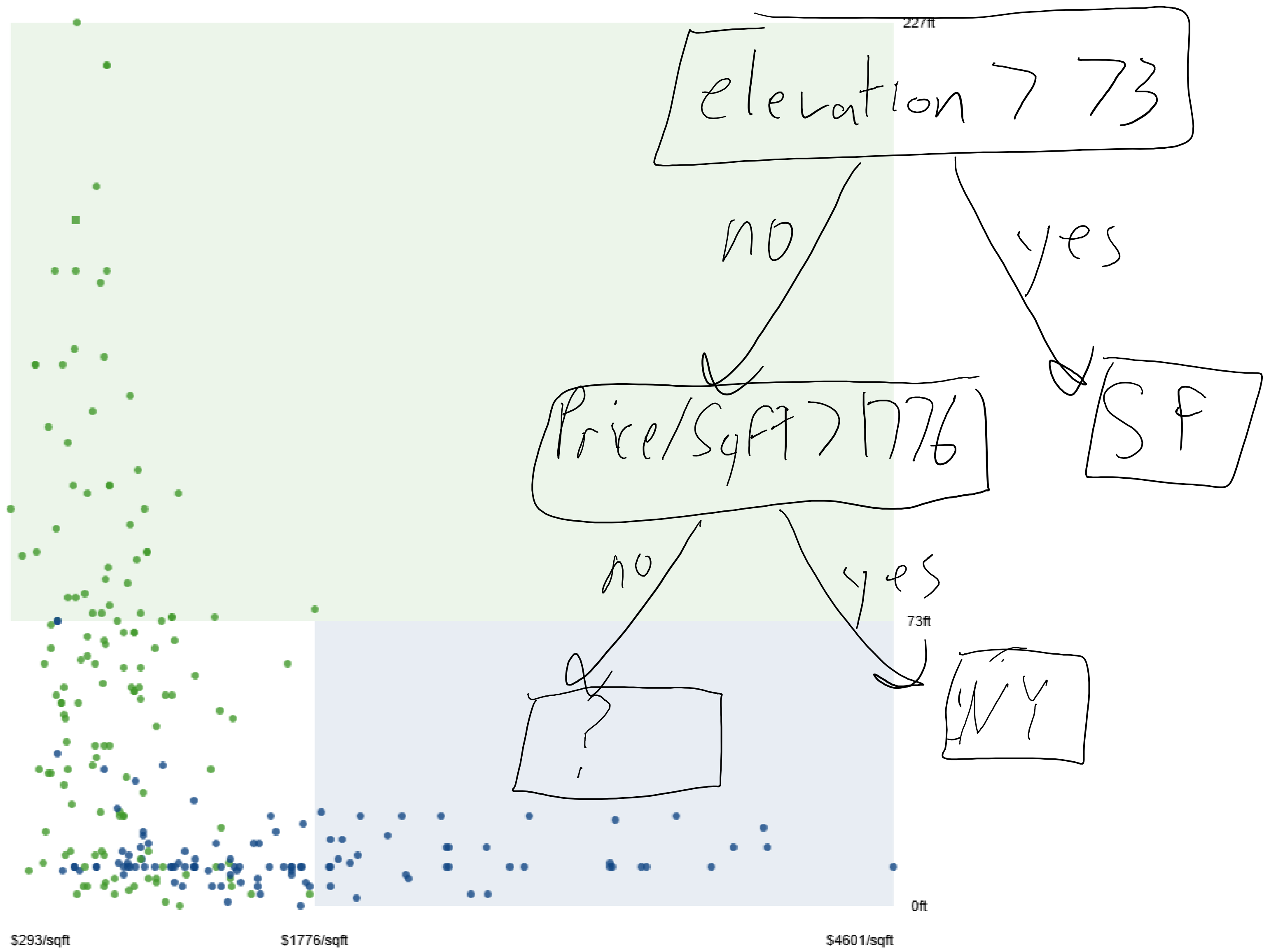
Plotting Elevation and Price/SqFt



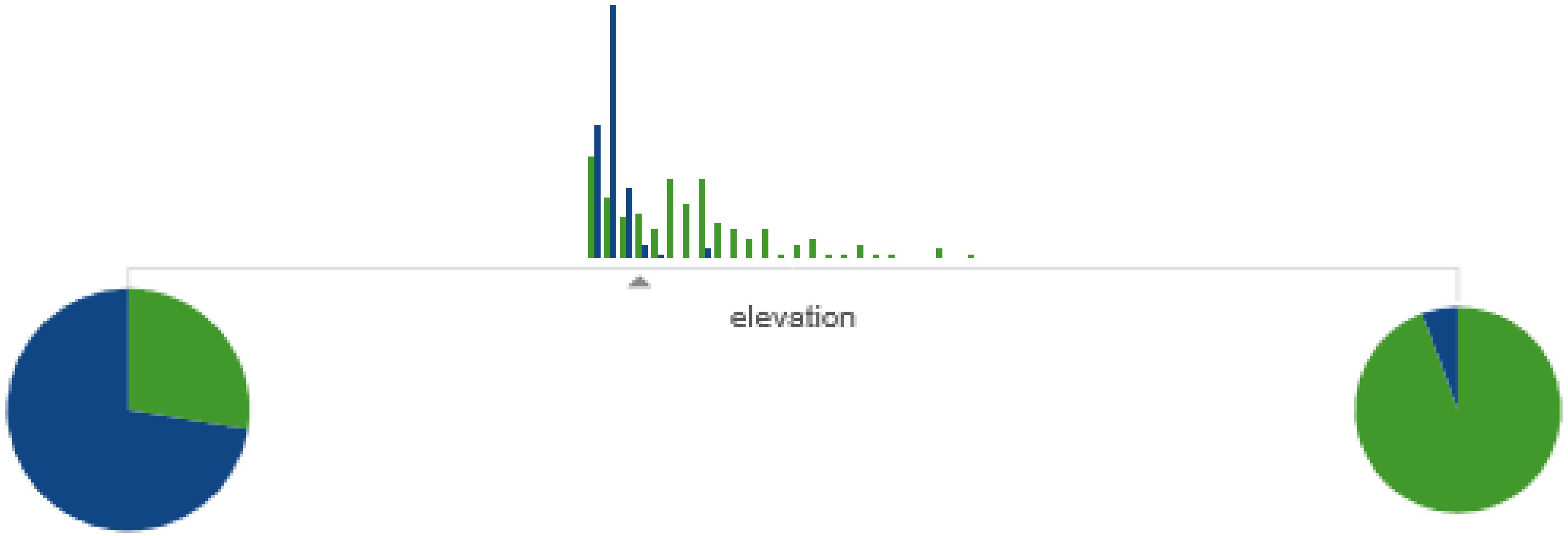
Simple Decision Tree Classification



Simple Decision Tree Classification

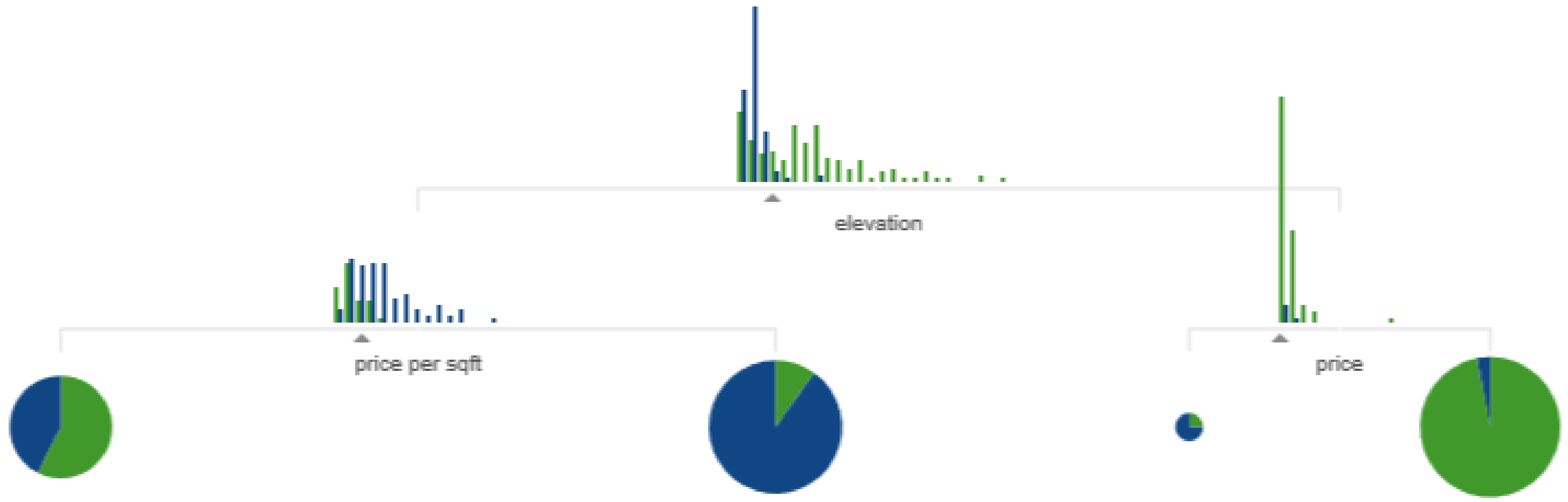


How does the depth affect accuracy?



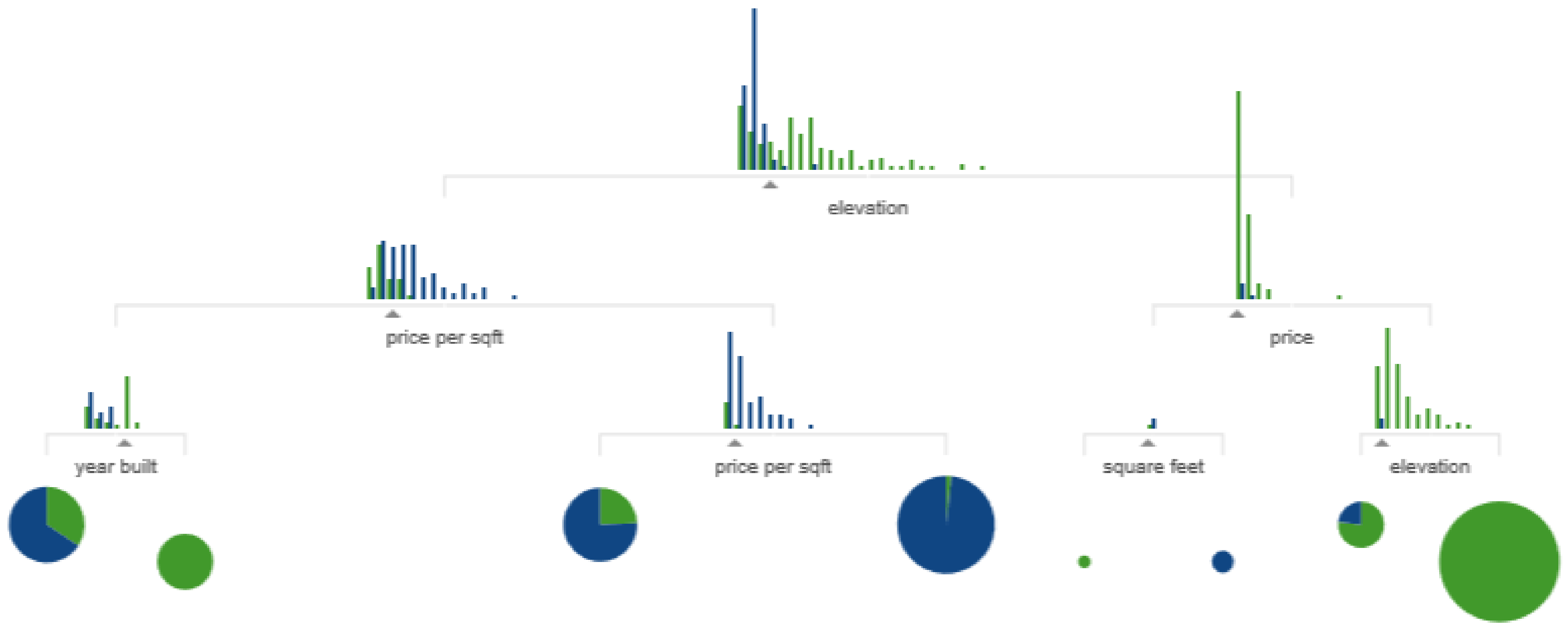
This is a good start (> 75% accuracy).

How does the depth affect accuracy?



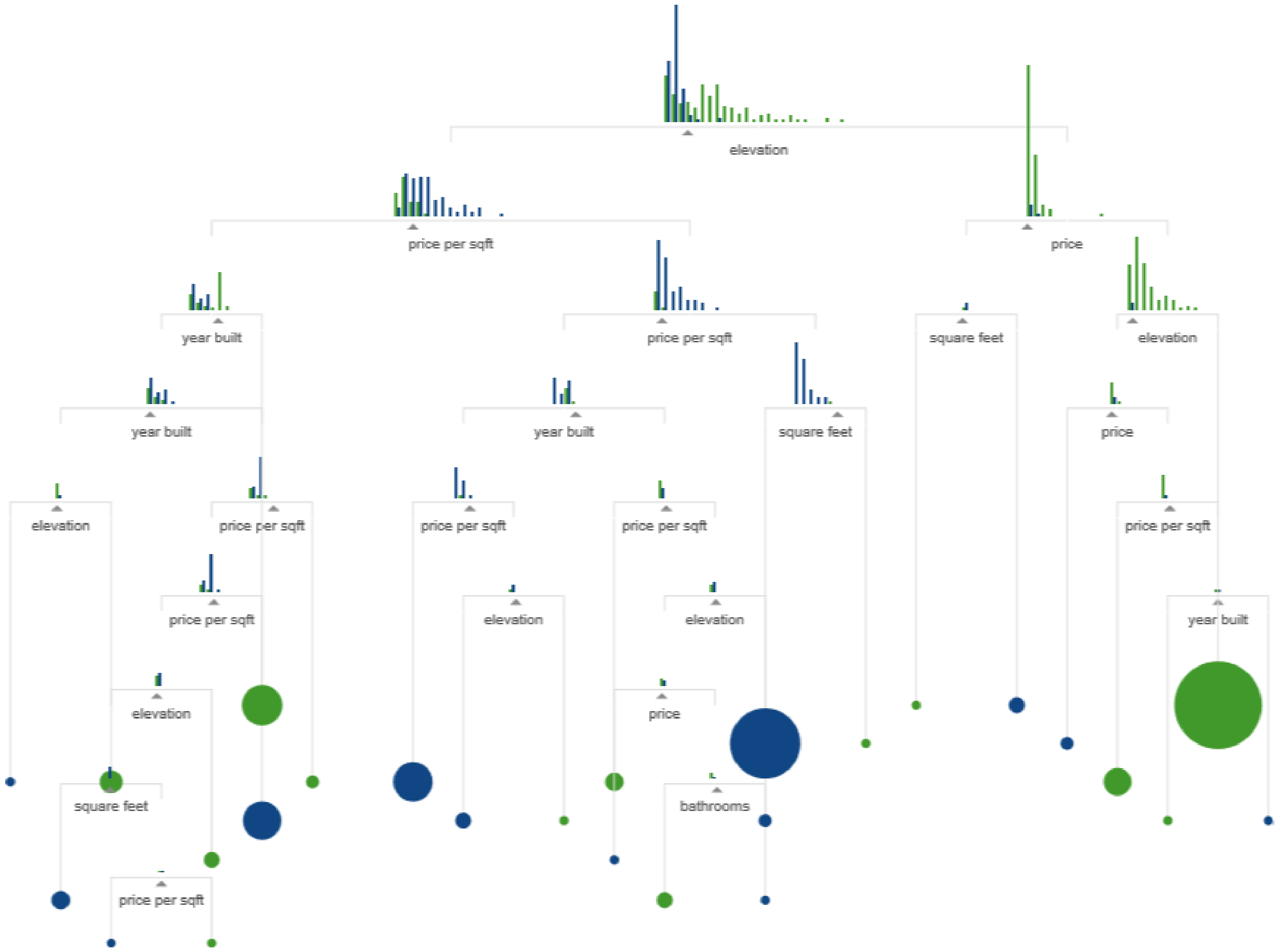
Start splitting the data recursively…

How does the depth affect accuracy?



Accuracy keeps increasing as we add depth.

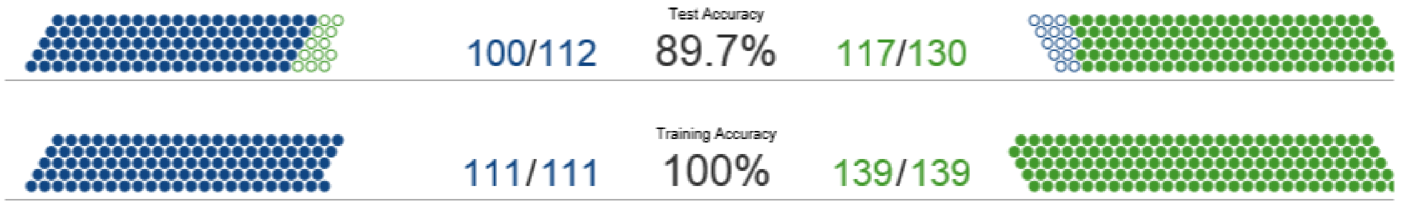
How does the depth affect accuracy?



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.

Supervised Learning Notation

- We are given training data where we know labels:

X =

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 =

Sick?
1
1
0
1
1

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

Xtest =

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	

ytest =

Sick?
?
?
?

Supervised Learning Notation

- Typical supervised learning steps:
  1. Build model based on training data X and y.
  2. Model makes predictions ‘yhat’ on test data Xtest.

$\hookrightarrow \hat{y}$

- Instead of training error, consider test error:
  - Is yhat similar to true unseen ytest?

Goal of Machine Learning

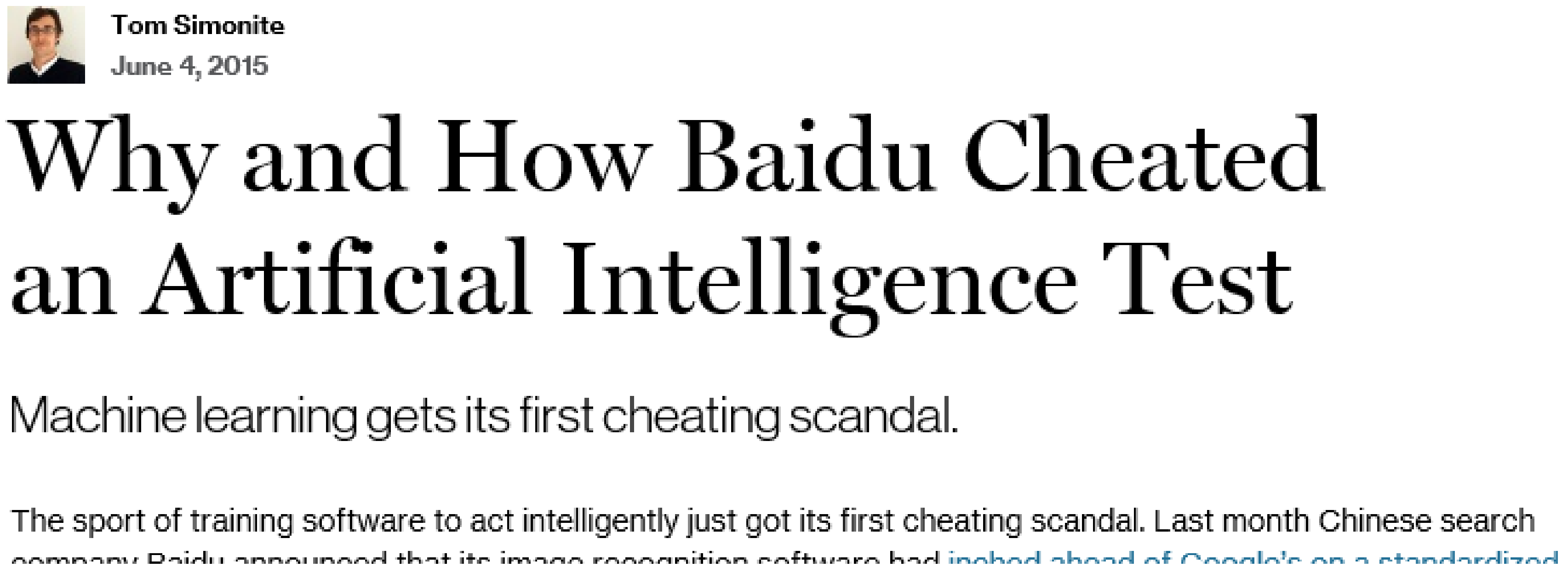
- In machine learning:
  - What we care about is the test error!
- Midterm analogy:
  - The training error is the practice midterm.
  - The test error is the actual midterm.
  - Goal: do well on actual midterm, not the practice one.
- Memorization vs learning:
  - Can do well on training data by memorizing it.
  - Only learned if you can do well in new situations.

Golden Rule of Machine Learning

- Even though what we care about is test error:
  - THE TEST DATA CANNOT INFLUENCE THE TRAINING PHASE IN ANY WAY.
- We’re measuring test error to see how well we do on new data:
  - If used during training, doesn’t measure this.
  - You can start to overfit if you use it during training.
  - Midterm analogy: you are cheating on the test.

Golden Rule of Machine Learning

- Even though what we care about is test error:
  - 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/

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

Xtest =

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

ytest =

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).

IID Assumption

- Training/test data is independent and identically distributed (IID) if:
  - All objects come from the same distribution (identically distributed).
  - The object 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 objects come from the same distribution?
  - Does the order of the objects 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?

Learning Theory

- Why does the IID assumption make learning possible?
  - Patterns in training examples are likely to be the same in test examples.
- Learning theory explores how training error is related to test error.
- The IID assumption is rarely true:
  - But it is often a good approximation.
  - There are other possible assumptions.
- Some keywords in learning theory:
  - Bias-variance decomposition, Hoeffding’s inequality and union bounds, sample complexity, probably approximately correct (PAC) learning, Vapnik-Chernovenkis (VC) dimension.

Fundamental Trade-Off

- Learning theory leads to a fundamental trade-off:
  1. How small you can make the training error.

     vs.
  2. How well training error approximates the test error.

- Different models make different trade-offs.
- Simple models (like decision stumps):
  - Training error is good approximation of test error:
    - Not very sensitive to the particular training set you have.
  - But don't fit training data well.
- Complex models (like deep decision trees):
  - Fit training data well.
  - Training error is poor approximation of test error:
    - Very sensitive to the particular training set you have.

Fundamental Trade-Off

- Learning theory leads to a fundamental trade-off:
  1. How small you can make the training error.

     vs.
  2. How well training error approximates the test error.

- Test error depends on the above and also the irreducible error:
  3. How low is it *possible* to make the test error?

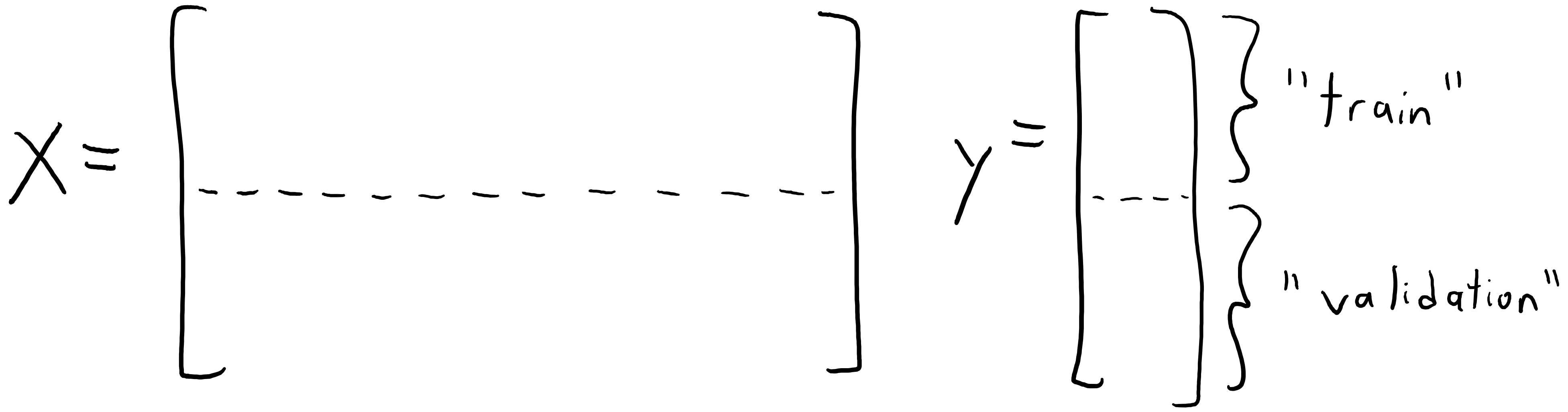
- You may have seen the bias-variance trade-off:
  - One form of fundamental trade-off for regression.
  - Part 1 shows up as bias, part 2 as variance, and part 3 is the same.
  - But it’s weird: involves *training sets you could have seen*, not your data.

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 your dataset to approximate test error.
- Split training objects into training set and validation set:
  - Train model based on the training data.
  - Test model based on the validation data.

Validation Error



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

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

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

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

Validation Error

- Validation error gives an 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, as an approximation of how you will do on the test.

- This leads to the following practical strategy:
  - 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.
  - …
  - Choose the depth with the lowest validation error.

Validation Error

- Validation error vs. training error for choosing depth:
  - Training error always decreases with depth.
  - Validation error initially decreases, but eventually increases (overfitting).

- Validation error is much less likely to lead to overfitting.

- But it can still overfit:
  - Validation error is only an unbiased approximation if you use it once.
  - If you minimize it to choose between models, introduces optimization bias.

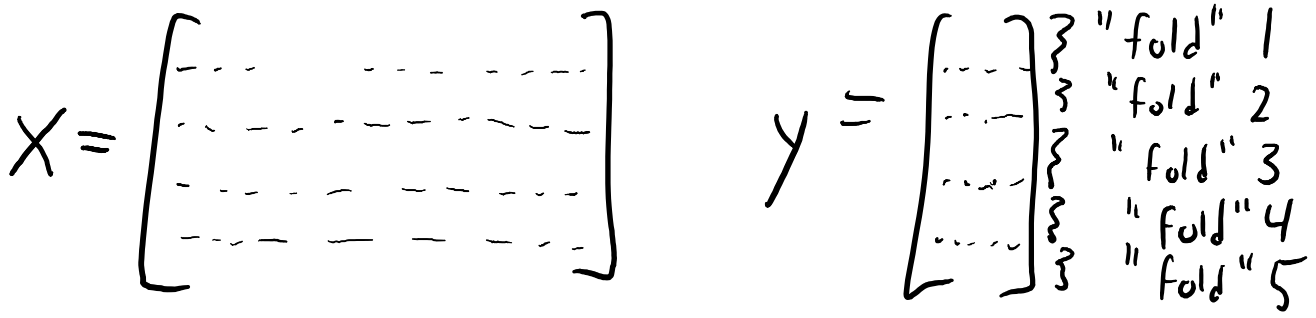
Validation Error and Optimization Bias

- Optimization bias is small if you only compare a few models:
  - Best decision tree on the training set among depths, 1, 2, 3,…, 10.
  - Here we're only using the validation set to pick between 10 models.
    - Validation likely still approximates test error.
    - Overfitting risk is low.

- Optimization bias is large if you compare a lot of models:
  - All possible decision trees of depth 10 or less.
  - Here we're using the validation set to pick between a billion+ models:
    - Some of these models likely have a low validation error by chance.
    - Overfitting risk is high.
  - If you did this, you might want a second validation set to detect overfitting.

Cross-Validation

- Isn’t it wasteful to only use part of your data?
- 5-fold cross-validation:
  - Train on 80% of the data, validate on the other 20%.
  - Repeat this 5 more times with different splits, and average the score.



1. Train on folds {1, 2, 3, 4}, compute error on fold 5.
2. Train on folds {1, 2, 3, 5}, compute error on fold 4.
3. Train on folds {1, 2, 4, 5}, compute error on fold 3.

⋮

6. Take average of the 5 errors.

Cross-Validation (CV)

- You can take this idea further:
  - 10-fold cross-validation: train on 90% of data and validate on 10%.
    - Repeat 10 times and average.
  - Leave-one-out cross-validation: train on all but one training example.
    - Repeat n times and average.
- Gets more accurate/expensive as you increase number of folds.

- We often re-train on the full dataset after picking depth.
- As before, if data is ordered then folds should be random splits.

Cross-Validation Theory

- Does CV give unbiased estimate of test error?
  - Yes: each data point is only used once in validation.
  - But again, that’s assuming you only do CV once.
- What about variance of CV?
  - Hard to characterize.
  - CV variance on ‘n’ data points is worse than with a validation set of size ‘n’.
    - But we believe it close!

Back to Decision Trees

- Instead of validation set, you can use CV to select tree depth.
- But you can also use these to decide whether to split:
  - Don’t split if validation/CV error doesn’t improve.
  - Different parts of the tree will have different depths.
- Or fit deep decision tree and use CV to prune:
  - Remove leaf nodes that don’t improve CV error.

- Popular implementations that have these tricks: C4.5, CART.

Summary

- Training error vs. testing error:
  - What we care about in machine learning is the testing error.
- Golden rule of machine learning:
  - The test data cannot influence training the model in any way.
- Fundamental trade-off:
  - Trade-off between getting low training error and having training error approximate test error.
- Validation sets and cross-validation:
  - We can use training data to approximate test error.

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

Bonus Slide: Bias-Variance Decomposition

- Analysis of expected test error of any learning algorithm:

Assume $y_i = f(x_i) + \epsilon$, for some function 'f' and random error ϵ with a mean of 0 and a variance of σ^2.

Assume we have a "learner" that can take a training set $\{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$ and use these to make predictions $\hat{f}(x_i)$.

Then for a new example (x_i, y_i) the error averaged over training sets is

$$E[(y_i - \hat{f}(x_i))^2] = \text{Bias}[\hat{f}(x_i)]^2 + \text{Var}[\hat{f}(x_i)] + \sigma^2$$

"Irreducible error": best we can hope for given the noise level.

Expected error due to having wrong model.

Where $\text{Bias}[\hat{f}(x_i)] = E[\hat{f}(x_i)] - f(x_i)$,

$\text{Var}[\hat{f}(x_i)] = E[(\hat{f}(x_i) - E[\hat{f}(x_i)])^2]$

How sensitive is the model to the particular training set?

Bonus Slide: Bias-Variance Decomposition

- 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.
- Bias-variance is a bit weird:
  - Considers expectation over possible training set.
  - But doesn't say anything about test error with *your* training set.
- There are other ways to estimate test error:
  - VC dimension bounds test error based on training error and model complexity.