

O'REILLY®

Security

BUILD BETTER DEFENSES

oreillysecuritycon.com

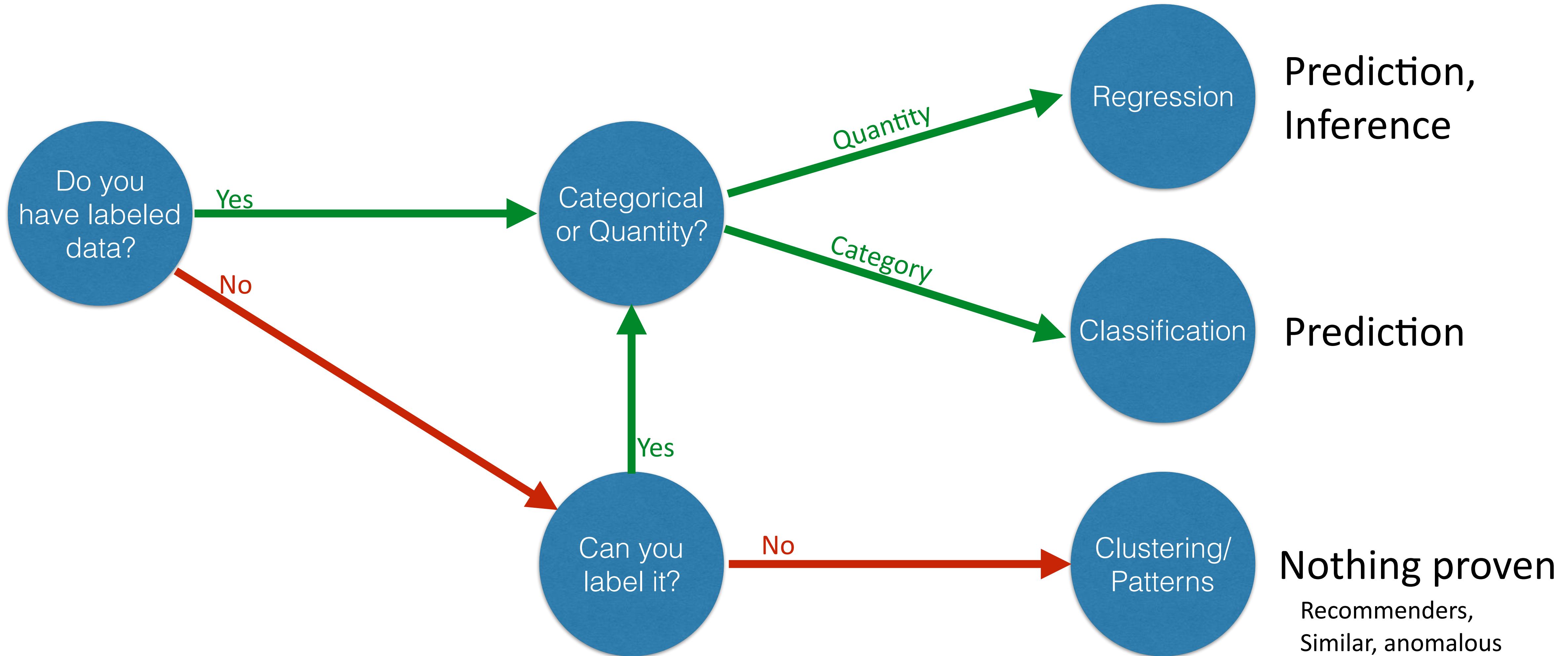
#oreillysecurity

Foundations of Security Data Science

Machine Learning: Supervised

Jay Jacobs
Charles Givre
Bob Rudis

What Data Science kind of is...



Data Science Topics

Core Statistics

Descriptive (range, median, mean)

Confidence Intervals

Correlation

Tests (t.test, ks.test, fitting)

Regression/Classification

Labeled Data

Can falsify claims

Prediction, maybe inference

No free lunch

Clustering/Unsupervised

Unlabeled data

Cannot falsify claims

Clustering

Anomaly Detection

Visualization

Visual cues, coord. system, scale, context

Pre-attentive processing, saccadic

Working and Long-term memory

Accuracy in Decoding, mumbling

Feature Engineering

1. Define the Object of Measurement
These will become the rows in your data (one per object)

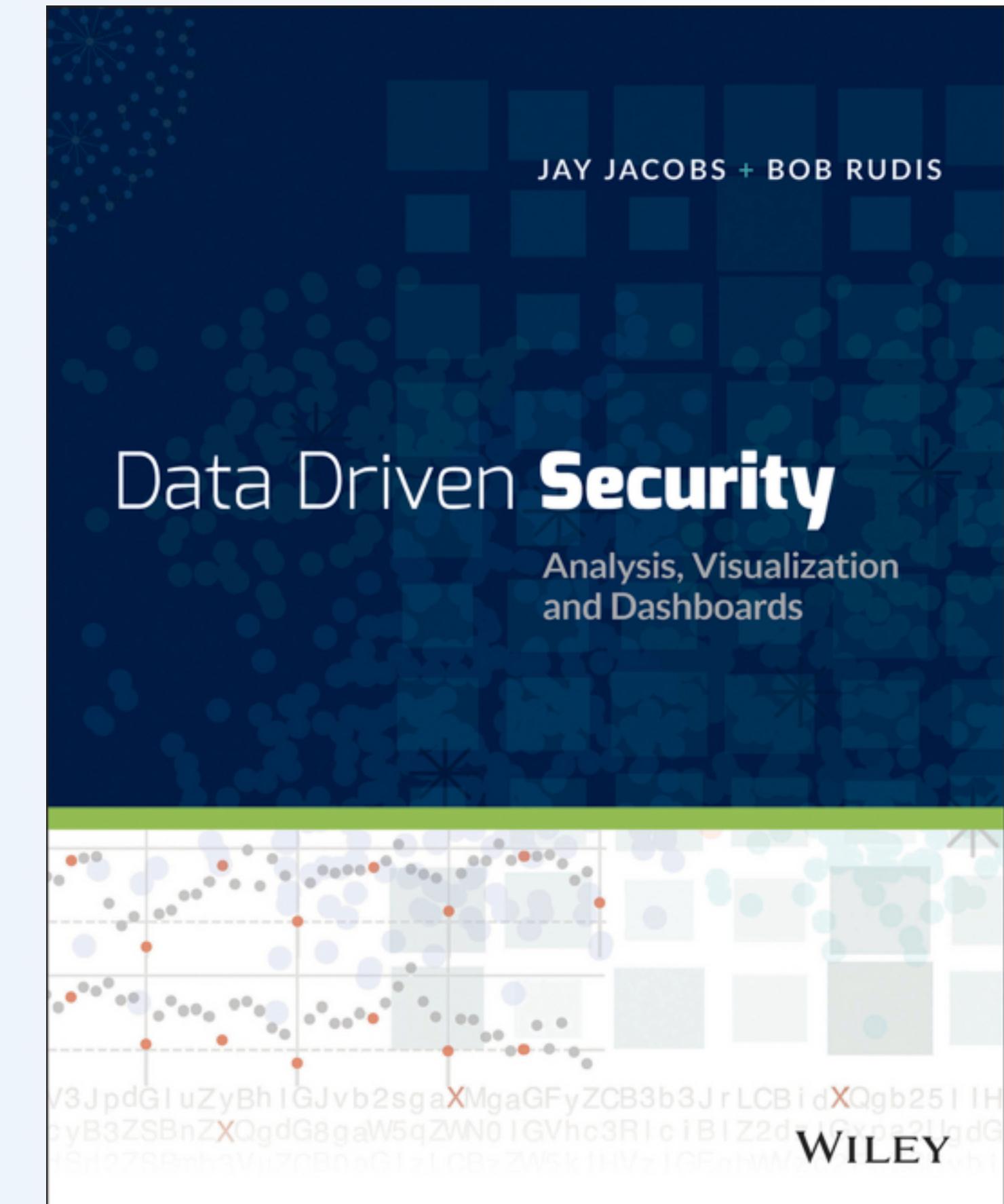
2. Define one or more “features” that describe each object
These will become the columns in your data

3. Run Algorithm (more on this later)

4. Optimally, measure feature contribution and model performance

Intuition vs. Expertise

“...It’s your experience with information security that will guide the direction of the analysis, provide context to the data, and help apply meaning to the results. In other words, domain expertise is beneficial in the beginning, middle, and end of all your data analysis efforts....”



Prediction vs. Interpretation

“While the primary interest of predictive modeling is to generate accurate predictions, a secondary interest may be to interpret the model and understand why it works. The unfortunate reality is that as we push towards higher accuracy, models become more complex and their interpretability becomes more difficult.”

<https://topepo.github.io/caret/available-models.html>

Max Kuhn · Kjell Johnson

Applied
Predictive
Modeling

 Springer

No Free Lunch / Measuring Quality

Why is it necessary to introduce so many different statistical learning approaches, rather than just a single *best* method?

There is no free lunch in statistics: no one method dominates all others over all possible data sets. Selecting the best approach can be one of the most challenging parts of performing statistical learning in practice.”

Springer Texts in Statistics

Gareth James
Daniela Witten
Trevor Hastie
Robert Tibshirani

An Introduction
to Statistical
Learning

with Applications in R

 Springer

O'REILLY®

Security

BUILD BETTER DEFENSES

oreillysecuritycon.com
#oreillysecurity

Brief Overview of Regression Analysis

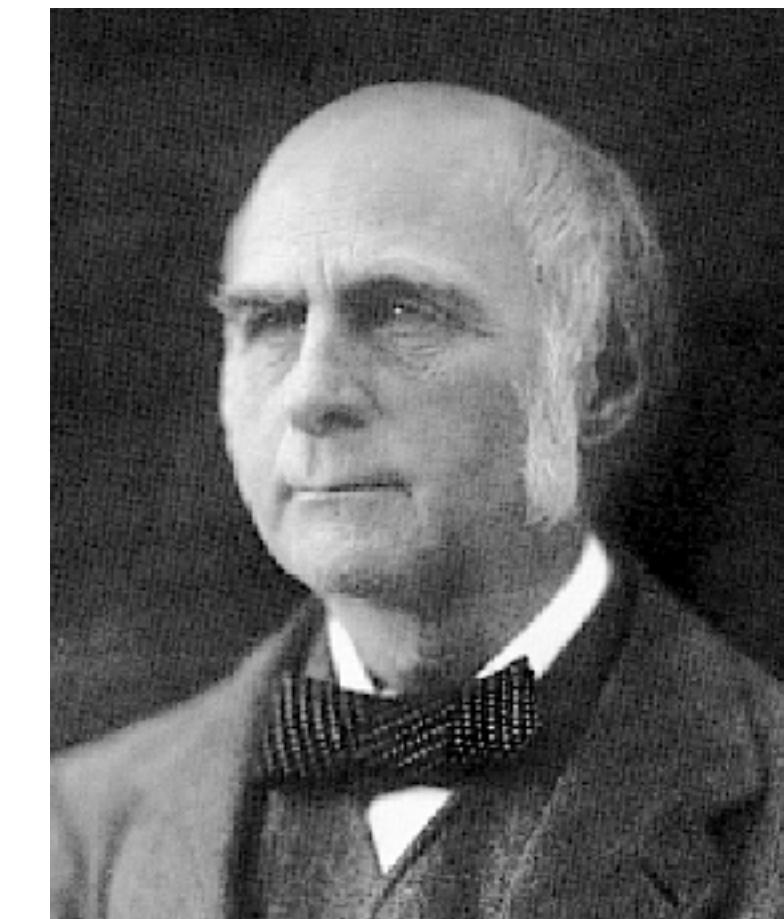
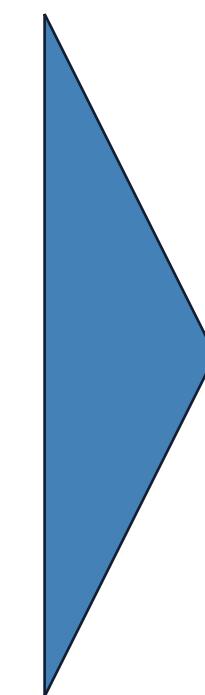
Regression: Standing on the Shoulders of Giants



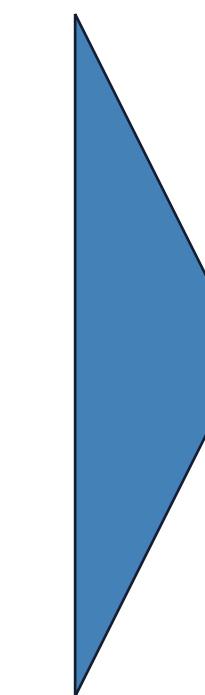
Adrien-Marie
Legendre (1805)



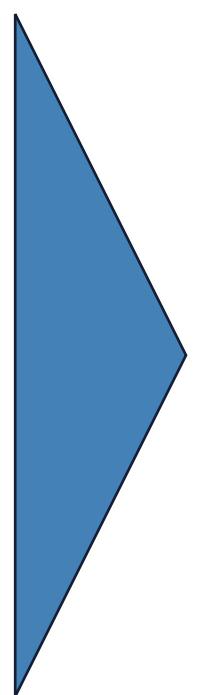
Carl Frederic
Gauss (1809)



Francis Galton
(1869)

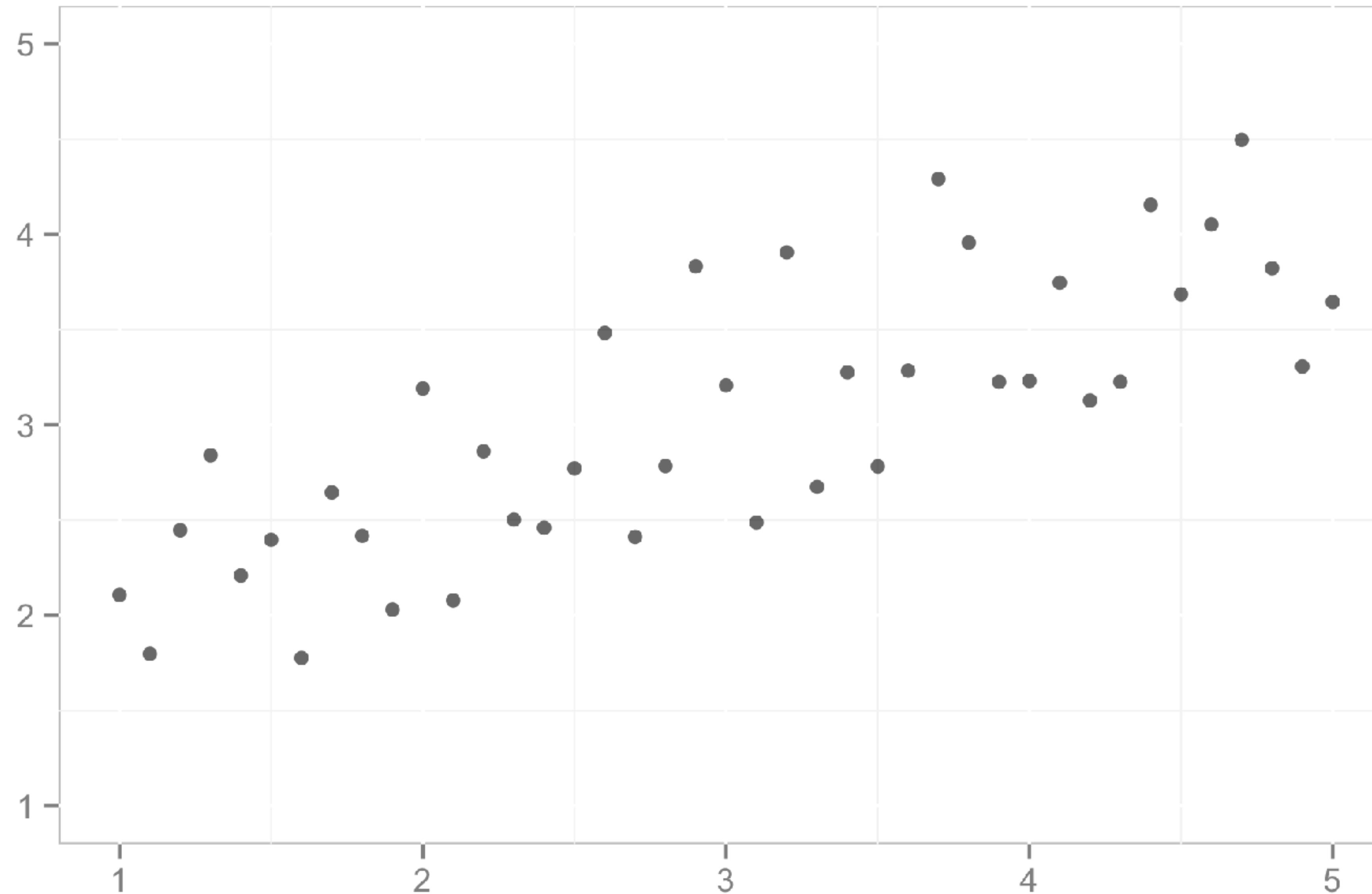


Karl Pearson
(1903)

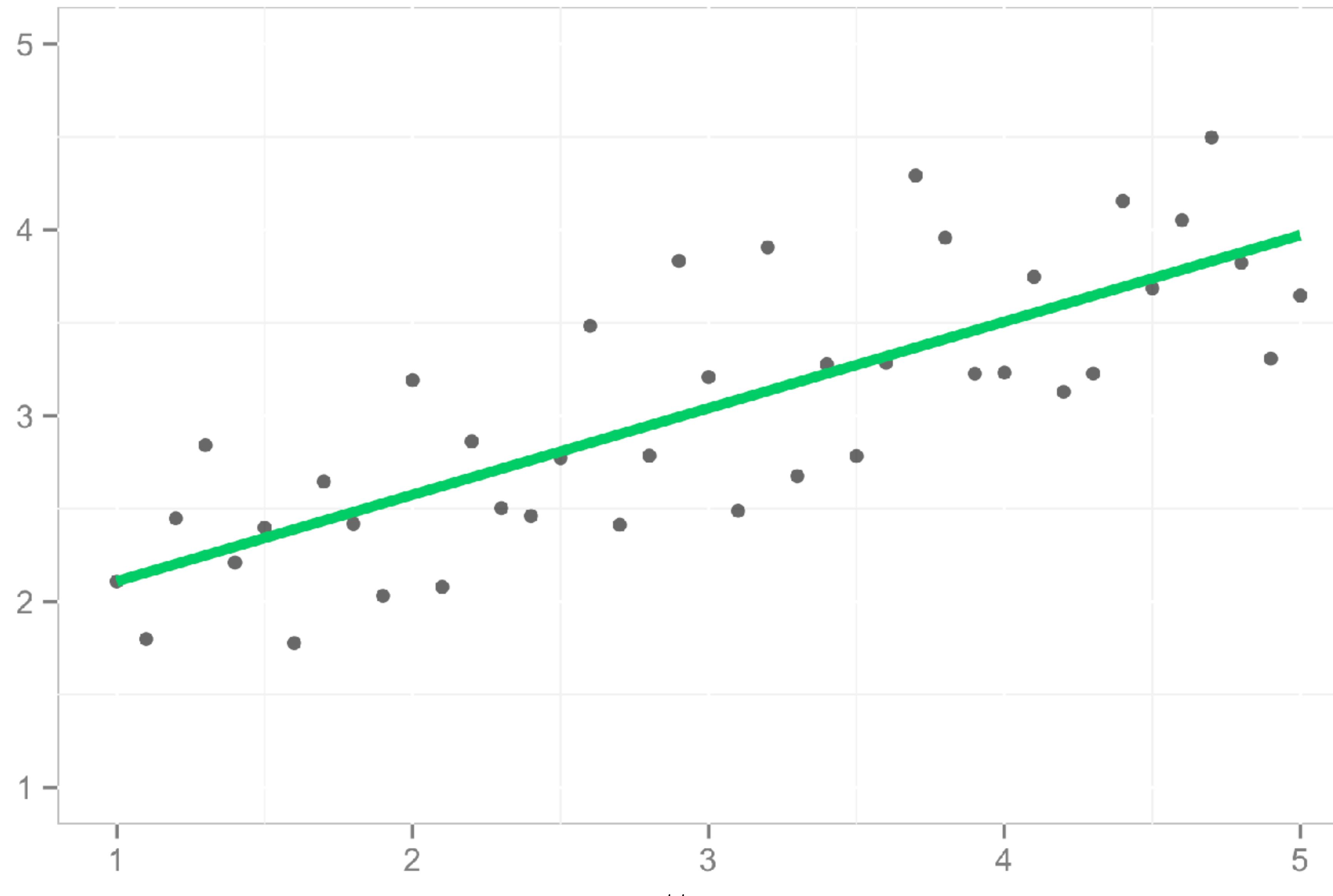


Ronald A. Fisher
(1922)

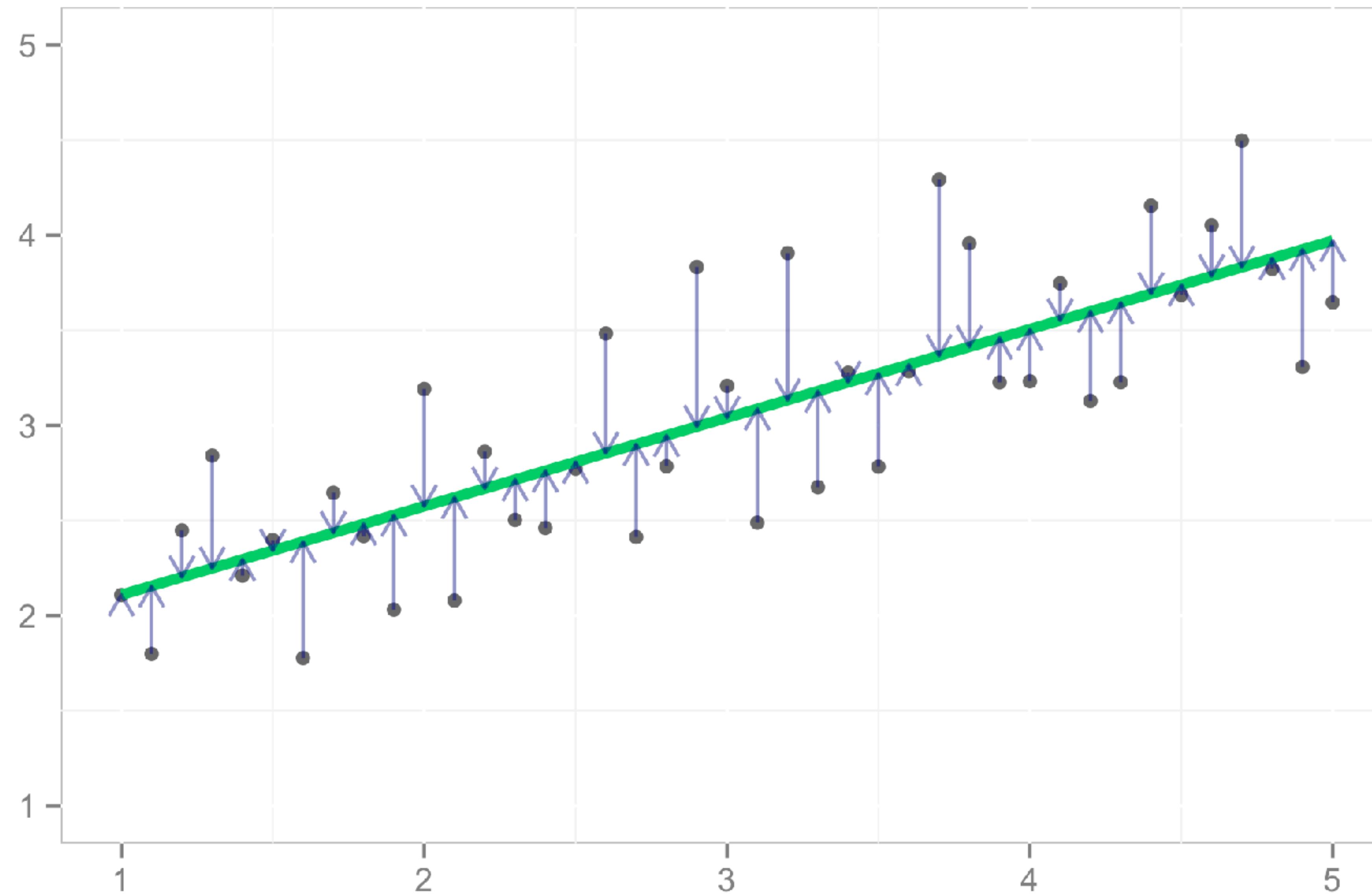
Most Likely Y, given X?



Least Squares



Least Squares



Simple Linear Regression

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$$

...Define the model beforehand

Simple Linear Regression

$$y_i = \underbrace{\beta_0 + \beta_1 x_i}_{\text{Linear Component}} + \underbrace{\varepsilon_i}_{\text{Random Component}}$$

...Define the model beforehand

Simple Linear Regression

$$y_i = \underbrace{\beta_0 + \beta_1 x_i}_{\text{Linear Component}} + \underbrace{\varepsilon_i}_{\text{Random Component}}$$

Dependent Variable
target, label,
outcome, response,
class

Intercept

Slope,
coefficient

Independent Variable
explanatory, predictors,
attributes, descriptors

Random Error

The diagram illustrates the simple linear regression equation $y_i = \beta_0 + \beta_1 x_i + \varepsilon_i$. It is divided into two main parts by a horizontal green line: the 'Linear Component' on the left and the 'Random Component' on the right. The 'Linear Component' consists of the intercept β_0 and the slope coefficient $\beta_1 x_i$. The 'Random Component' consists of the error term ε_i . Arrows point from each part to its corresponding definition: 'Dependent Variable target, label, outcome, response, class' points to the y_i term; 'Intercept' points to the β_0 term; 'Slope, coefficient' points to the $\beta_1 x_i$ term; 'Independent Variable explanatory, predictors, attributes, descriptors' points to the x_i term; and 'Random Error' points to the ε_i term.

...Define the model beforehand

Simple Linear Regression

$$y_i = \underline{\beta_0 + \beta_1 x_i} + \underline{\varepsilon_i}$$

Dependent Variable
target, label

Intercept

Slope, explanatory, features
coefficient

Independent Variable

Random Error

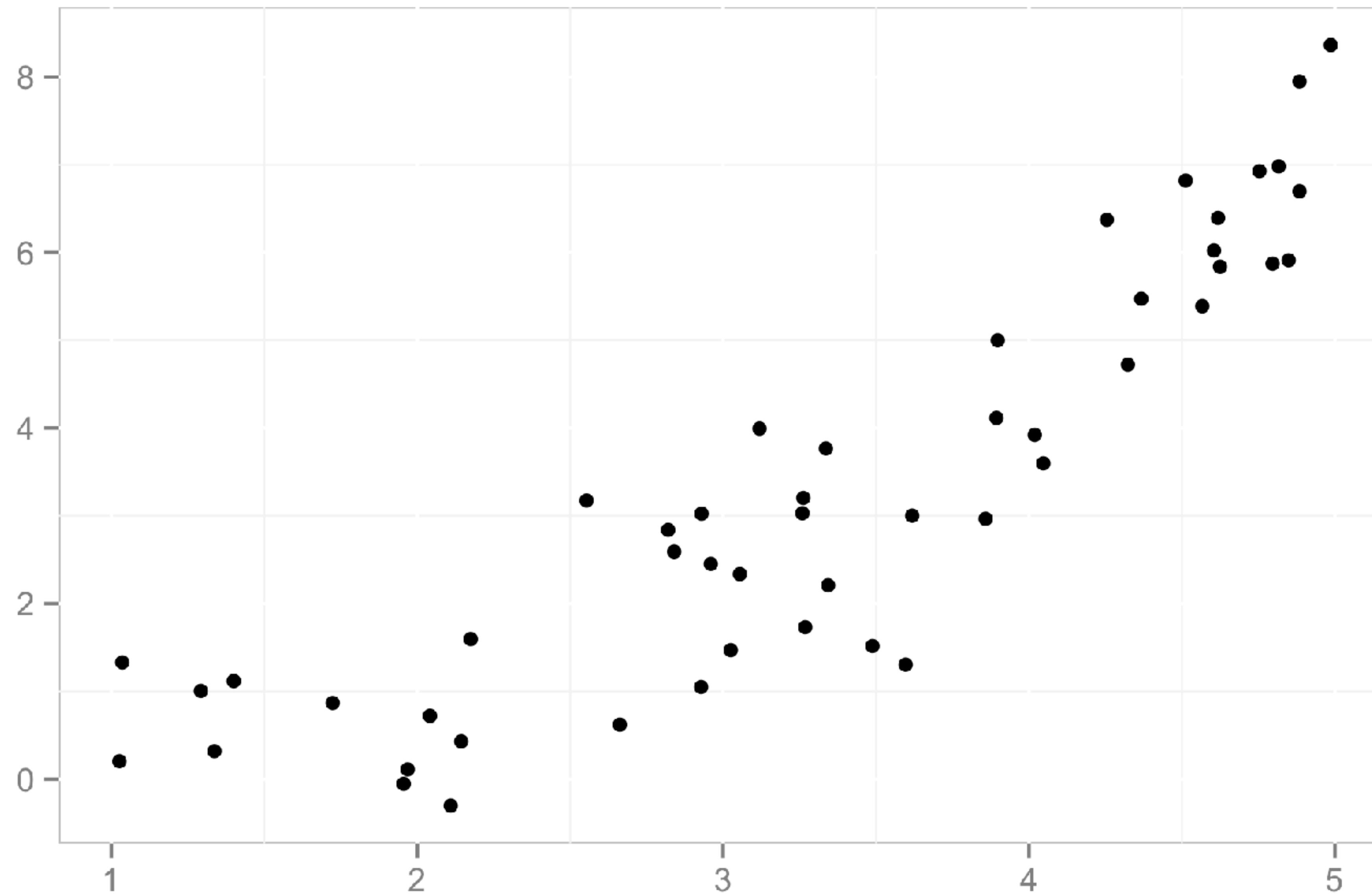
Linear Component

Random Component

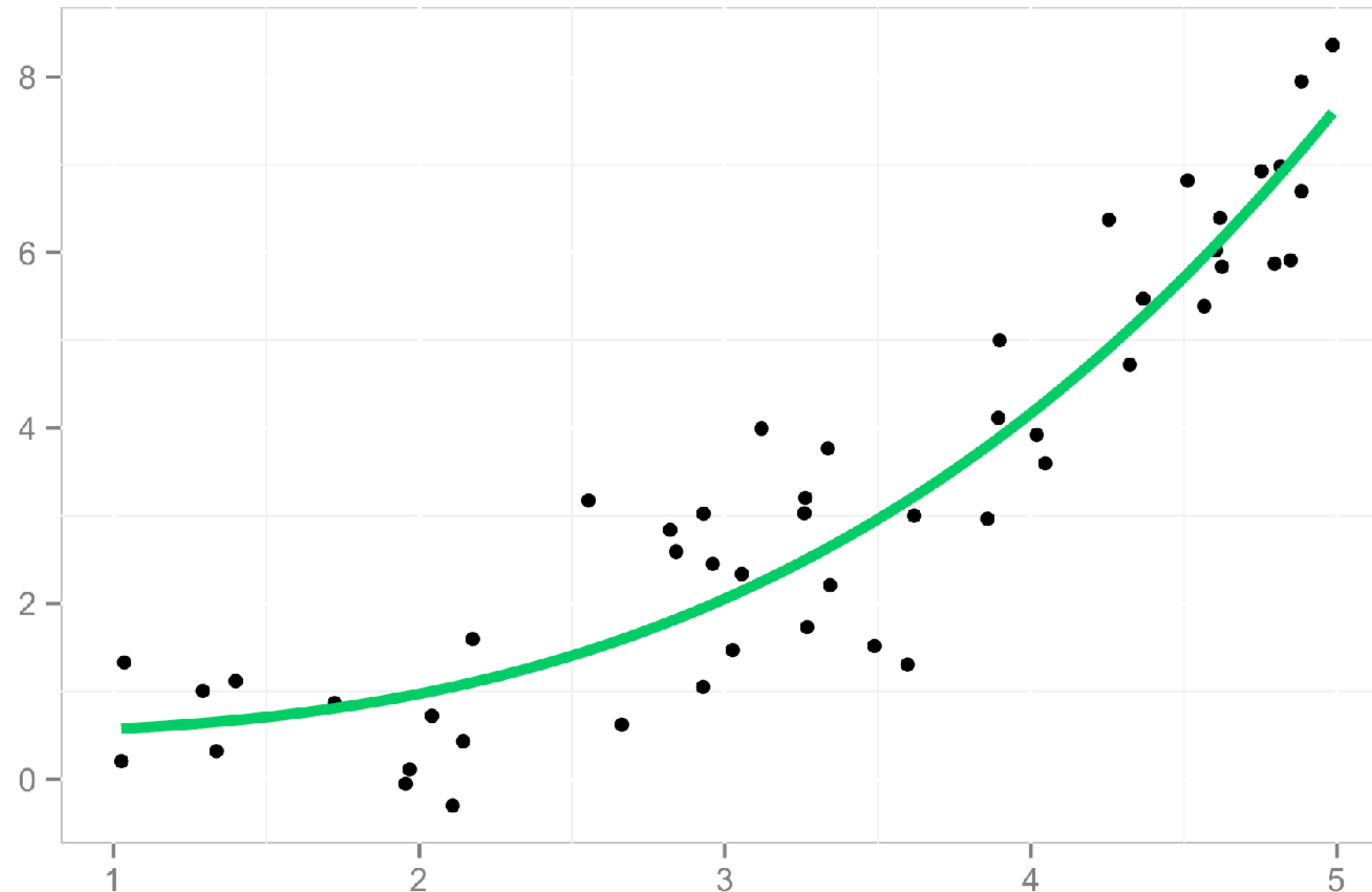
May not be linear

...Define the model beforehand

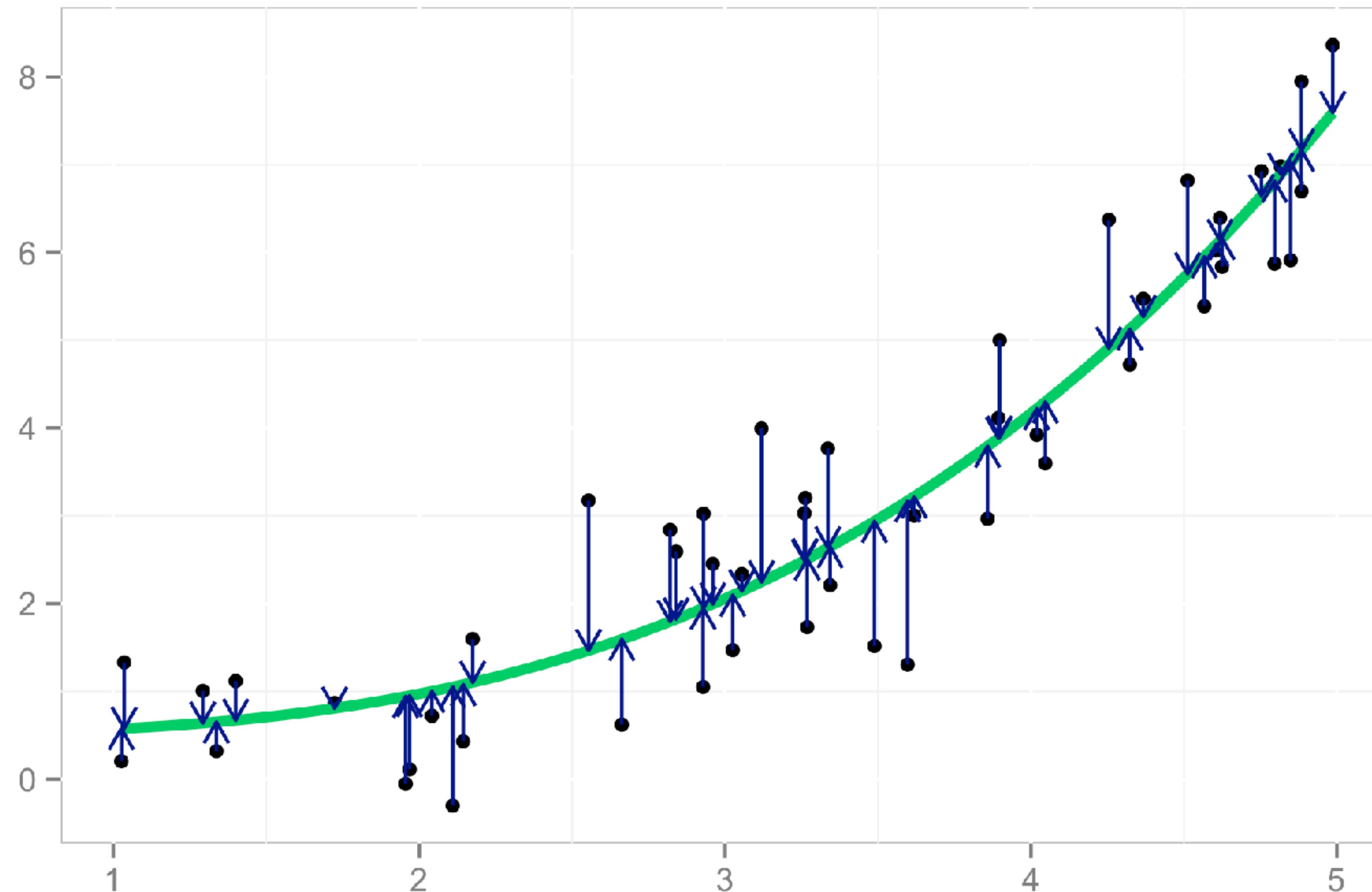
Linear Regression



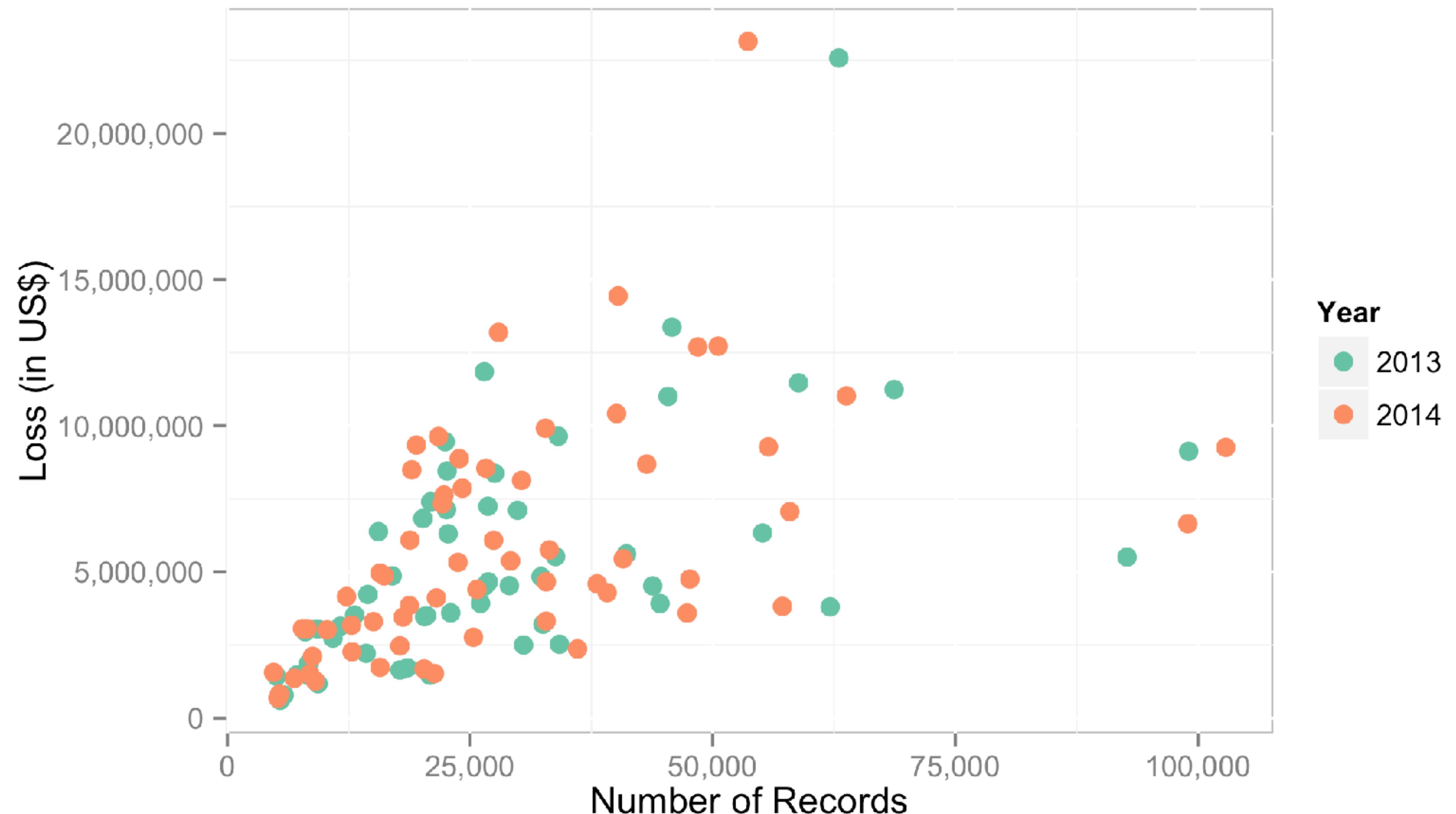
Linear Regression



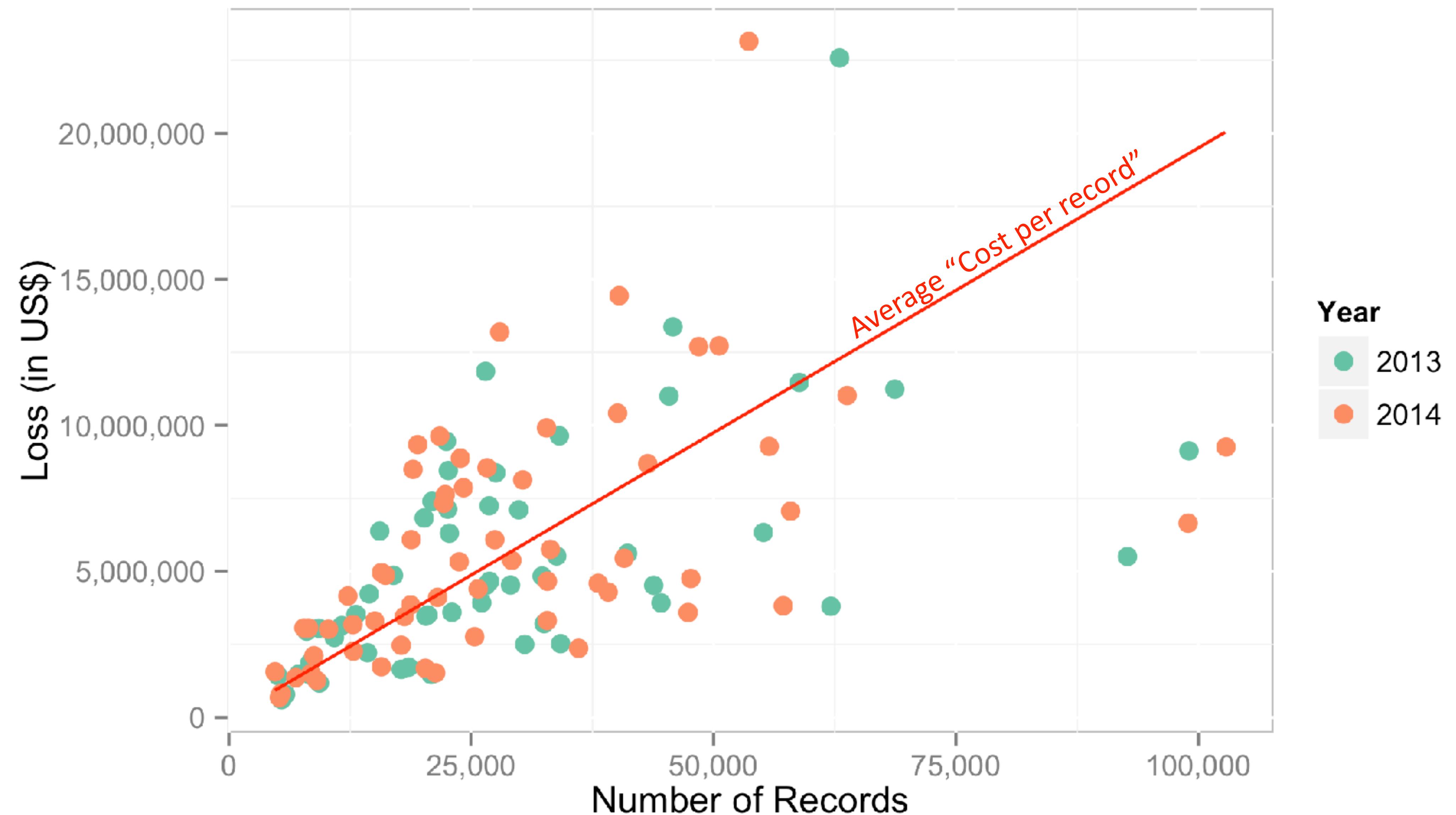
Linear Regression



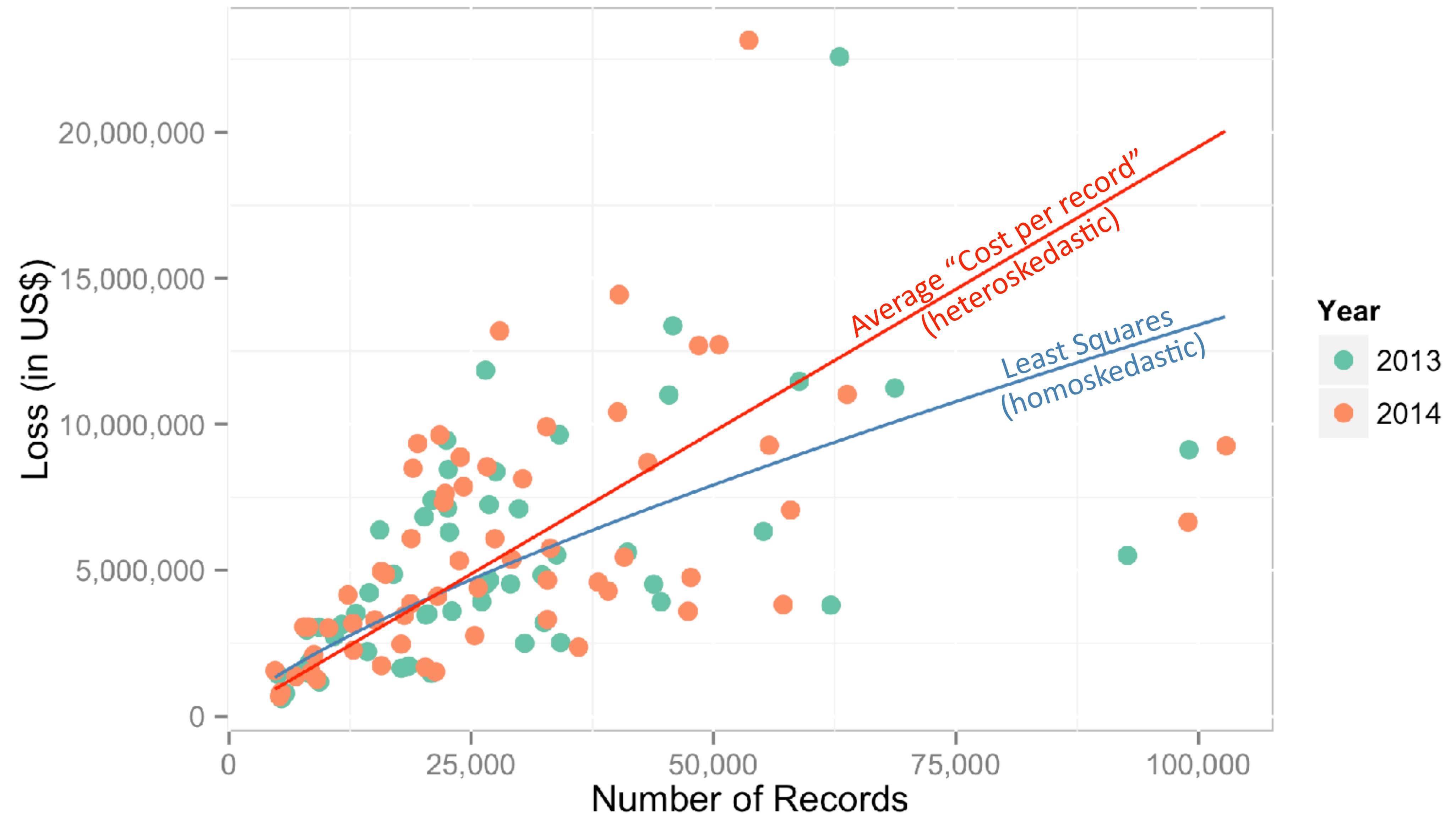
Cost per Record



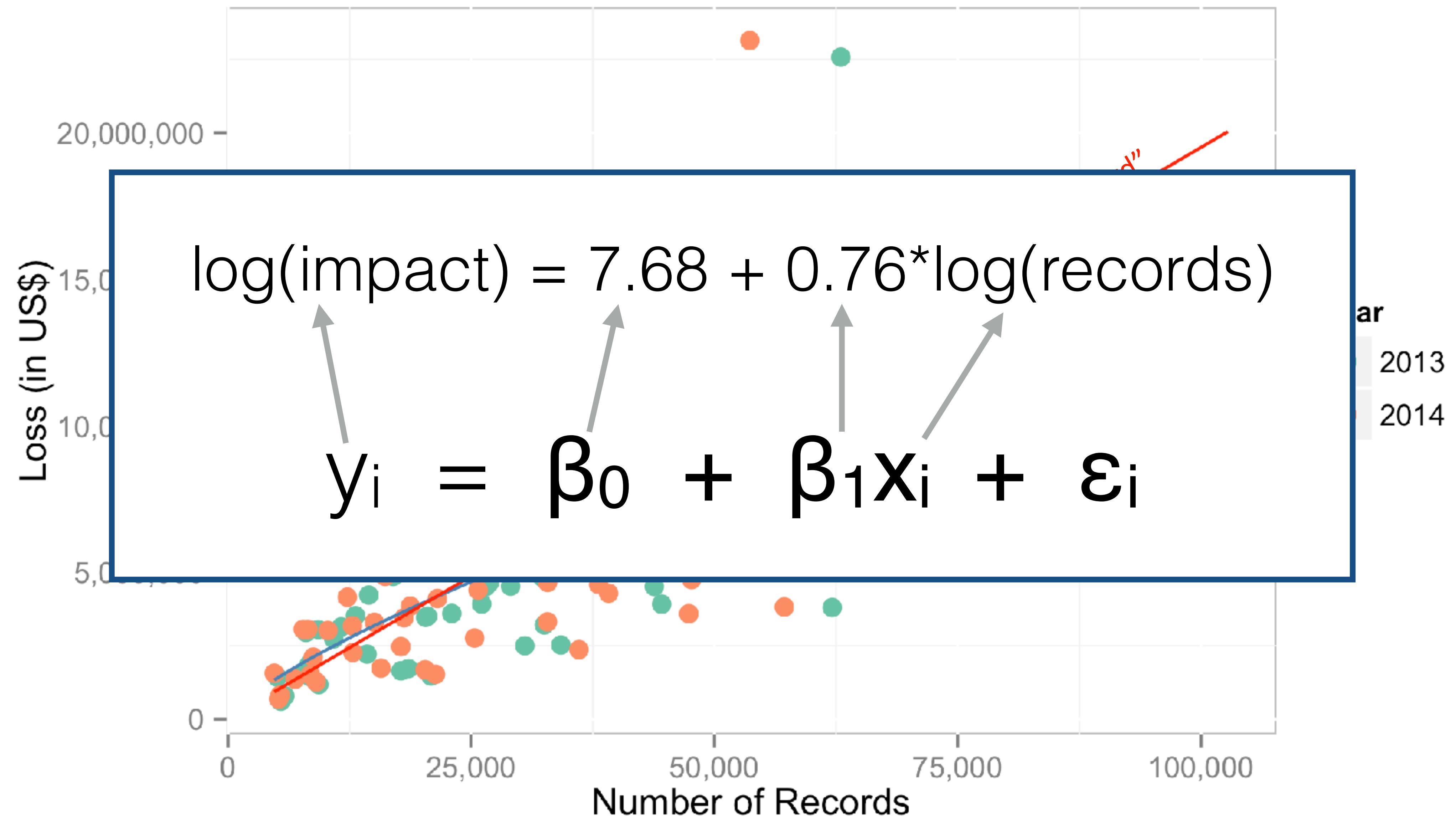
Cost per Record



Cost per Record



Cost per Record



linear regression in R

```
> lm(log(total) ~ log(records), data=ponemon)
##
## Call:
## lm(formula = log(total) ~ log(records), data = ponemon)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.0243 -0.3792  0.0204  0.4197  1.0188
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t| )
## (Intercept) 7.6800    0.7013   10.9   <2e-16 ***
## log(records) 0.7584    0.0697   10.9   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.523 on 113 degrees of freedom
## Multiple R-squared:  0.512, Adjusted R-squared:  0.508
## F-statistic: 119 on 1 and 113 DF,  p-value: <2e-16
```

```
> lm(log(total) ~ log(records), data=ponemon)
##
## Call:
## lm(formula = log(total) ~ log(records), data = ponemon)
##
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -1.0243 -0.3792  0.0204  0.4197  1.0188 
##
## Coefficients: Coefficients
##                 Estimate Std. Error t value Pr(>|t|)    
## (Intercept)    7.6800    0.7013   10.9 <2e-16 ***  
## log(records)  0.7584    0.0697   10.9 <2e-16 ***  
## 
## --- 
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.523 on 113 degrees of freedom
## Multiple R-squared:  0.512, Adjusted R-squared:  0.508 
## F-statistic: 119 on 1 and 113 DF, p-value: <2e-16
```

Look for uncorrelated, constant variance, mean=0,

P-Values of independent variables

R² value

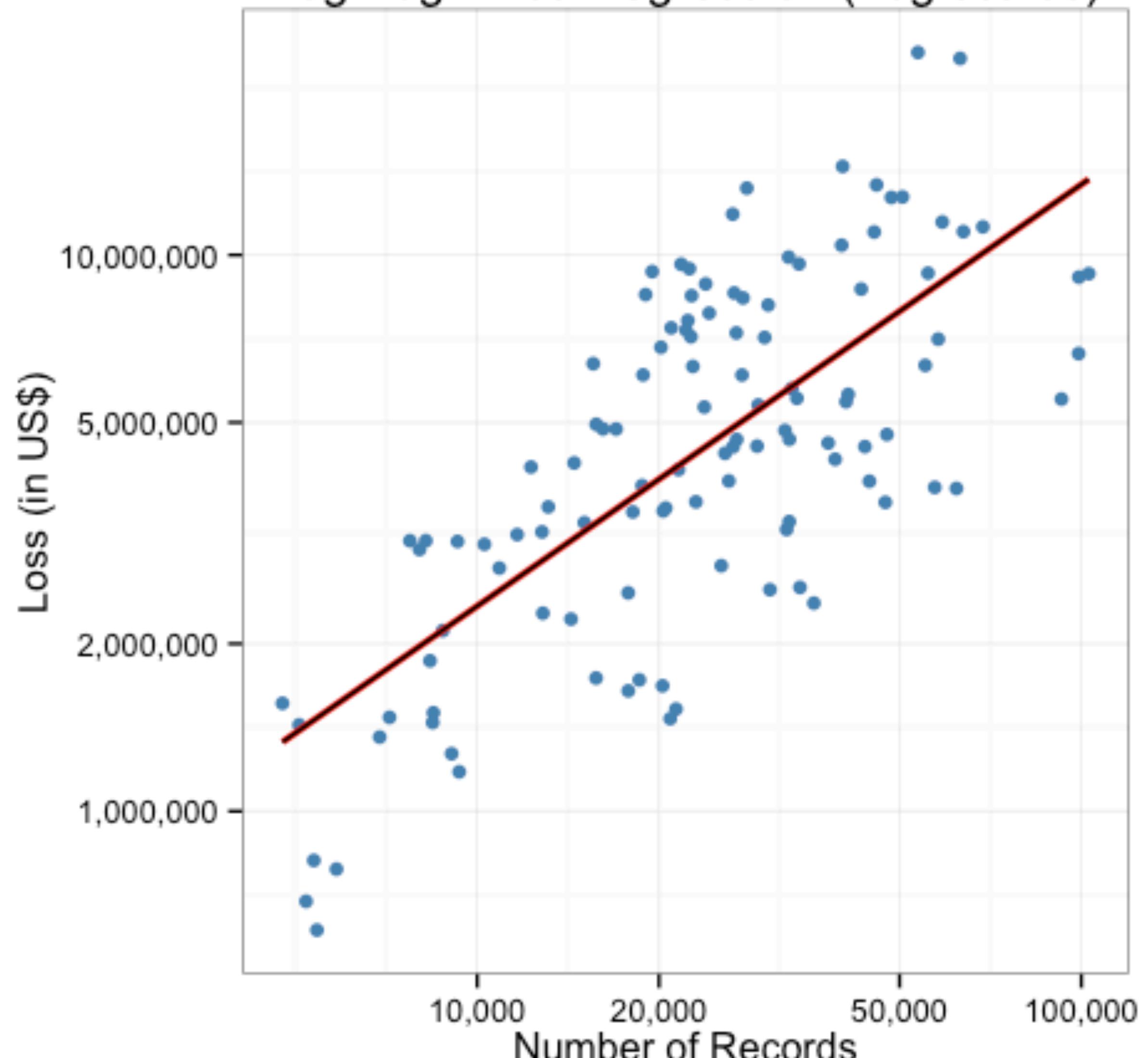
P-Value of model

Cost per Record

$$\log(\text{impact}) = 7.68 + 0.76 * \log(\text{records})$$

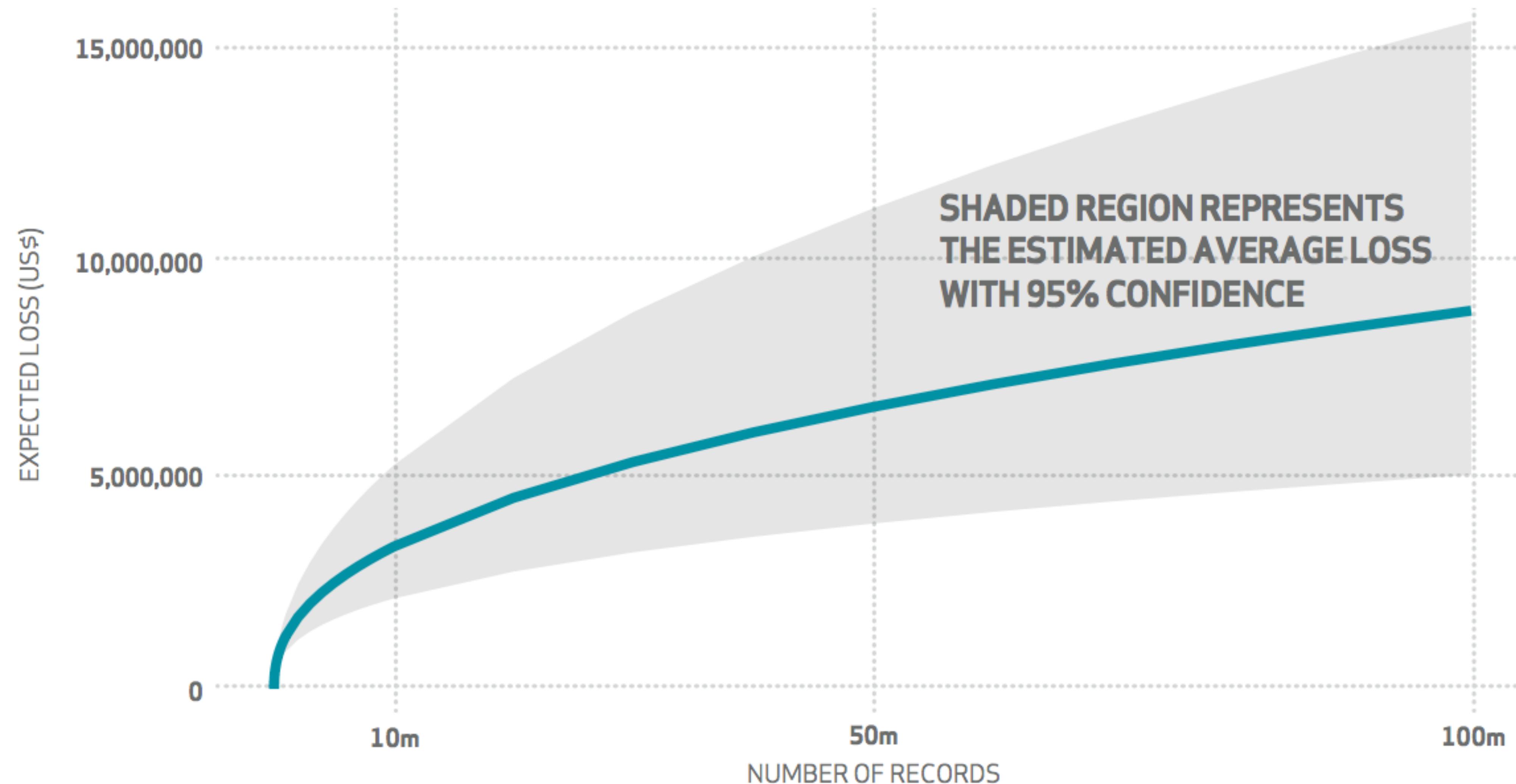
For every 1% increase in records, there is a 0.76% increase in loss.

Log-Log Linear regression (Log scales)

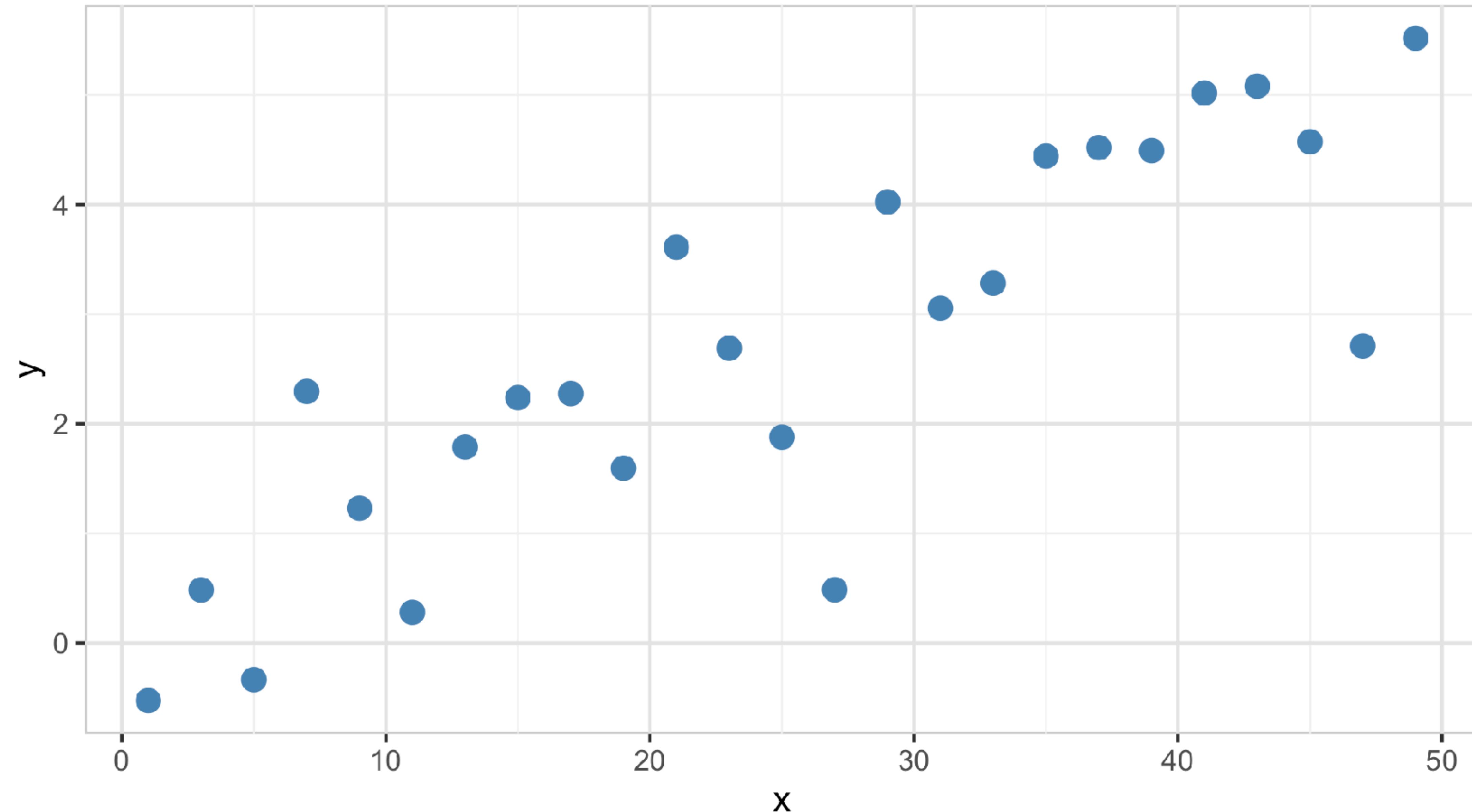


Different data, Still a log-log model

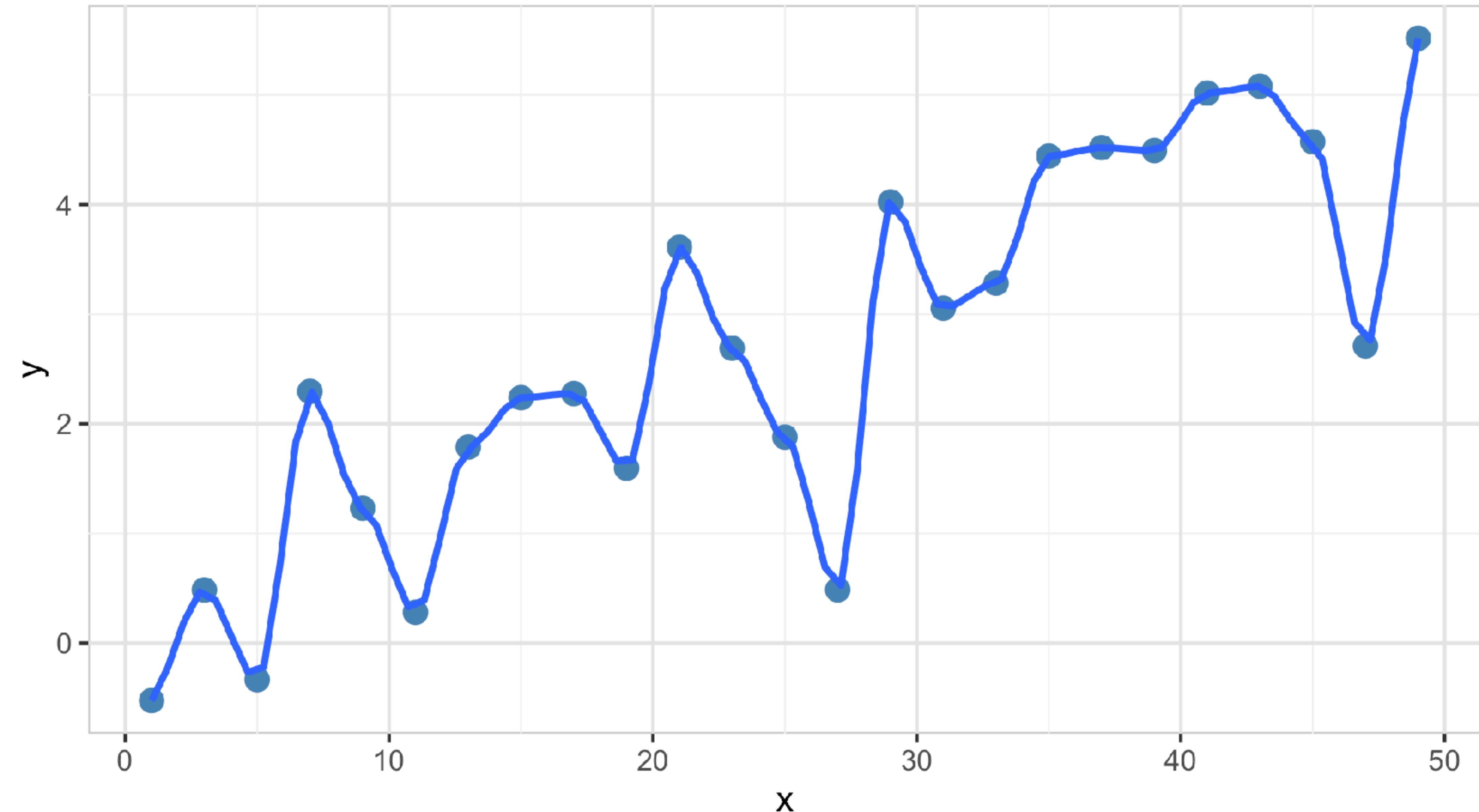
“...there’s a lot of stuff contributing to the cost of breaches besides the number of records lost.”



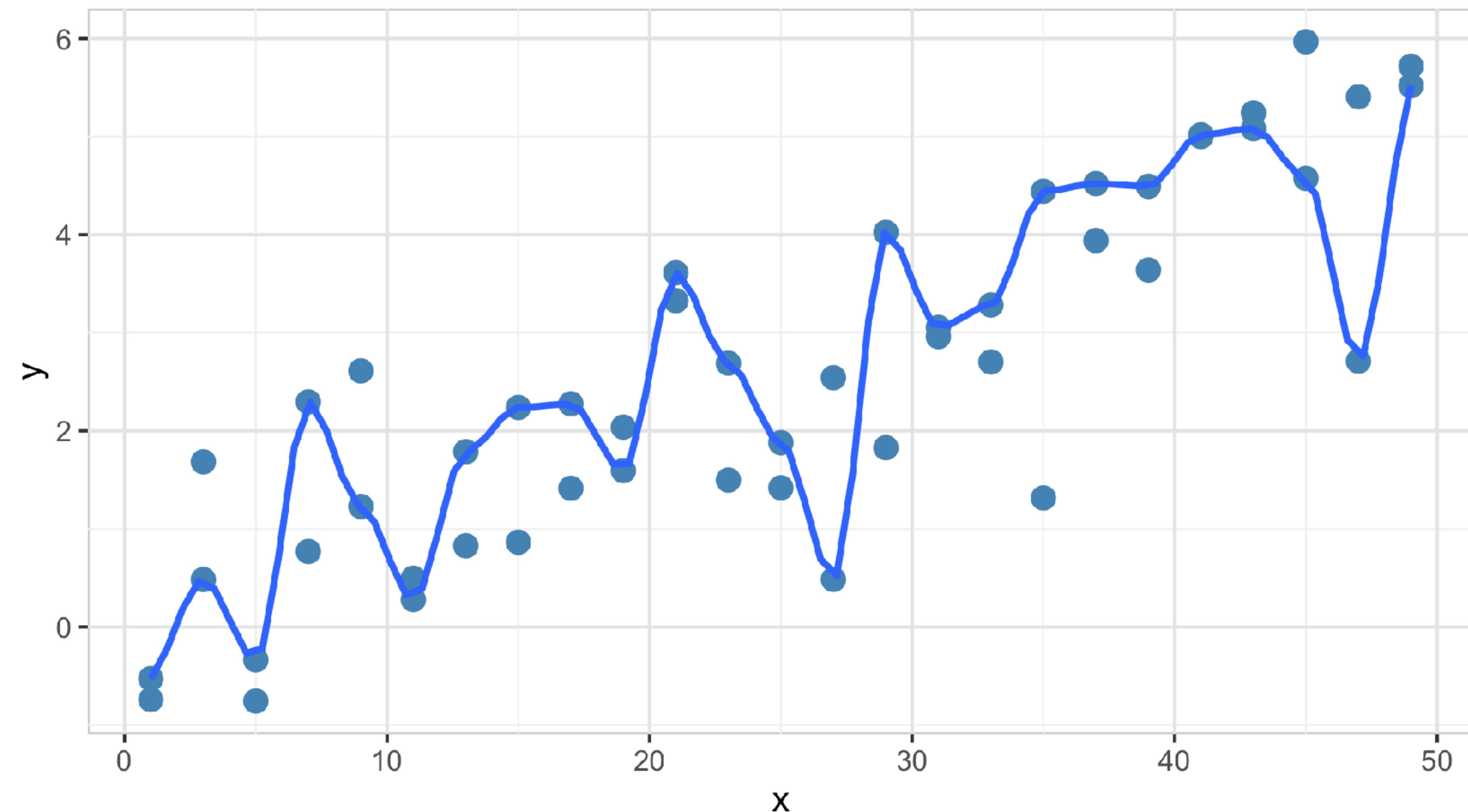
The challenge of over fitting



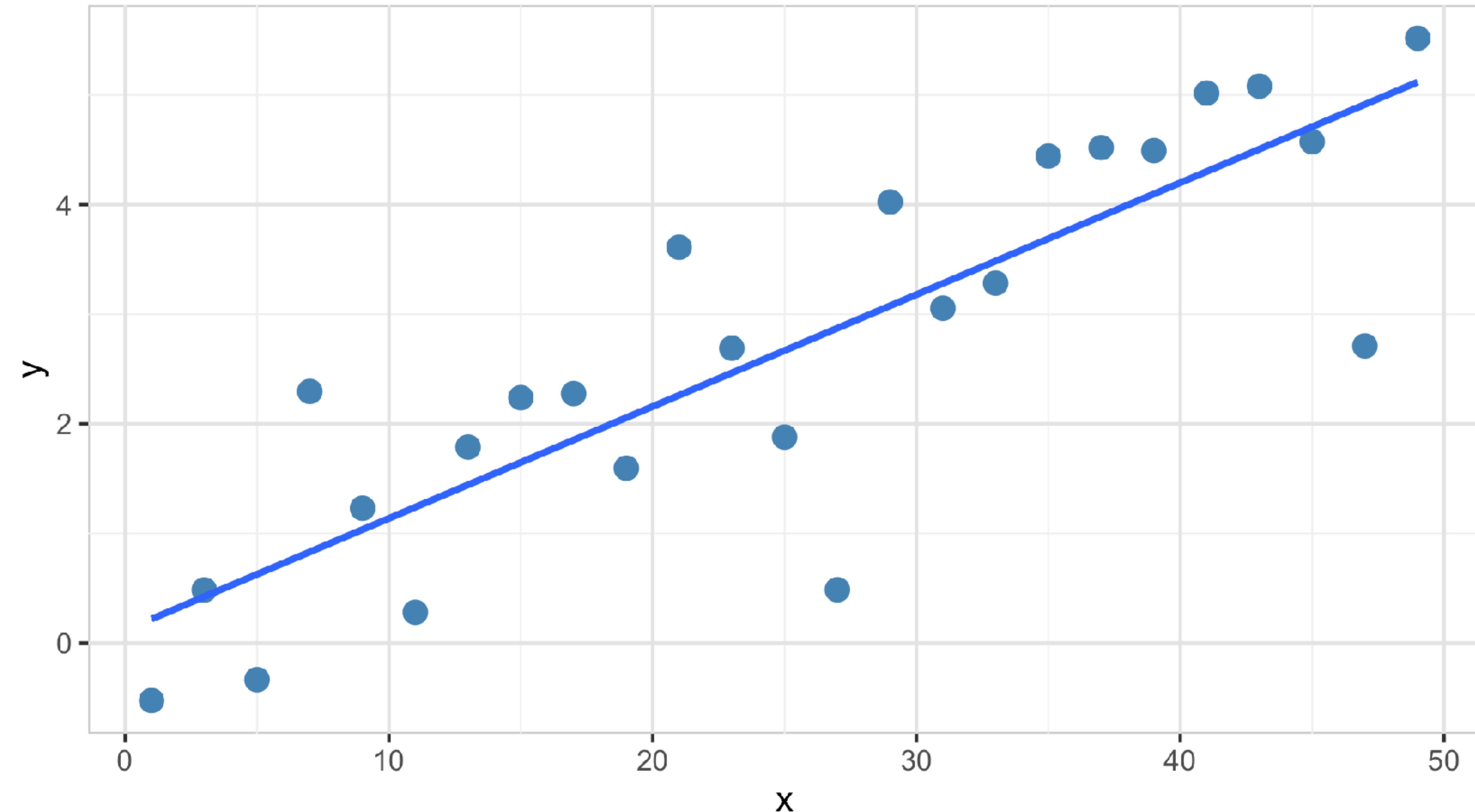
The challenge of over fitting



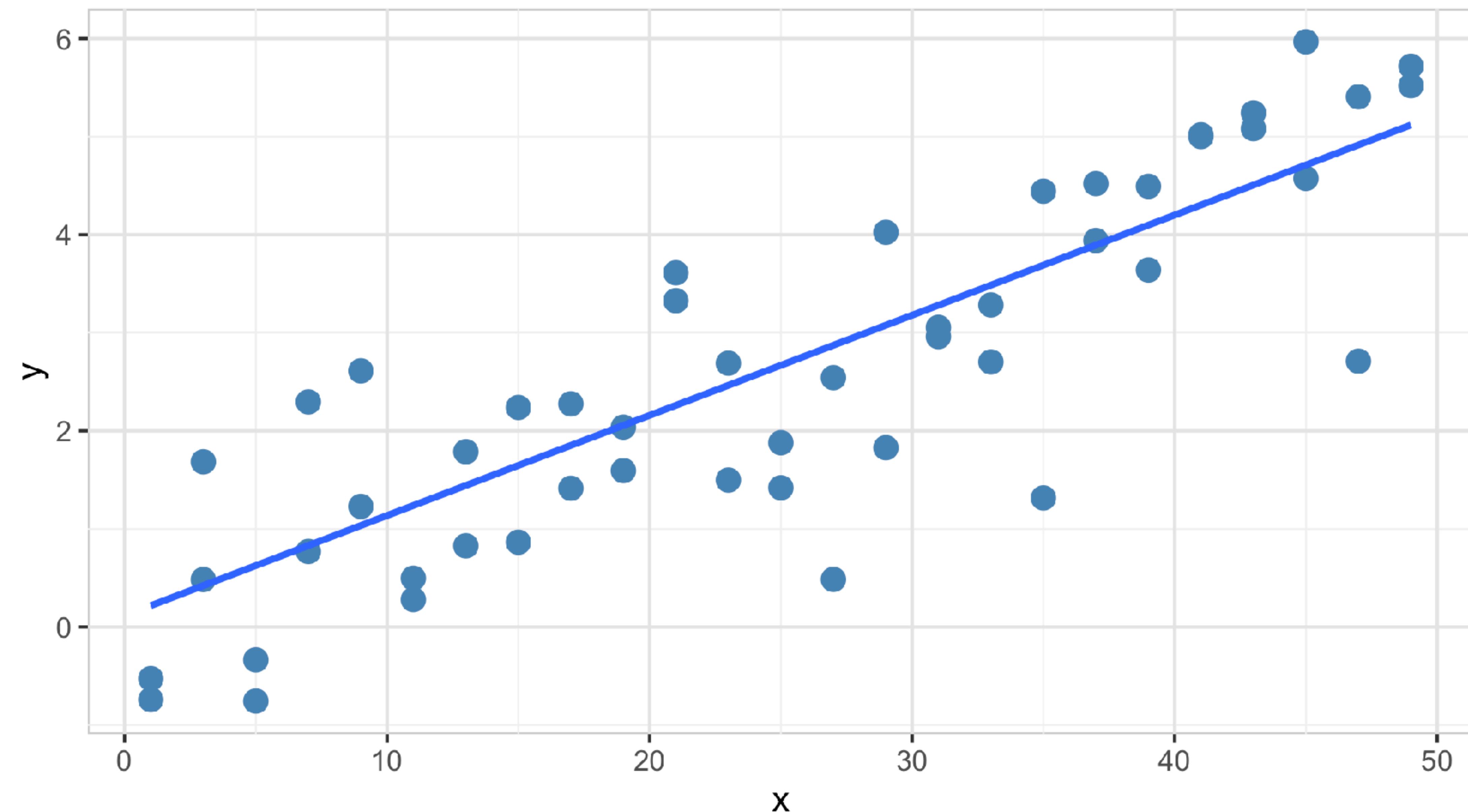
The challenge of over fitting



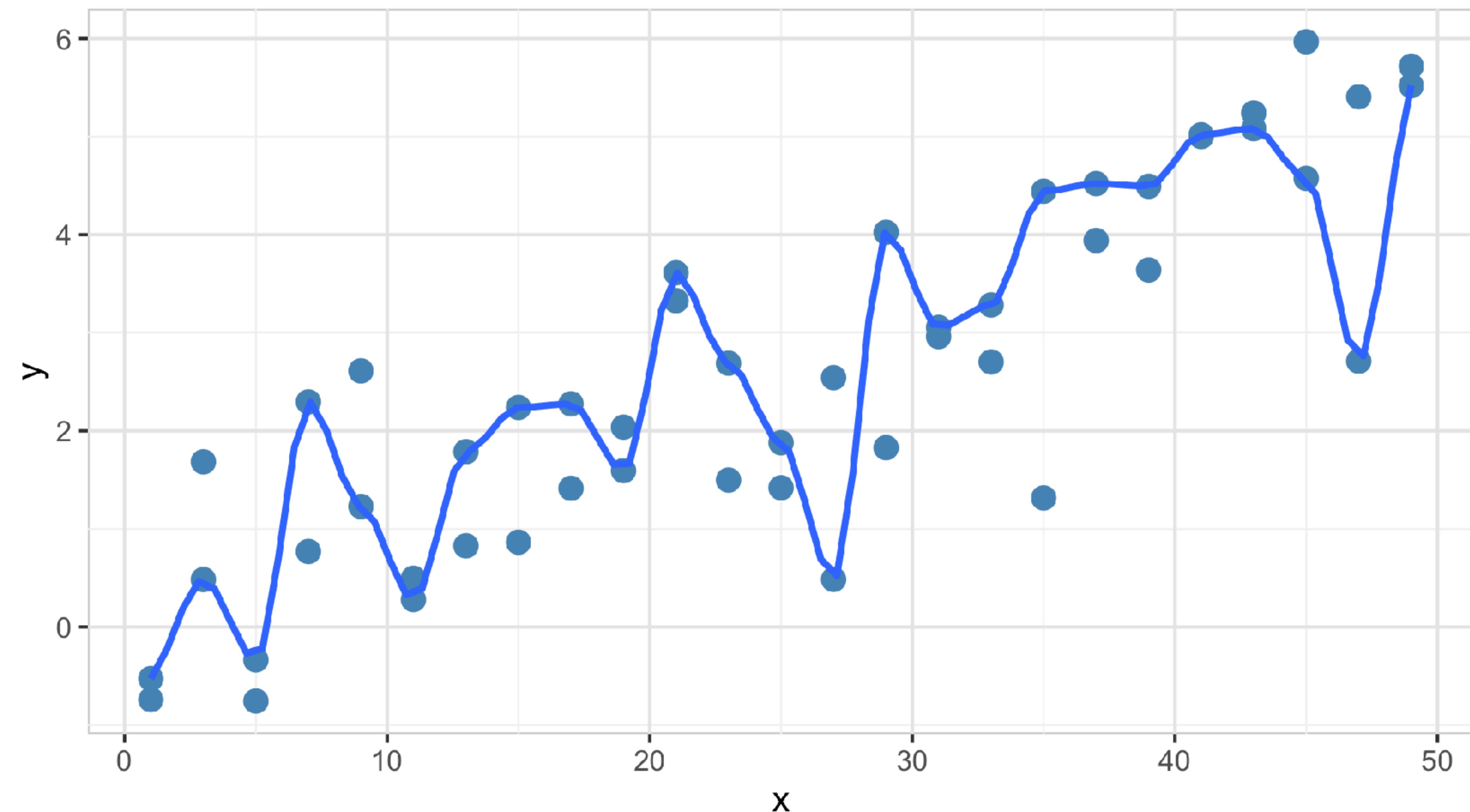
The challenge of over fitting



The challenge of over fitting



The challenge of over fitting



How to screw-up regression analysis

- Non-Linear relationships
 - Assumes relationship is linear and additive
- Skewed residuals
 - normality of the error distribution
- Variables are not Independent
 - should have little to no multicollinearity
- Homoscedasticity
 - Variance of error is consistently constant

O'REILLY®

Security

BUILD BETTER DEFENSES

Training and Testing

oreillysecuritycon.com

#oreillysecurity

Question:

How could we avoid overfitting a model?

Answer:

Question:

How could we avoid overfitting a model?

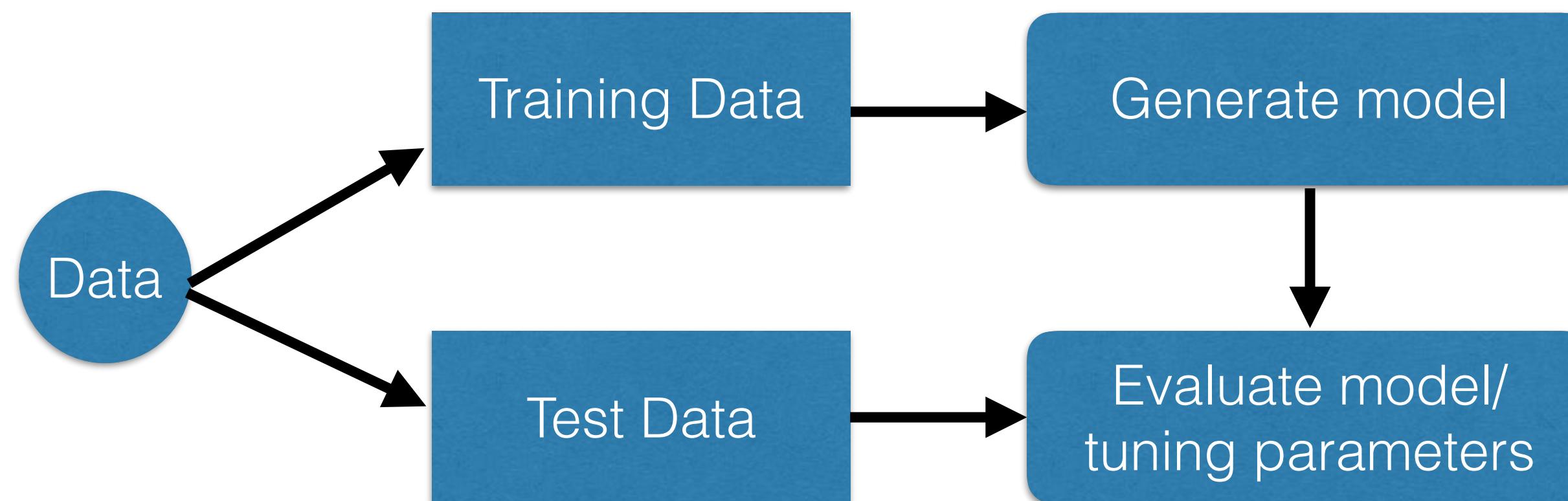
Answer:

Evaluate the performance of model with new data
(or data that was not used to train the model)

Data Splitting

Create two data sets by randomly assigning observations into either a training or testing data set.

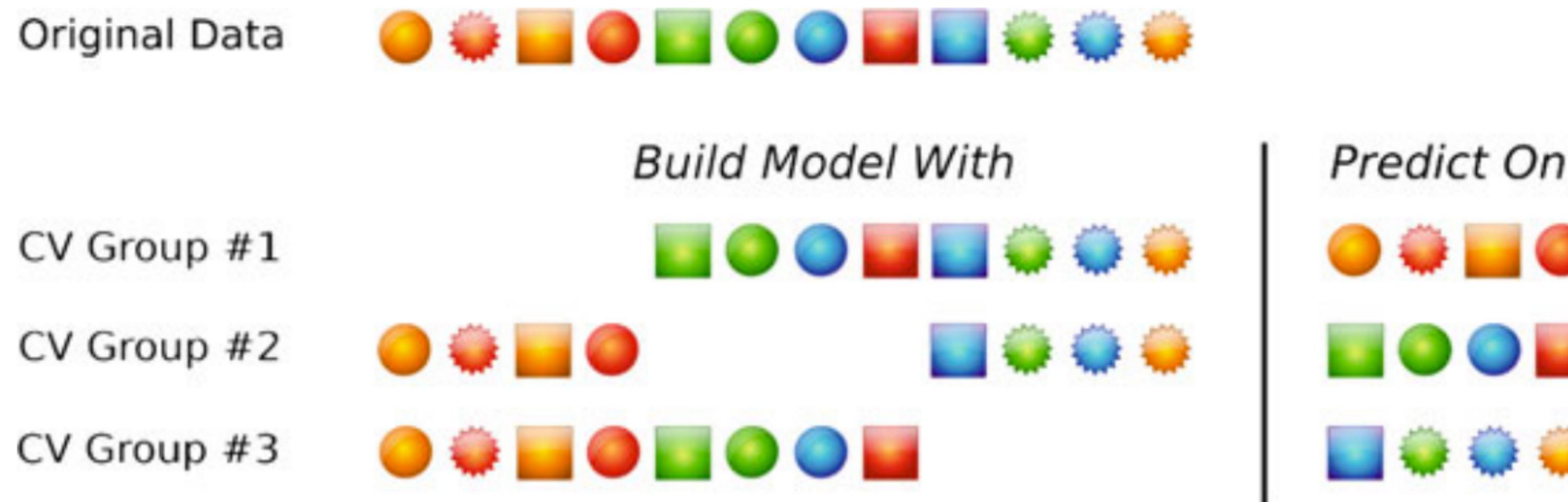
- **Training Data:** observations used to produce a model
- **Test Data:** observations used to measure model performance and set tuning parameters



K-Fold Cross-Validation

Cross-Validation is useful to compare models or find tuning parameters

- Data is partitioned into K sets of roughly equal size
- loop through k-times training with the main set, testing with the hold out set
- when $k = n$, “leave-one-out” cross-validation



Other ways to (not) split

Bias may be introduced with non-random samples, but some alternatives:

- Split on some element of time
 - If collected by days, use all but the last day to train and test on the last day.
 - Split on another variable (example: data from departments)

A Non-Random split makes the **assumption** that there is no difference between whatever is being split upon.

(e.g. if all the data was collected on weekdays, but the last day is a weekend, or departments have differences in the data collected)

Missing Values

Security Data is generally EXTREMELY messy and is often a “convenience” sample, there will be **missing values...**

- See if you can figure *why* they are missing
 - Indication of larger problem
 - Maybe missing values are informative (missing p2p)
- Ways to deal with missing data:
 - Leave it missing (some models deal with [NA/None] well)
 - Drop incomplete cases within the data
 - *Impute* new data (create a model for missing data) (knn)

Other Considerations

- Zero-Variance Predictors (or NZV)
- Collinearity and Multicollinearity
- Dummy Variables (factors in R)
- Adding/Deriving variables
- Binning (resist the urge)

O'REILLY®

Security

BUILD BETTER DEFENSES

Overview of supervised learning

oreillysecuritycon.com

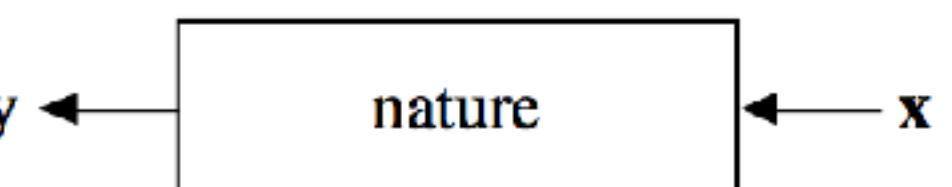
#oreillysecurity

Statistical Modeling: The Two Cultures

Leo Breiman

1. INTRODUCTION

Statistics starts with data. Think of the data as being generated by a black box in which a vector of input variables \mathbf{x} (independent variables) go in one side, and on the other side the response variables \mathbf{y} come out. Inside the black box, nature functions to associate the predictor variables with the response variables, so the picture is like this:



There are two goals in analyzing the data:

Prediction. To be able to predict what the responses are going to be to future input variables;

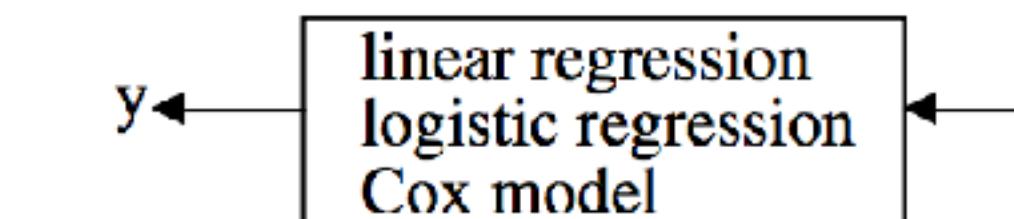
Information. To extract some information about how nature is associating the response variables to the input variables.

There are two different approaches toward these goals:

The Data Modeling Culture

The analysis in this culture starts with assuming a stochastic data model for the inside of the black box. For example, a common data model is that data are generated by independent draws from response variables = $f(\text{predictor variables}, \text{random noise, parameters})$

The values of the parameters are estimated from the data and the model then used for information and/or prediction. Thus the black box is filled in like this:

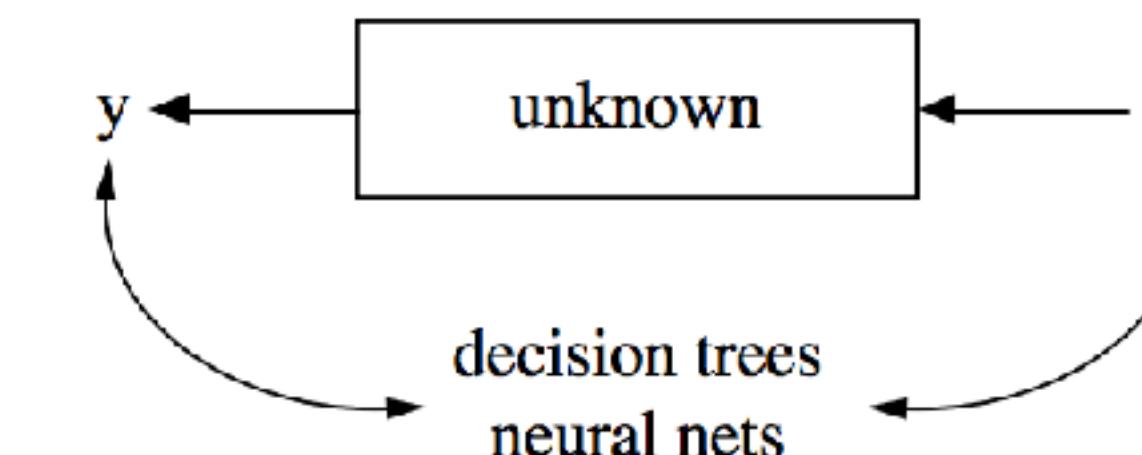


Model validation. Yes–no using goodness-of-fit tests and residual examination.

Estimated culture population. 98% of all statisticians.

The Algorithmic Modeling Culture

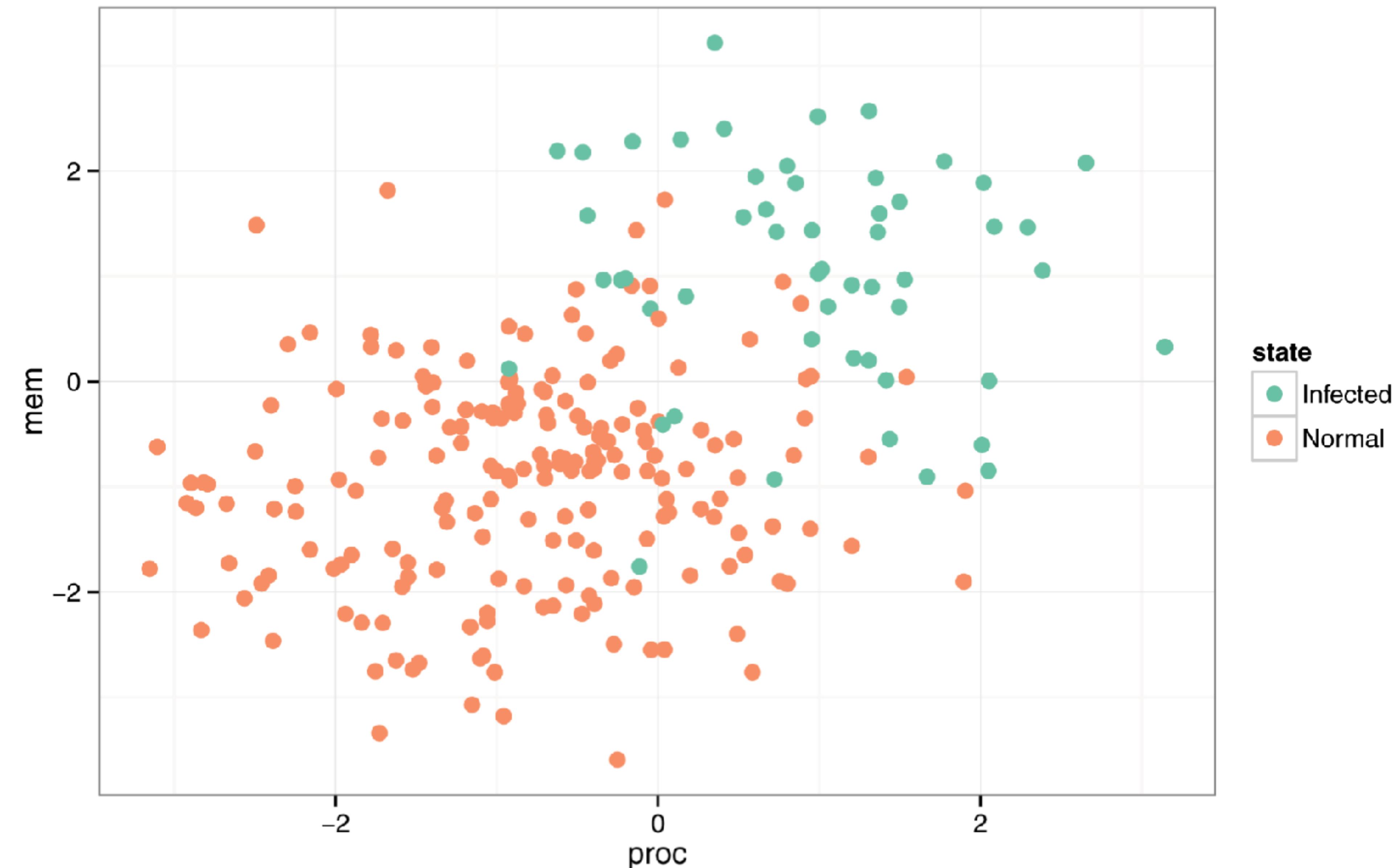
The analysis in this culture considers the inside of the box complex and unknown. Their approach is to find a function $f(\mathbf{x})$ —an algorithm that operates on \mathbf{x} to predict the responses \mathbf{y} . Their black box looks like this:



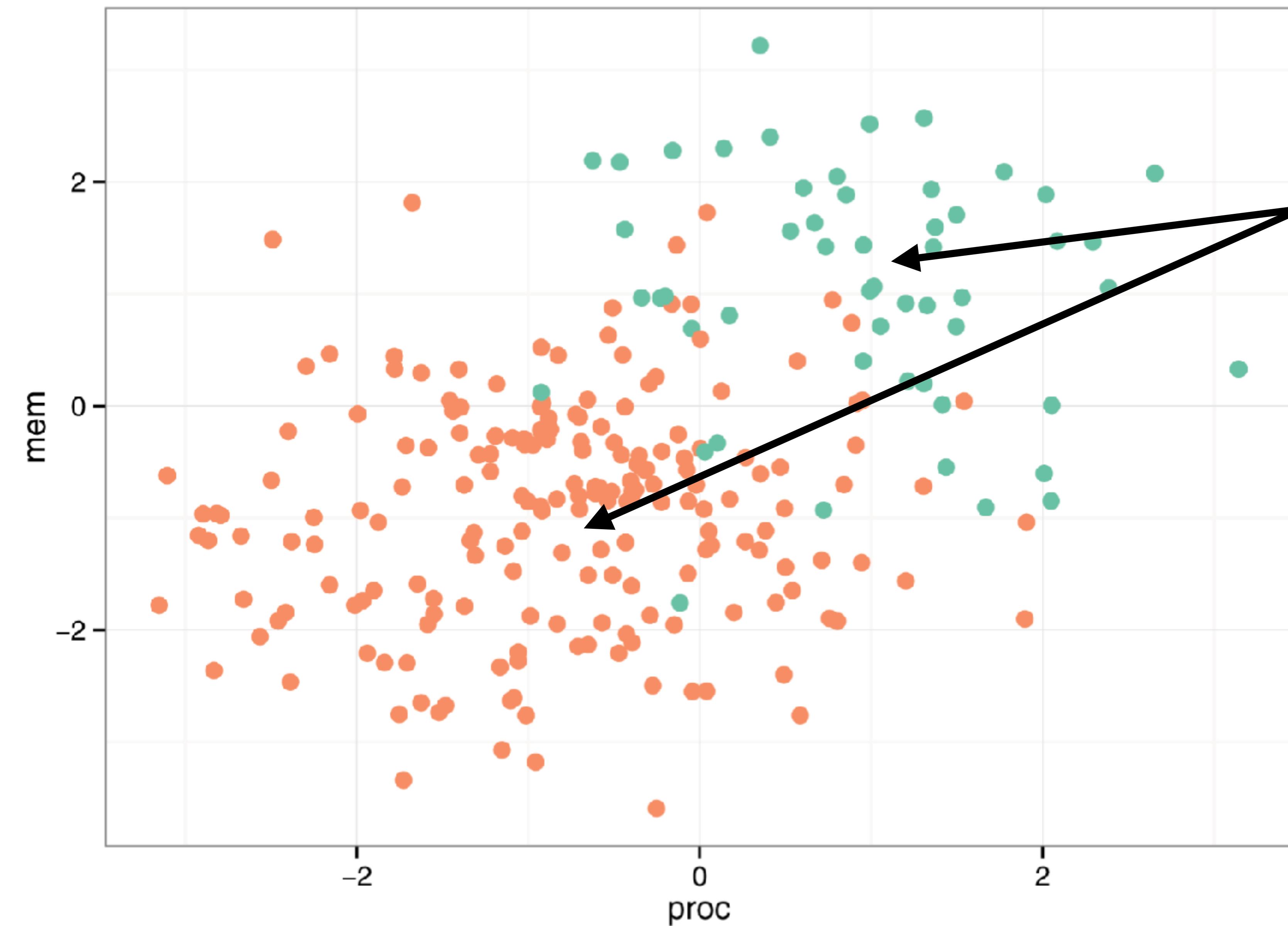
Model validation. Measured by predictive accuracy.

Estimated culture population. 2% of statisticians, many in other fields.

Teaching the machine to learn



Teaching the machine to learn



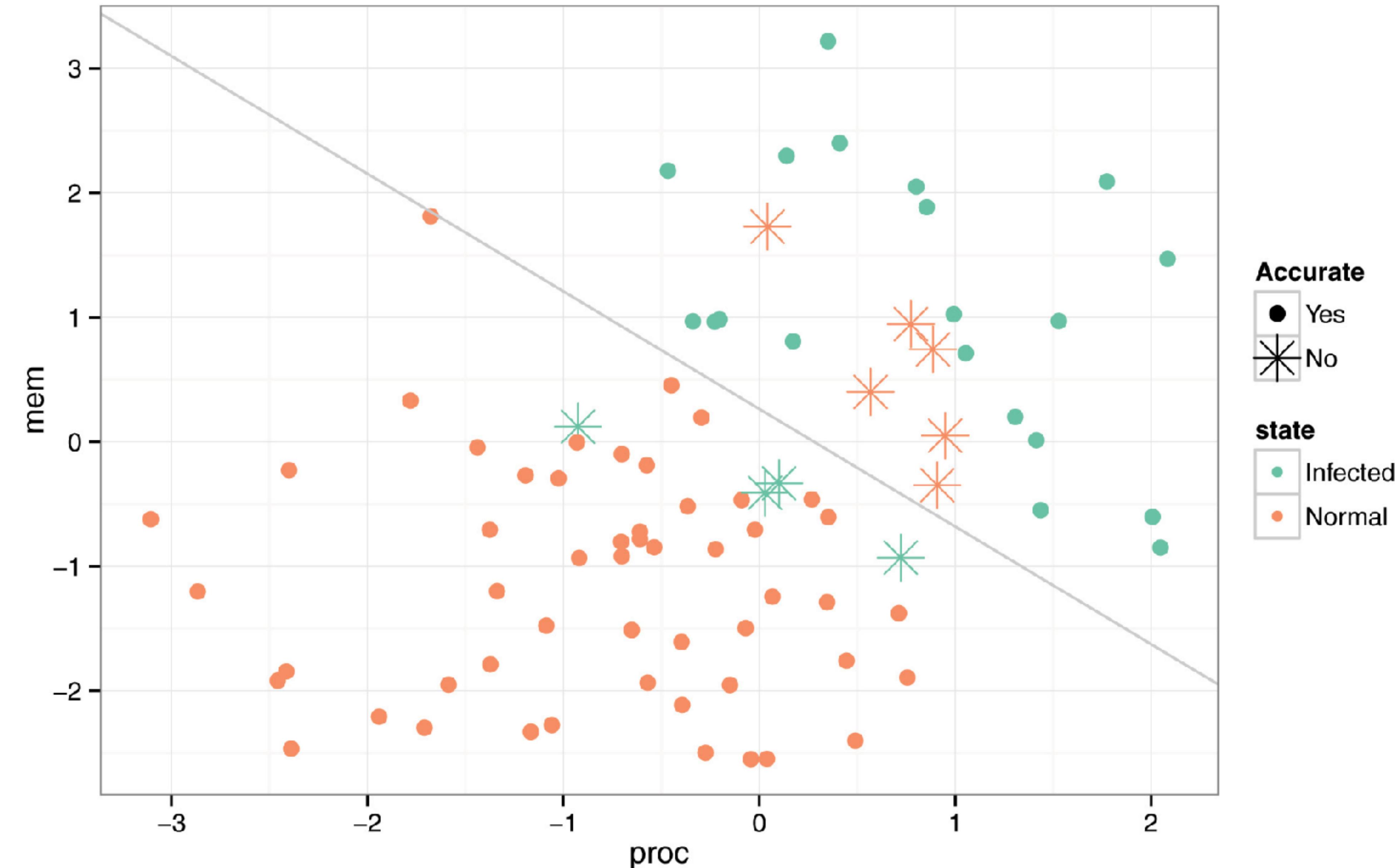
1) Get mean of *proc* and *mem* for both classes

state

- Infected
- Normal

2) Assign class based on which centroid is closer

Teaching the machine to learn...



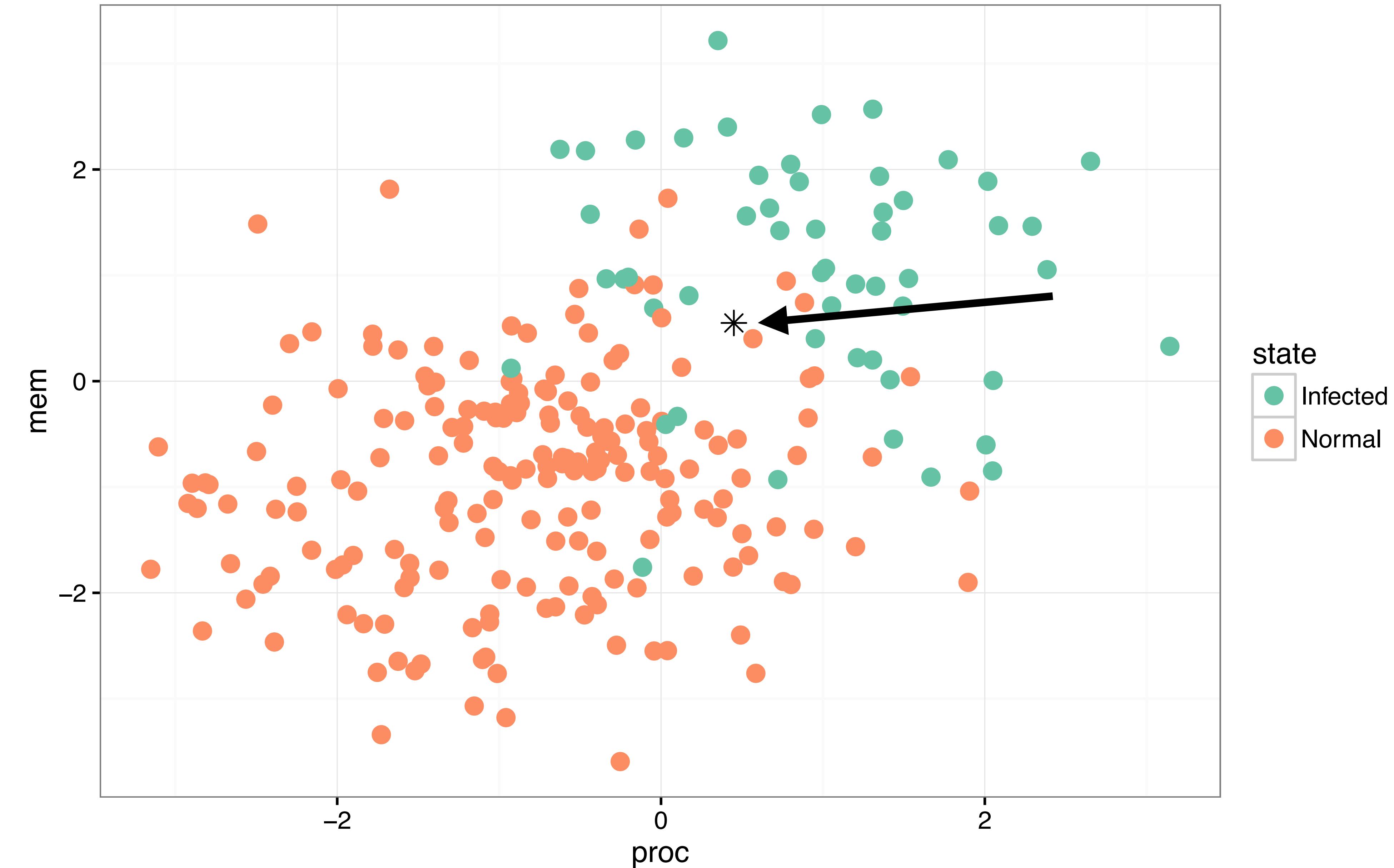
What if rather than look at all of the variables as a group, we just assume that any unknown value will probably be like its neighbors?

Example: k-nearest neighbors

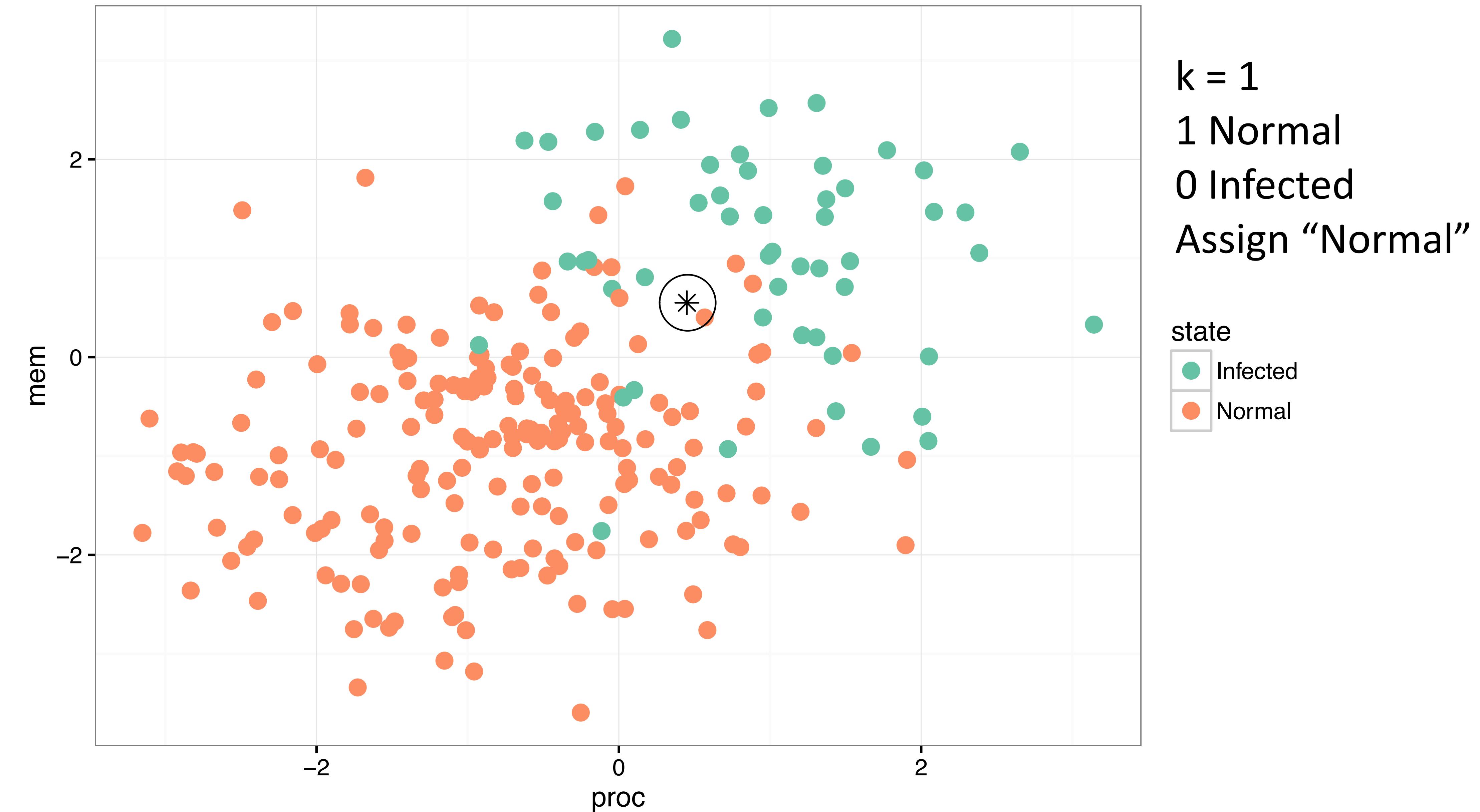
Define K, and for every unknown observation:

1. Find k-nearest neighbors (by *distance*)
2. Assign class based on dominate class
3. Optional weight by distance

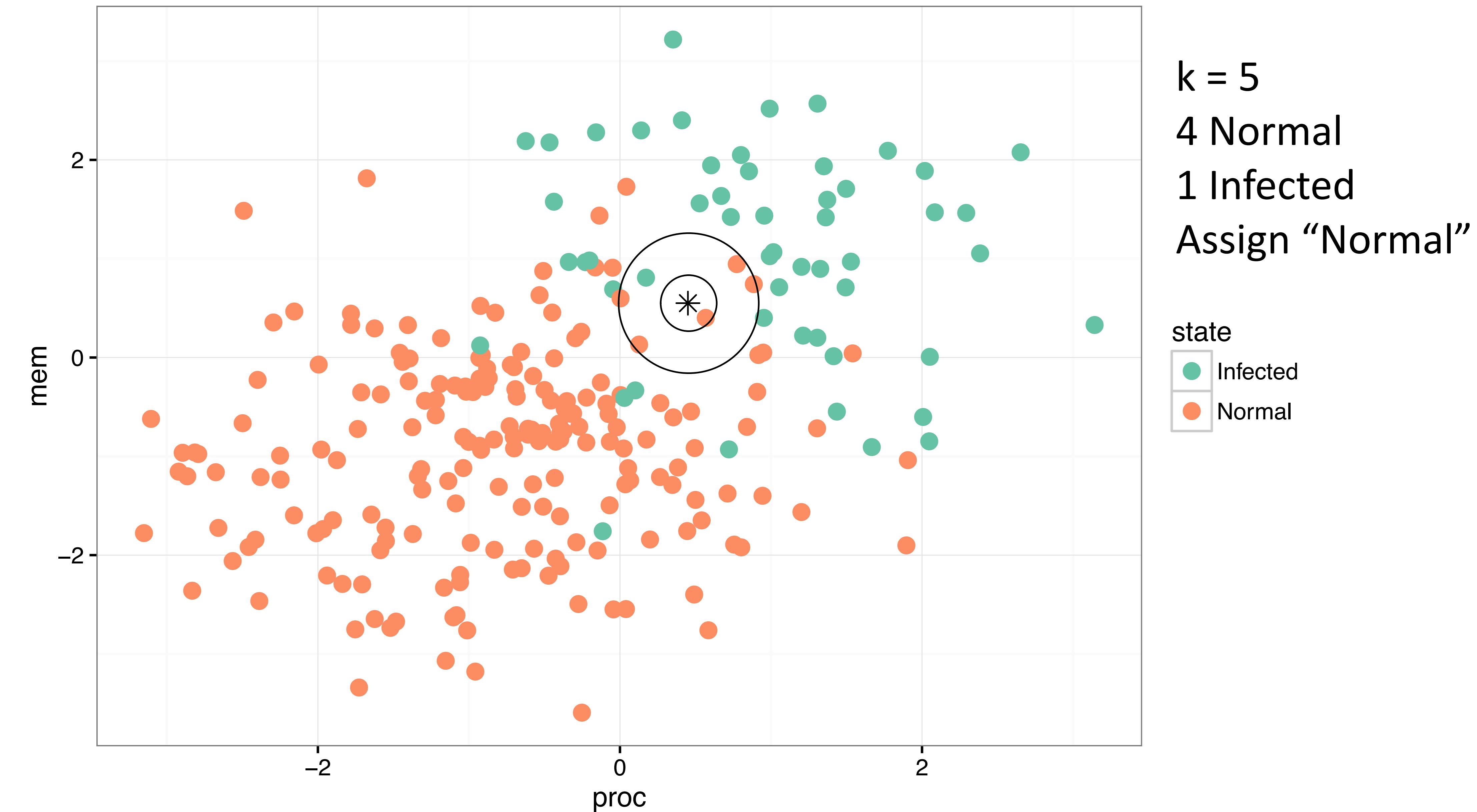
k-nearest neighbors



k-nearest neighbors

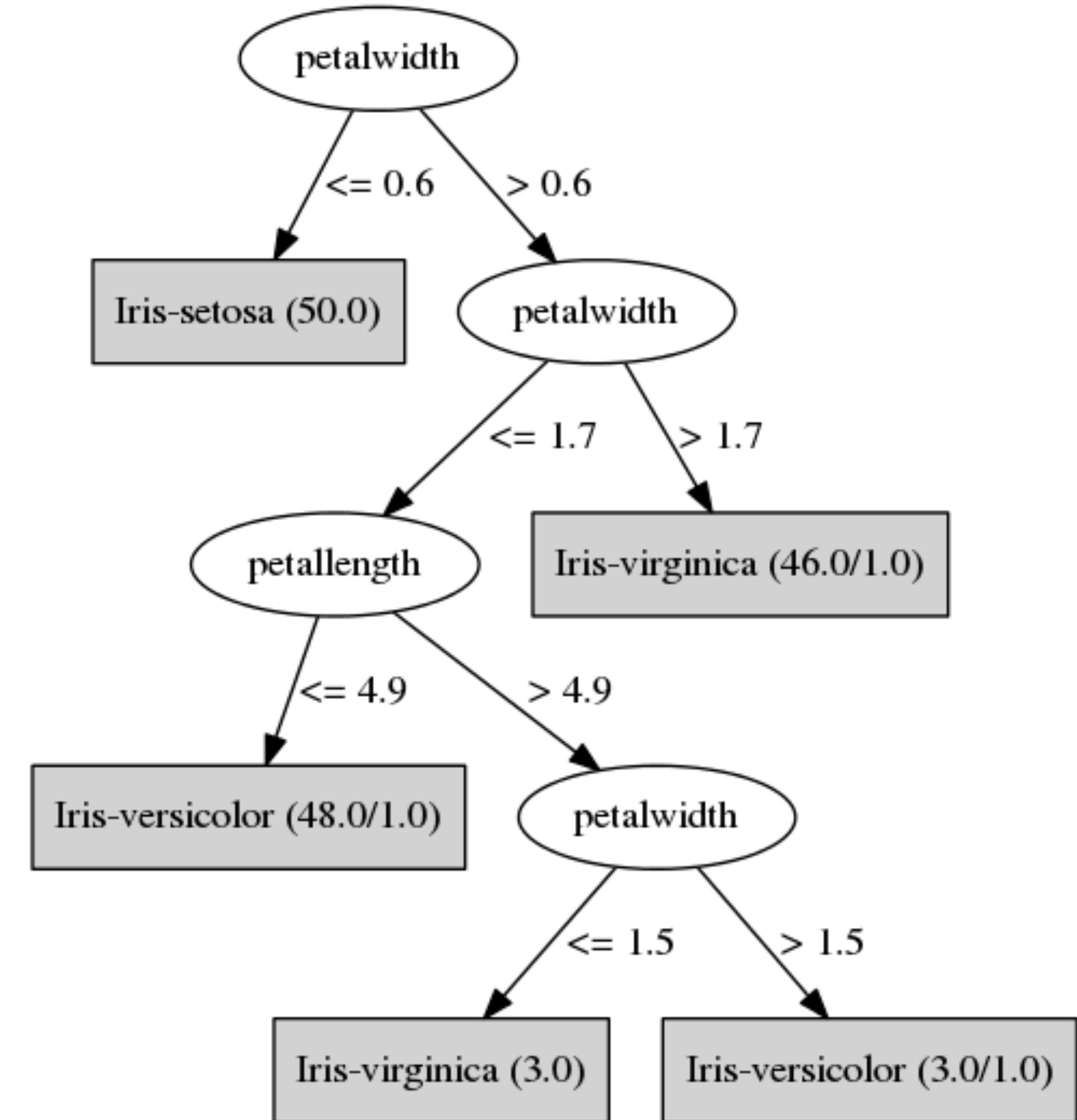


k-nearest neighbors

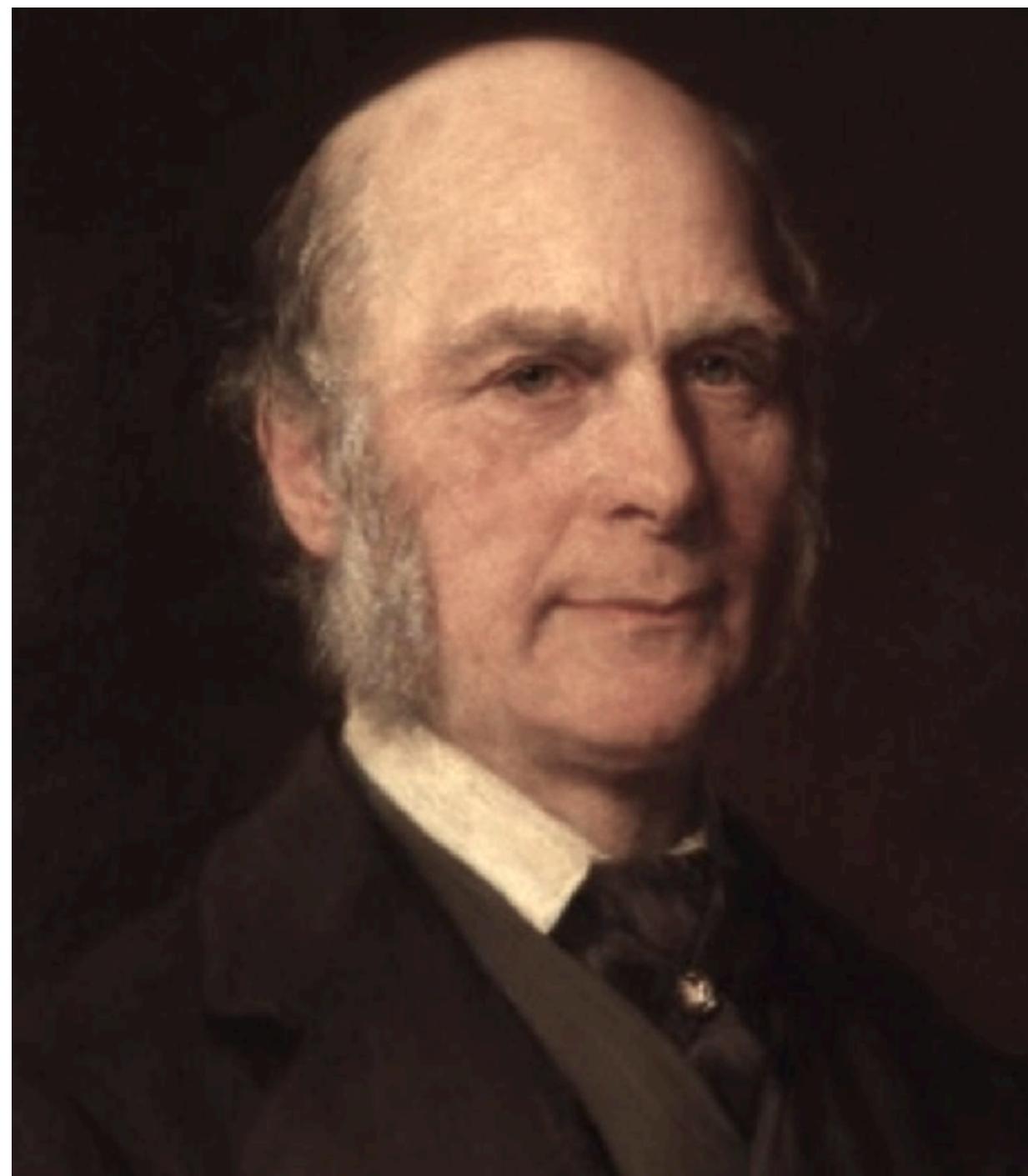


Decision Tree

- “uses a tree-like graph or model of decisions and their possible consequences”
- Use one or more variable, one or more branches
- Can split on continuous or categorical variable
- Final result is probability (between 0-1)



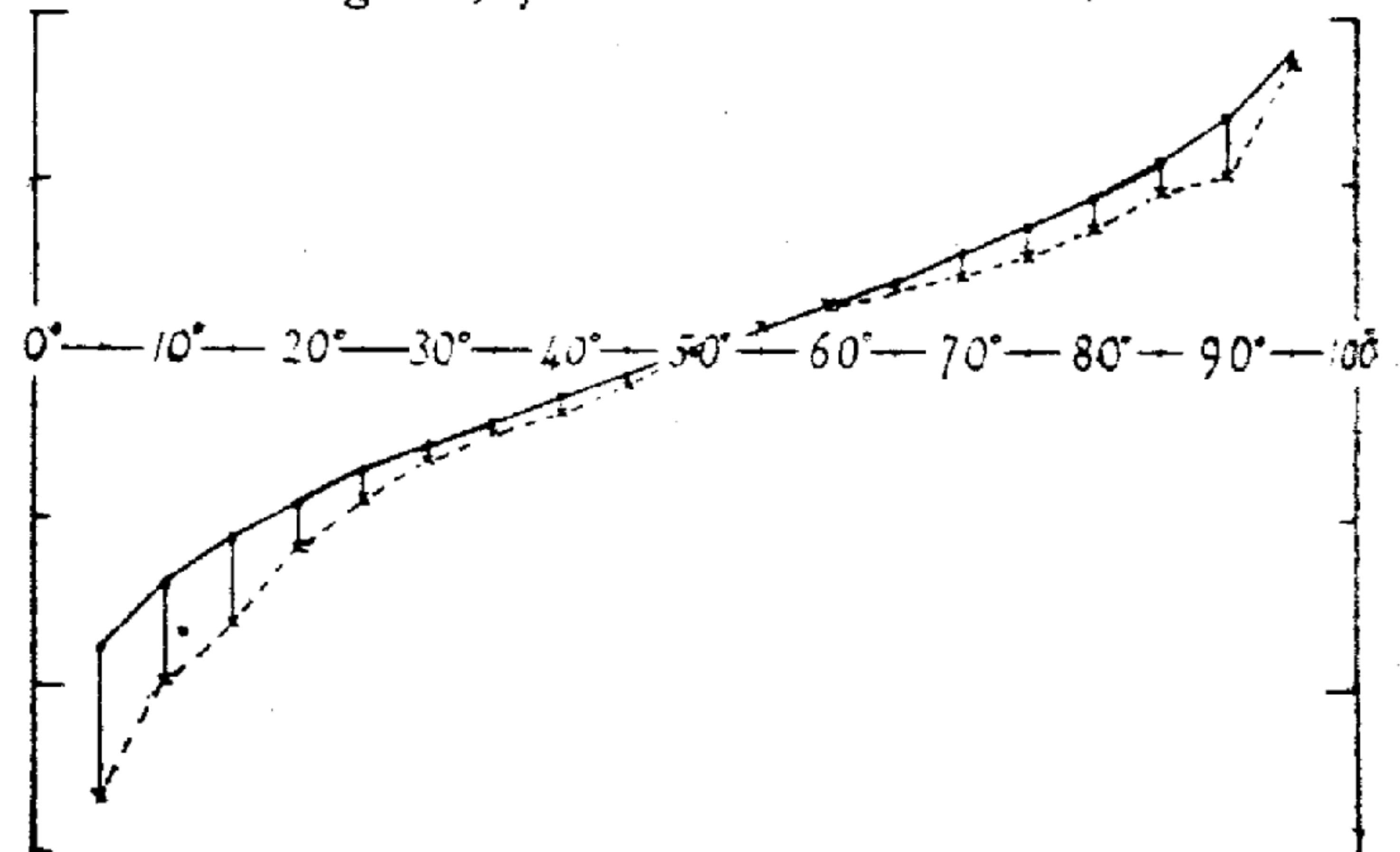
Wisdom of the crowds (ensemble learning)



“The material about to be discussed refers to a small matter...”

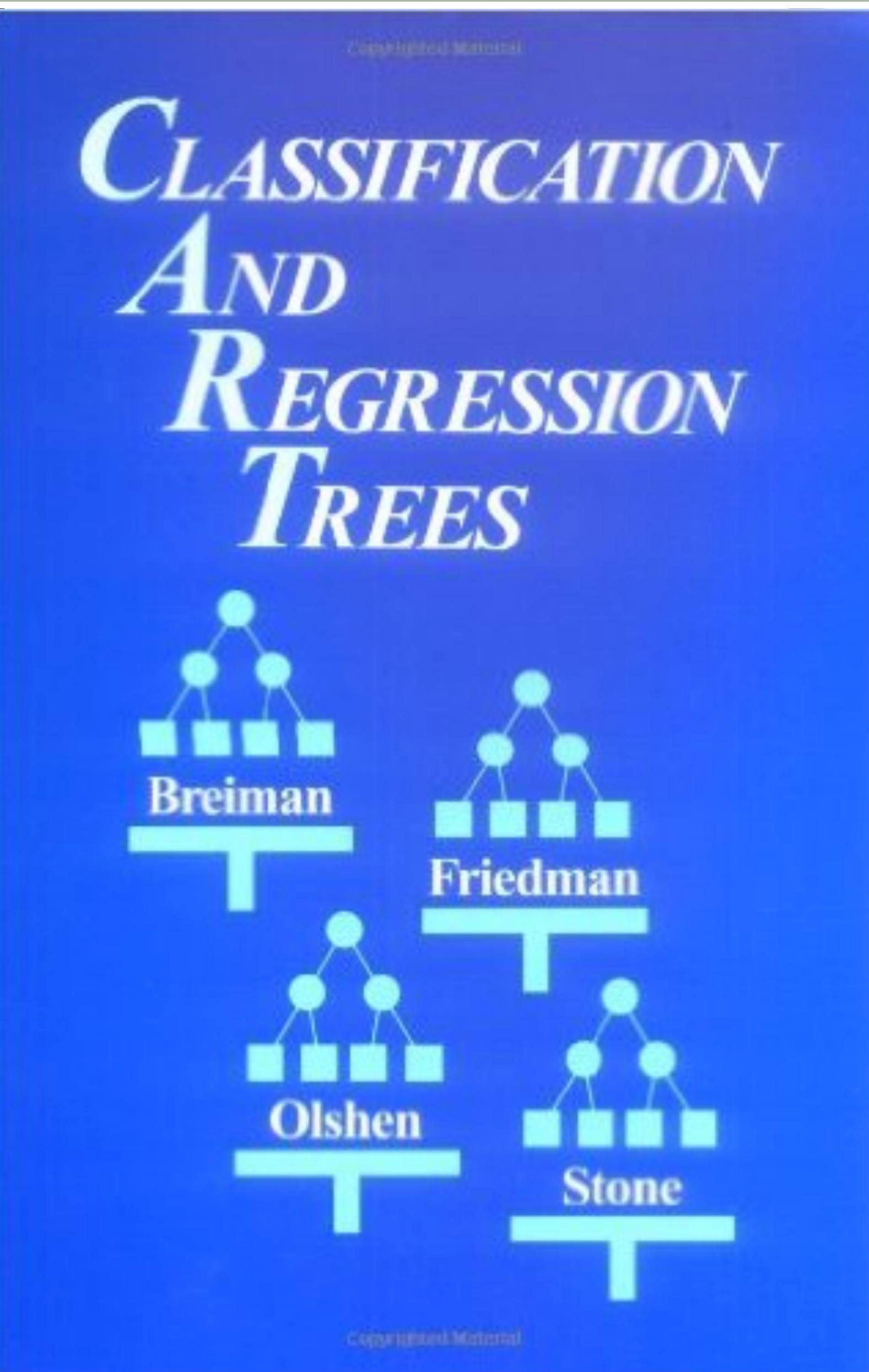
— Sir Francis Galton

Diagram, from the tabular values.



The continuous line is the normal curve with p.e.=37.
The broken line is drawn from the observations.
The lines connecting them show the differences between the observed and the normal.

Leo Breiman



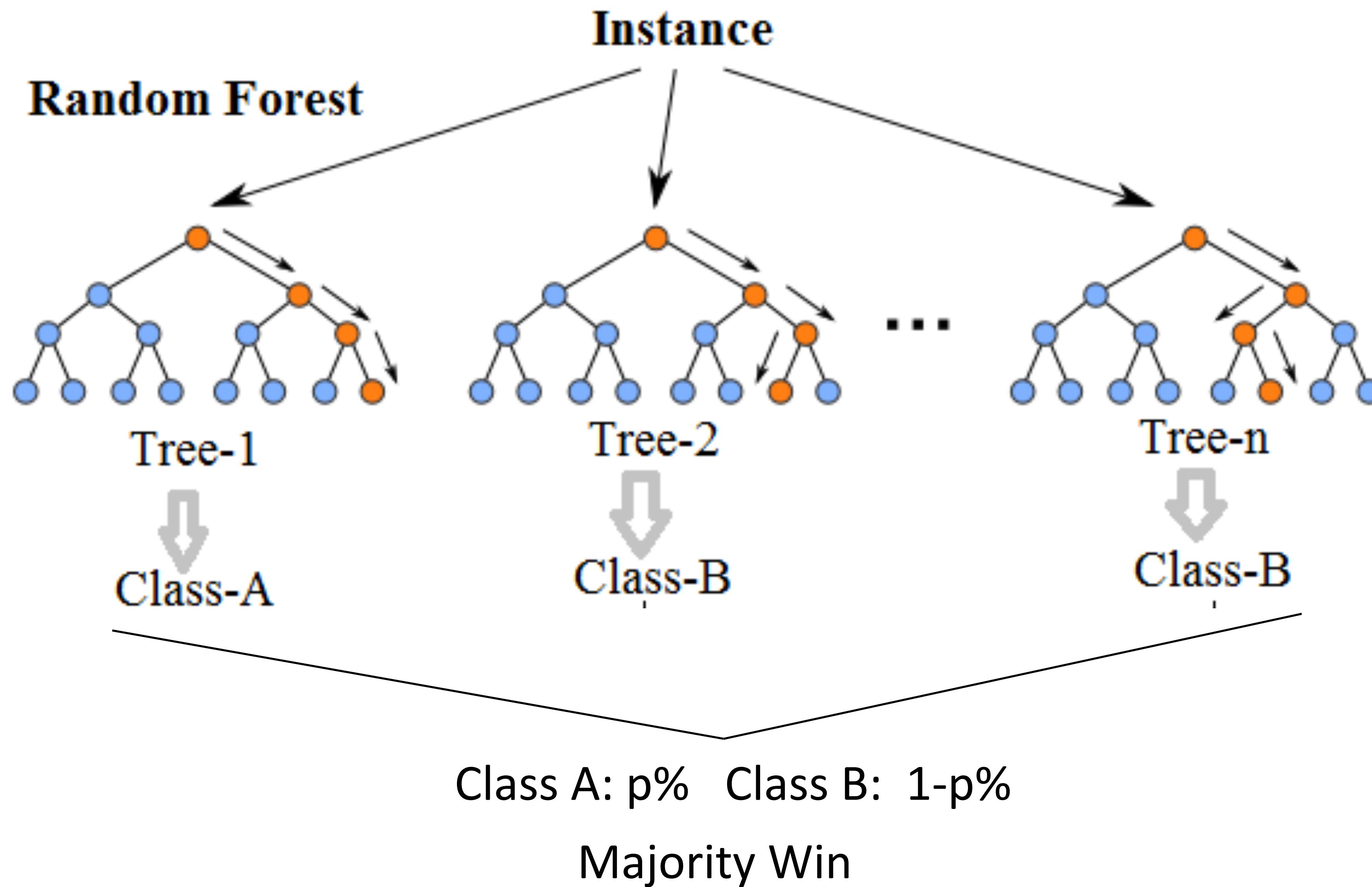
How Random Forest Works

1. Set how many decision trees to grow (ntree)
2. For each tree:
 1. Grab m predictor variables (\sqrt{p}) at random
 2. Set break point in continuous or class if categorical
 3. Derive a “vote” for each branch of the tree

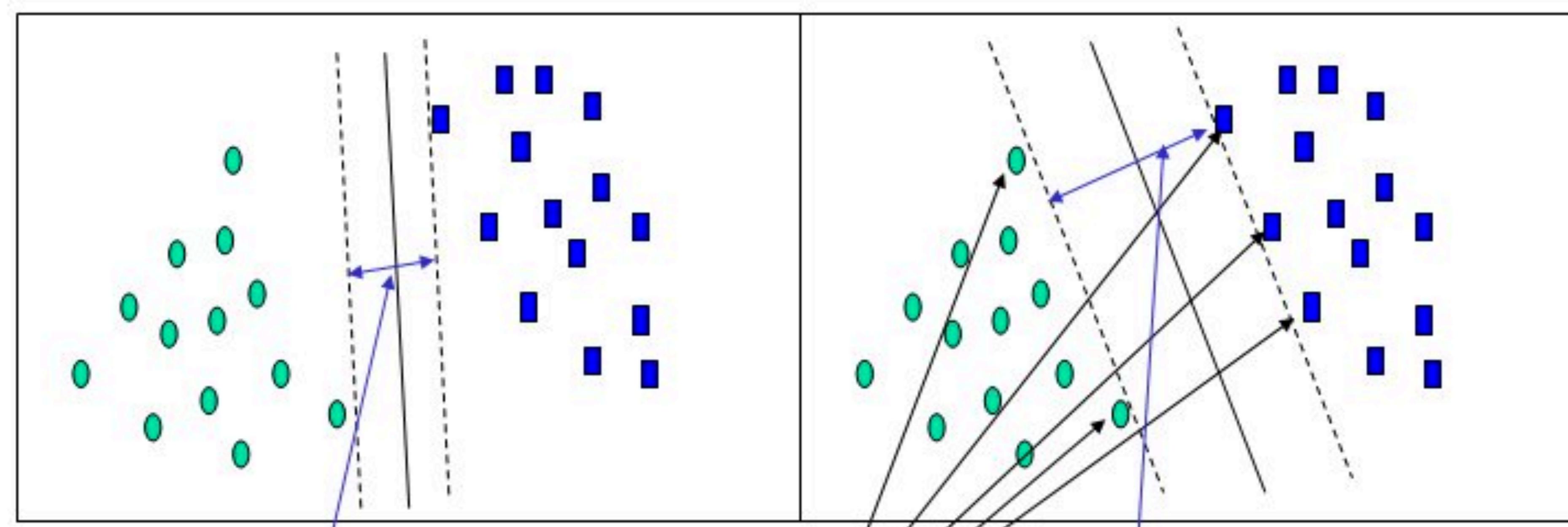
To predict a new observation:

1. Allow each tree to “vote” given observation

Random Forest



Support Vector Machine



Small Margin

Large Margin

Support Vectors

...Can make some fancy margins with kernel densities
and hyperplanes in multidimensional data.

O'REILLY®

Security

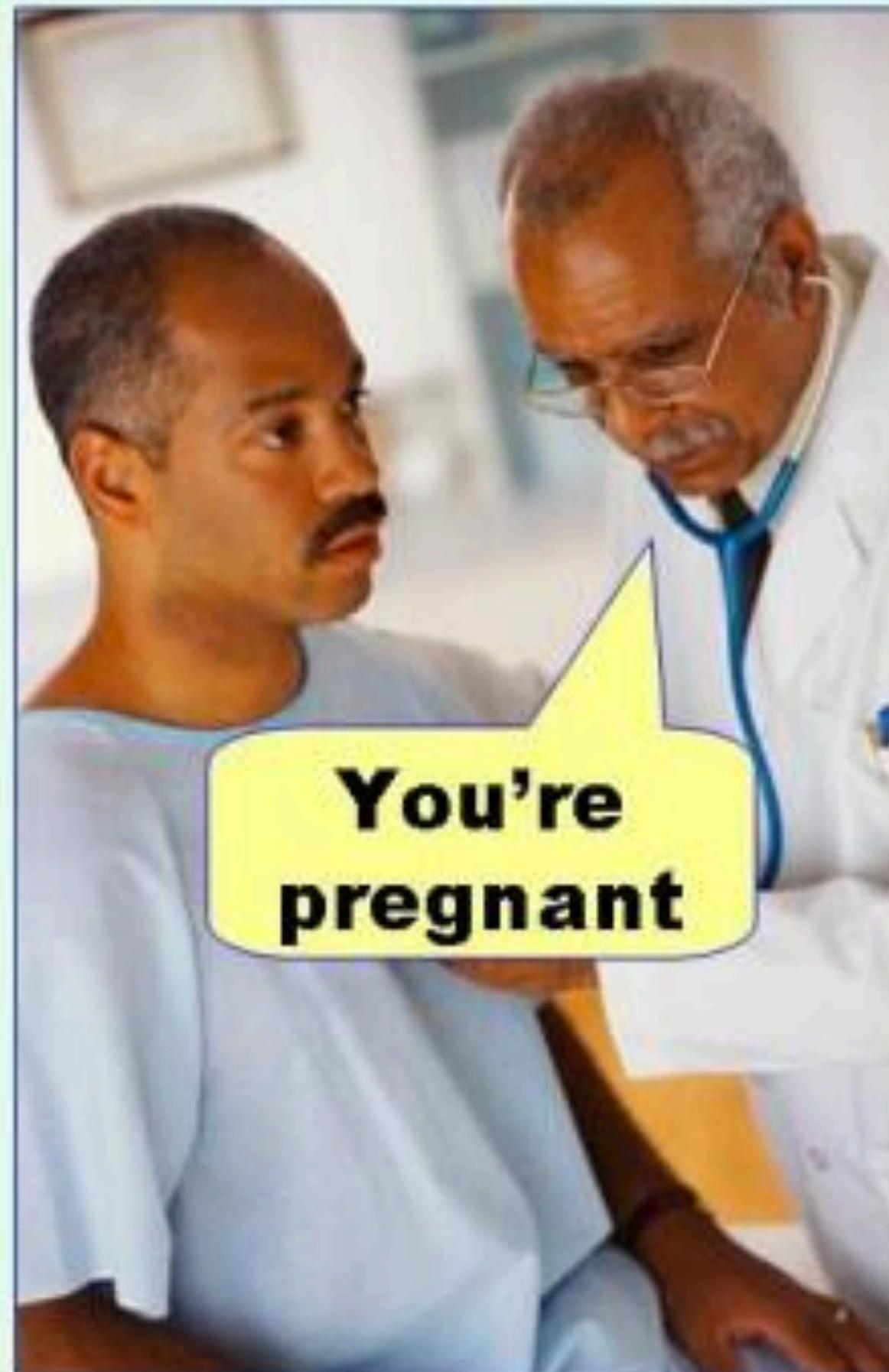
BUILD BETTER DEFENSES

oreillysecuritycon.com
#oreillysecurity

Measuring Performance of a Classifier

Evaluating how well a classifier performs

Type I error
(false positive)



Type II error
(false negative)

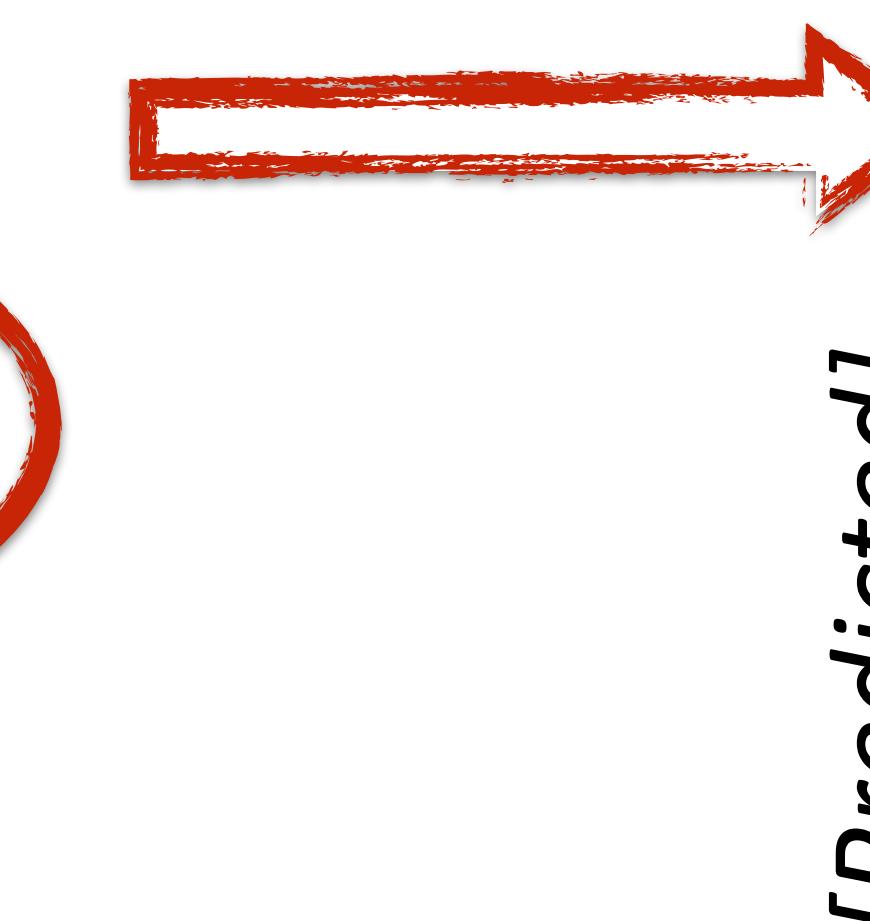


Evaluating how well a classifier performs: Confusion Matrix

Every package/tool should create some type of “confusion matrix”

Confusion Matrix and Statistics

Reference	nivdort	alexa
Prediction	nivdort	alexa
nivdort	6701	1591
alexa	283	5409



		[Reality]	
		Positive	Negative
Variants Called by the Algorithm	Positive	True Positive (TP) Correct variant allele or position call	False Positive (FP) Incorrect variant allele or position call.
	Negative	False Negative (FN) Incorrect reference genotype or no call.	True Negative (TN) Correct reference genotype or no call.

Evaluating how well a classifier performs

Confusion Matrix and Statistics

Reference

Prediction	nivdort	alexa
nivdort	6701	1591
alexa	283	5409

Accuracy : 0.866

95% CI : (0.8602, 0.8716)

No Information Rate : 0.5006

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.732

Mcnemar's Test P-Value : < 2.2e-16

Sensitivity : 0.9595

Specificity : 0.7727

Pos Pred Value : 0.8081

Neg Pred Value : 0.9503

Prevalence : 0.4994

Detection Rate : 0.4792

Detection Prevalence : 0.5930

Balanced Accuracy : 0.8661

'Positive' Class : nivdort

Overall Accuracy of the Classifier

95% confidence intervals around accuracy

% of prevalent class (if we say "all dga")

p-value if Accuracy is > No-Info Rate

Evaluating how well a classifier performs

Confusion Matrix and Statistics

Reference		Prediction	
nivdort	alexa	nivdort	alexa
6701	1591		
283	5409		

Accuracy : 0.866
95% CI : (0.8602, 0.8716)

No Information Rate : 0.5006

P-Value [Acc > NIR] : < 2.2e-16

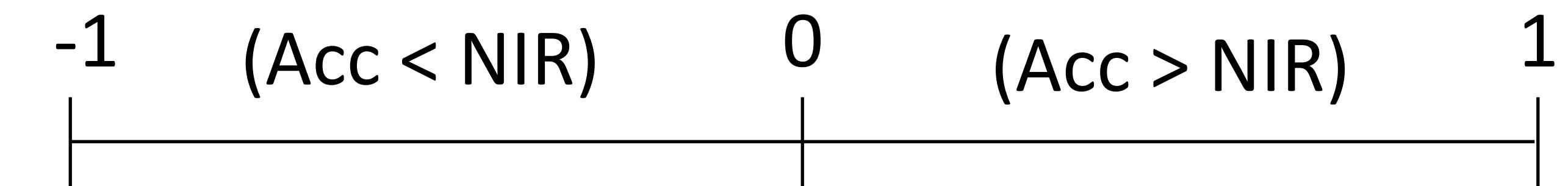
Kappa : 0.732
Mcnemar's Test P-Value : < 2.2e-16

Sensitivity : 0.9595
Specificity : 0.7727
Pos Pred Value : 0.8081
Neg Pred Value : 0.9503
Prevalence : 0.4994
Detection Rate : 0.4792
Detection Prevalence : 0.5930
Balanced Accuracy : 0.8661

'Positive' Class : nivdort

Kappa:

compares an Observed Accuracy with an Expected Accuracy (random chance).



Perfectly
Worse

No
Different
(Acc = NIR)

Perfectly
Better

p-value with $H_0: \text{kappa} = \text{NIR}$

Evaluating how well a classifier performs

Confusion Matrix and Statistics

		Reference	
Prediction		nivdort	alexa
nivdort	6701	1591	
alexa	283	5409	

Accuracy : 0.866
95% CI : (0.8602, 0.8716)
No Information Rate : 0.5006
P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.732
Mcnemar's Test P-Value : < 2.2e-16

Sensitivity : 0.9595
Specificity : 0.7727
Pos Pred Value : 0.8081
Neg Pred Value : 0.9503
Prevalence : 0.4994
Detection Rate : 0.4792
Detection Prevalence : 0.5930
Balanced Accuracy : 0.8661

'Positive' Class : nivdort

		Reference	
Predicted		Event	No Event
Event	A	B	
	C	D	

The formulas used here are:

$$\text{Sensitivity} = \frac{A}{A + C}$$

$$\text{Specificity} = \frac{D}{B + D}$$

$$\text{Prevalence} = \frac{A + C}{A + B + C + D}$$

Sensitivity: ability to identify “positive” class
Specificity: ability to identify non-positive class

Evaluating how well a classifier performs

Confusion Matrix and Statistics

Reference			
Prediction	nivdort	alexa	
nivdort	6701	1591	
alexa	283	5409	

Accuracy : 0.866
95% CI : (0.8602, 0.8716)

No Information Rate : 0.5006
P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.732
McNemar's Test P-Value : < 2.2e-16

Sensitivity : 0.9595
Specificity : 0.7727
Pos Pred Value : 0.8081
Neg Pred Value : 0.9503
Prevalence : 0.4994
Detection Rate : 0.4792
Detection Prevalence : 0.5930
Balanced Accuracy : 0.8661
'Positive' Class : nivdort

		Reference	
		Predicted	
		Event	No Event
		A	B
		C	D

$$PPV = \frac{sensitivity \times prevalence}{((sensitivity \times prevalence) + ((1 - specificity) \times (1 - prevalence)))}$$

$$NPV = \frac{specificity \times (1 - prevalence)}{((1 - sensitivity) \times prevalence) + ((specificity) \times (1 - prevalence))}$$

$$Detection\ Rate = \frac{A}{A + B + C + D}$$

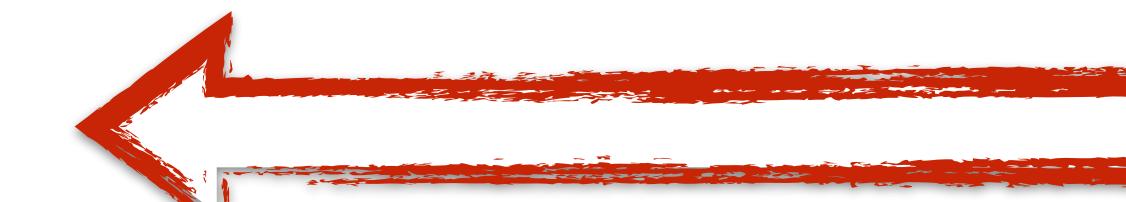
$$Detection\ Prevalence = \frac{A + B}{A + B + C + D}$$

$$Balanced\ Accuracy = (sensitivity + specificity)/2$$

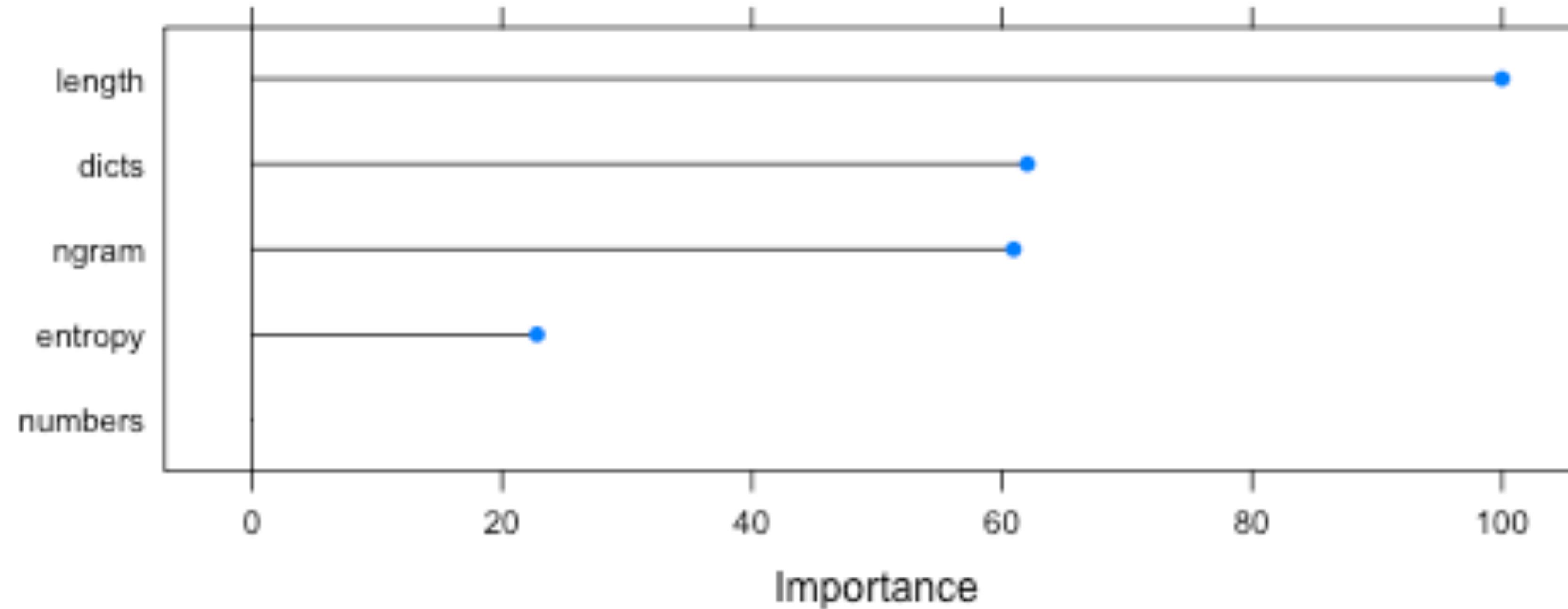
$$Precision = \frac{A}{A + B}$$

$$Recall = \frac{A}{A + C}$$

$$F1 = \frac{(1 + \beta^2) \times precision \times recall}{(\beta^2 \times precision) + recall}$$



Variable importance...



```
plot(varImp(rfFit))
```

Note: SVM does not have a "varImp" function

{ LAB } time

Look over the “example” worksheets
Evaluate a Classifier on “nvdort” and “alexa”

