BPS5231

Artificial Intelligence

For

Sustainable Building Design

L5

Ang Yu Qian

Assistant Professor

Project

Interim 1

Deliverables – Code or Repo + Slides + Write-up / Conf. Paper

Interim 1 (15%): Due 5 October

Font 11 | Single Spacing | Aptos Display | 5 Pages

1. Introduction
2. Problem Statement / Research Problem
   - You can: answer a problem, develop a tool, research a phenomenon, etc.
   - Can also be one of Design Beyond themes: Form, Climate, Energy, etc.
3. Data
4. Exploratory Data Analysis (Code + Visuals/Charts)

Recess week is a good time to work on it

This should be a continuation towards your final submission.

Problem Statement from Center for Liveable Cities

Centre for Liveable Cities Singapore

To distil, create and share knowledge on liveable and sustainable cities.

https://www.clc.gov.sg/events/lecture/creating-a-regenerative-city

Problem Statement from Center for Liveable Cities

Advancing conventional sustainability, regenerative urban development integrates climate-positive strategies such as nature-based solutions, low-carbon development and resource circularity to achieve co-benefits across liveability, resiliency and resource outcomes.

1. How can AI unlock new understanding and approaches to optimising resources (land, carbon, fiscal etc.) in urban development?

2. How can AI maximise regenerative co-benefits for people and environment?

3. How can AI methods support the scaling up of regenerative approaches and co-benefits across different development scales and contexts?

https://www.clc.gov.sg/events/lecture/creating-a-regenerative-city

L05.1
Supervised 5

Bagging

Random Forest

Boosting

XGBoost

L05.2
Visualizations

Exploratory Data Analysis

L05.3
Simulations

Rhino

ClimateStudio

L05.1
Supervised 5

Bagging

Random Forest

Boosting

XGBoost

L05.2
Visualizations

Exploratory Data Analysis

L05.3
Simulations

Rhino

ClimateStudio

Today, we will cover

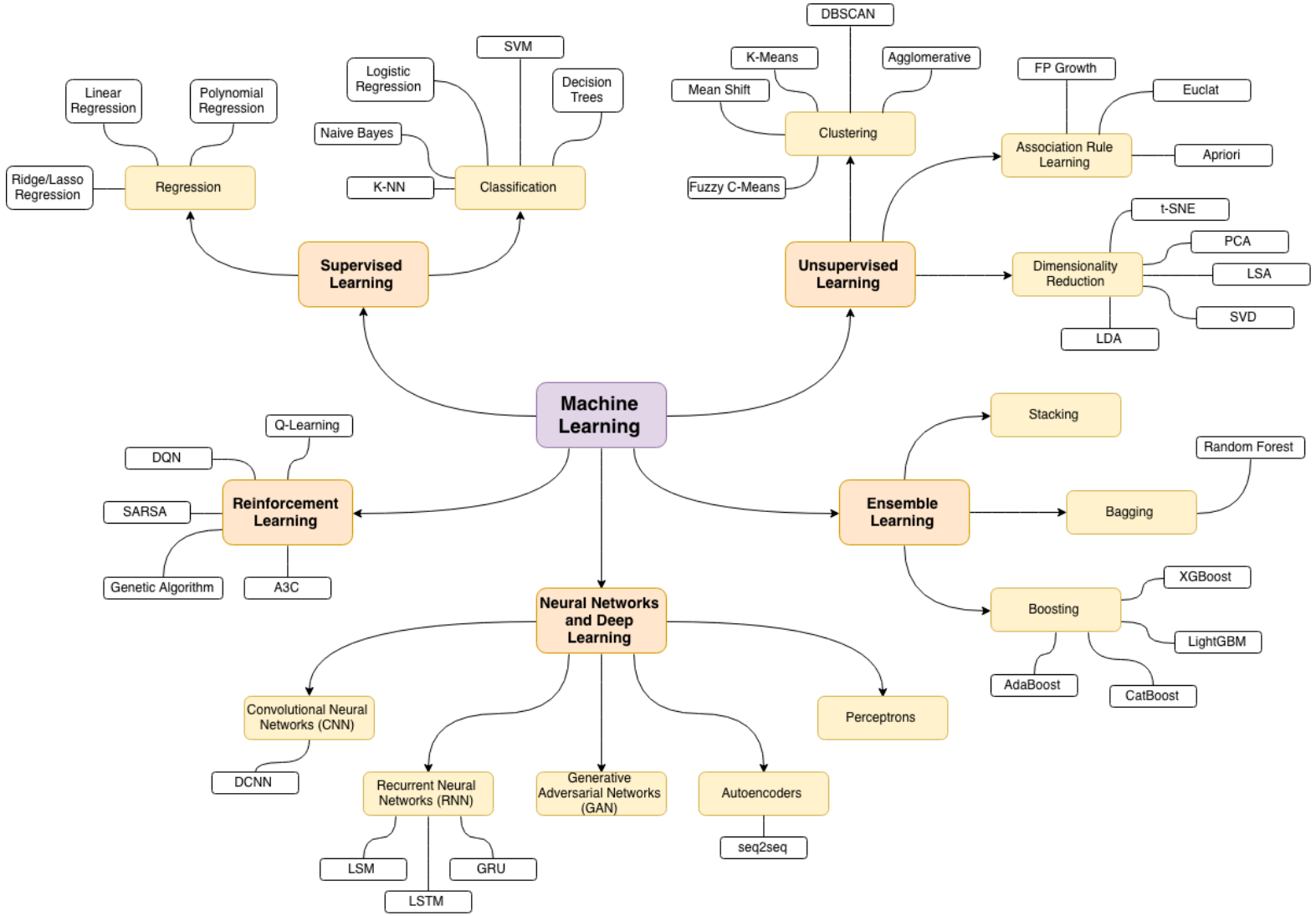
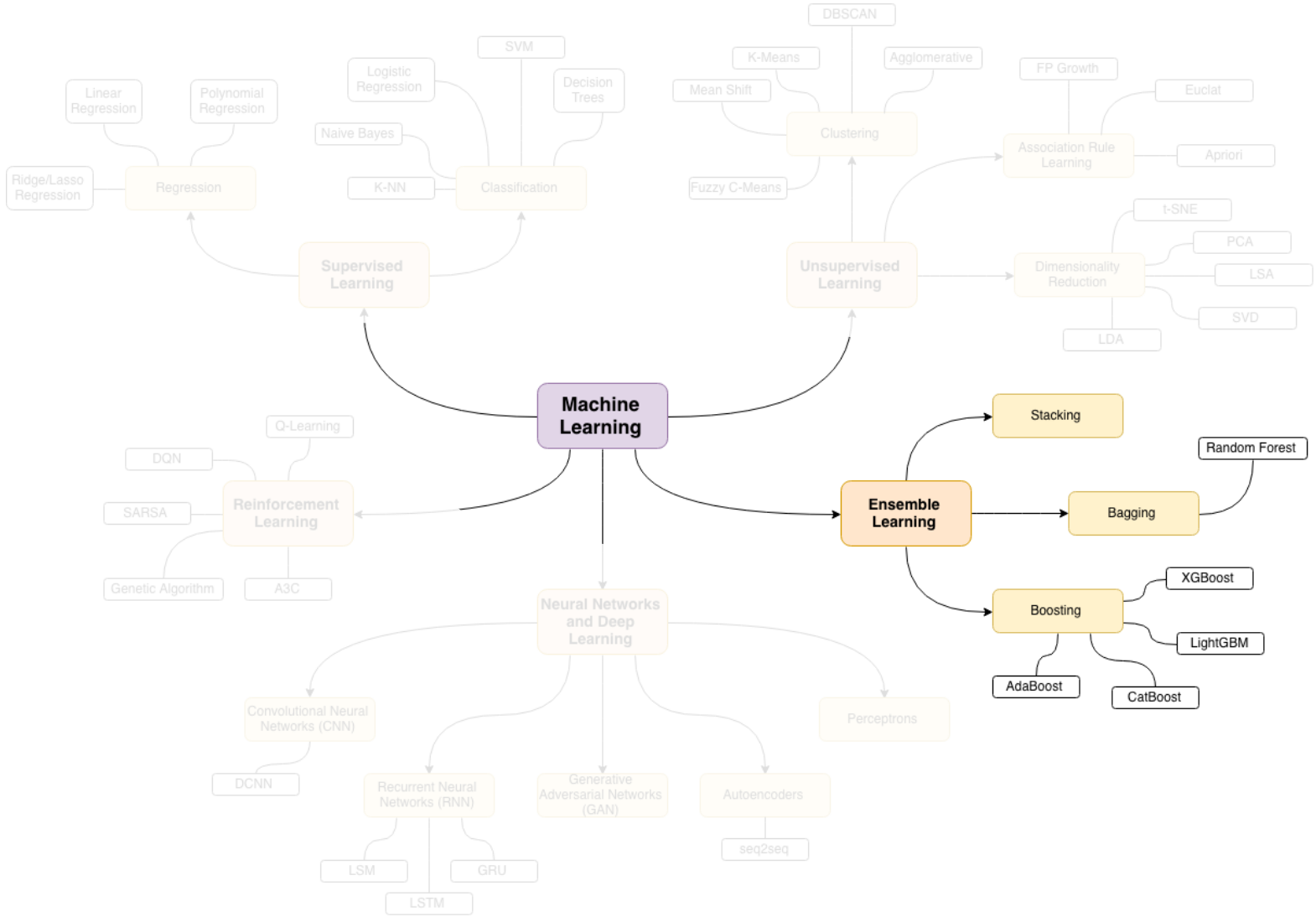


Image from Anil Jain, Google

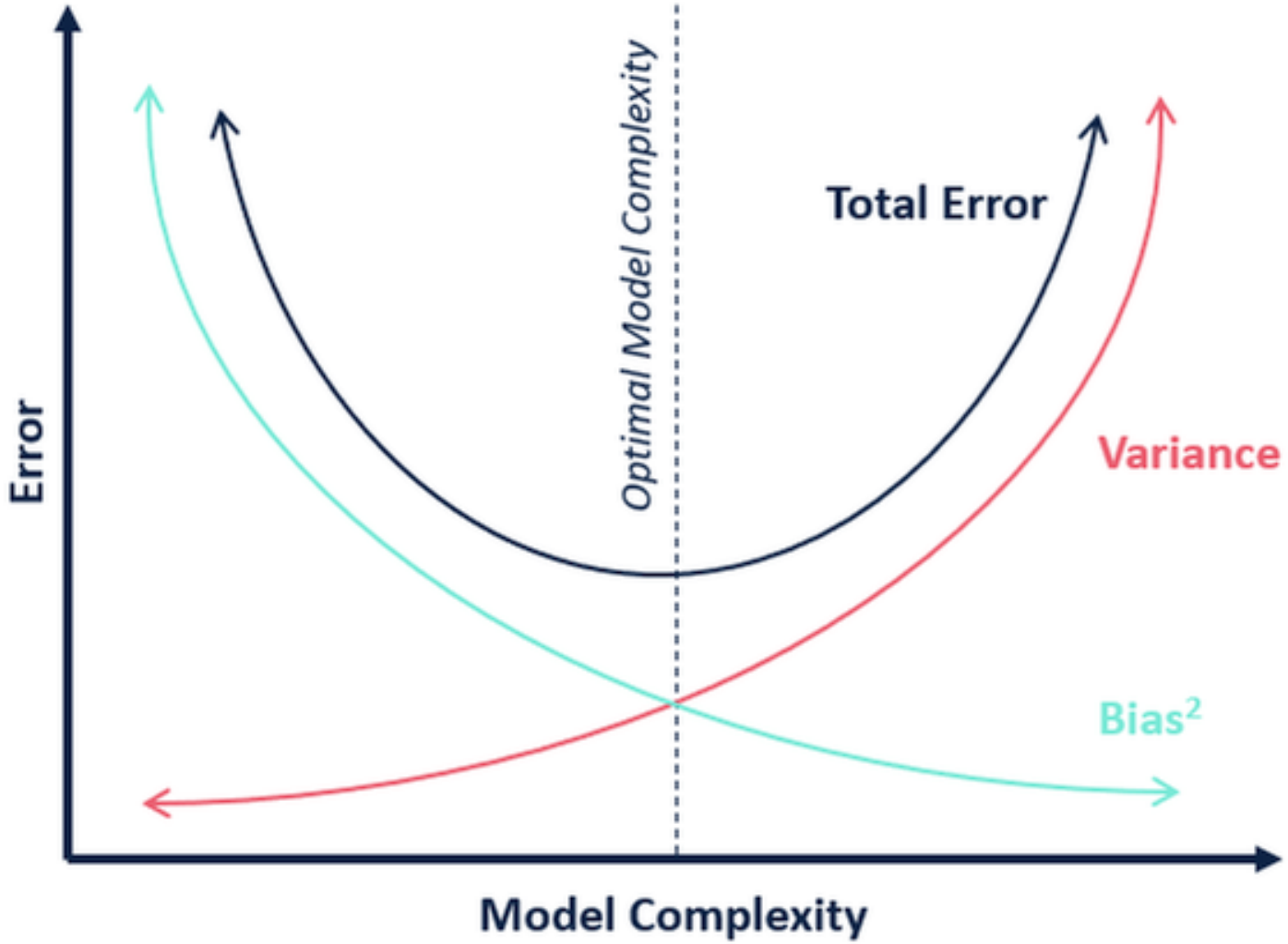
Today, we will cover



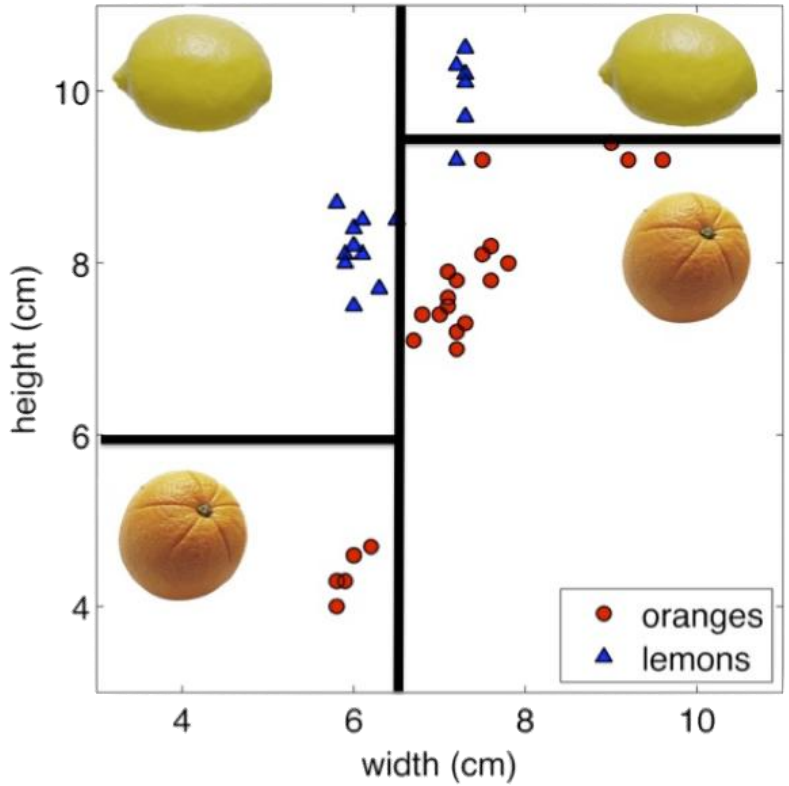
← Today

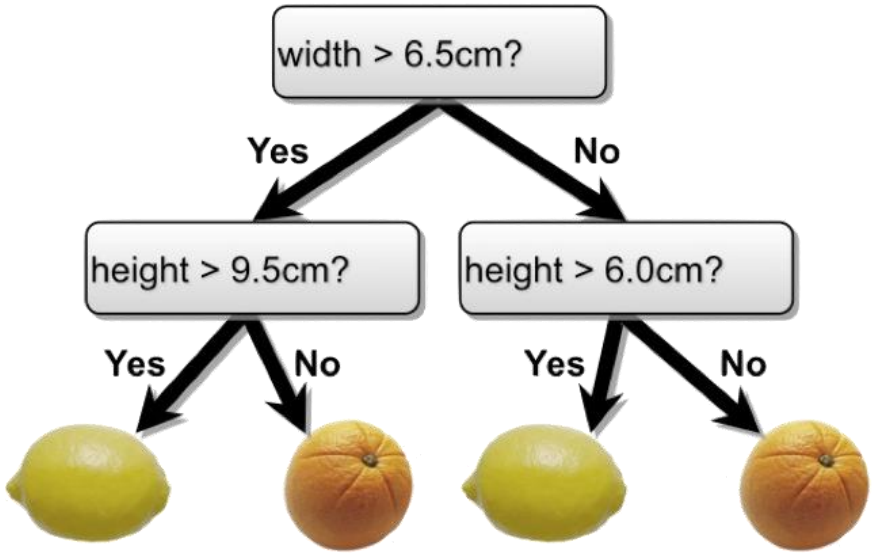
Image from Anil Jain, Google

Recap – Decision Trees

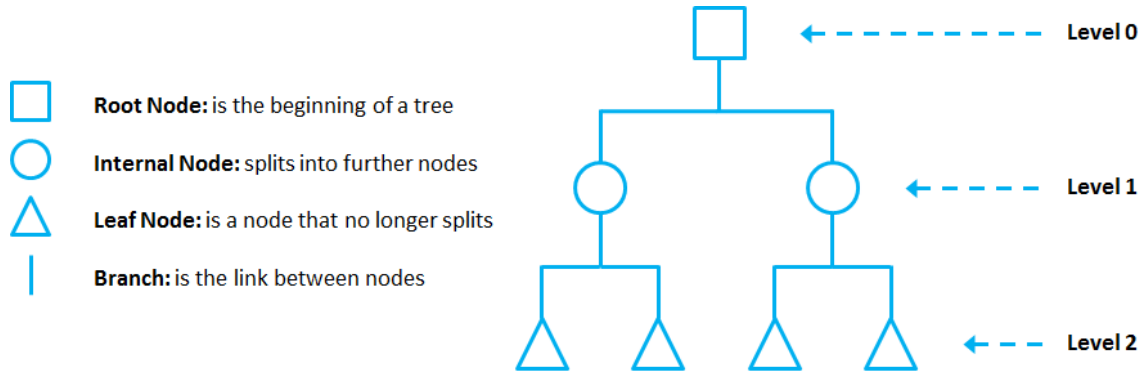


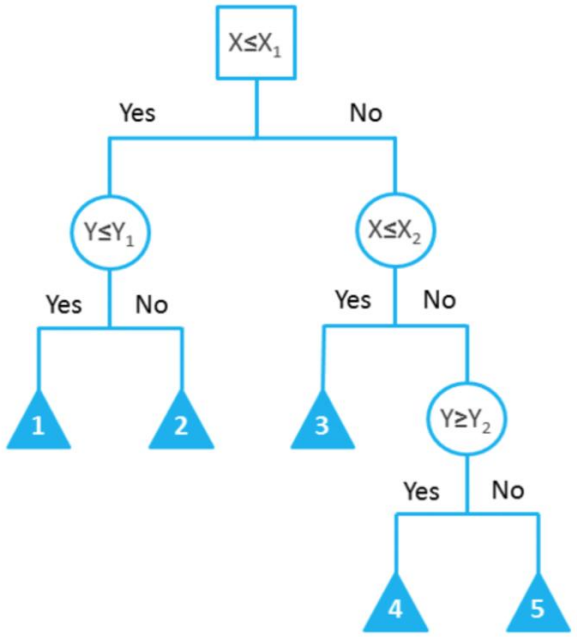
Each comparison and branching represents splitting a region in the feature space on a single feature. Typically, at each iteration, we split once along one dimension (one predictor). Why?

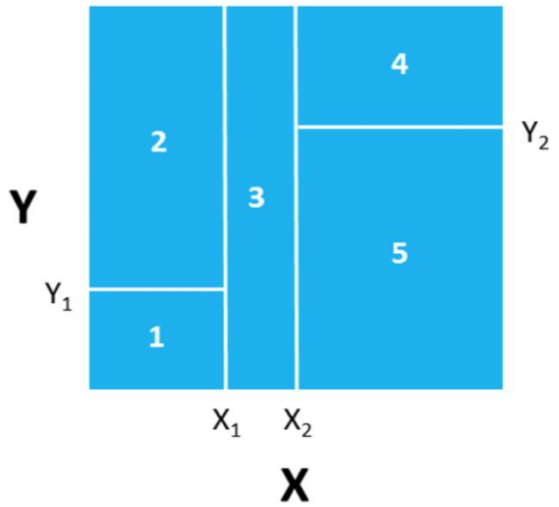




Recap – Decision Trees







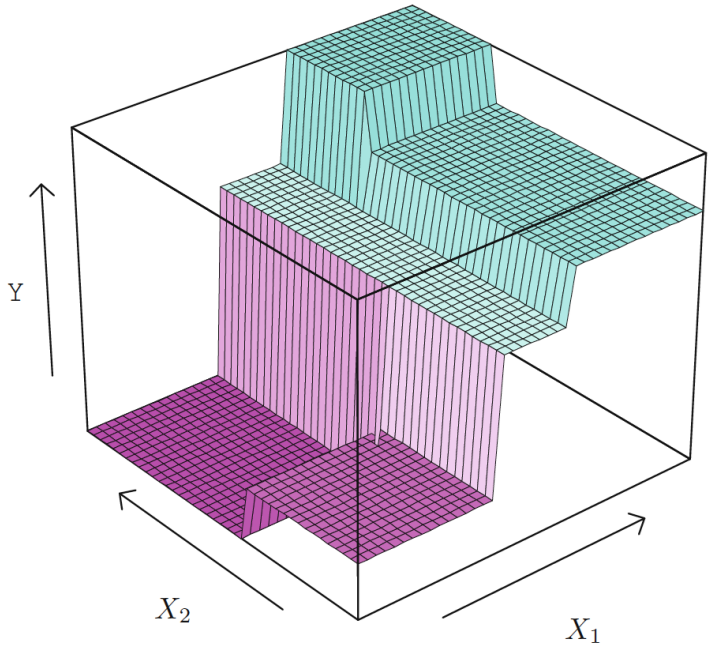


Image from Open Data Science

Learning the Model:

We will seek a reasonable model using a greedy algorithm.

1. Start with an empty decision tree (undivided feature space)
2. Choose the 'optimal' predictor on which to split and choose the 'optimal' threshold value for splitting.
3. Recurse on each new node until stopping condition is met

Now, we need only define our splitting criterion and stopping condition.

For classification, we label each region in the model with the label of the class to which the plurality of the points within the region belong.

For regression, we predict with the average of the output values of the training points contained in the region.

Optimality of Splitting

While there is no ‘correct’ way to define an optimal split, there are some common sensical guidelines for every splitting criterion:

the regions in the feature space should grow progressively more pure with the number of splits. That is, we should see each region ‘specialize’ towards a single class.

the fitness metric of a split should take a differentiable form (making optimization possible).

we shouldn’t end up with empty regions - regions containing no training points.

The splitting criteria we've examined each minimize a loss function

A. For classification, purity of the regions is a good indicator the performance of the model. Entropy as a splitting criterial minimizes the cross-entropy (greedy).

B. For regression, we want to select a splitting criterion that promotes splits that improves the predictive accuracy of the model as measured by the MSE.

Bagging – Core idea = Reduce Variance?

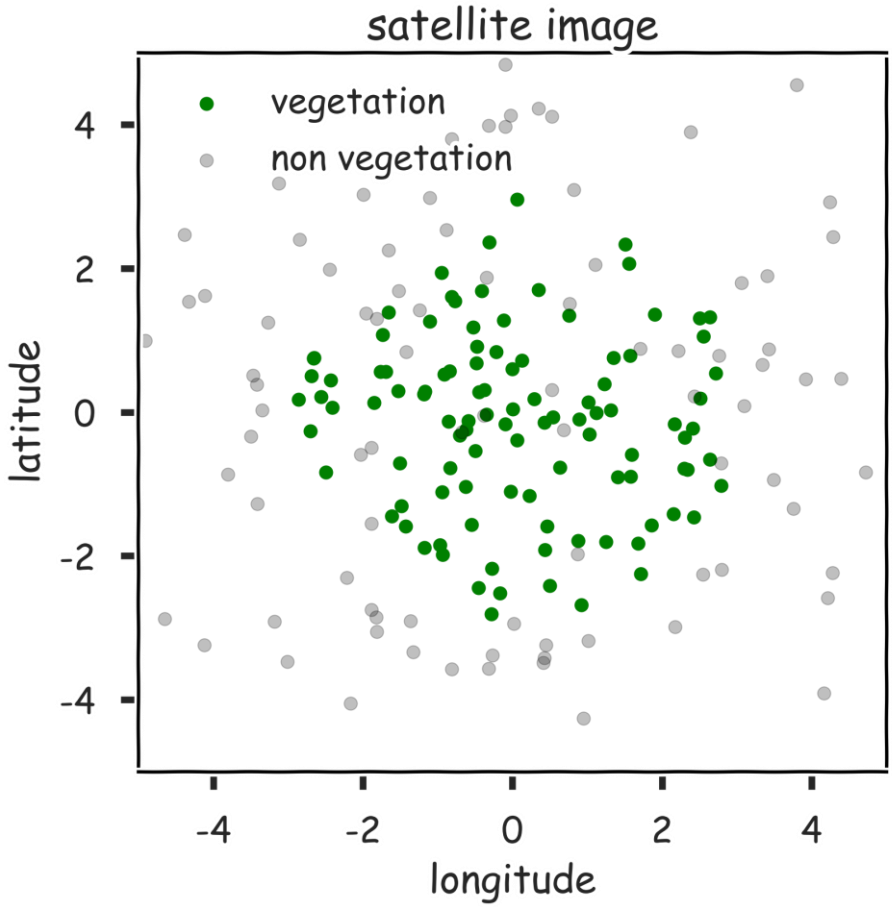


Image from Palvos Protopapas

Hyperparameter: Depth

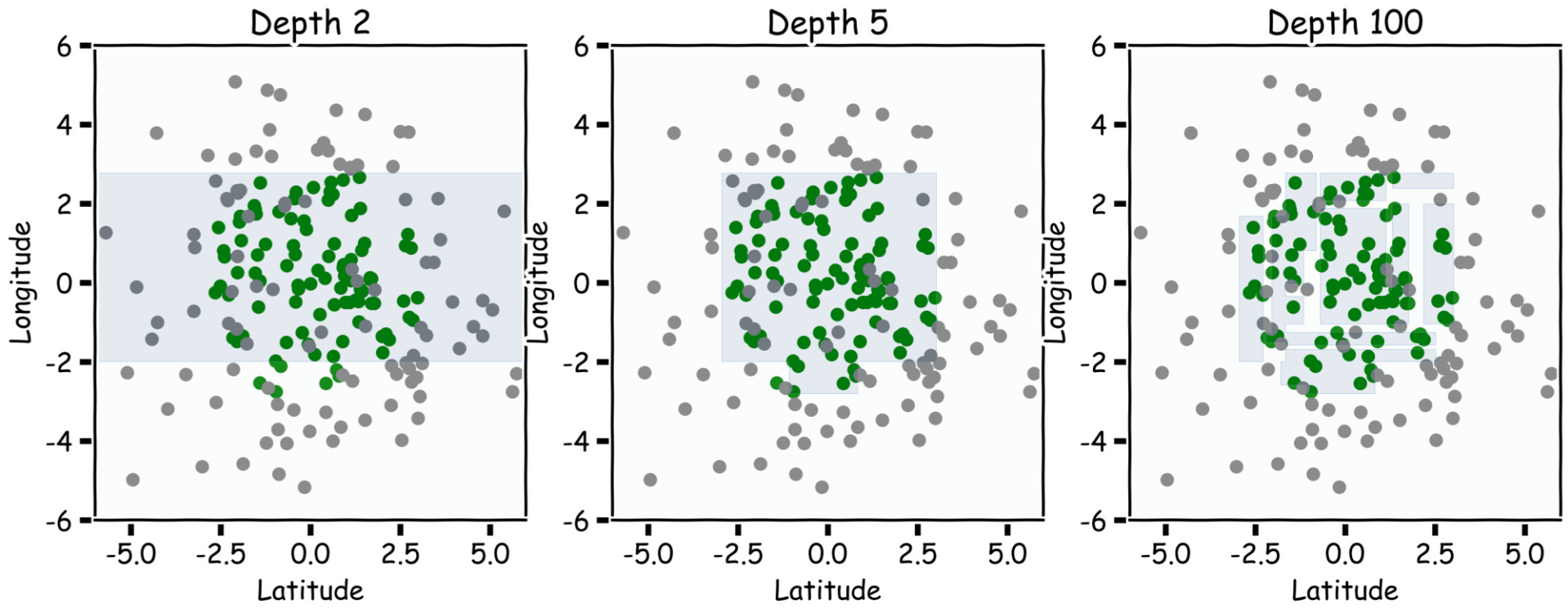


Image from Palvos Protopapas

Hyperparameter: Use train/validation or cross-validation to estimate best depth

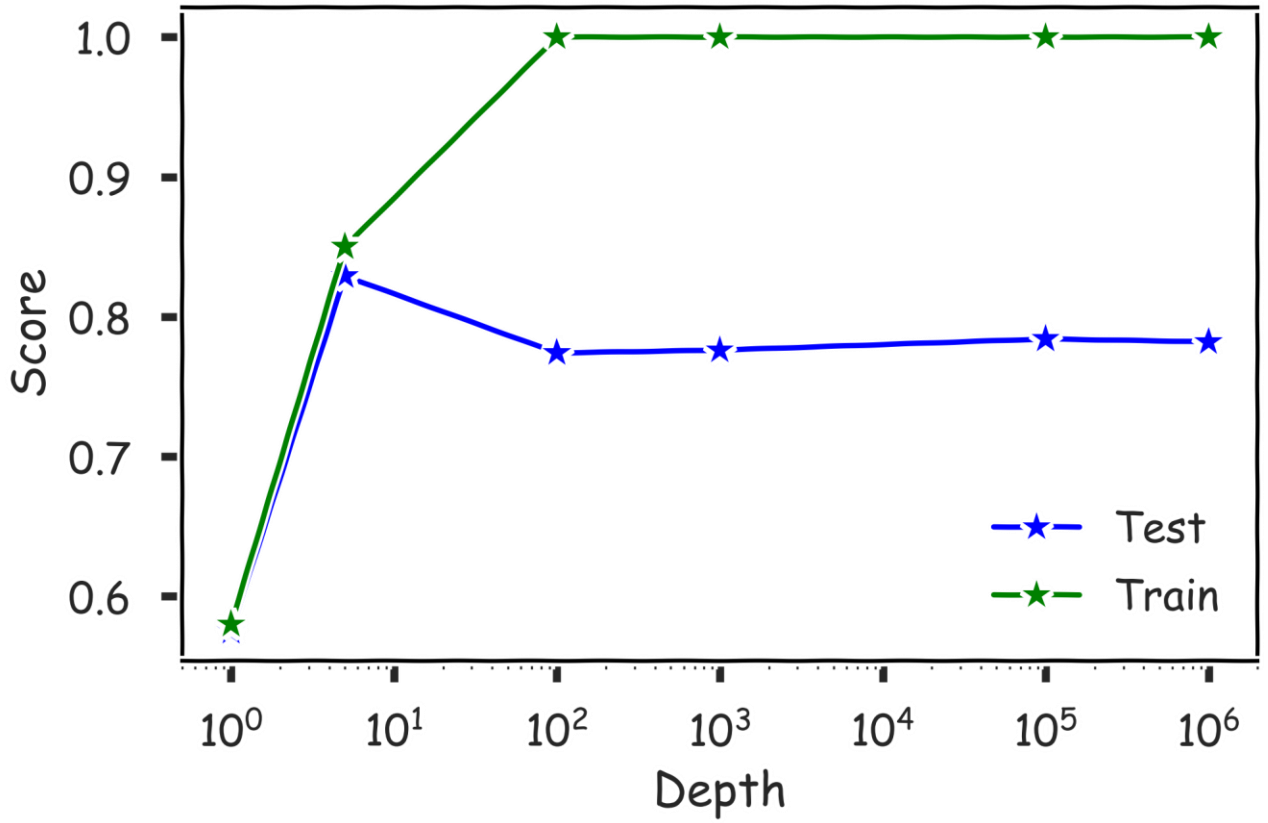


Image from Palvos Protopapas

Magic realism: bootstrap

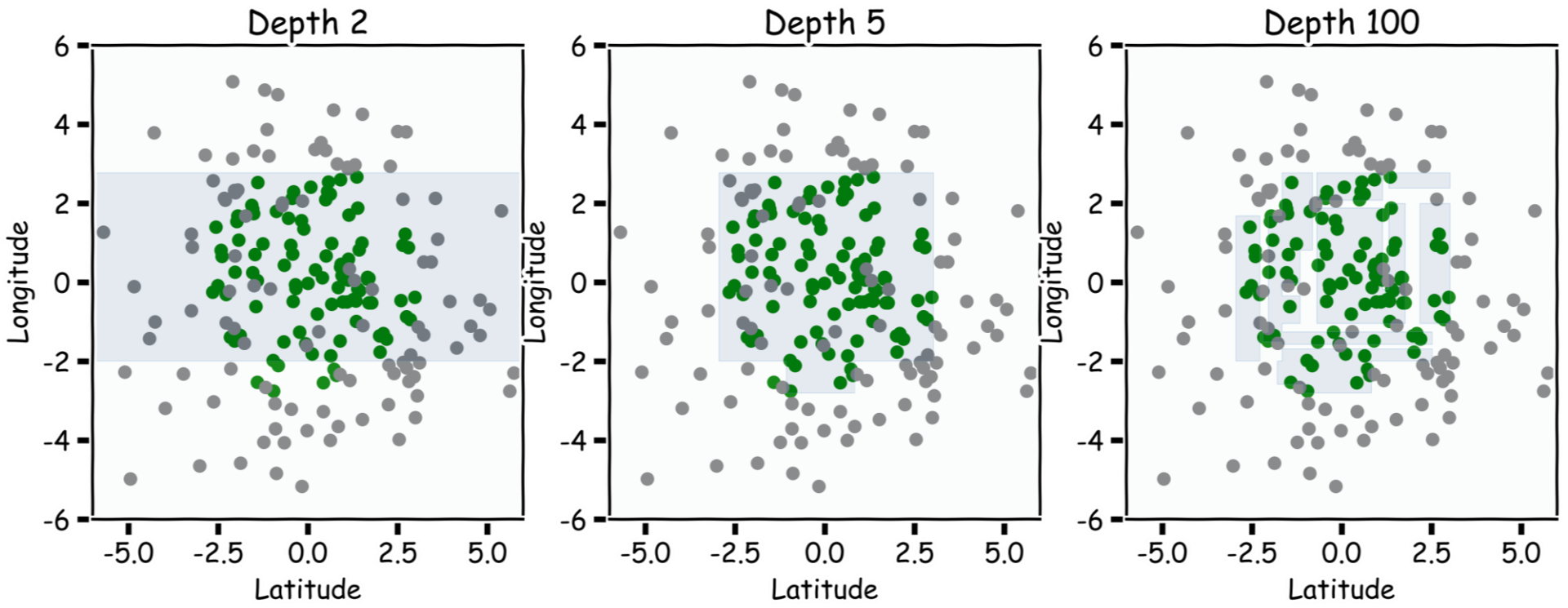


Image from Palvos Protopapas

Magic realism: bootstrap

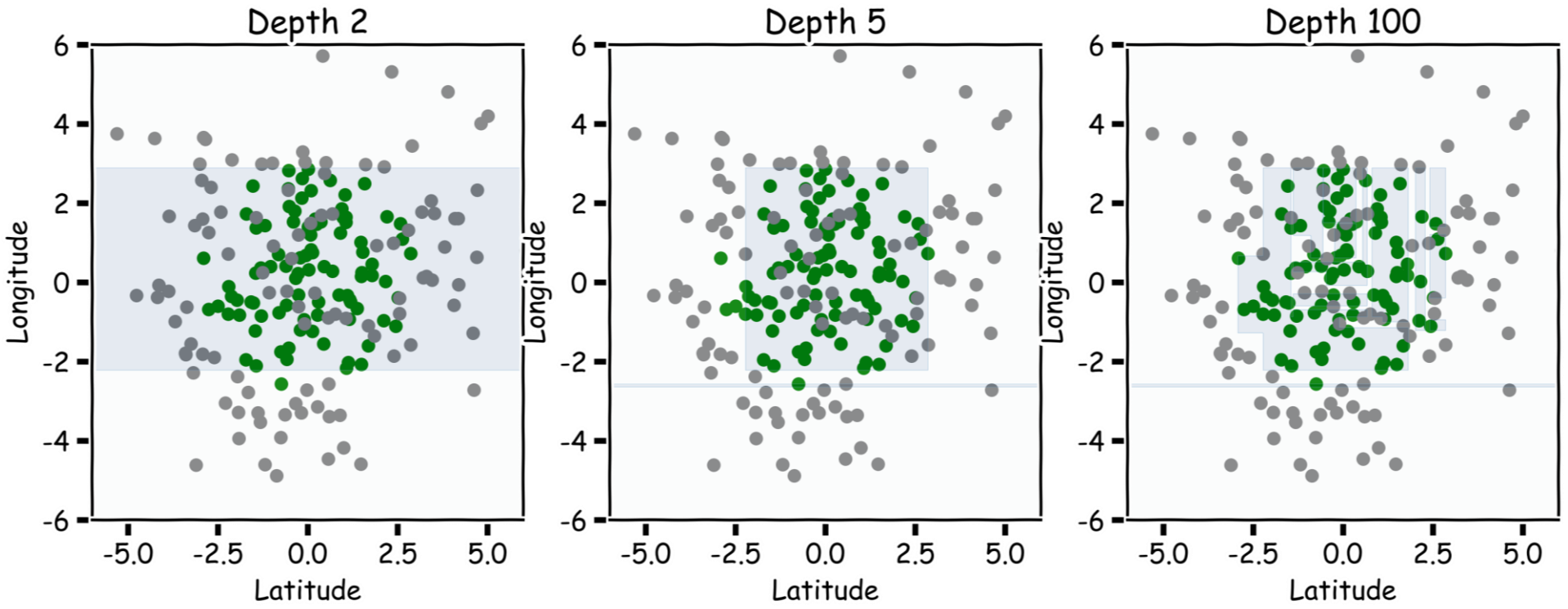


Image from Palvos Protopapas

Bagging

Decision trees models are highly interpretable and fast to train, using our greedy learning algorithm.

However, in order to capture a complex decision boundary (or to approximate a complex function), we need to use a large tree (since each time we can only make axis aligned splits).

We've seen that large trees have high variance and are prone to overfitting.

For these reasons, in practice, decision tree models often underperforms when compared with other classification or regression methods.

Bootstrap (Statistics)

Bootstrapping is a statistical procedure that resamples a single dataset to create many simulated samples.

This process allows for the calculation of standard errors, confidence intervals, and hypothesis testing (Forst)

Bootstrap (Statistics)

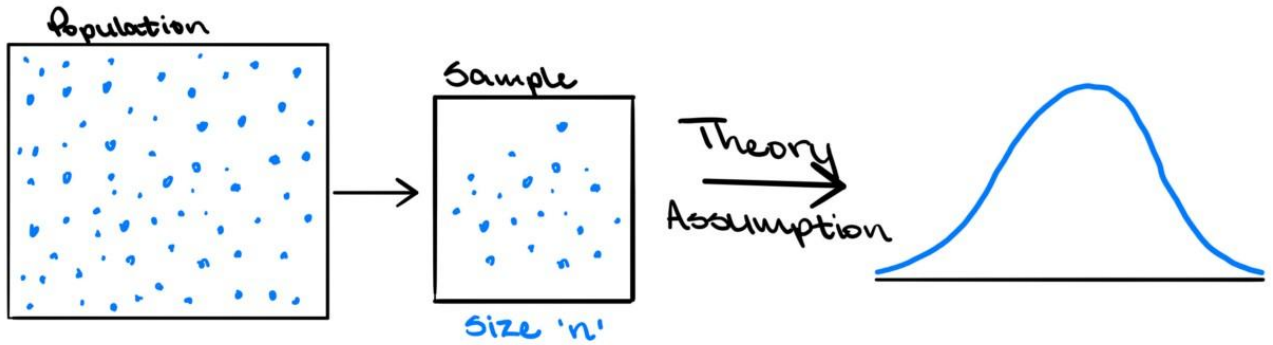


Image from Jonas Dieckmann

Bootstrap (Statistics)

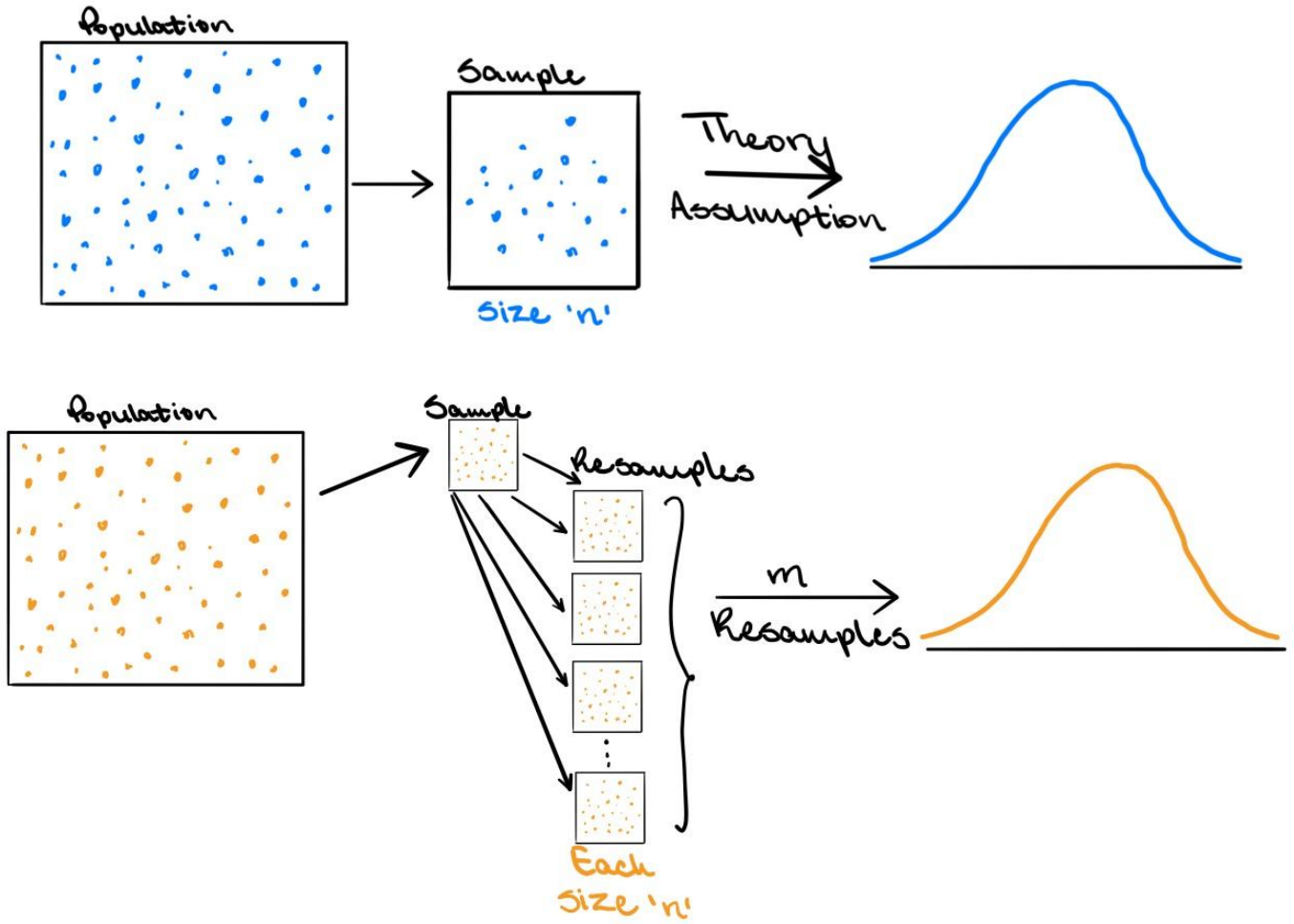


Image from Jonas Dieckmann

Bootstrap (Statistics)

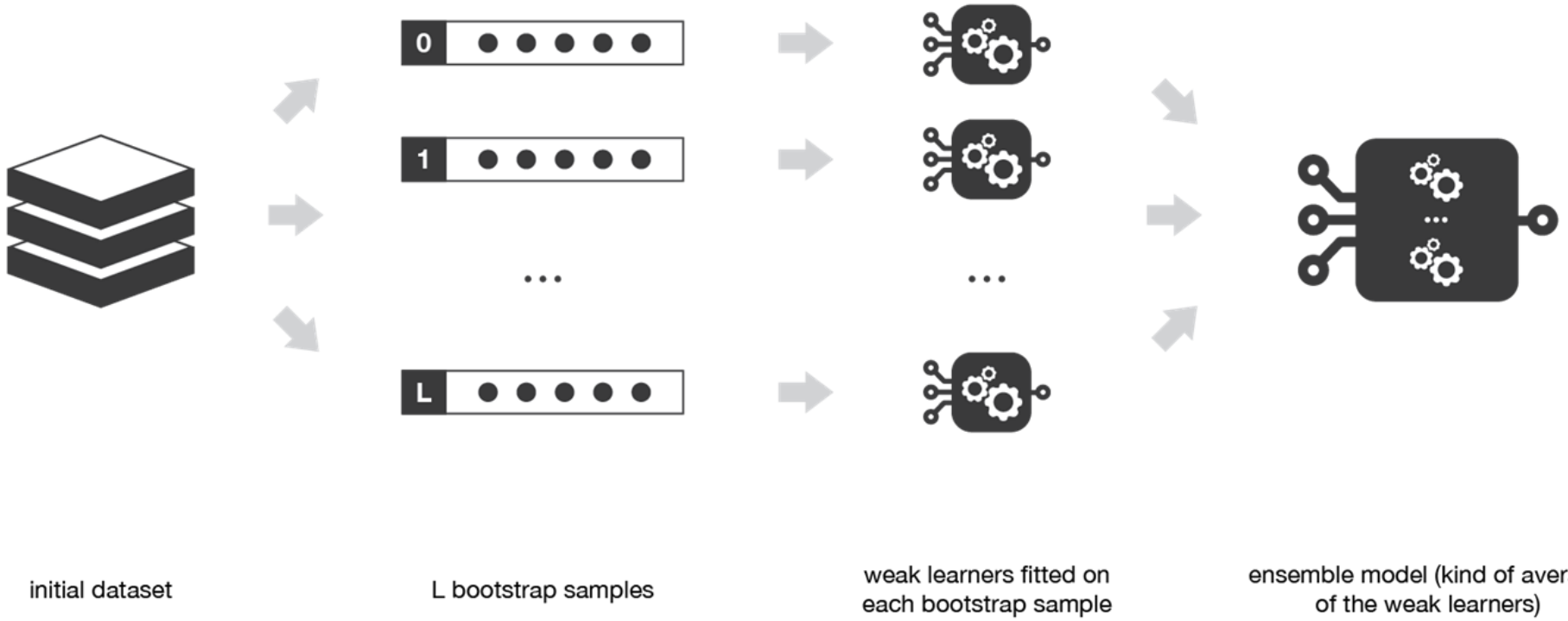


Image from Jonas Dieckmann

Bootstrap (Statistics)

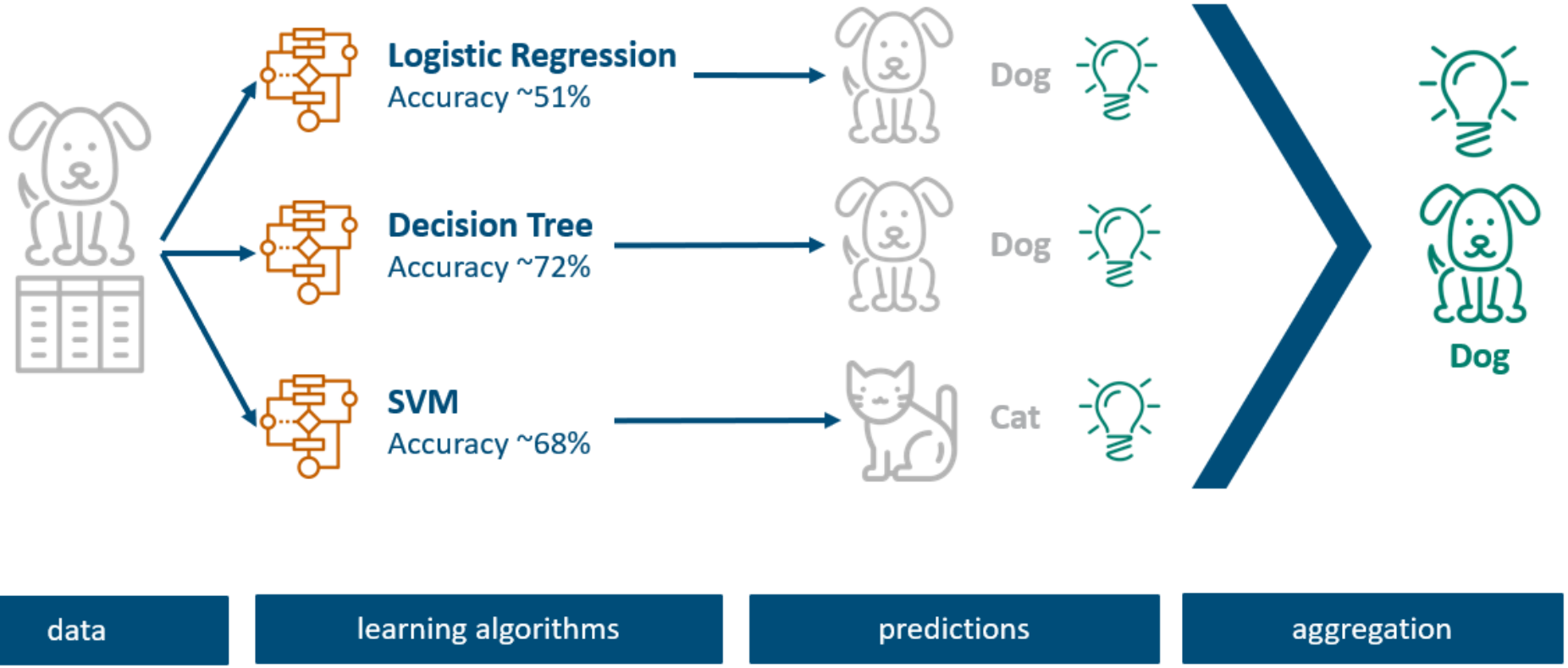


Image from Jonas Dieckmann

Bagging

Combine them? 2 magic realisms

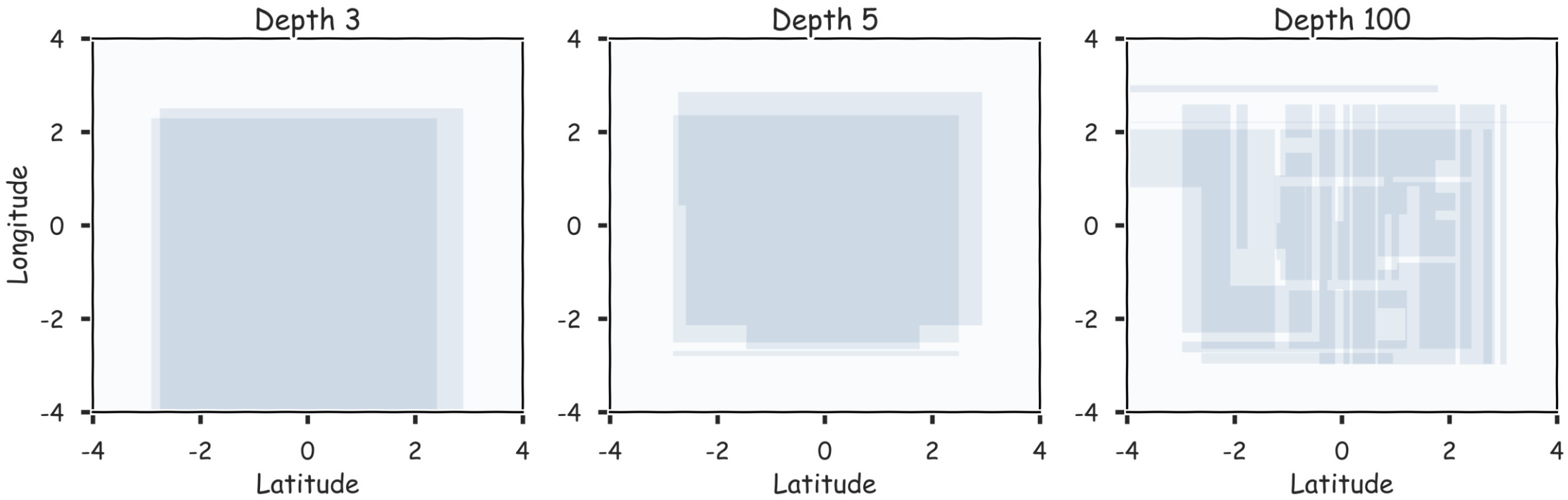


Image from Palvos Protopapas

Combine them? 20 magic realisms

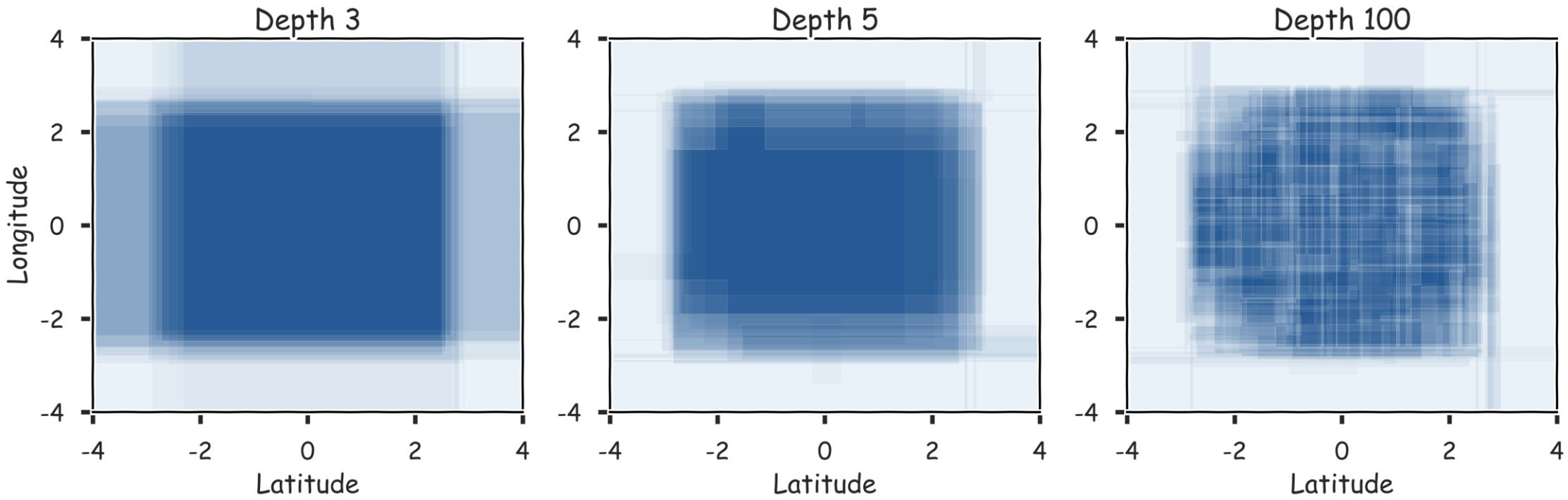


Image from Palvos Protopapas

Bagging

Combine them? 100 magic realisms

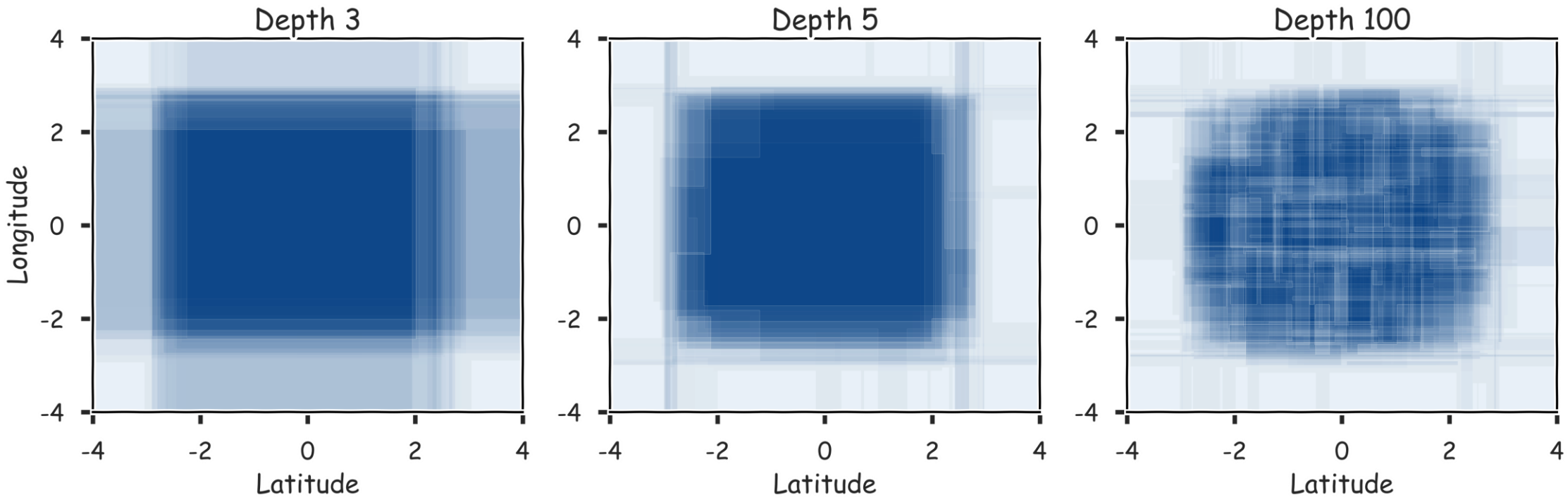


Image from Palvos Protopapas

Bagging

Combine them? 300 magic realisms

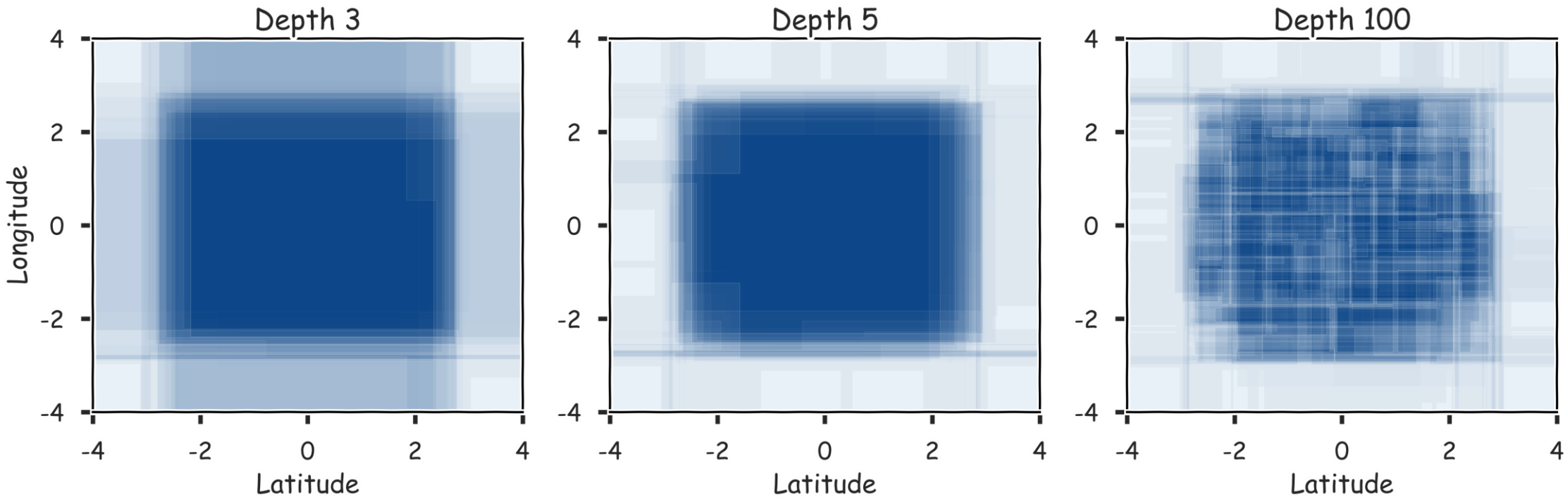


Image from Palvos Protopapas

Bagging

One way to adjust for the high variance of the output of an experiment is to perform the experiment multiple times and then average the results.

The same idea can be applied to high variance models:

1. (Bootstrap) we generate multiple samples of training data, via bootstrapping. We train a full decision tree on each sample of data.
2. (Aggregate) for a given input, we output the averaged outputs of all the models for that input.

For classification, we return the class that is outputted by the plurality of the models. For regression we return the average of the outputs for each tree.

This method is called Bagging (Breiman, 1996), short for, of course,

Bootstrap Aggregating.

Note that bagging enjoys the benefits of:

1. High expressiveness - by using full trees each model is able to approximate complex functions and decision boundaries.

2. Low variance - averaging the prediction of all the models reduces the variance in the final prediction, assuming that we choose a sufficiently large number of trees.

Bagging

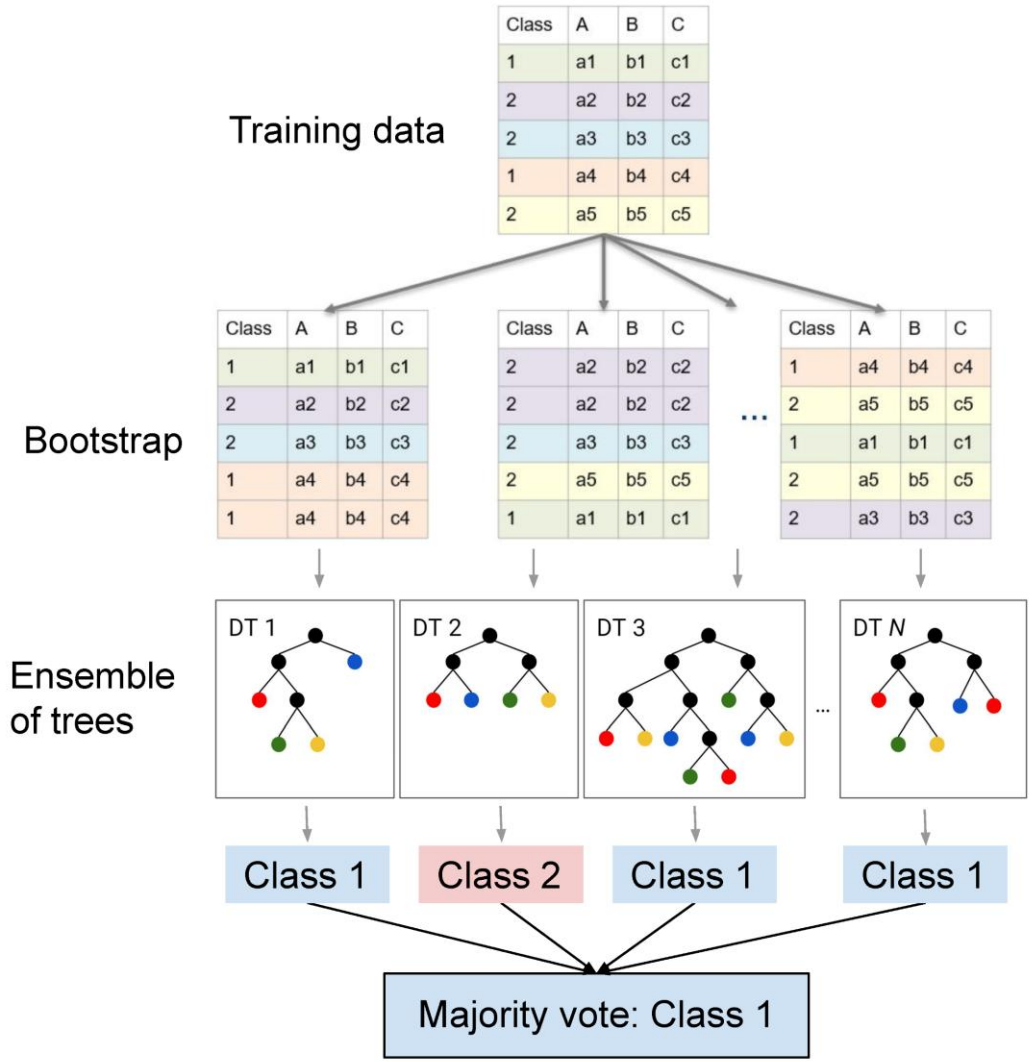


Image from Palvos Protopapas

Bagging

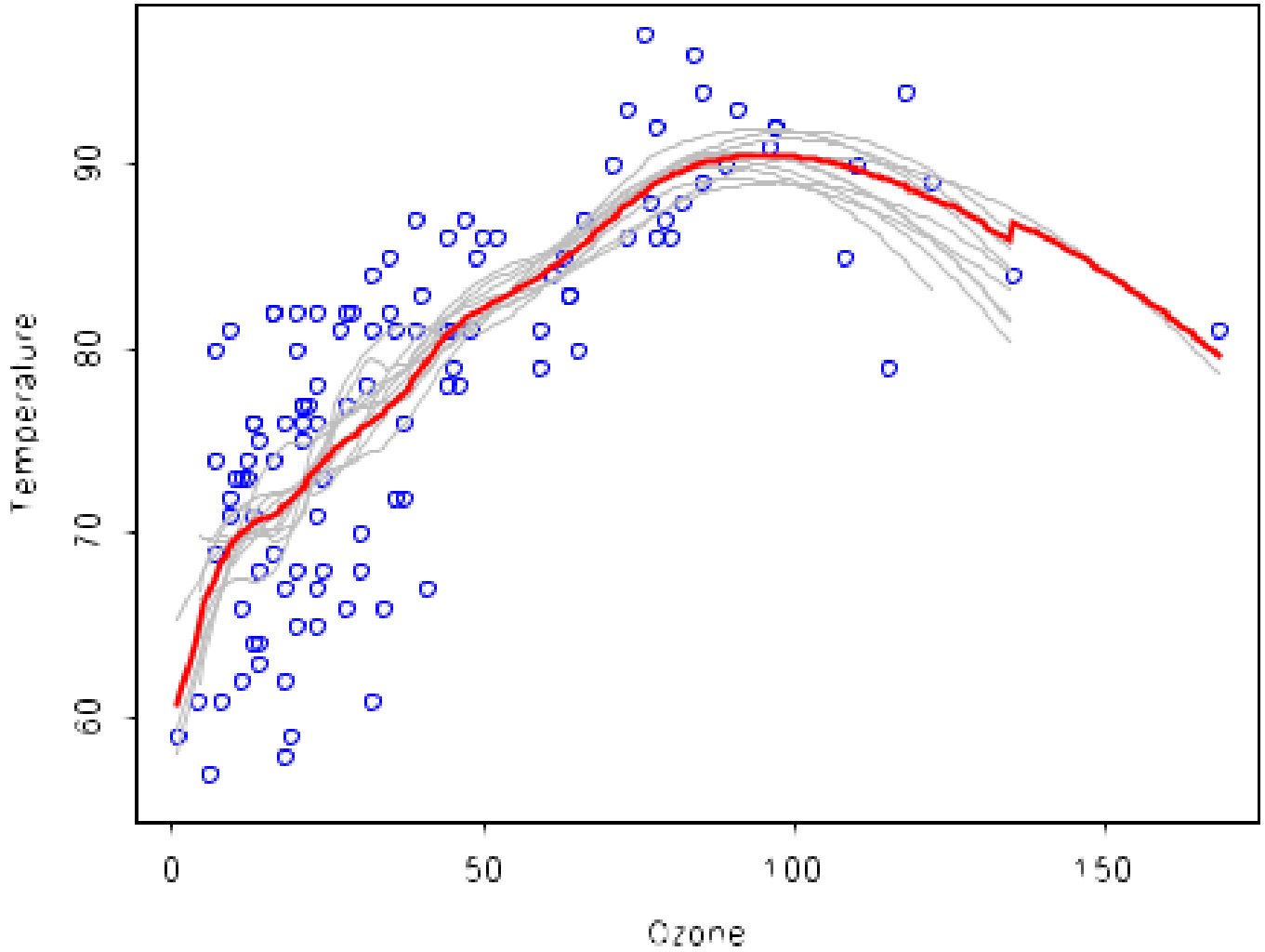


Image from Palvos Protopapas

Bagging

Question: Do you see any problems?

Bagging

Question: Do you see any problems?

Still some overfitting if the trees are too large.

If trees are too shallow it can still underfits.

Interpretability:

The major drawback of bagging (and other ensemble methods that we will study) is that the averaged model is no longer easily interpretable - i.e. one can no longer trace the 'logic' of an output through a series of decisions based on predictor values!

In practice, the ensembles of trees in Bagging tend to be highly correlated.

Suppose we have an extremely strong predictor, x_j , in the training set amongst moderate predictors. Then the greedy learning algorithm ensures that most of the models in the ensemble will choose to split on x_j in early iterations.

That is, each tree in the ensemble is identically distributed, with the expected output of the averaged model the same as the expected output of any one of the trees.

Random Forest is a modified form of bagging that creates ensembles of independent decision trees.

To de-correlate the trees, we:

1. train each tree on a separate bootstrap sample of the full training set (same as in bagging)
2. for each tree, at each split, we randomly select a set of J' predictors from the full set of predictors.

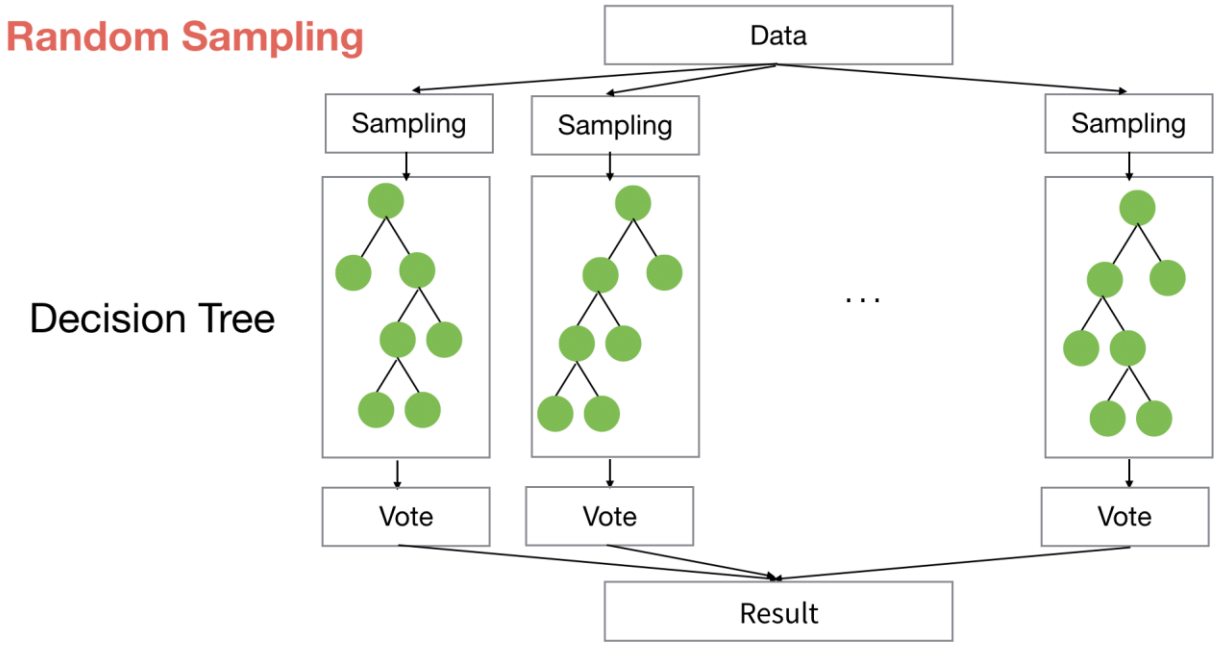
From amongst the J' predictors, we select the optimal predictor and the optimal corresponding threshold for the split.

Random forest models have multiple hyper-parameters to tune:

1. the number of predictors to randomly select at each split
2. the total number of trees in the ensemble
3. the minimum leaf node size

In theory, each tree in the random forest is full, but in practice this can be computationally expensive (and added redundancies in the model), thus, imposing a minimum node size is not unusual.

Variable/Feature Importance



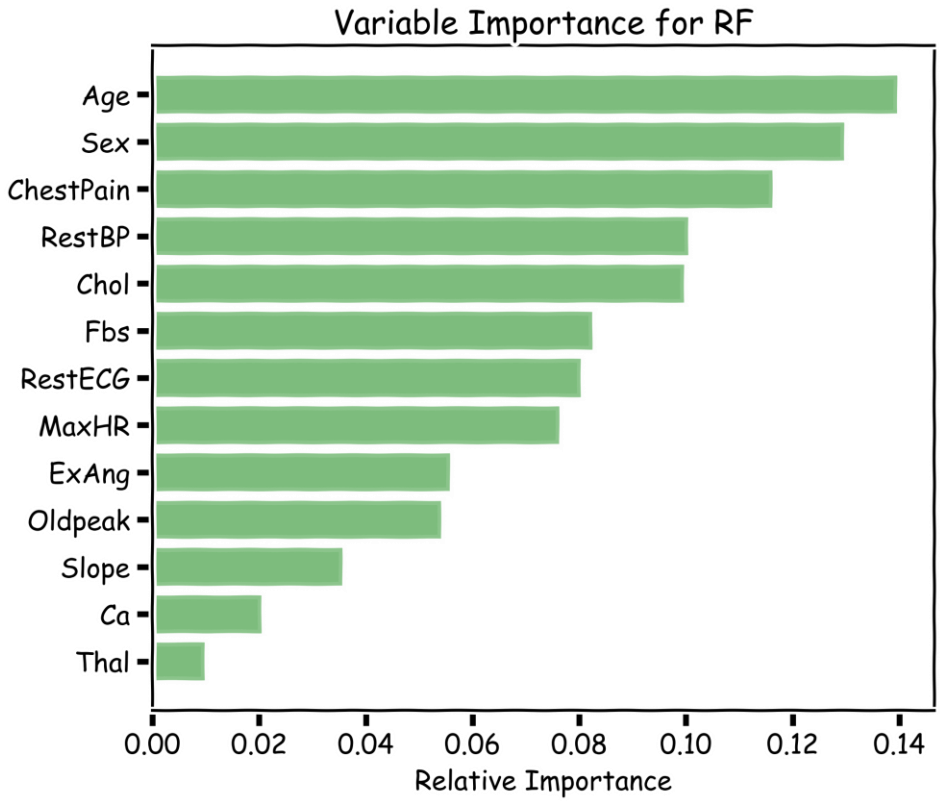


Image from Palvos Protopapas

When the number of predictors is large, but the number of relevant predictors is small, random forests can perform poorly.

Question: Why?

In each split, the chances of selected a relevant predictor will be low and hence most trees in the ensemble will be weak models.

Google Colab
https://bit.ly/BPS5231-L5

When the number of predictors is large, but the number of relevant predictors is small, random forests can perform poorly.

Question: Why?

In each split, the chances of selected a relevant predictor will be low and hence most trees in the ensemble will be weak models.

Question: Could we address the shortcomings of single decision trees models in some other way?

For example, rather than performing variance reduction on complex trees, can we decrease the bias of simple trees - make them more expressive?

Can we learn from our mistakes?

A solution to this problem, making an expressive model from simple trees, is another class of ensemble methods called boosting.

Goal: turn many simple models into one strong model
Build the model additively: start simple, fix what's wrong, repeat
Works for regression and classification
Core concept: learn from residuals (errors)

Why Boosting (vs Bagging/Random Forests)?
Bagging: many models in parallel → average them (reduces variance)
Boosting: models in sequence → each fixes previous mistakes (reduces bias)

Often achieves higher accuracy but needs more careful tuning
More sensitive to hyperparameters and noise

The key intuition behind boosting is that one can take an ensemble of simple models $\{T_h\}_{h \in H}$ and additively combine them into a single, more complex model.

Each model T_h might be a poor fit for the data, but a linear combination of the ensemble

$$T = \sum_h \lambda_h T_H$$

can be expressive/flexible.

Question: But which models should we include in our ensemble? What should the coefficients or weights in the linear combination be?

Gradient boosting is a method for iteratively building a complex regression model T by adding simple models. Each new simple model added to the ensemble compensates for the weaknesses of the current ensemble.

1. Fit a simple model $T^{(0)}$ on the training data

$$\{(x_1, y_1), \ldots, (x_N, y_N)\}$$

Set $T \leftarrow T^{(0)}$. Compute the residuals $\{r_1, \ldots, r_N\}$ for T.

2. Fit a simple model, $T^{(1)}$, to the current residuals, i.e. train using

$$\{(x_1, r_1), \ldots, (x_N, r_N)\}$$

3. Set $T \leftarrow T + \lambda T^{(1)}$
4. Compute residuals, set $r_n \leftarrow r_n - \lambda T^i(x_n),\ n = 1, \ldots, N$
5. Repeat steps 2-4 until stopping condition met.

where λ is a constant called the learning rate.

Boosting

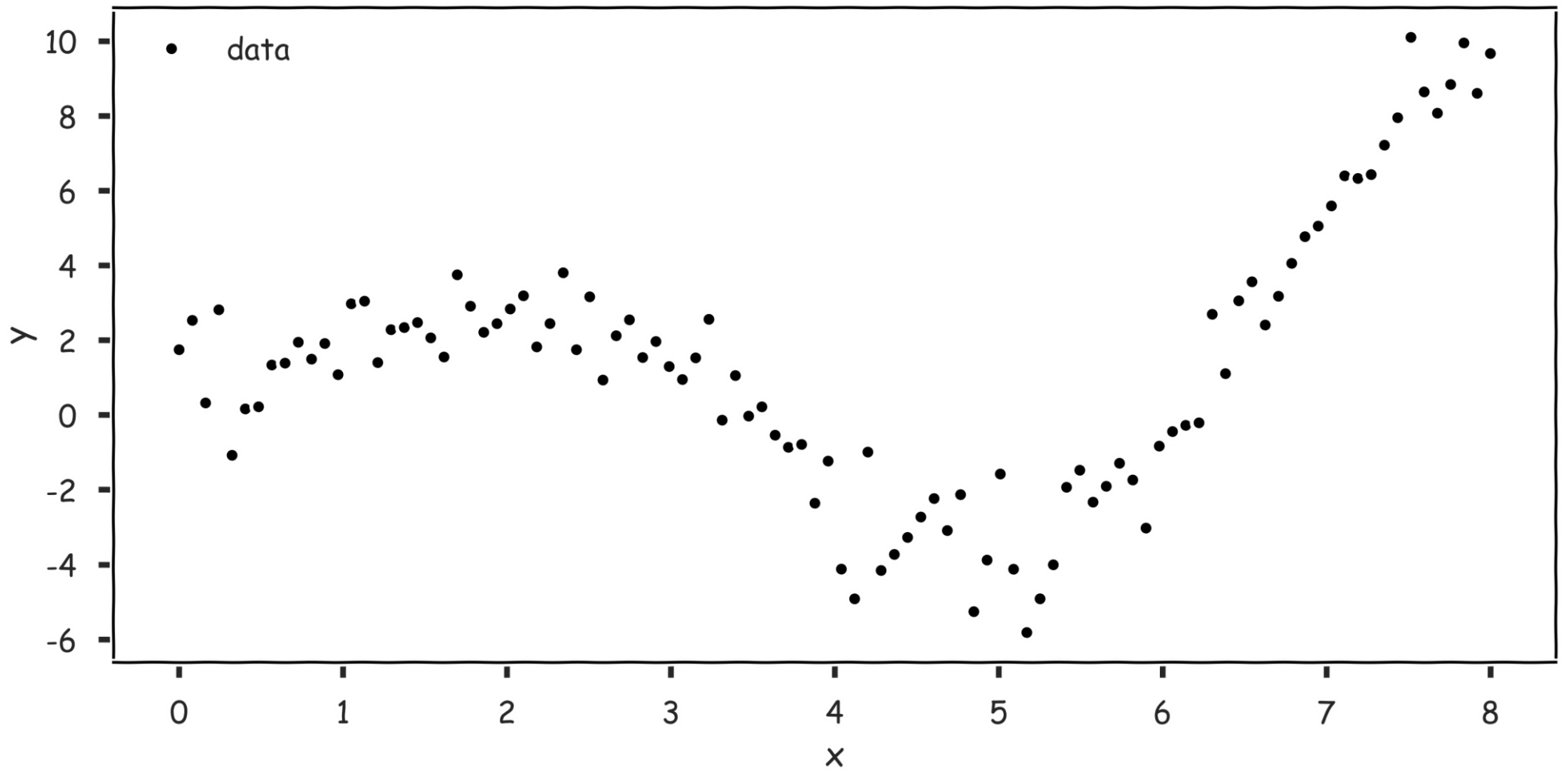


Image from Palvos Protopapas

Boosting

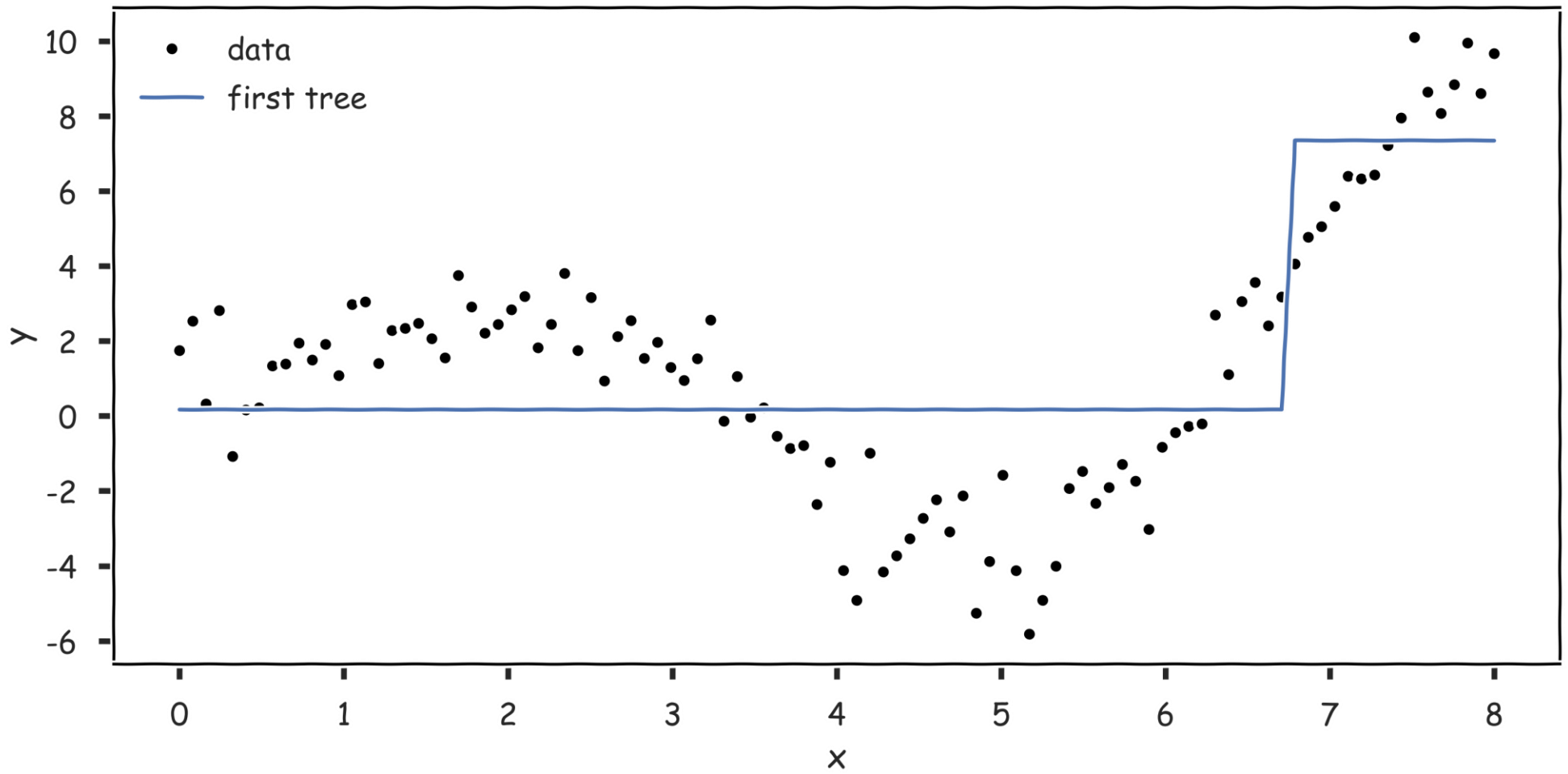


Image from Palvos Protopapas

Boosting

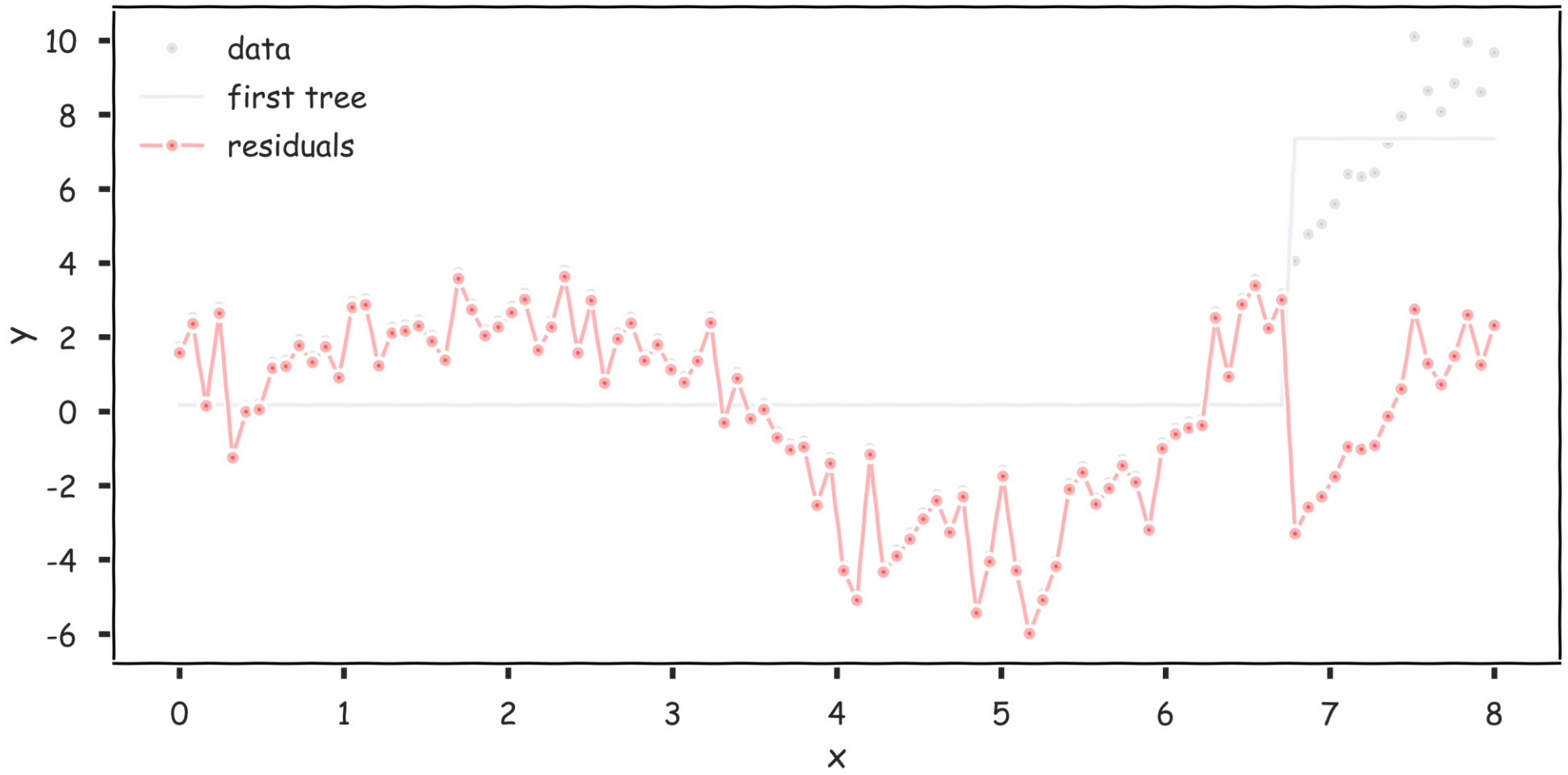


Image from Palvos Protopapas

Boosting

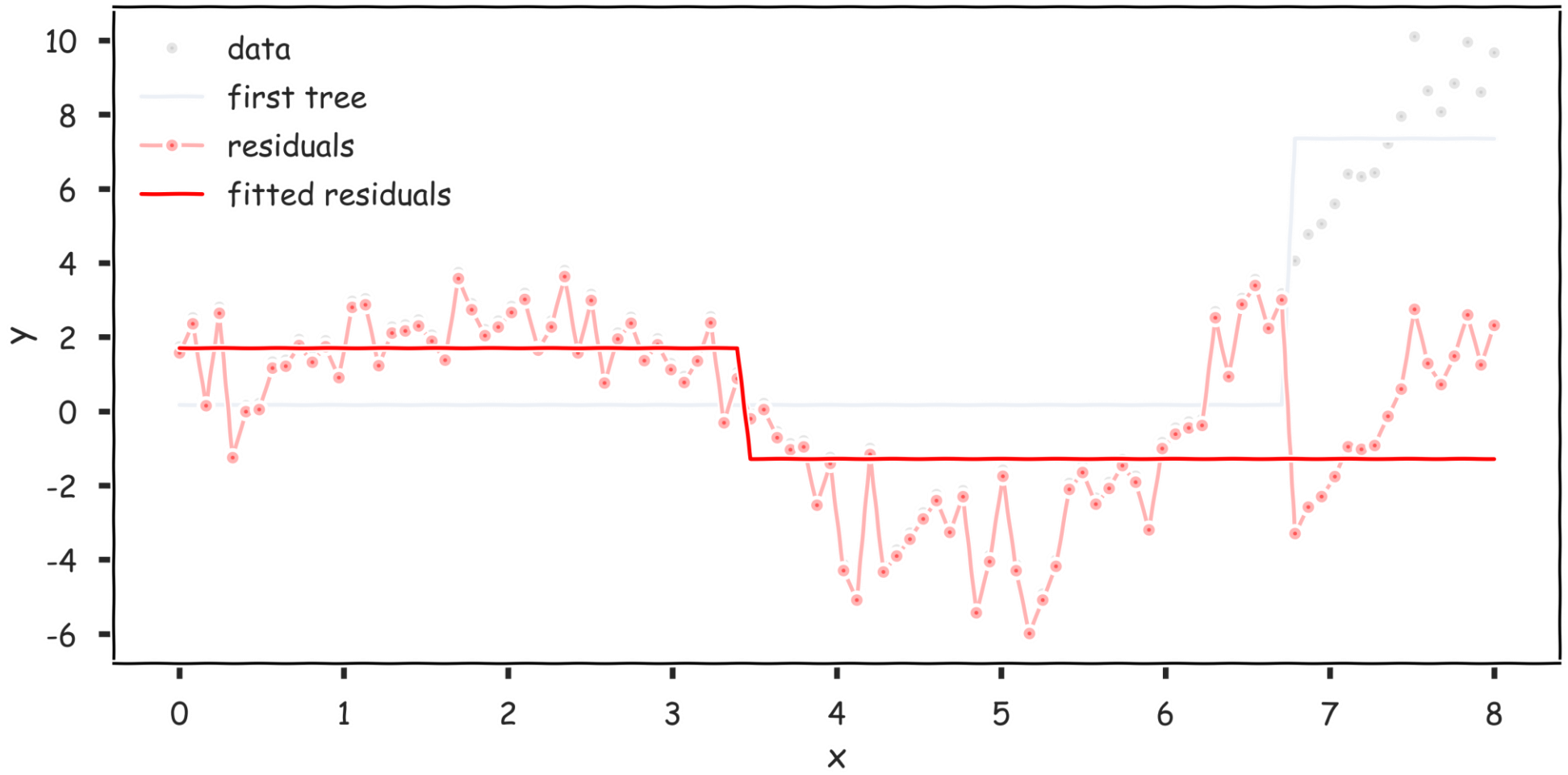


Image from Palvos Protopapas

Boosting

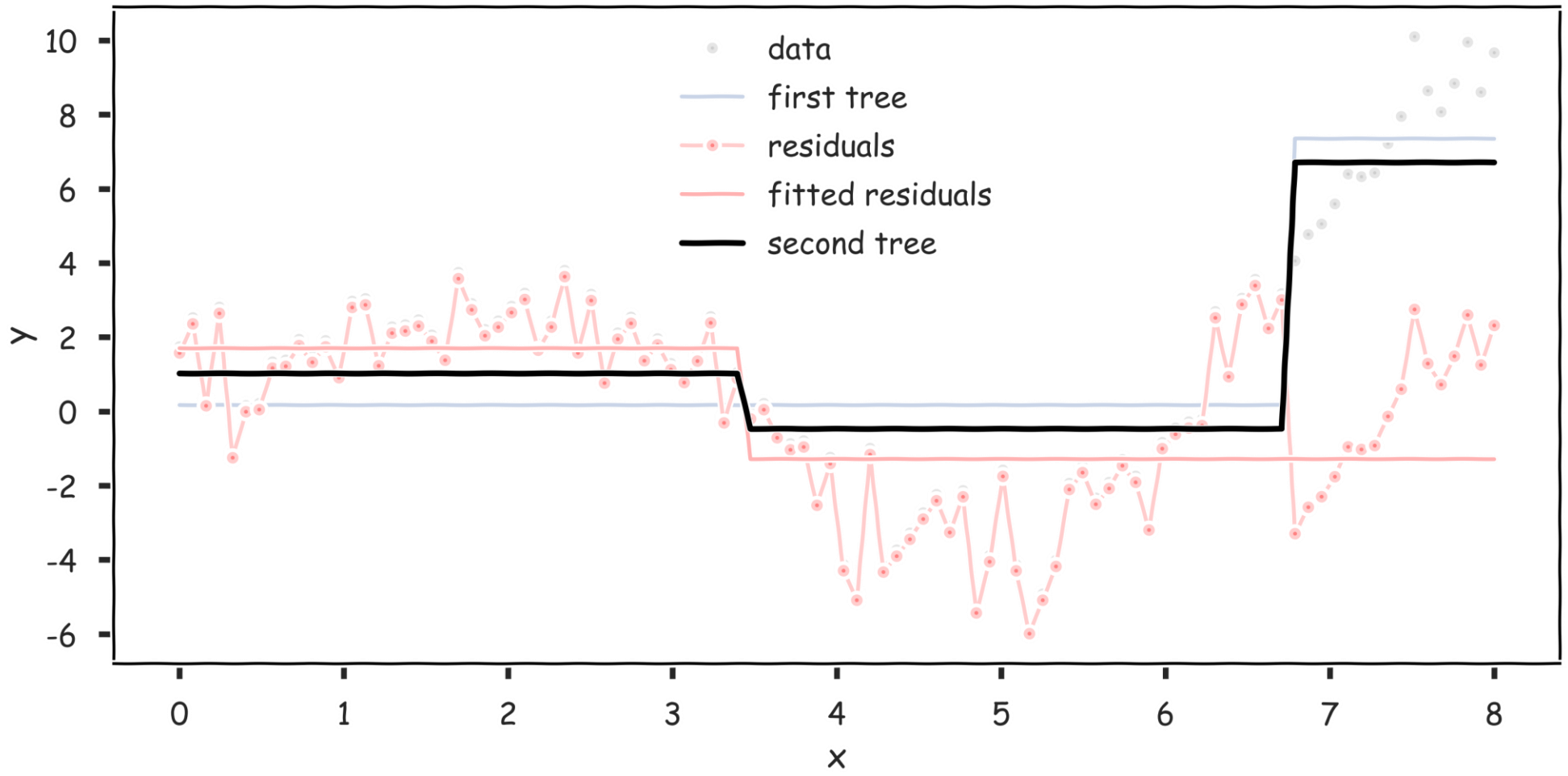


Image from Palvos Protopapas

Boosting

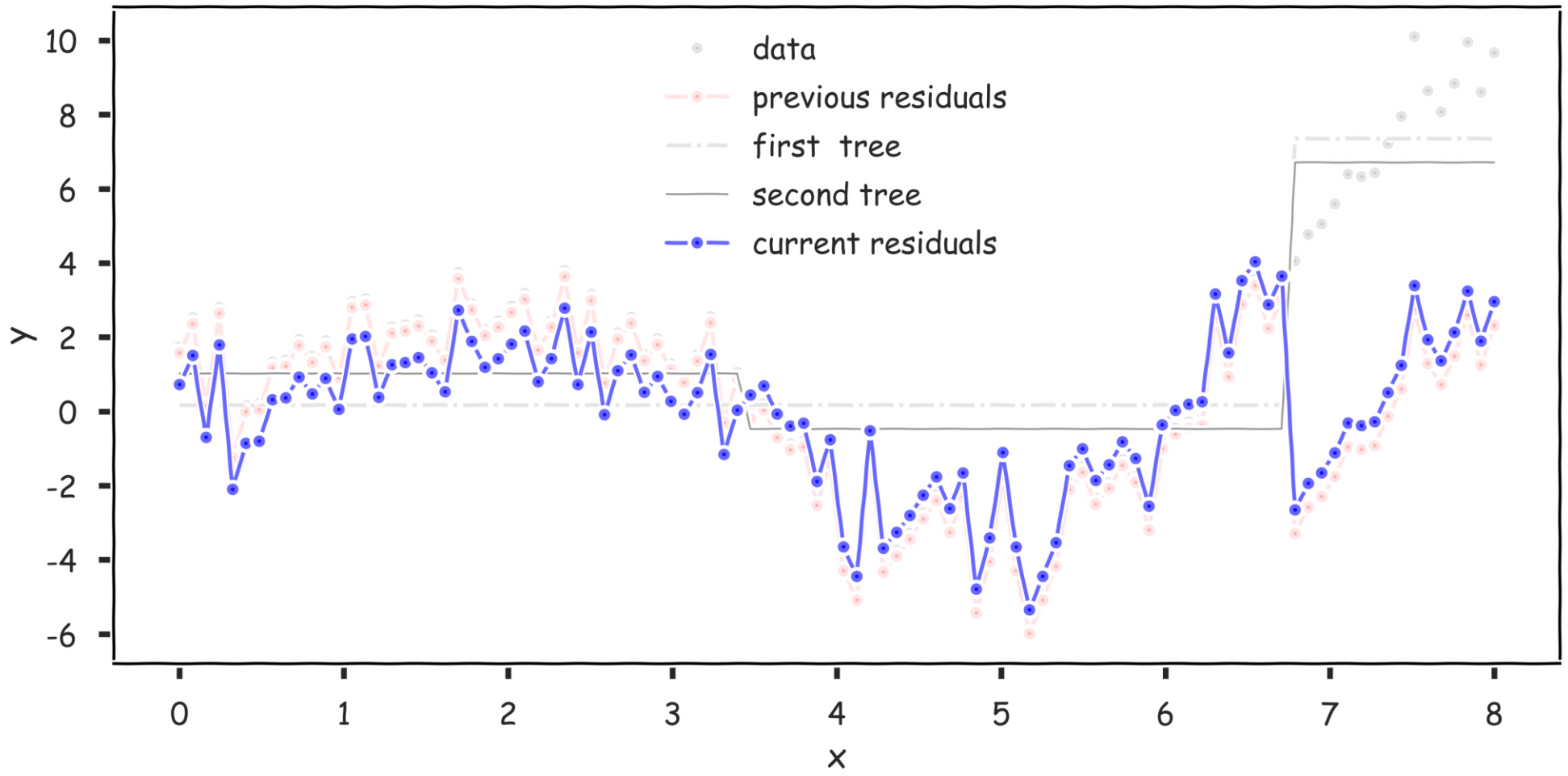


Image from Palvos Protopapas

Boosting

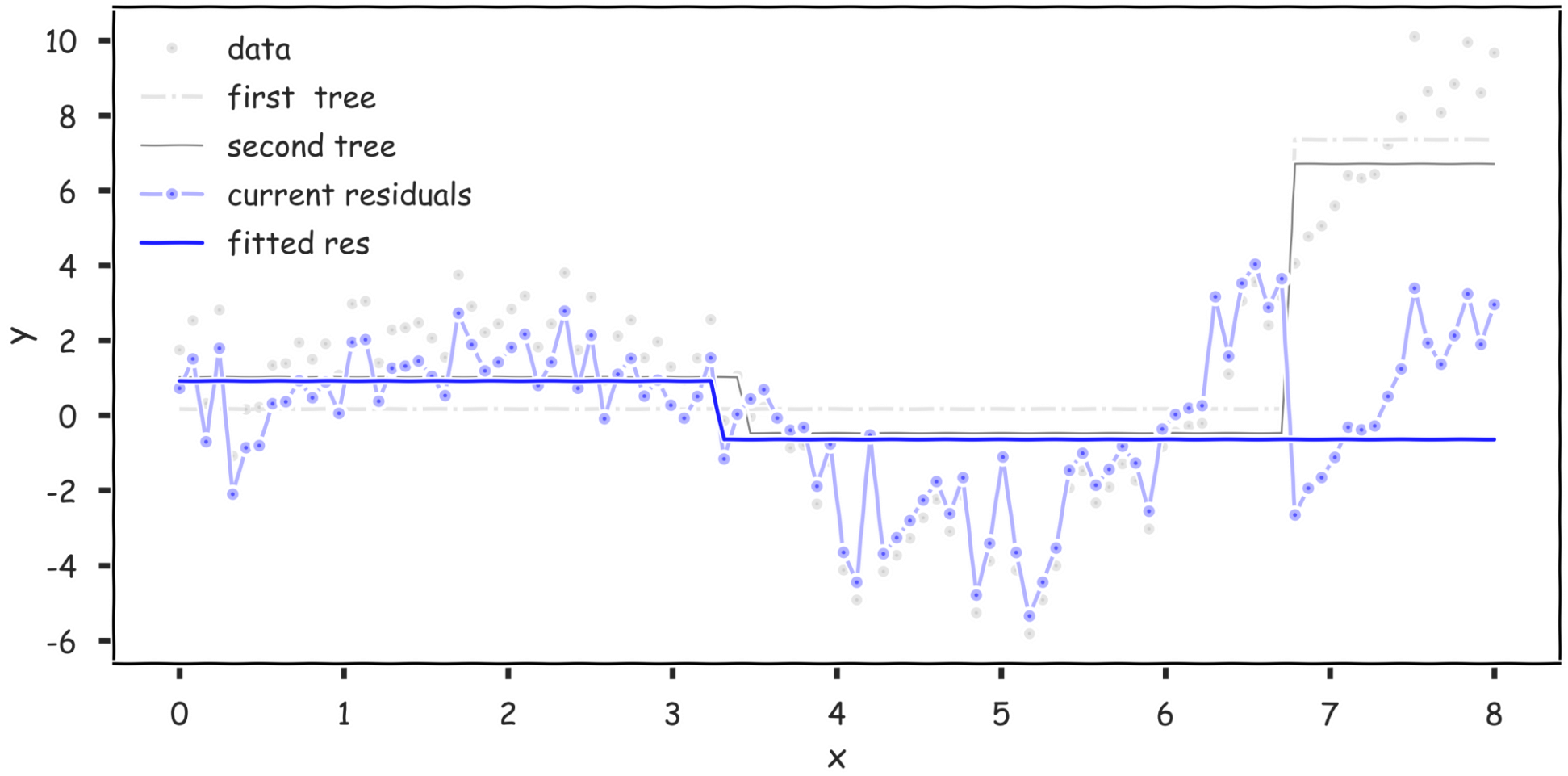


Image from Palvos Protopapas

Boosting

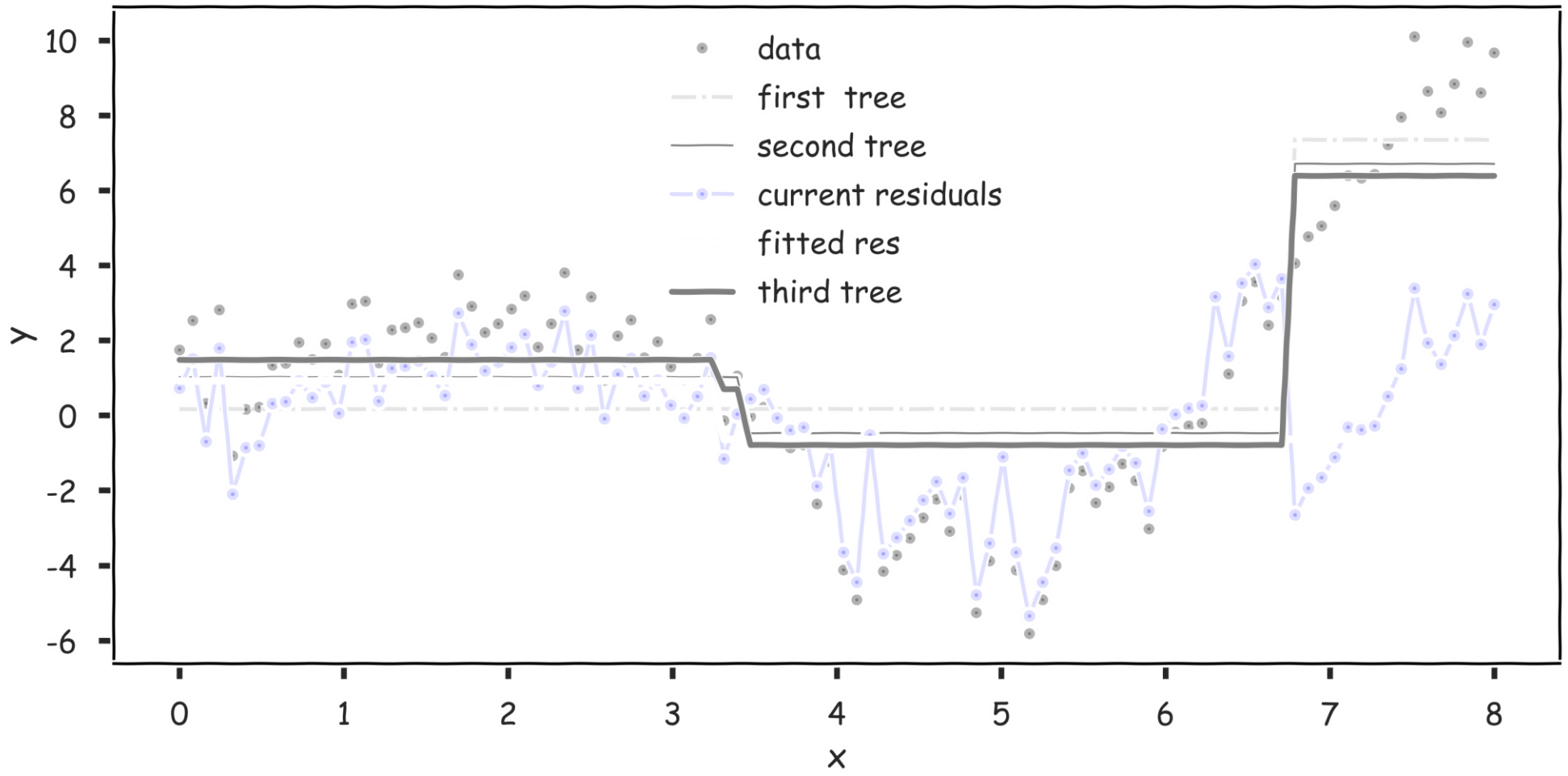


Image from Palvos Protopapas

Intuitively, each simple model $T^{(i)}$ we add to our ensemble model T, models the errors of T.
Thus, with each addition of $T^{(i)}$, the residual is reduced

$$r_n - \lambda T^{(i)}(x_n)$$

Note that gradient boosting has a tuning parameter, λ.

If we want to easily reason about how to choose λ and investigate the effect of λ on the model T, we need a bit more mathematical formalism. In particular, how can we effectively descend through this optimization via an iterative algorithm?

We can formulate gradient boosting as a type of gradient descent.

What is (perhaps) the most famous / popular boosting model?

XGBoost: A Scalable Tree Boosting System

Tianqi Chen, Carlos Guestrin

Tree boosting is a highly effective and widely used machine learning method. In this paper, we describe a scalable end-to-end tree boosting system called XGBoost, which is used widely by data scientists to achieve state-of-the-art results on many machine learning challenges. We propose a novel sparsity-aware algorithm for sparse data and weighted quantile sketch for approximate tree learning. More importantly, we provide insights on cache access patterns, data compression and sharding to build a scalable tree boosting system. By combining these insights, XGBoost scales beyond billions of examples using far fewer resources than existing systems.

Comments: KDD'16 changed all figures to type1
Subjects: **Machine Learning (cs.LG)**
Cite as: arXiv:1603.02754 **[cs.LG]**
(or arXiv:1603.02754v3 **[cs.LG]** for this version)
https://doi.org/10.48550/arXiv.1603.02754
Related DOI: https://doi.org/10.1145/2939672.2939785

Kaggle Champions Swear by XGBoost — And You Can Too

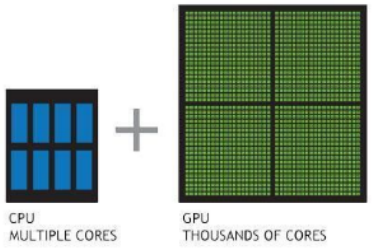
by **superorange0707** · June 12th, 2025

TLDR >

NVIDIA Products Solutions Industries Shop Drivers Support

How XGBoost Runs Better with GPUs

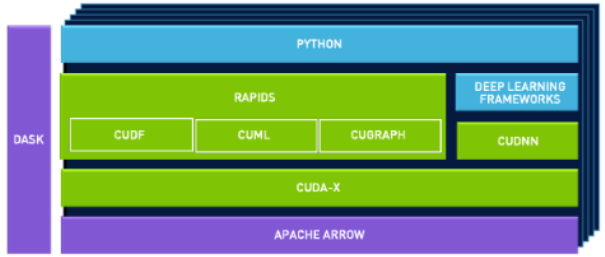
CPU-powered machine learning tasks with XGBoost can literally take hours to run. That's because creating highly accurate, state-of-the-art prediction results involves the creation of thousands of decision trees and the testing of large numbers of parameter combinations. Graphics processing units, or GPUs, with their massively parallel architecture consisting of thousands of small efficient cores, can launch thousands of parallel threads simultaneously to supercharge compute-intensive tasks.



NVIDIA developed NVIDIA RAPIDS™—an open-source data analytics and machine learning acceleration platform—or executing end-to-end data science training pipelines completely in GPUs. It relies on NVIDIA CUDA® primitives for low-level compute optimization, but exposes that GPU parallelism and high memory bandwidth through user-friendly Python interfaces.

Focusing on common data preparation tasks for analytics and data science, RAPIDS offers a familiar DataFrame API that integrates with scikit-learn and a variety of machine learning algorithms without paying typical serialization costs. This allows acceleration for end-to-end pipelines—from data prep to machine learning to deep learning. RAPIDS also includes support for multi-node, multi-GPU deployments, enabling vastly accelerated processing and training on much larger dataset sizes.

Machine Learning to Deep Learning: All on GPU



Boosting Zoo

Method	Core idea	Strengths	Watch-outs / Tips
XGBoost	2nd-order objective + L1/L2 on leaves; exact/approx splits; missing-value path.	Strong on tabular data; robust, feature-rich; good CPU/GPU.	Can overfit with deep trees; tune eta, max_depth, min_child_weight, subsample, colsample, lambda/alpha.
CatBoost	Ordered boosting + native categorical encoding (no leakage).	Best with many categoricals; strong defaults; less tuning.	Slower on pure numeric; set loss_function, depth, l2_leaf_reg; use GPU if available.
LightGBM	Histogram splits + leaf-wise (best-first) growth; GOSS/EFB speeding.	Very fast on large/high-dim data; memory-efficient; great defaults.	Leaf-wise can overfit—raise min_data_in_leaf, cap max_depth; mind small datasets.
AdaBoost	Reweight misclassified points; closed-form stage weights (exp. loss).	Very simple; works well with stumps; fast.	Sensitive to label noise/outliers; mainly for classification; fewer knobs to regularize.

General tabular SOTA baseline → XGBoost
Lots of categorical features → CatBoost
Large datasets / speed → LightGBM (Developed by Microsoft)
Simple tasks, classifications → AdaBoost

Google Colab
https://bit.ly/BPS5231-L5

L05.1
Supervised 5

Bagging

Random Forest

Boosting

XGBoost

L05.2
Visualizations

Exploratory Data Analysis

Communication

L05.3
Simulations

Rhino

ClimateStudio

Visualization

- Exploratory Data Analysis (EDA)
- Communication

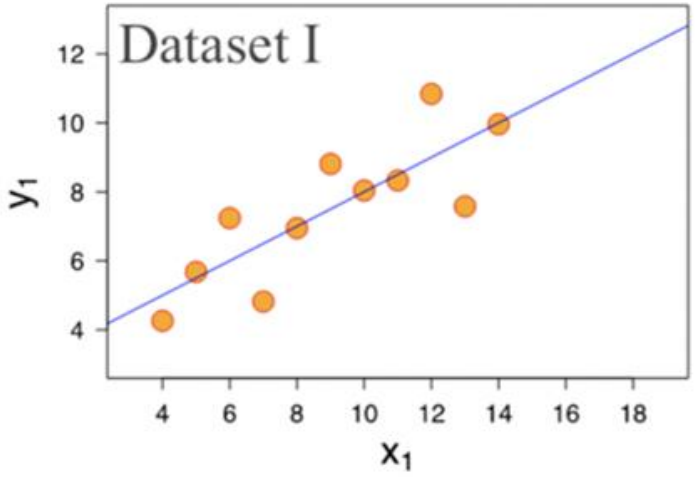
Anscombe's Data

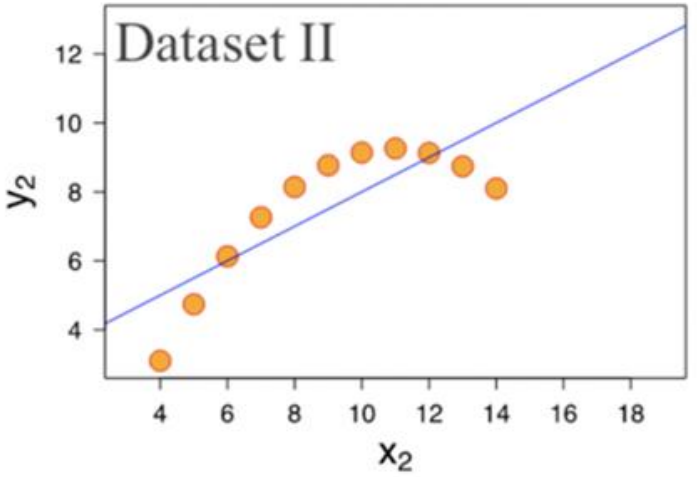
	Dataset I		Dataset II		Dataset III		Dataset IV	
	x	y	x	y	x	y	x	y
	10	8.04	10	9.14	10	7.46	8	6.58
	8	6.95	8	8.14	8	6.77	8	5.76
	13	7.58	13	8.74	13	12.74	8	7.71
	9	8.81	9	8.77	9	7.11	8	8.84
	11	8.33	11	9.26	11	7.81	8	8.47
	14	9.96	14	8.1	14	8.84	8	7.04
	6	7.24	6	6.13	6	6.08	8	5.25
	4	4.26	4	3.1	4	5.39	19	12.5
	12	10.84	12	9.13	12	8.15	8	5.56
	7	4.82	7	7.26	7	6.42	8	7.91
	5	5.68	5	4.74	5	5.73	8	6.89
Sum:	99.00	82.51	99.00	82.51	99.00	82.51	99.00	82.51
Avg:	9.00	7.50	9.00	7.50	9.00	7.50	9.00	7.50
Std:	3.32	2.03	3.32	2.03	3.32	2.03	3.32	2.03

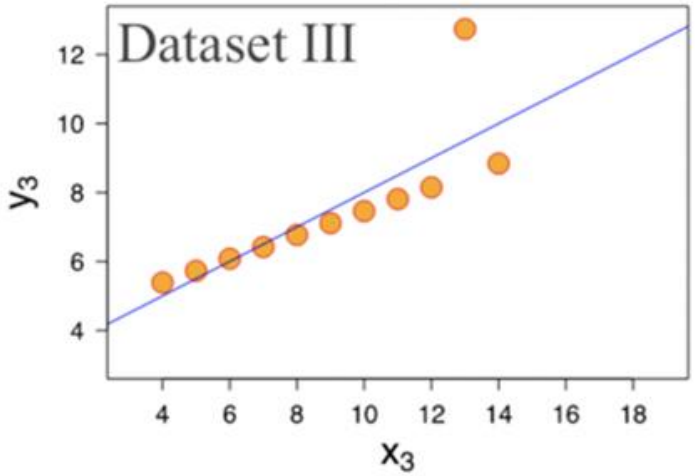
Anscombe's Data

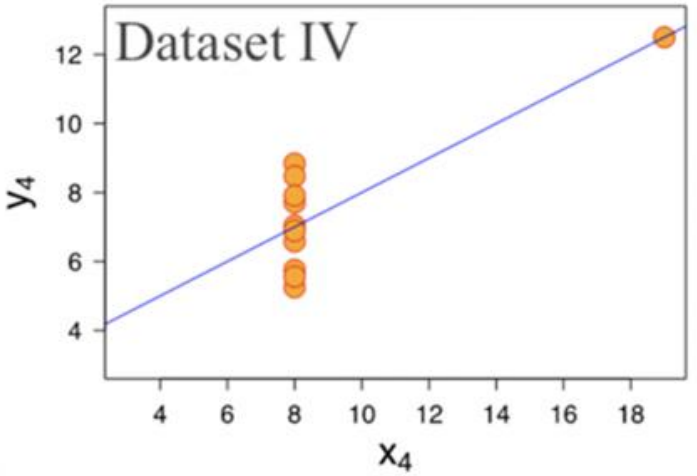
Summary Statistics clearly don't tell the story of underlying data.

Picture can be worth a thousand words

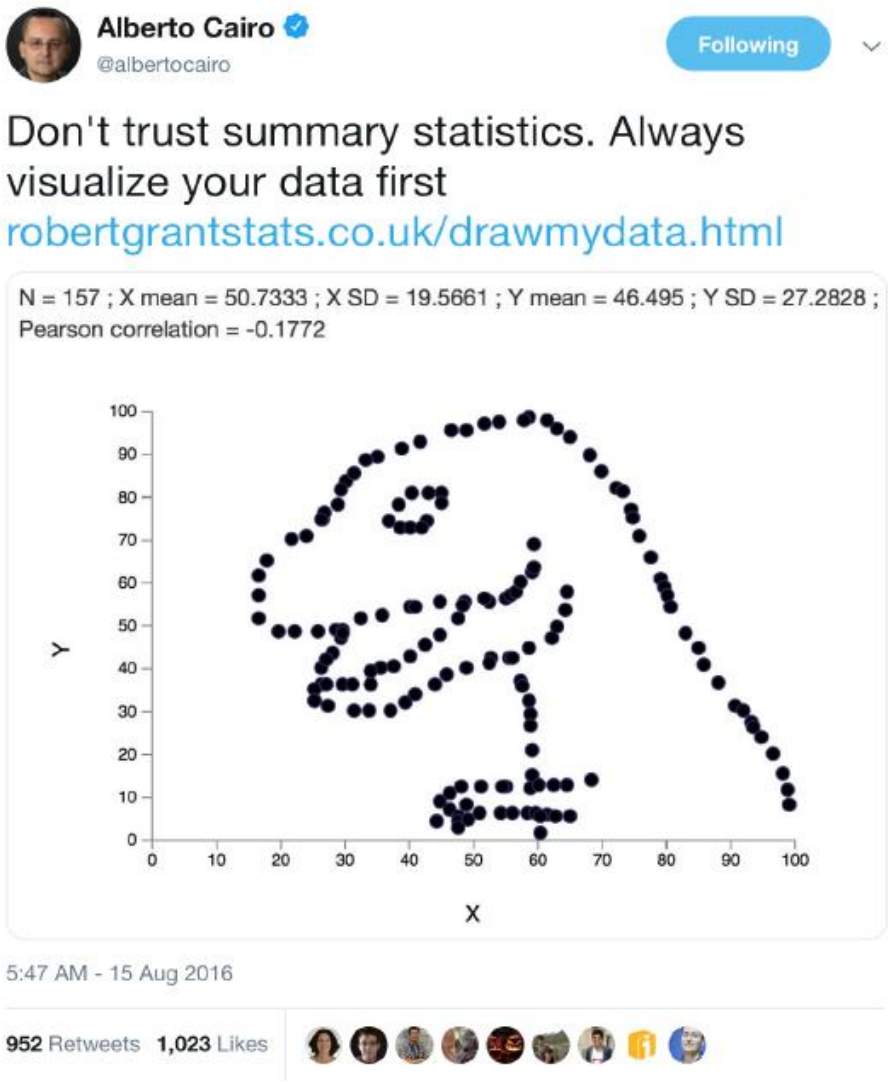








Visualizations



Analyze (exploratory)

- Explore the data
- Assess a situation
- Determine how to proceed
- Decide what to do

Communicate (explanatory)

- Present data and ideas
- Explain and inform
- Provide evidence and support
- Influence and persuade

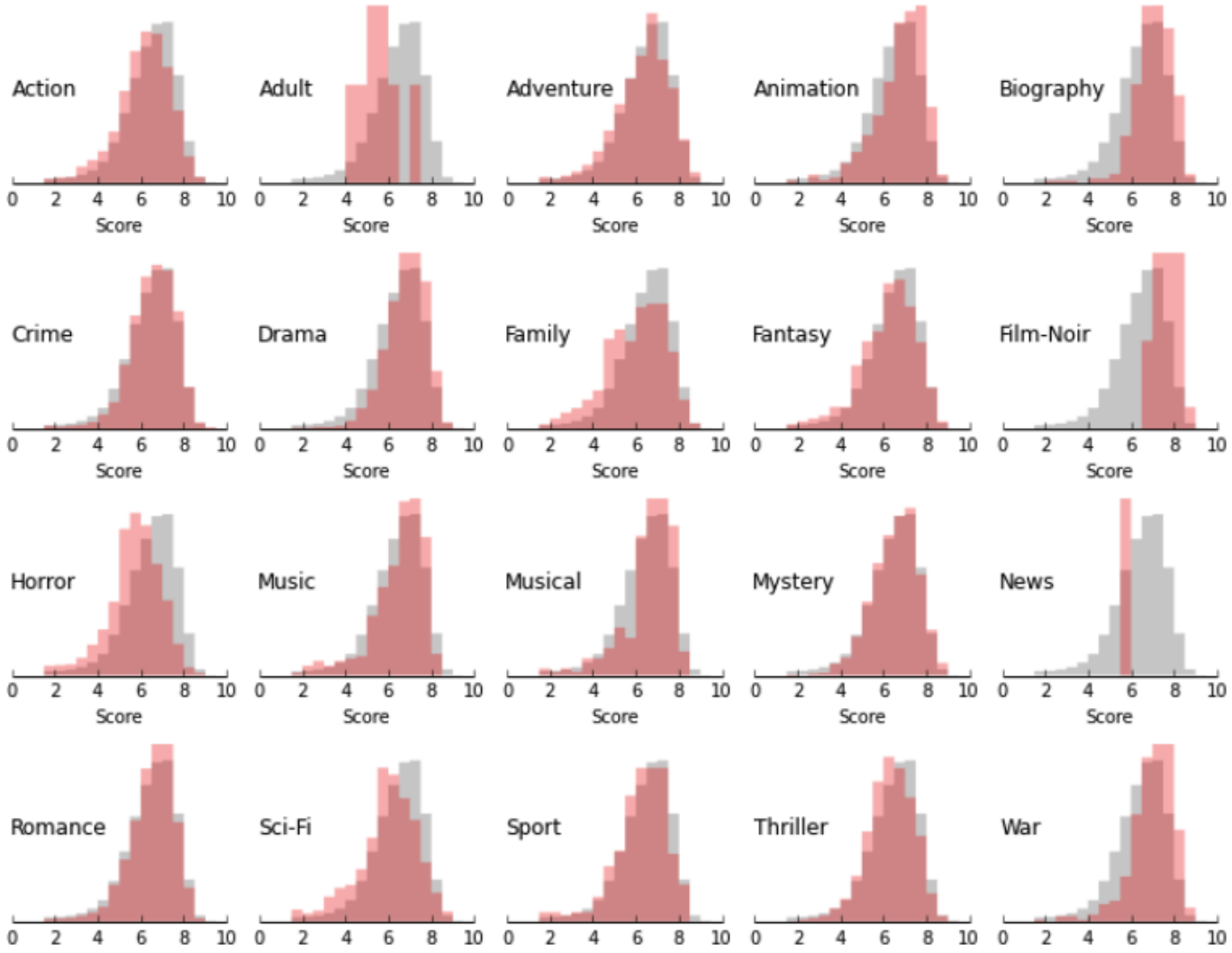
Purposes of EDA

- Maximize insight into a dataset
- Uncover underlying structure
- Detect outliers
- Test underlying assumptions
- Develop parsimonious models

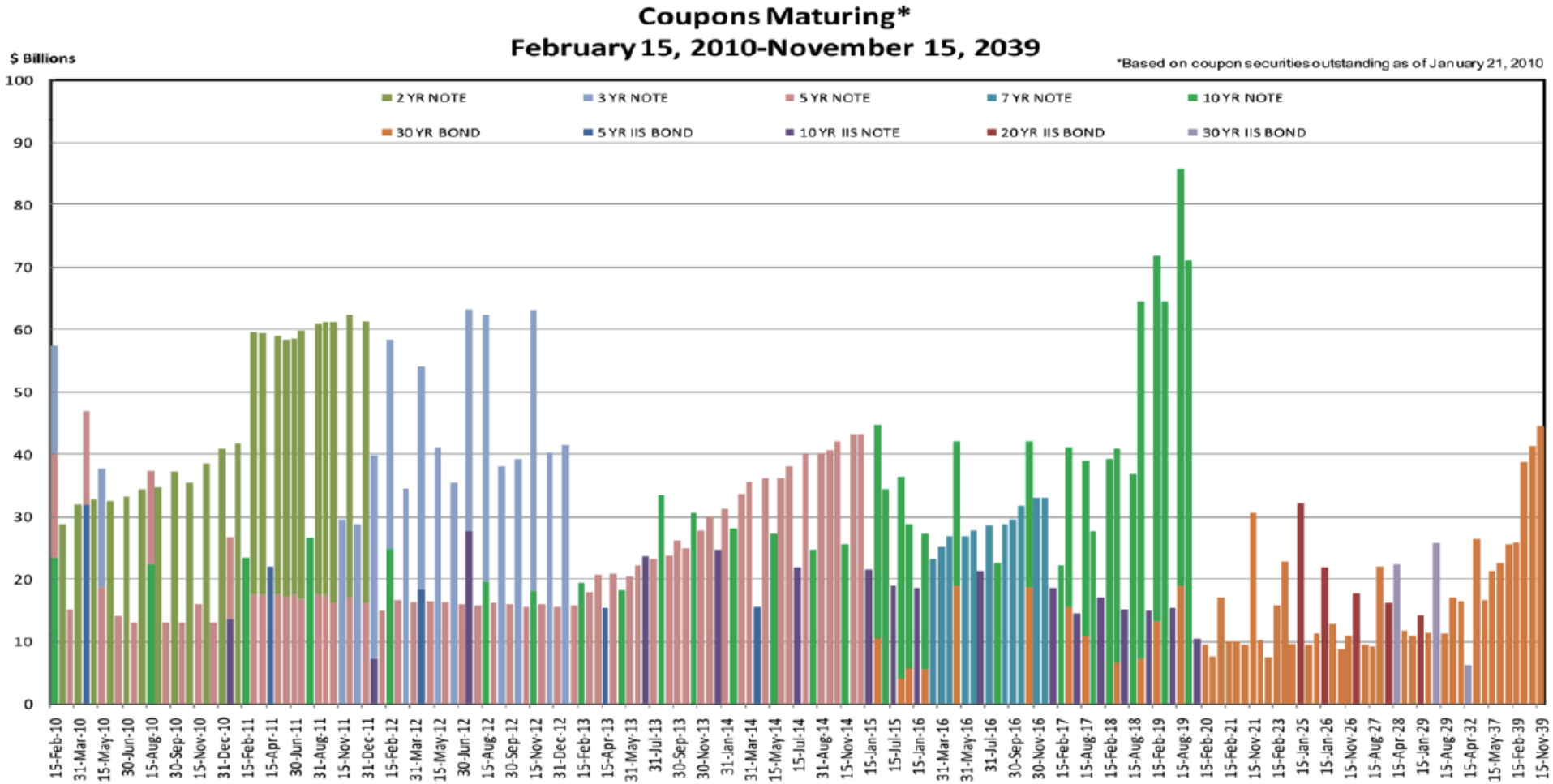
Effective Visualization

- Graphical Integrity
- Scope
- Displays
- Sensible Design

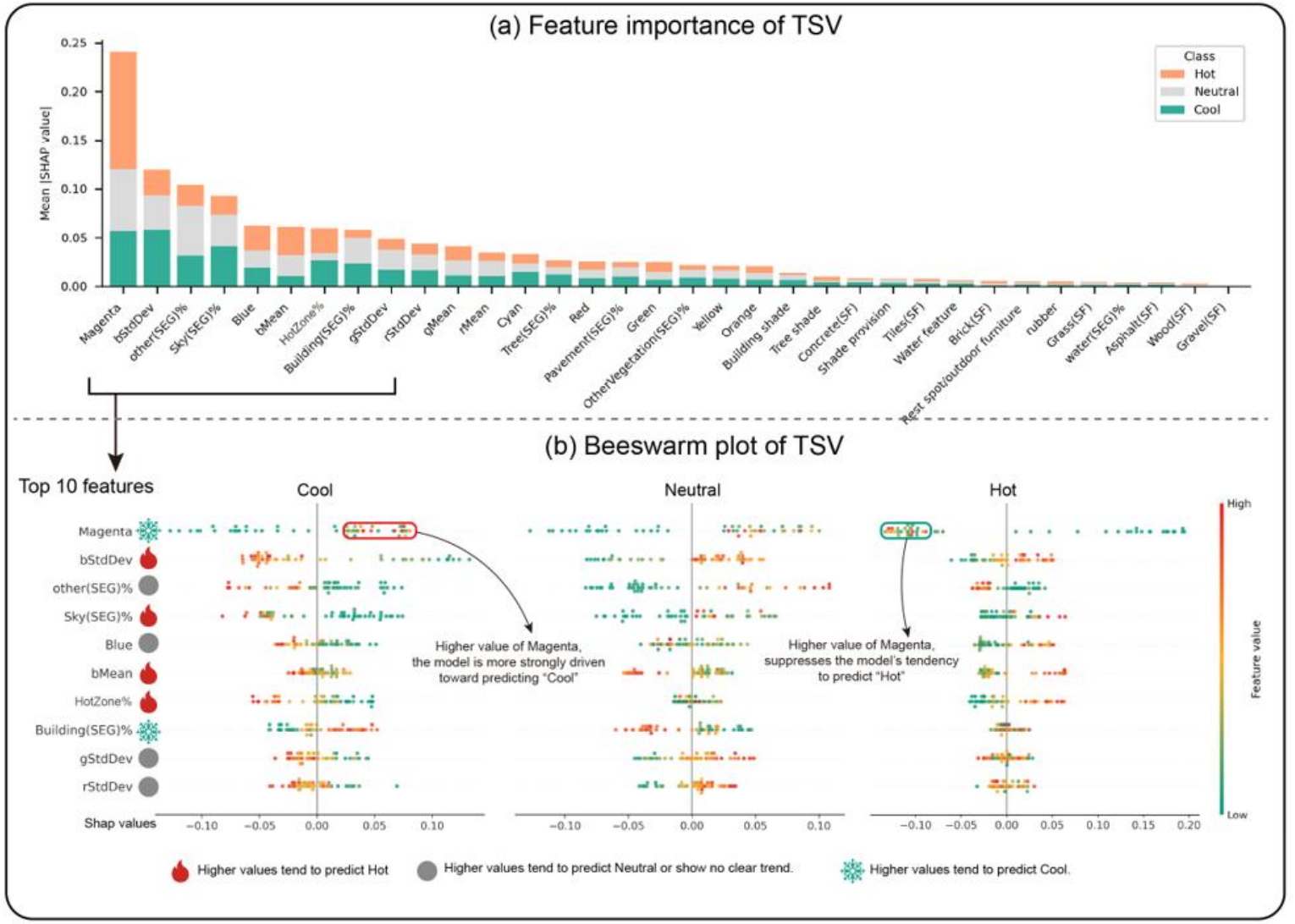
Explore


Action
Adult
Adventure
Animation
Biography
Crime
Drama
Family
Fantasy
Film-Noir
Horror
Music
Musical
Mystery
News
Romance
Sci-Fi
Sport
Thriller
War
0 2 4 6 8 10
Score

Is this chart effective?



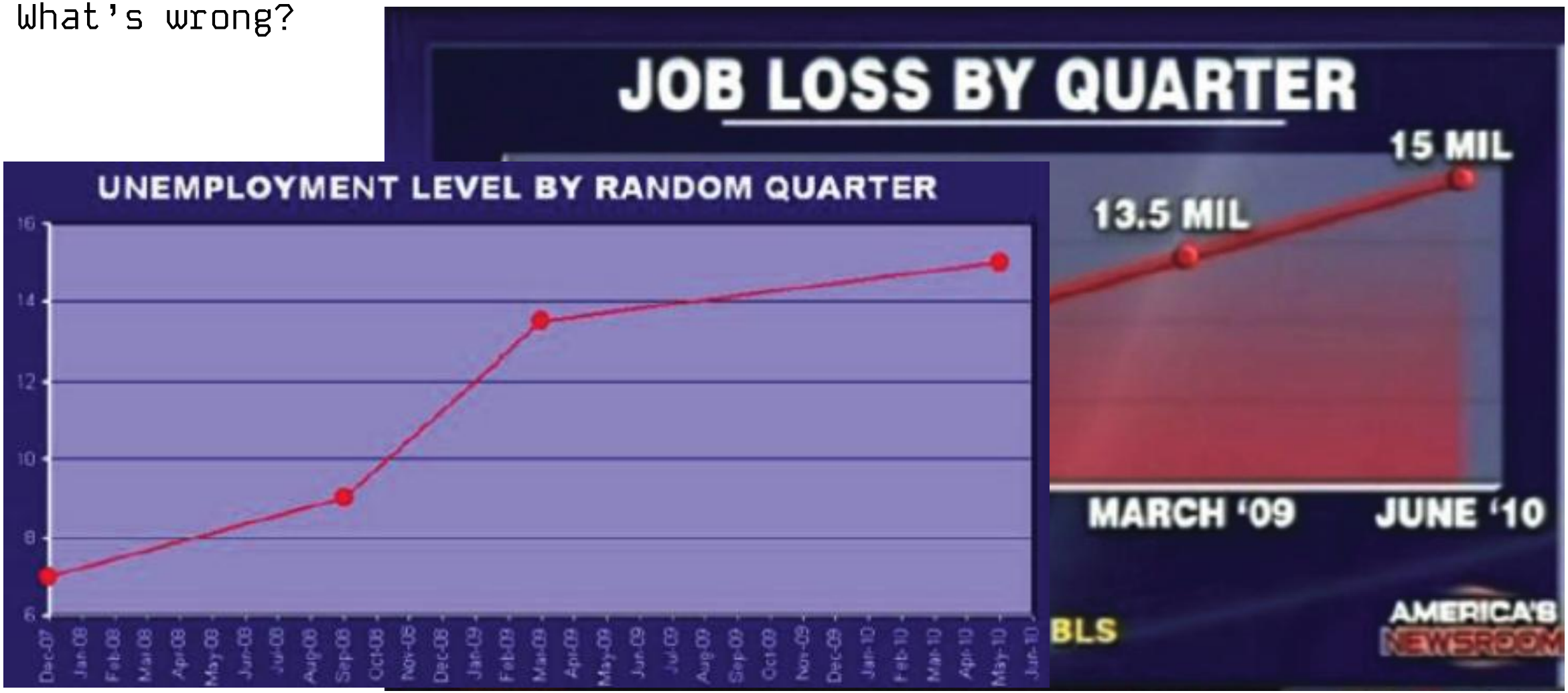
This?


(a) Feature importance of TSV
Mean |SHAP value|
Class
Hot
Neutral
Cool
Magenta
bStdDev
other(SEG)%
Sky(SEG)%
Blue
bMean
HotZone%
Building(SEG)%
gStdDev
rStdDev
gMean
rMean
Cyan
Tree(SEG)%
Red
Pavement(SEG)%
Green
OtherVegetation(SEG)%
Yellow
Orange
Building shade
Tree shade
Concrete(SF)
Shade provision
Tiles(SF)
Water feature
Brick(SF)
Rest spot/outdoor furniture
rubber
Grass(SF)
water(SEG)%
Asphalt(SF)
Wood(SF)
Gravel(SF)
(b) Beeswarm plot of TSV
Top 10 features
Cool
Neutral
Hot
Higher value of Magenta, the model is more strongly driven toward predicting "Cool"
Higher value of Magenta, suppresses the model's tendency to predict "Hot"
Shap values
High
Feature value
Low
Higher values tend to predict Hot
Higher values tend to predict Neutral or show no clear trend.
Higher values tend to predict Cool.

What's wrong?


JOB LOSS BY QUARTER
15 MIL
13.5 MIL
9 MIL
7 MIL
DEC '07
SEPT '08
MARCH '09
JUNE '10
FOX NEWS
FOX NEWS
.COM
SOURCE: BLS
AMERICA'S NEWSROOM

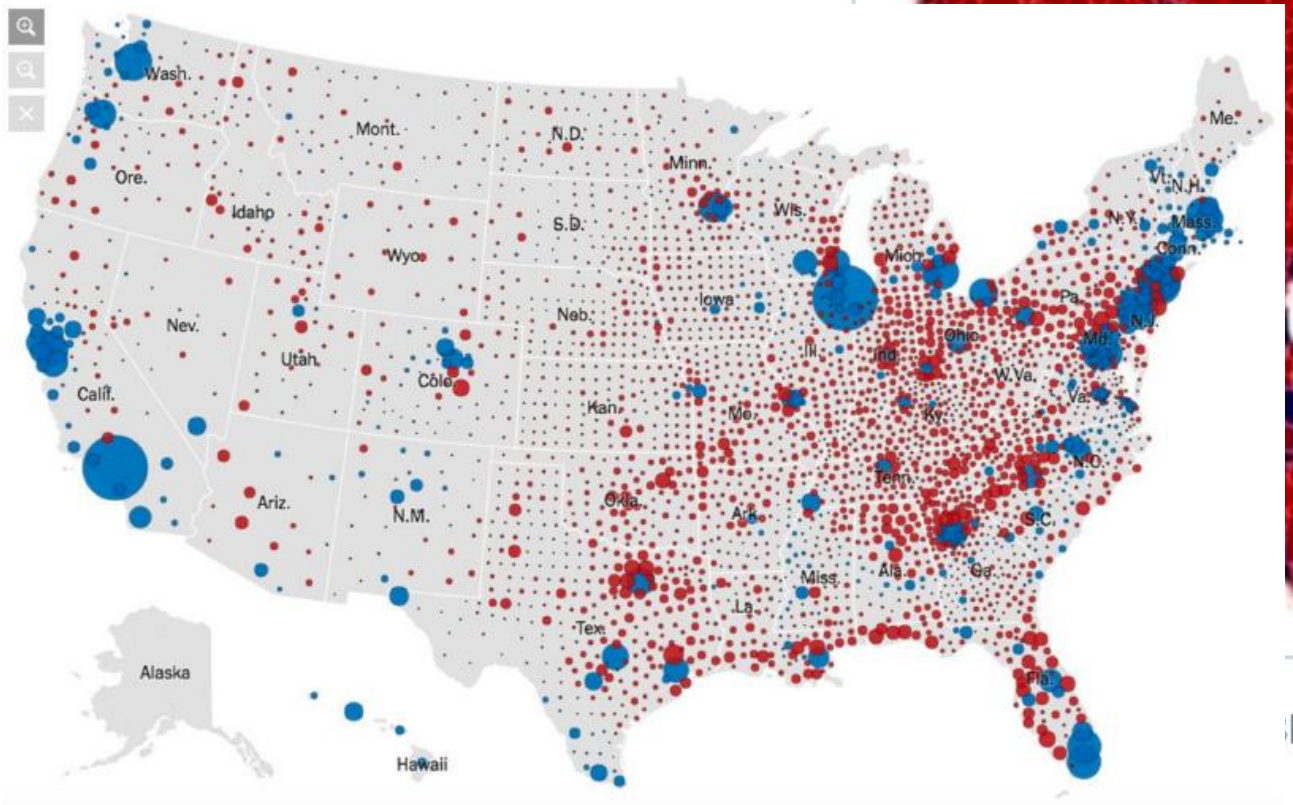
What's wrong?


JOB LOSS BY QUARTER
15 MIL
13.5 MIL
MARCH '09
JUNE '10
BLS
AMERICA'S
NEWSROOM
UNEMPLOYMENT LEVEL BY RANDOM QUARTER
16
14
12
10
8
6
Dec-07
Jan-08
Feb-08
Mar-08
Apr-08
May-08
Jun-08
Jul-08
Aug-08
Sep-08
Oct-08
Nov-08
Dec-08
Jan-09
Feb-09
Mar-09
Apr-09
May-09
Jun-09
Jul-09
Aug-09
Sep-09
Oct-09
Nov-09
Dec-09
Jan-10
Feb-10
Mar-10
Apr-10
May-10
Jun-10

What's wrong?

October 1, 2019

What's wrong?



October 1, 2019

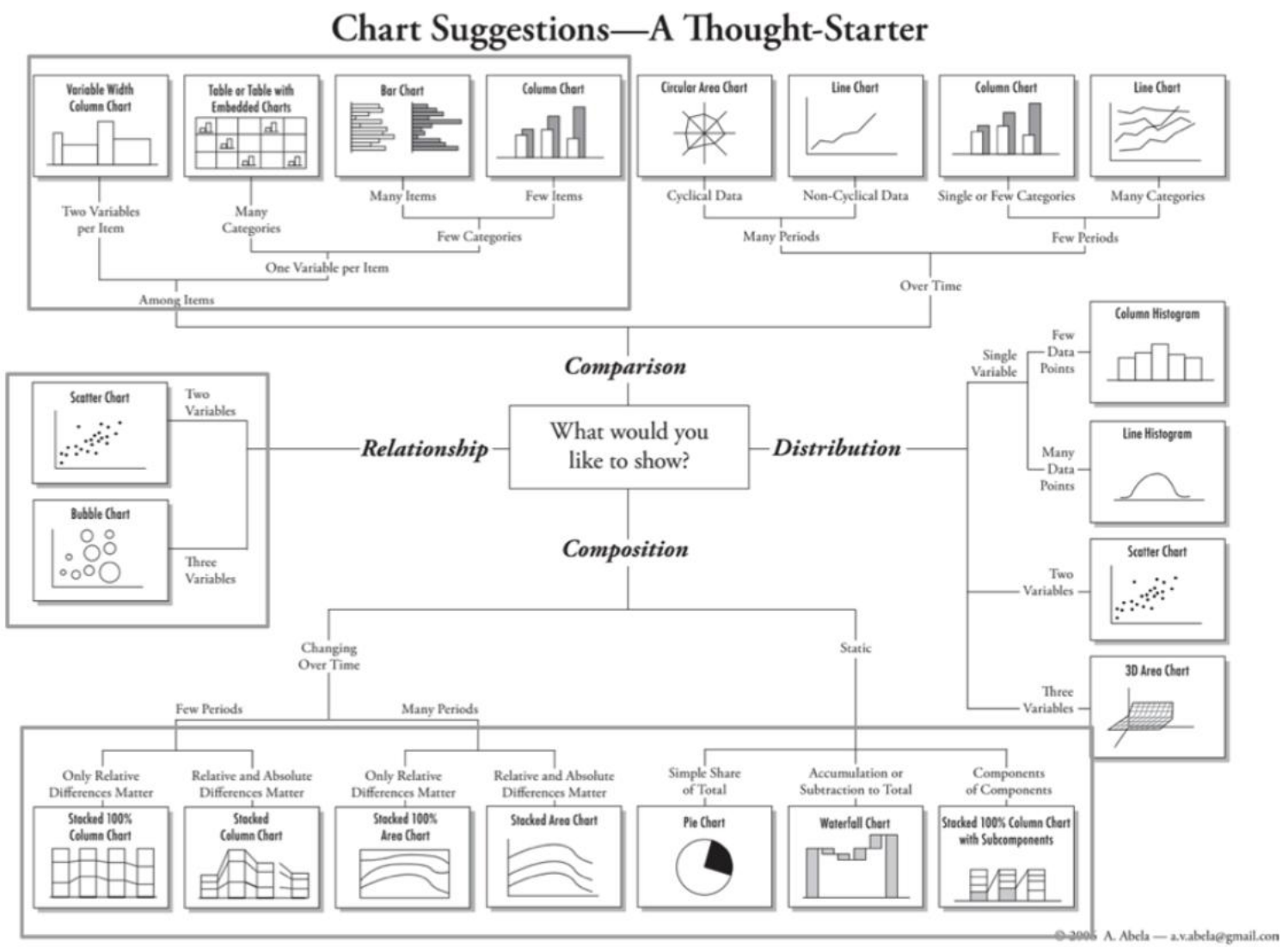
Making plots is effectively providing an answer to an implicit question

You get to pick the answer

Ensure the answer doesn't leave the viewer with uncertainty as to what it's answering or the completeness of the answer

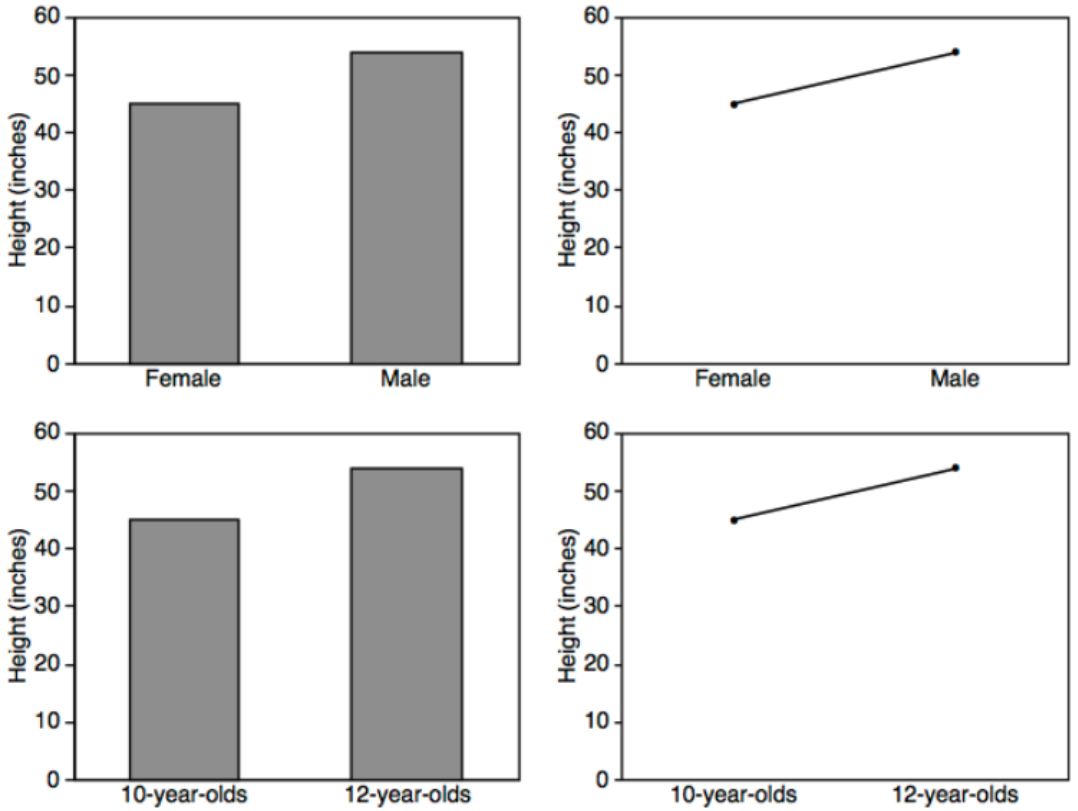
A good plot should invoke and inspire new questions

Visualizations (Scope)



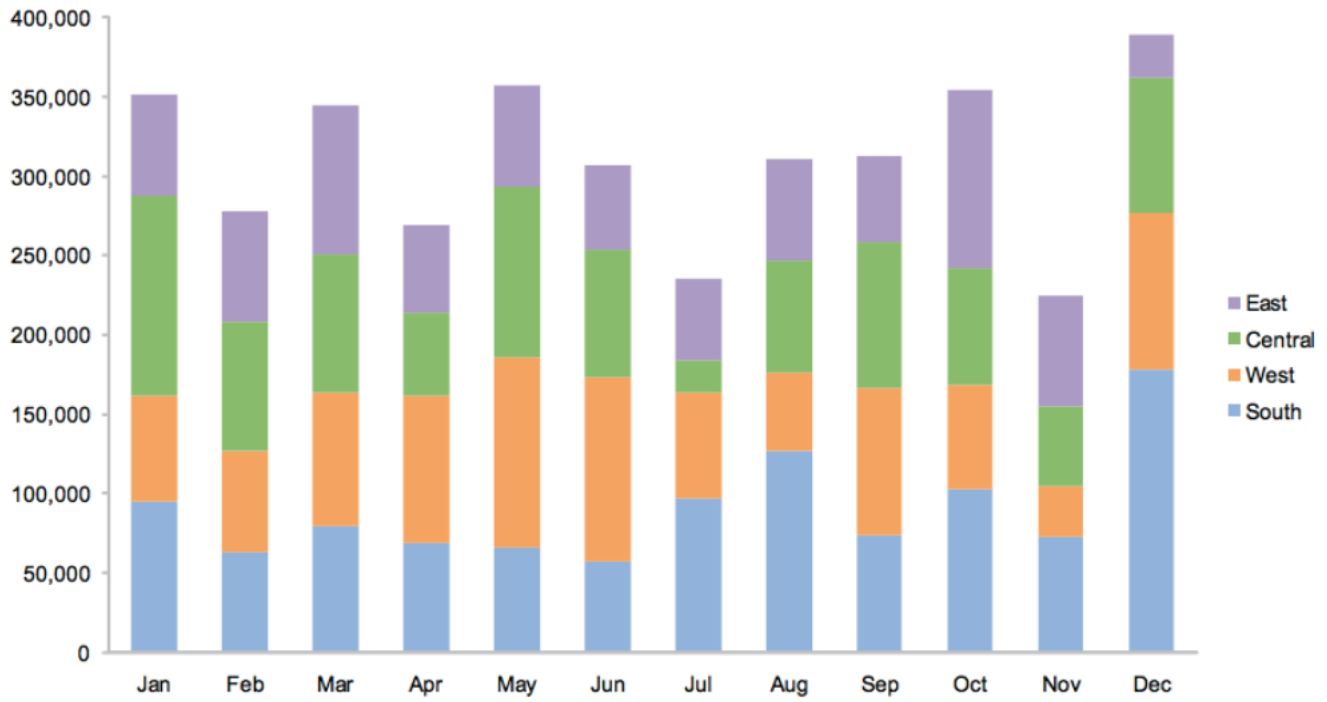
Comparisons

Bars vs. Lines

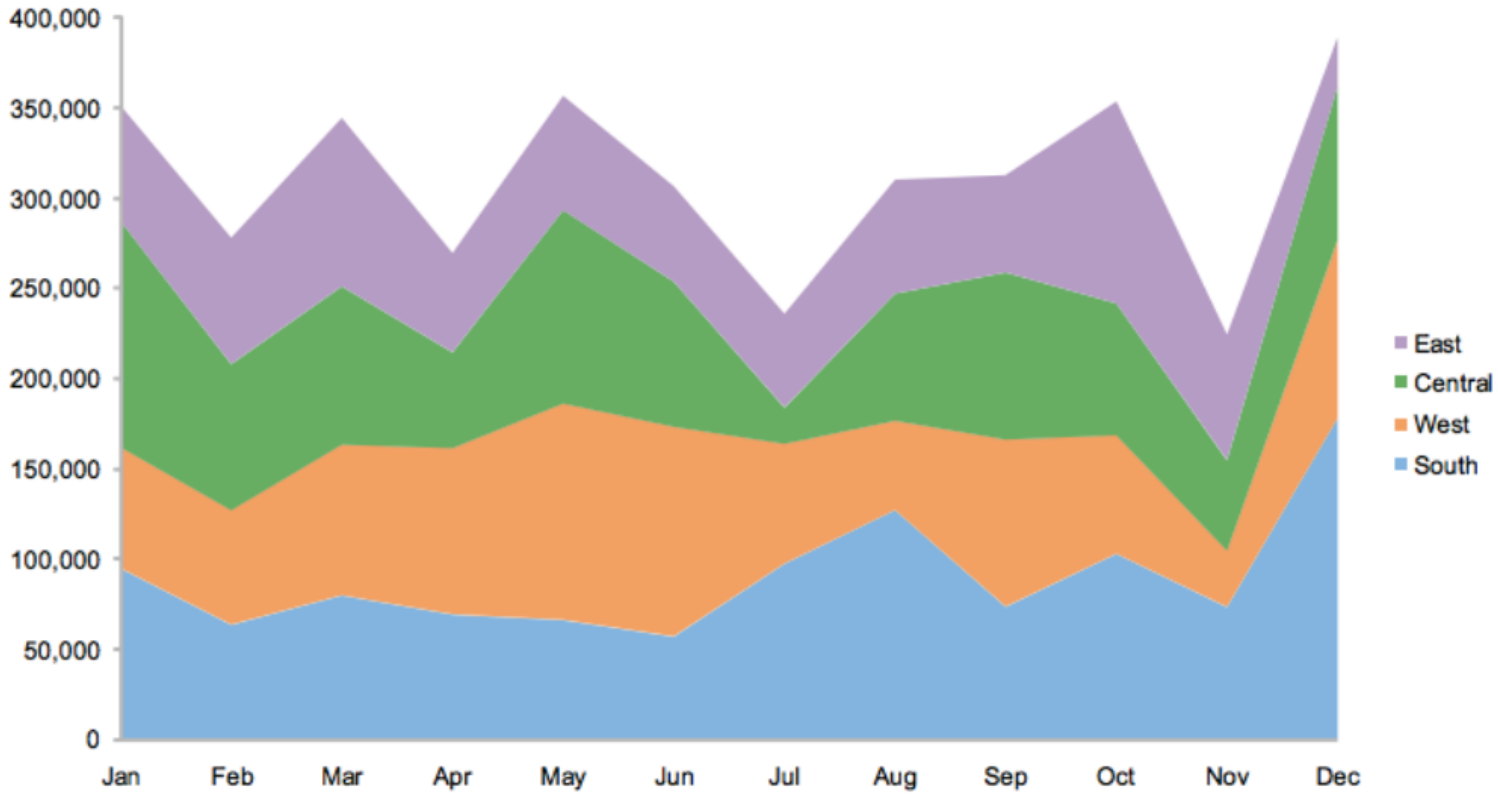


Proportions

Proportions



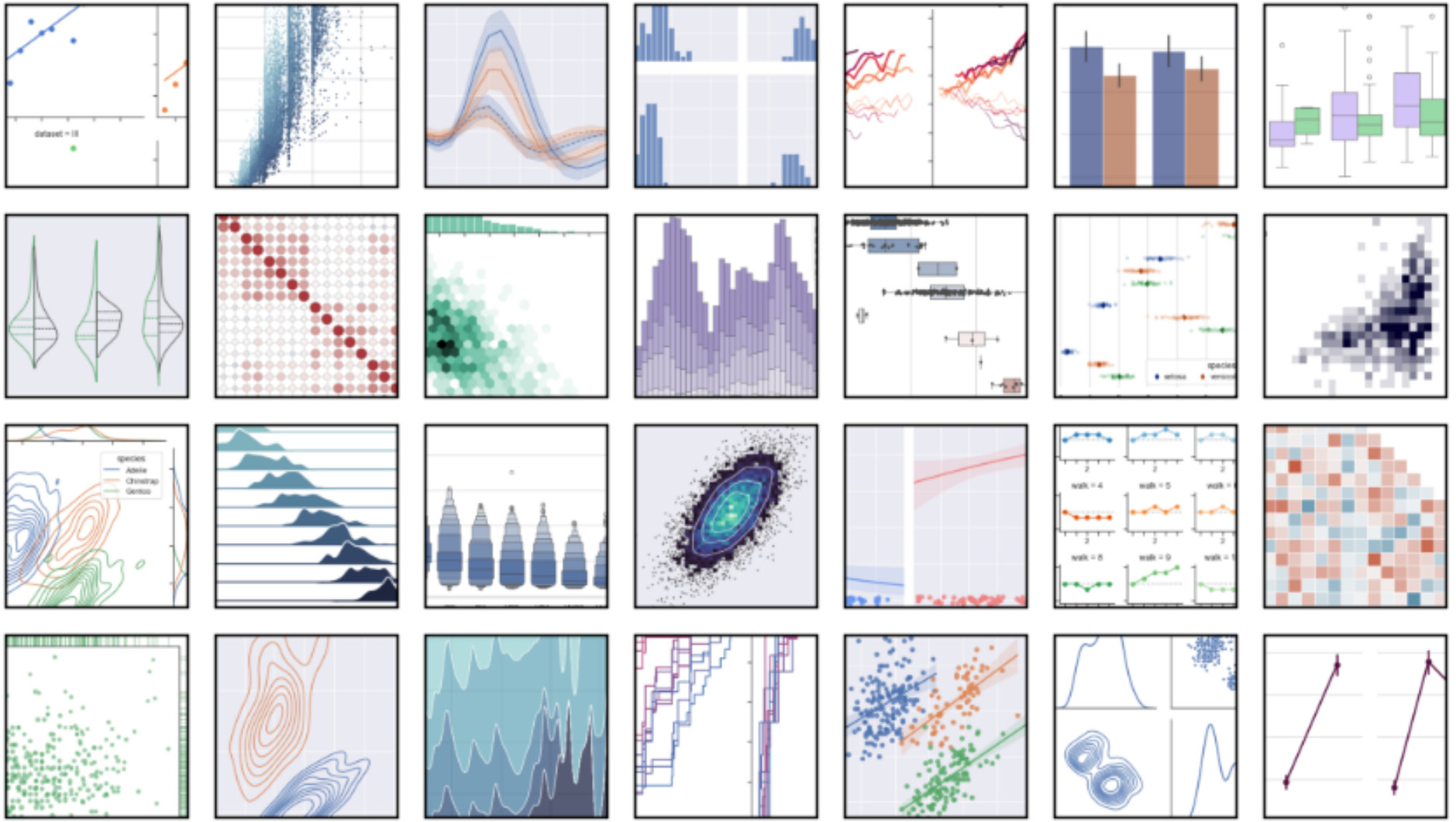
Proportions



Trends

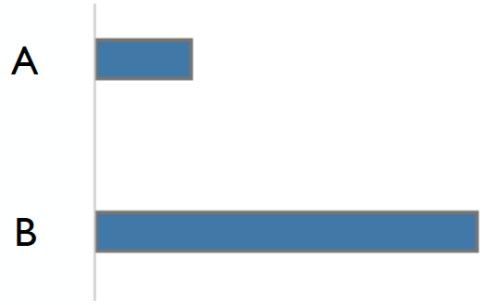

NVIDIA Corporation
NVDA · NASDAQ ·
+ Follow
Price Alert
$177.82 +$0.65 ↗ 0.37%
At close: Sep 12, 4:00:01 PM EDT
$173.42 -$4.40 ↘ 2.47%
After hours: Sep 15, 6:31:33 AM EDT
1D 5D 1M 6M YTD 1Y 5Y MAX
+$164.83 ↗ 1,269%
150
100
50
$13.94
Sep 1, 2022
2021 2022 2023 2024 2025
Prev Close $177.17
52W Range $86.62 - $184.48
Market Cap $4.33T
Open $177.77
P/E Ratio 50.81
Dividend Yield 0.0225%
Day Range $176.45 - $178.59
Volume 125M
EPS $3.50

Seaborn

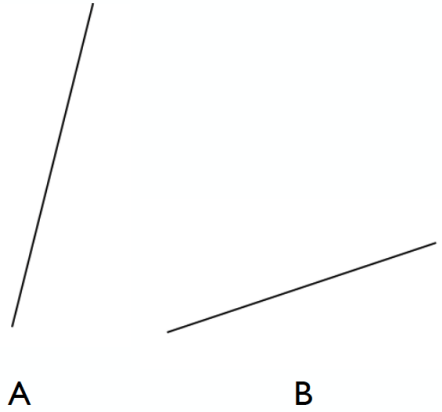


Perceptual Effectiveness

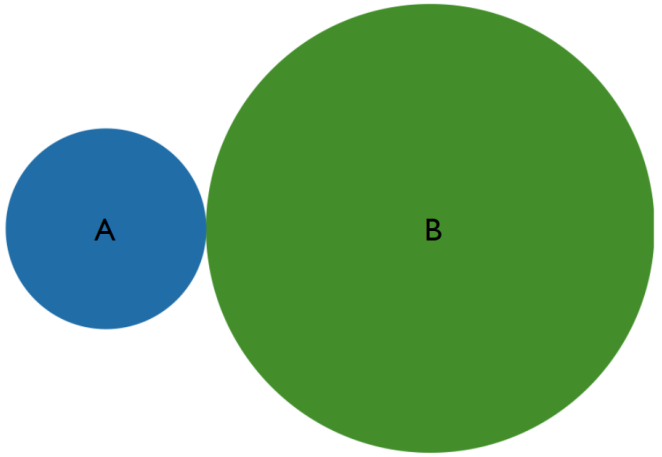
How much longer?



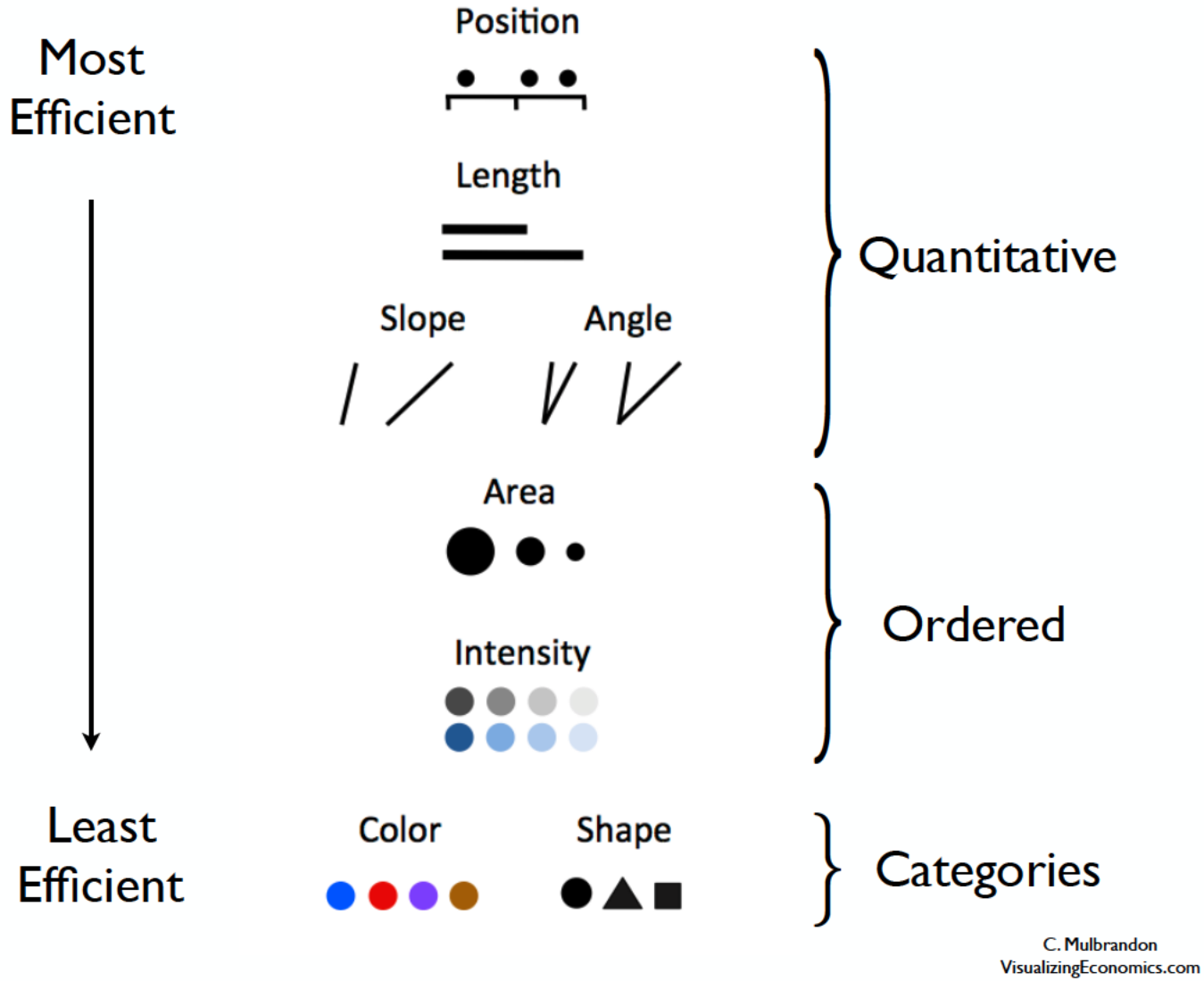
How much steeper slope?



How much larger area?

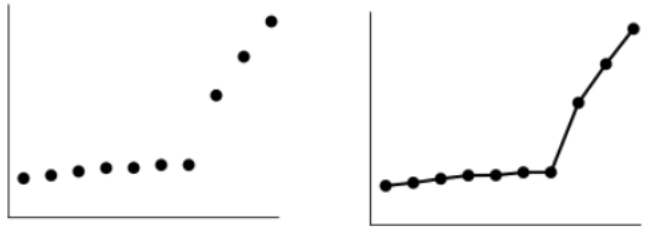


Perceptual Effectiveness

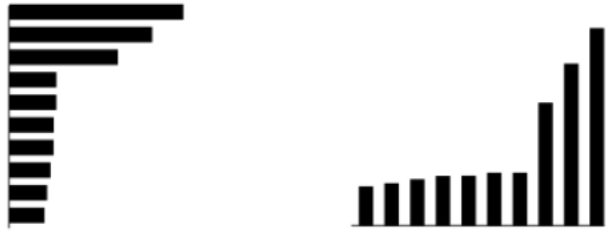


Perceptual Effectiveness

Most Effective



Position

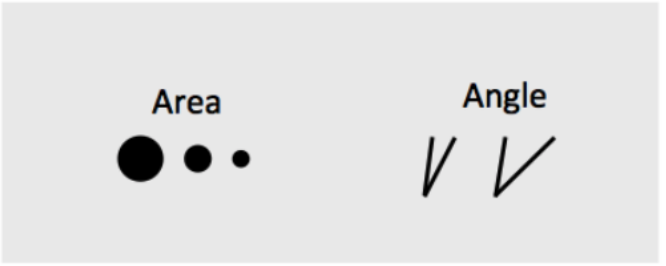


Length

VisualizingEconomics.com

Perceptual Effectiveness

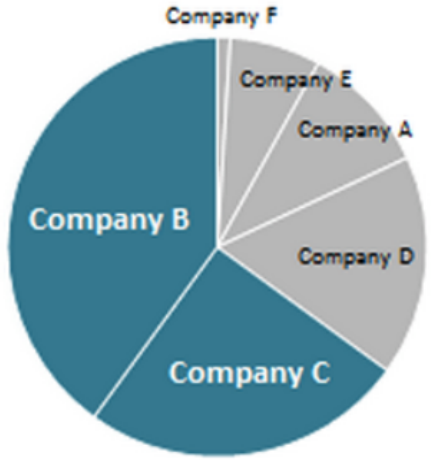
Less Effective

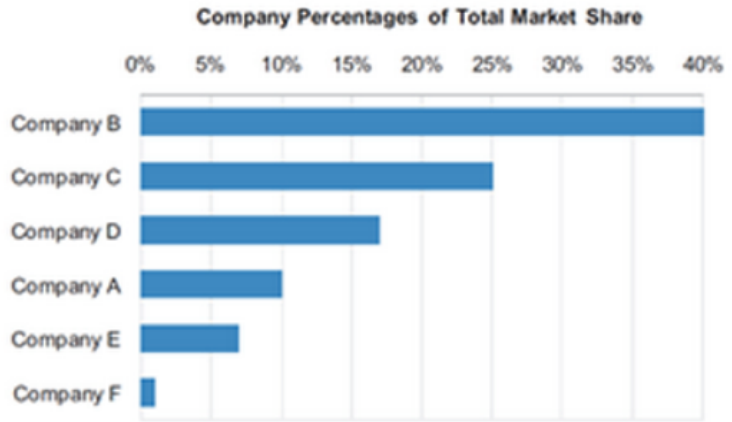


Perceptual Effectiveness

Pie vs. Bar Charts

65% of the market is controlled by companies B and C

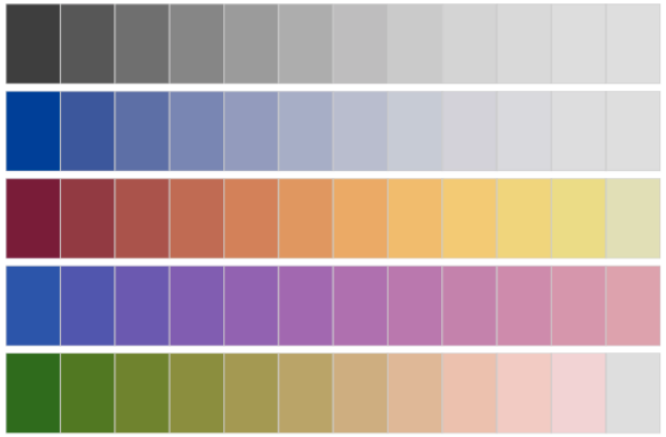




Colors

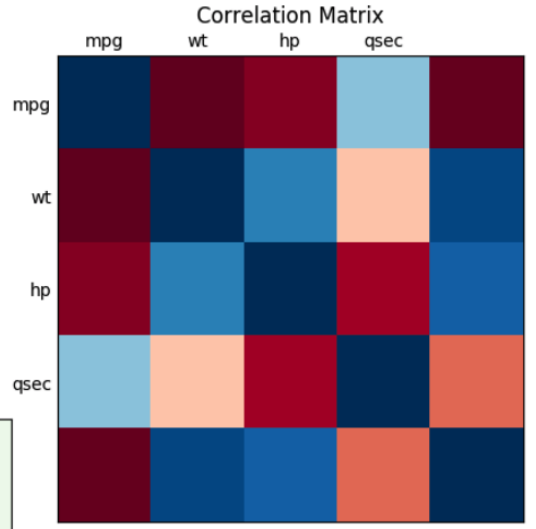
Colors for Ordinal Data

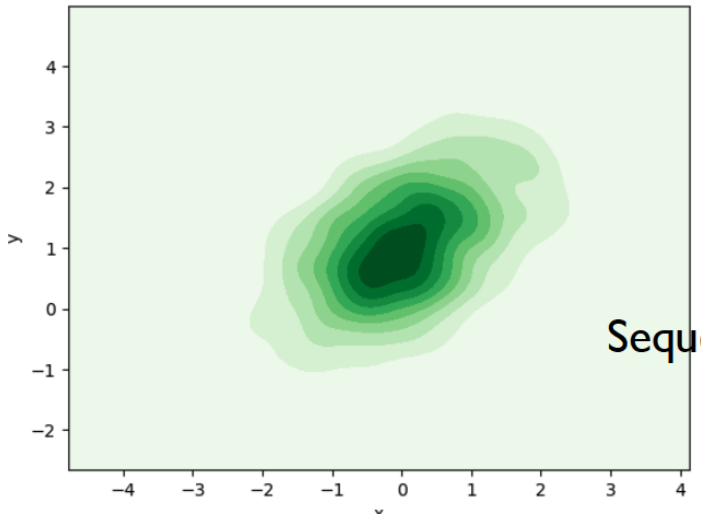
Vary luminance and saturation



Zeilis et al, 2009, "Escaping RGBland: Selecting Colors for Statistical Graphics"

Diverging Palette for Quantitative or Ordinal





Sequential Palette for Densities

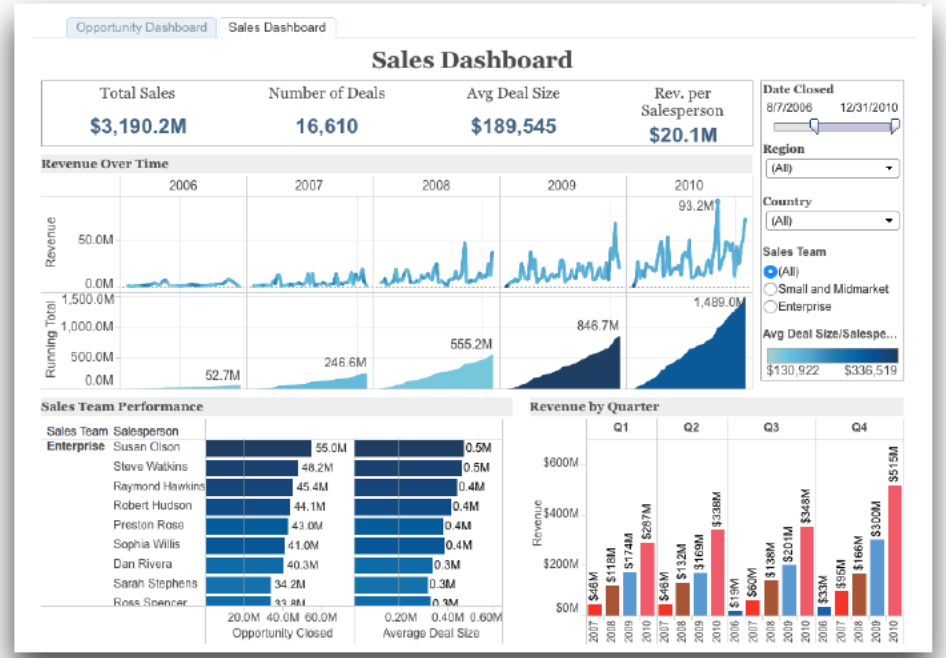
Colors

Colors for Categories

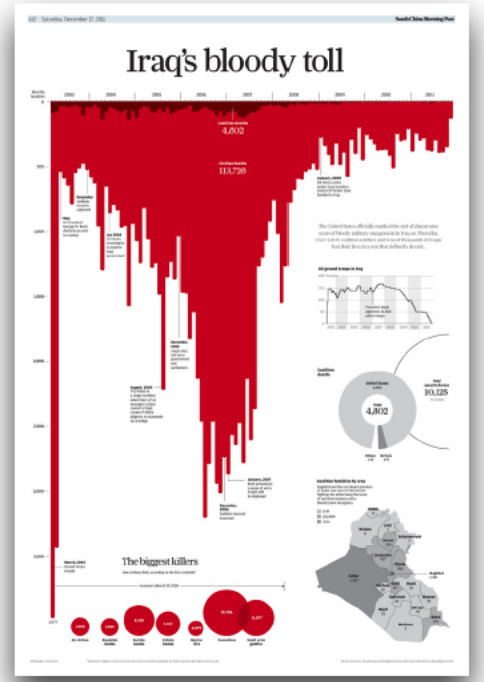
Do not use more than 5-8 colors at once

What is the message?

Exploratory
Neutral

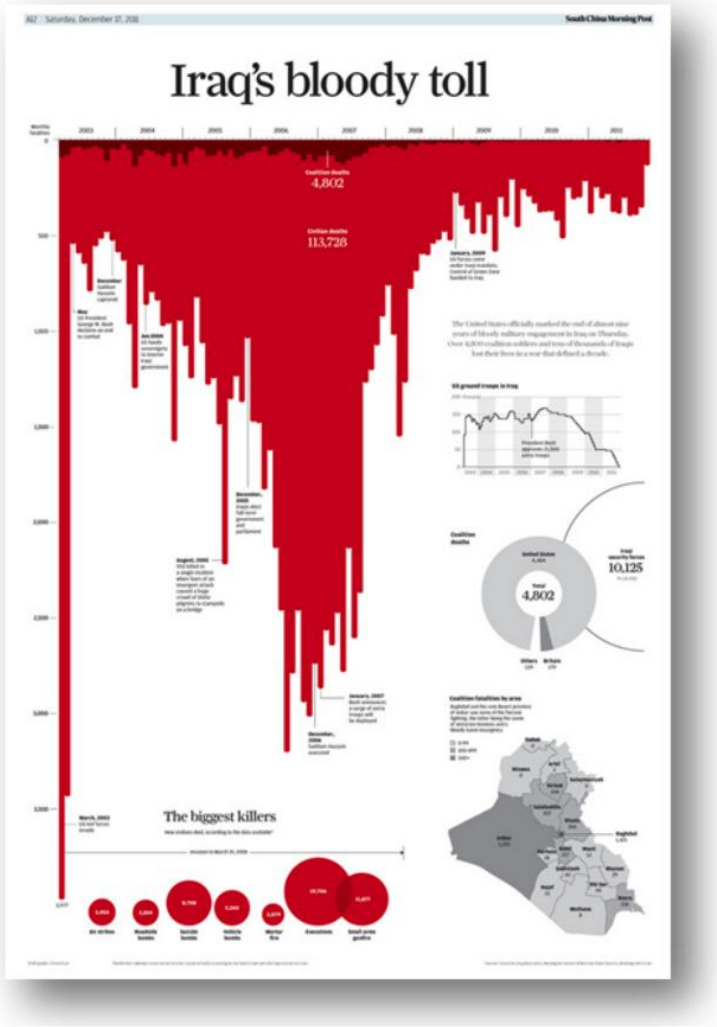


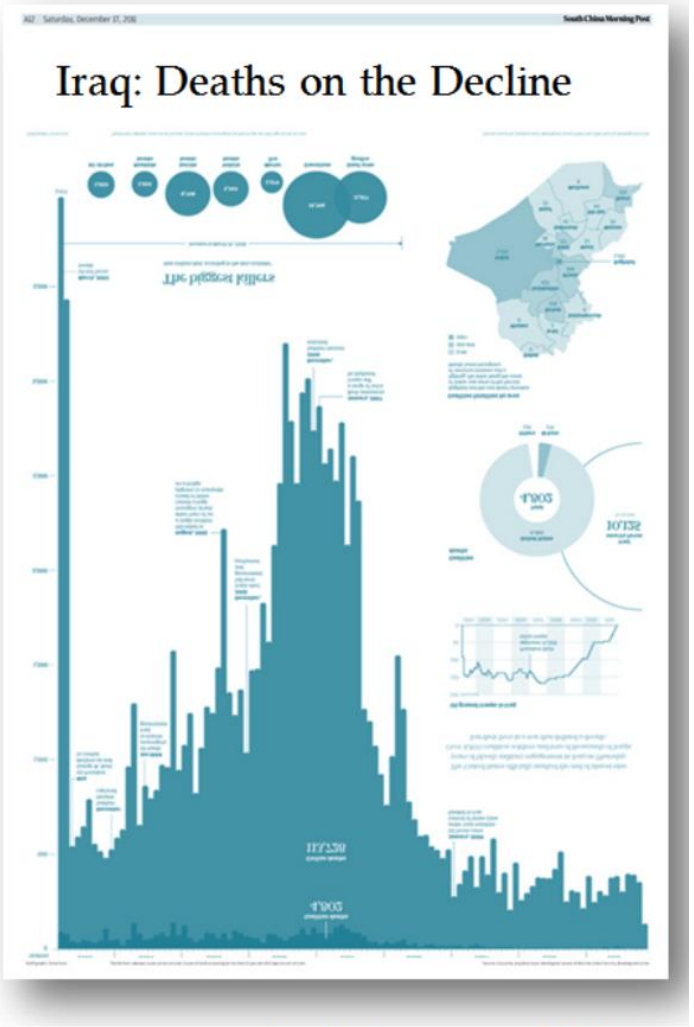
Explanatory
Opinionated



SUPERVISED 5 VISUALIZATION DESIGN SPACE

Visualizations (Communication)





Andy Cotgreave, Tableau

Takeaways

How you choose to display your data greatly influences how people interpret the data

Humans are visual, *emotional* creations; make graphs that don't make others feel confused, insulted, etc

Your graphs should illicit good feelings and effectively convey your narrative

Channels: Expressiveness Types and Effectiveness Ranks

Magnitude Channels: O or Q attributes

- Position on common scale
- Position on unaligned scale
- Length (1D size)
- Tilt/angle
- Area (2D size)
- Depth (3D position)
- Color luminance
- Color saturation
- Curvature
- Volume (3D size)

Most → Least Effectiveness

Same (Color luminance, Color saturation)

Same (Curvature, Volume (3D size))

Identity Channels : N attributes

- Spatial region
- Color hue
- Motion
- Shape

Tamara Munzner, *Visualization Analysis and Design*

L05.1
Supervised 5

Bagging

Random Forest

Boosting

XGBoost

L05.2
Visualizations

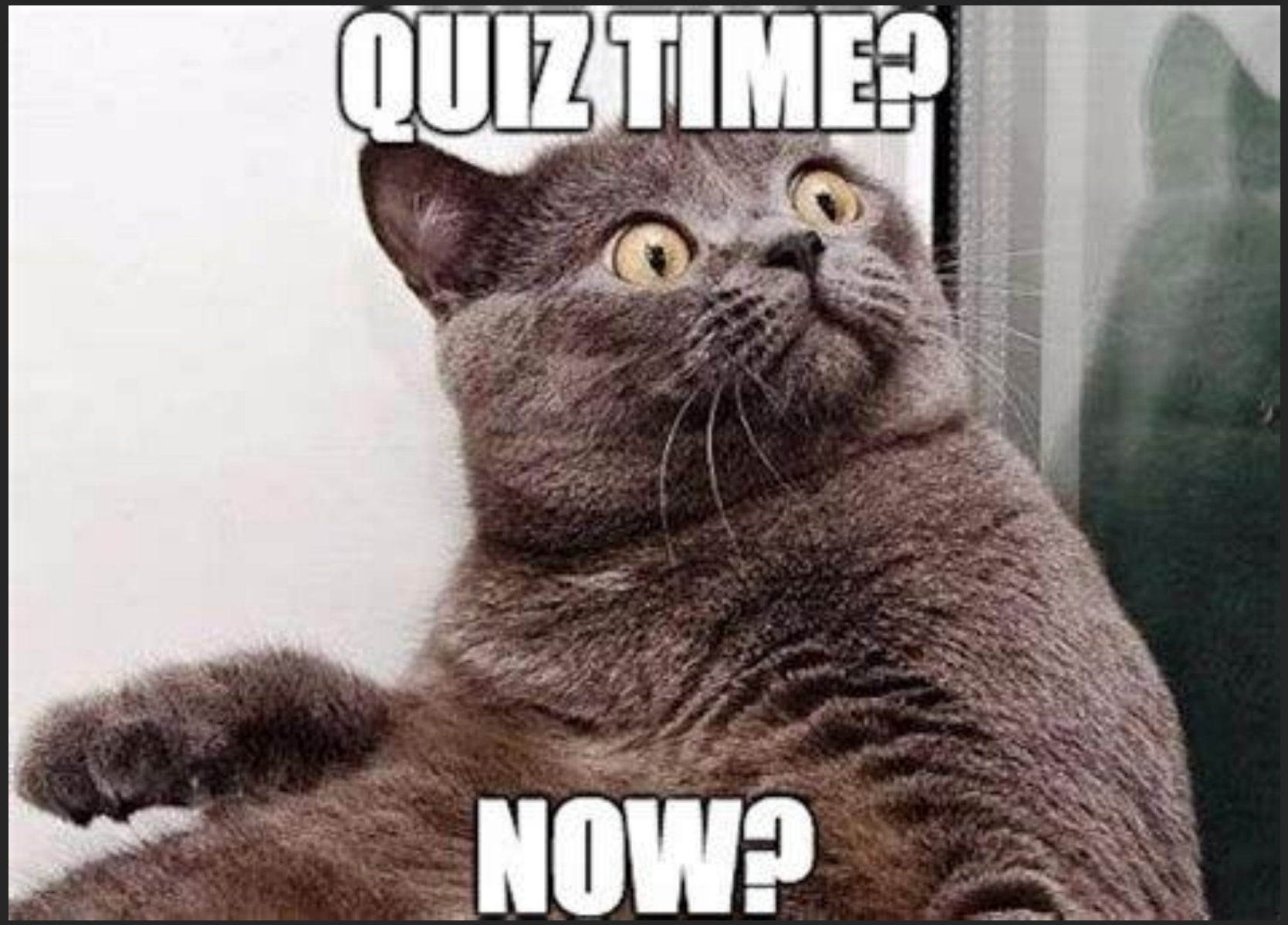
Exploratory Data Analysis

Communication

L05.3
Simulations

Rhino

ClimateStudio


QUIZ TIME?
NOW?