Ensembles

Making Trees ***Unstoppable***

Poul Petersen
CIO, BigML, Inc

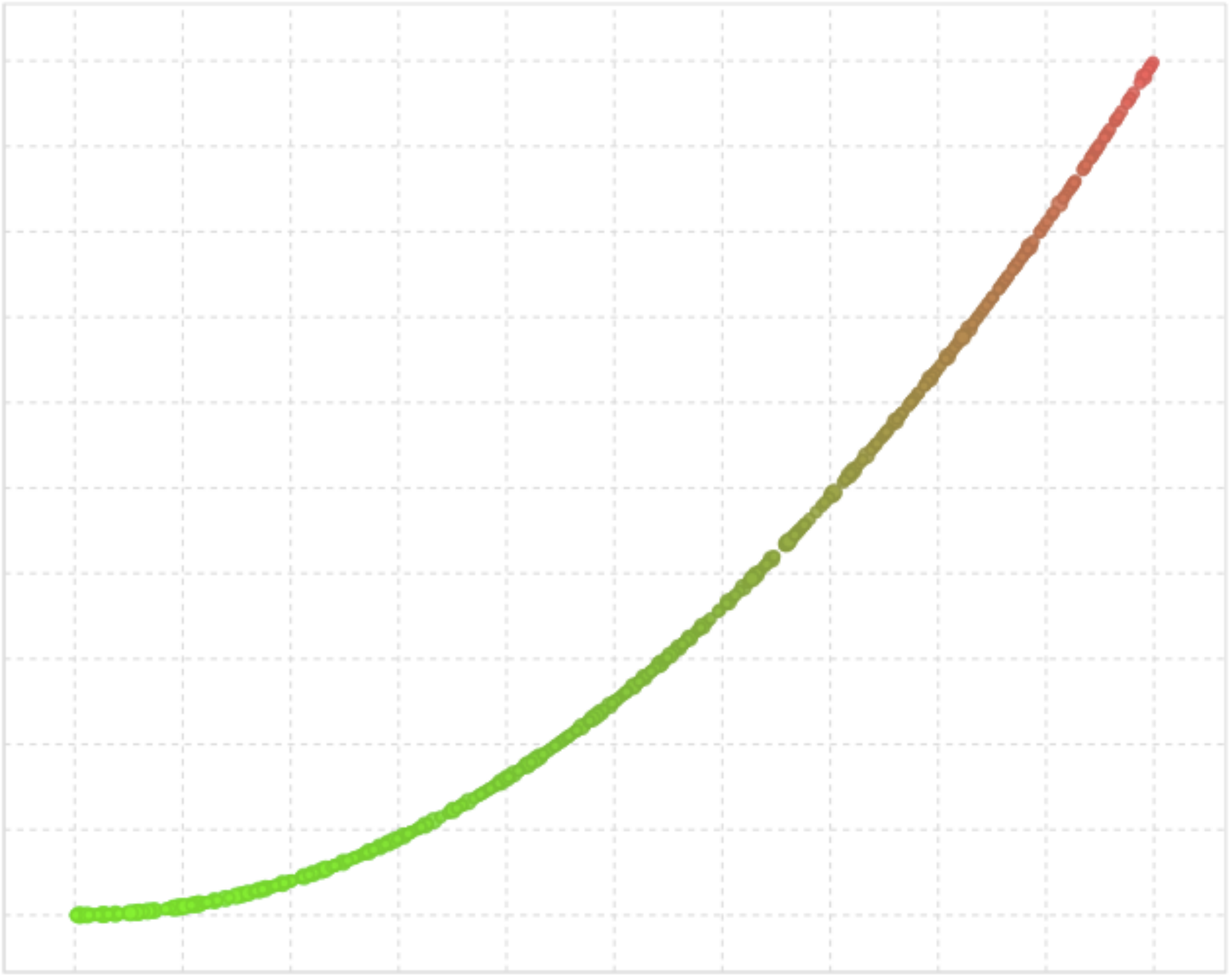
What is an Ensemble?

- Rather than build a single model…
- Combine the output of several typically “weaker” models into a powerful ensemble…
- Q1: Why is this necessary?
- Q2: How do we build “weaker” models?
- Q3: How do we “combine” models?

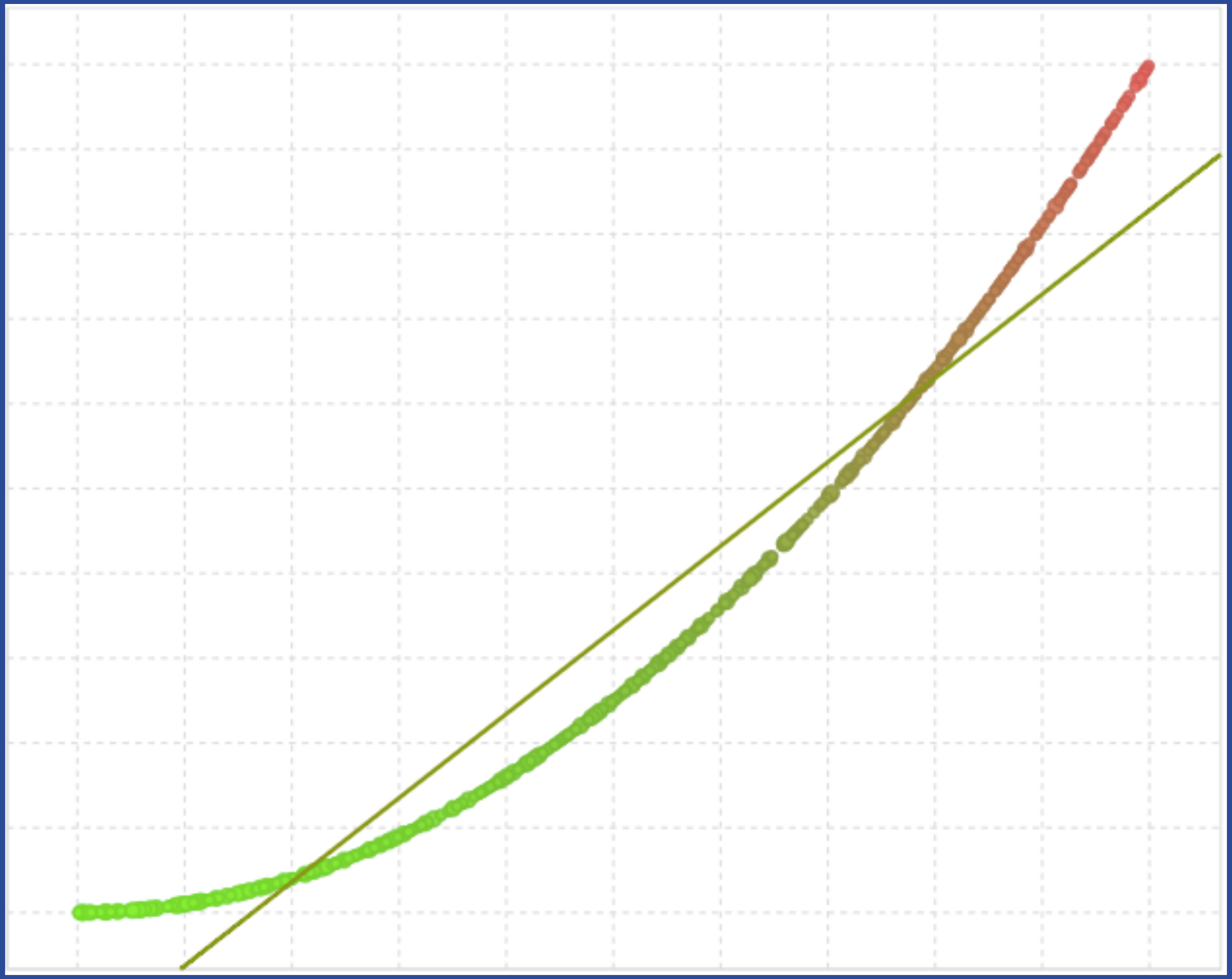
No Model is Perfect

- A given ML algorithm may simply ***not be able*** to exactly model the “real solution” of a particular dataset.
    - Try to fit a line to a curve

Simple Example - Fit a Line



Simple Example - Fit a Line



No Model is Perfect

- A given ML algorithm may simply ***not be able*** to exactly model the "real solution" of a particular dataset.
  - Try to fit a line to a curve
- Even if the model is very capable, the "real solution" may be elusive
  - DT/NN can model any decision boundary with enough training data, but the solution is NP-hard
  - Practical algorithms involve random processes and may arrive at different, yet equally good, "solutions" depending on the starting conditions, local optima, etc.
- If that wasn't bad enough…

No Data is Perfect

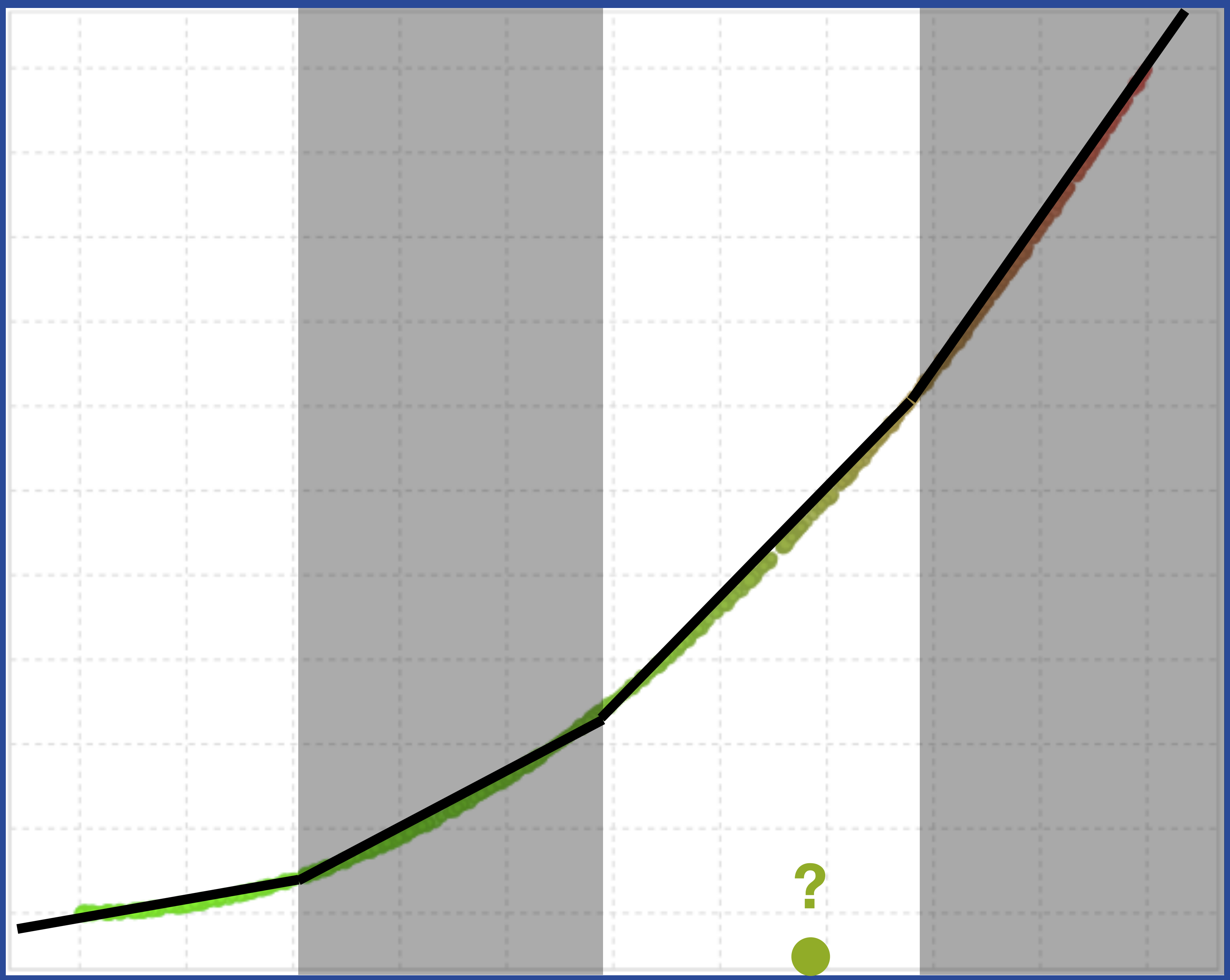
- Not enough data!
  - Always working with finite training data
  - Therefore, every “model” is an approximation of the “real solution” and there may be several good approximations.
- Anomalies / Outliers
  - The model is trying to generalize from discrete training data.
  - Outliers can “skew” the model, by overfitting
- Mistakes in your data
  - Does the model have to do everything for you?
  - But really, there is ***always*** mistakes in your data

Ensemble Techniques

- **Key Idea**:
  - By combining several good “models”, the combination may be closer to the best possible “model”
  - we want to ensure diversity. It’s not useful to use an ensemble of 100 models that are all the same
- Training Data Tricks
  - Build several models, each with only some of the data
  - Introduce randomness directly into the algorithm
  - Add training weights to “focus” the additional models on the mistakes made
- Prediction Tricks
  - Model the mistakes
  - Model the output of several different algorithms

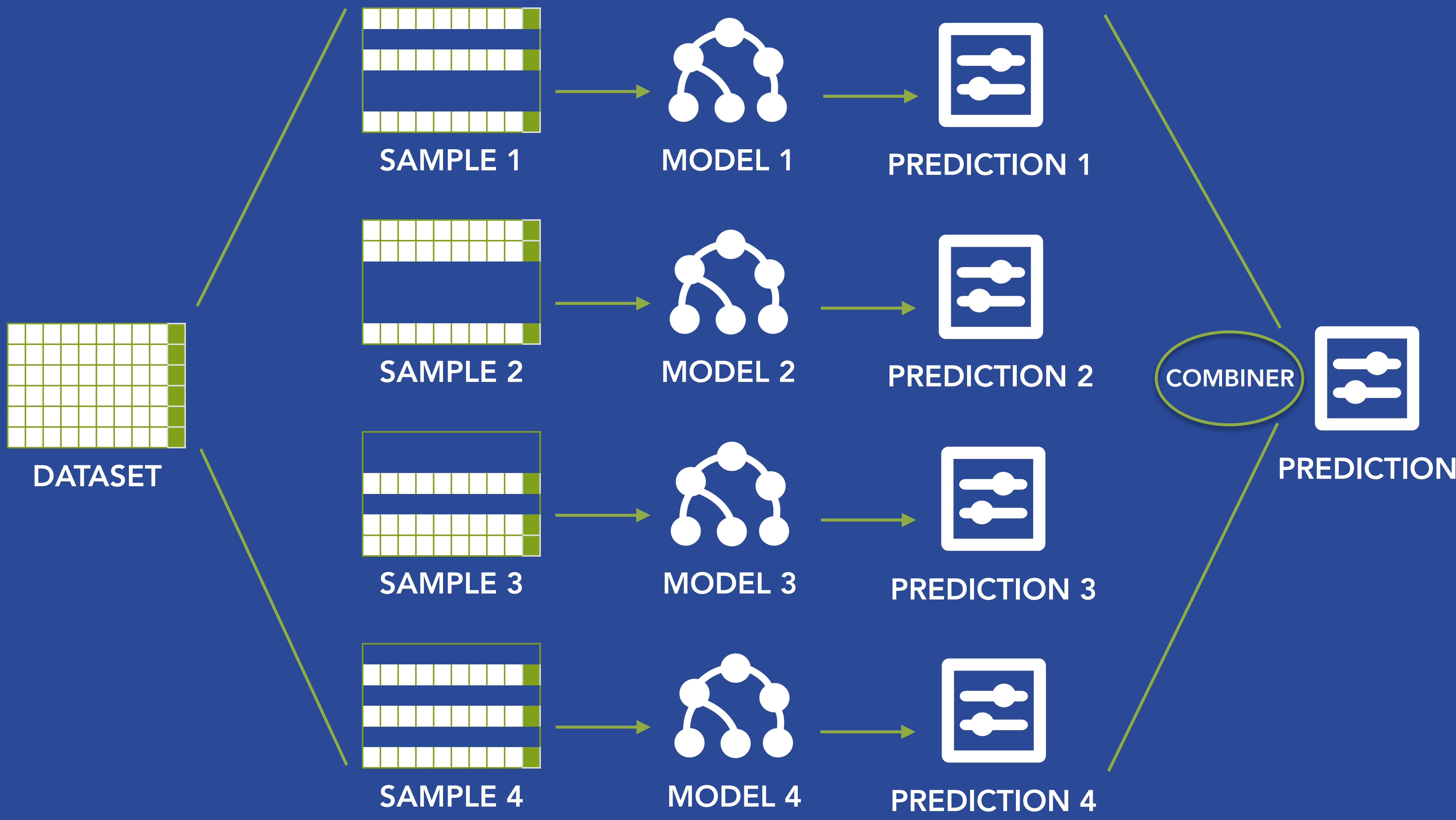
Simple Example - Fit a Line

Partition the data… then model each partition…



For predictions, use the model for the same partition

Decision Forest



Decision Forest Config

Things are getting harder to configure…

- Individual tree parameters are still available
  - Balanced objective, Missing splits, Node Depth, etc.
- **Number of models**: How many trees to build
- Sampling options:
  - **Deterministic** / **Random**
  - **Replacement**:
    - Allows sampling the same instance more than once
    - Effectively the same as ≈ 63.21%
    - “Full size” samples with zero covariance (good thing)
- At prediction time
  - Combiner…

Outlier Example

Diameter	Color	Shape	Fruit
4	red	round	plum
5	red	round	apple
5	red	round	apple
6	red	round	plum
7	red	round	apple

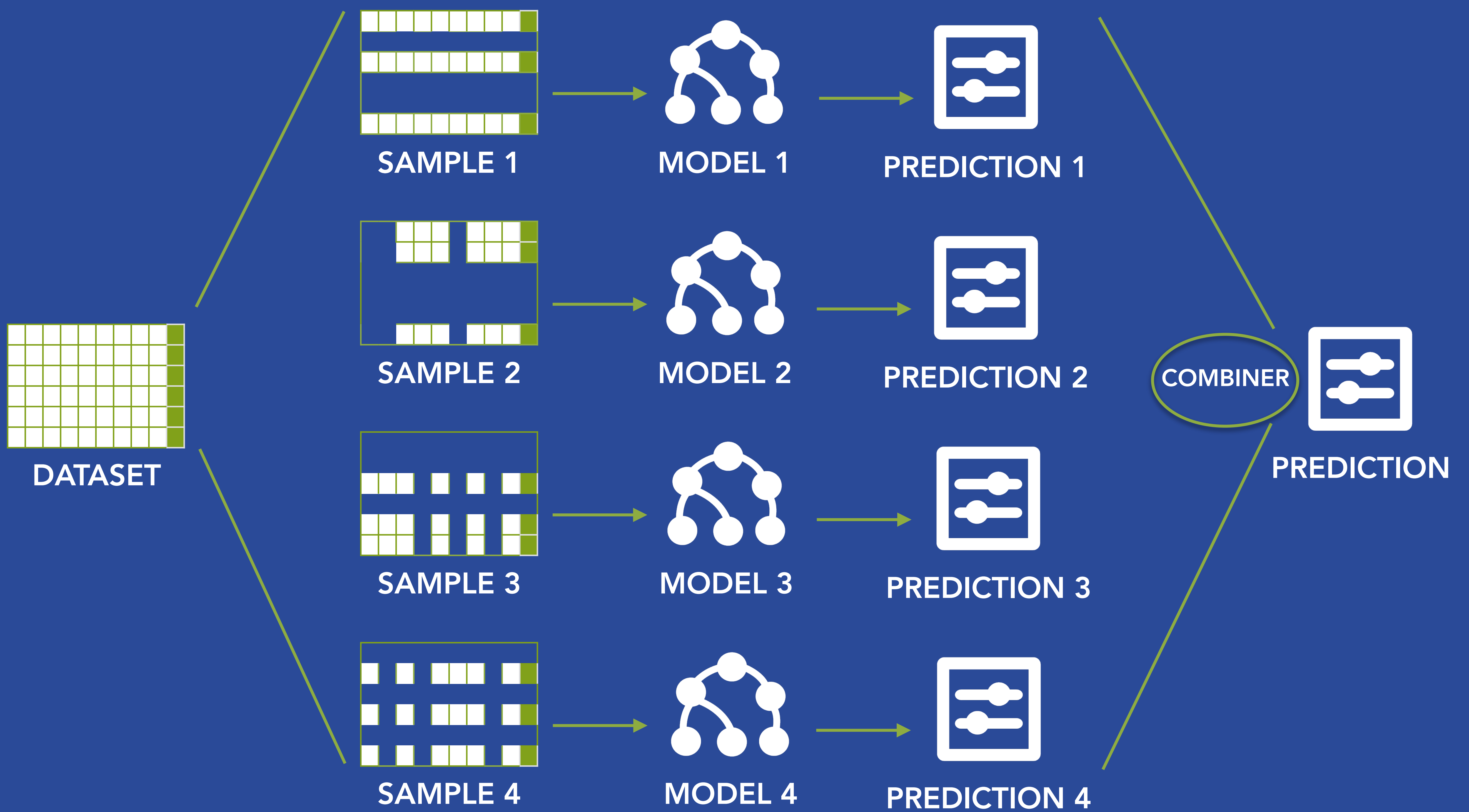
What is a round, red 6cm fruit?

All Data: “plum”

Sample 1: “plum”
Sample 2: “apple”
Sample 3: “apple”

“apple”

Random Decision Forest



RDF Config

Things are getting harder to configure…

- Individual tree parameters are still there…
  - Balanced objective, Missing splits, Node Depth, etc.
- Decision Forest parameters still available
  - Number of model, Sampling, etc
- Random candidates:
  - The number of features to consider at each split

Boosting

ADDRESS	BEDS	BATHS	SQFT	LOT SIZE	YEAR BUILT	LATITUDE	LONGITUDE	LAST SALE PRICE
1522 NW Jonquil	4	3	2424	5227	1991	44.594828	-123.269328	360000
7360 NW Valley Vw	3	2	1785	25700	1979	44.643876	-123.238189	307500
4748 NW Veronica	5	3.5	4135	6098	2004	44.5929659	-123.306916	600000
411 NW 16th		3	2825	4792	1938	44.570883	-123.272113	435350

MODEL 1

PREDICTED SALE PRICE
360750
306875
587500
435350

ERROR
750
-625
-12500
0

ADDRESS	BEDS	BATHS	SQFT	LOT SIZE	YEAR BUILT	LATITUDE	LONGITUDE	ERROR
1522 NW Jonquil	4	3	2424	5227	1991	44.594828	-123.269328	750
7360 NW Valley Vw	3	2	1785	25700	1979	44.643876	-123.238189	625
4748 NW Veronica	5	3.5	4135	6098	2004	44.5929659	-123.306916	12500
411 NW 16th		3	2825	4792	1938	44.570883	-123.272113	0

MODEL 2

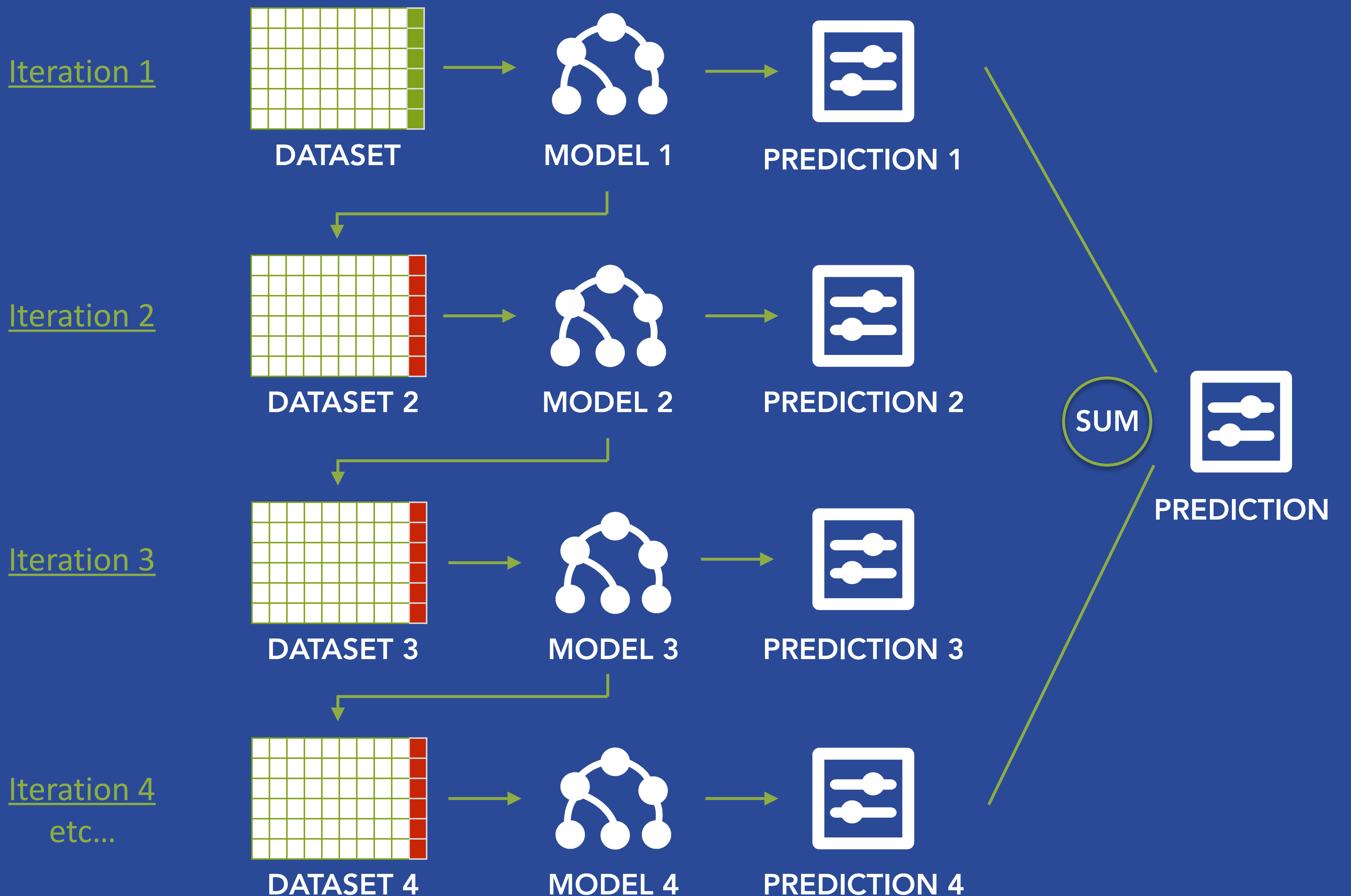
PREDICTED ERROR
750
625
12393.83333
6879.67857

"Hey Model 1, what do you predict is the sale price of this home?"

"Hey Model 2, how much error do you predict Model 1 just made?"

Why stop at one iteration?

Boosting



Iteration 1
DATASET
MODEL 1
PREDICTION 1
Iteration 2
DATASET 2
MODEL 2
PREDICTION 2
SUM
PREDICTION
Iteration 3
DATASET 3
MODEL 3
PREDICTION 3
Iteration 4
etc…
DATASET 4
MODEL 4
PREDICTION 4

Wait a Second…

… what about classification?

pregnancies	plasma glucose	blood pressure	triceps skin thickness	insulin	bmi	diabetes pedigree	age	diabetes
6	148	72	35	0	33.6	0.627	50	TRUE
1	85	66	29	0	26.6	0.351	31	FALSE
8	183	64	0	0	23.3	0.672	32	TRUE
1	89	66	23	94	28.1	0.167	21	FALSE

→ MODEL 1 →

predicted diabetes
TRUE
TRUE
FALSE
FALSE

→

ERROR
?
?
?
?

… we could try

pregnancies	plasma glucose	blood pressure	triceps skin thickness	insulin	bmi	diabetes pedigree	age	diabetes
6	148	72	35	0	33.6	0.627	50	1
1	85	66	29	0	26.6	0.351	31	0
8	183	64	0	0	23.3	0.672	32	1
1	89	66	23	94	28.1	0.167	21	0

→ MODEL 1 →

predicted diabetes
1
1
0
0

→

ERROR
0
-1
1
0

Wait a Second…

… but then what about multiple classes?

pregnancies	plasma glucose	blood pressure	triceps skin thickness	insulin	bmi	diabetes pedigree	age	favorite color
6	148	72	35	0	33.6	0.627	50	RED
1	85	66	29	0	26.6	0.351	31	GREEN
8	183	64	0	0	23.3	0.672	32	BLUE
1	89	66	23	94	28.1	0.167	21	RED

→ **MODEL 1** →

predicted favorite color
BLUE
GREEN
RED
GREEN

→

ERROR
?
?
?
?

Boosting Classification

Iteration 1

pregnancies	plasma glucose	blood pressure	triceps skin thickness	insulin	bmi	diabetes pedigree	age	favorite color
6	148	72	35	0	33.6	0.627	50	RED
1	85	66	29	0	26.6	0.351	31	GREEN
8	183	64	0	0	23.3	0.672	32	BLUE
1	89	66	23	94	28.1	0.167	21	RED

MODEL 1
RED/NOT RED

Class RED Probability
0.9
0.7
0.46
0.12

Class RED ERROR
0.1
-0.7
0.54
-0.12

MODEL 1
BLUE/NOT BLUE

...and repeat for each class at each iteration

Class BLUE Probability
0.1
0.3
0.54
0.88

Class BLUE ERROR
-0.1
0.7
-0.54
0.12

Iteration 2

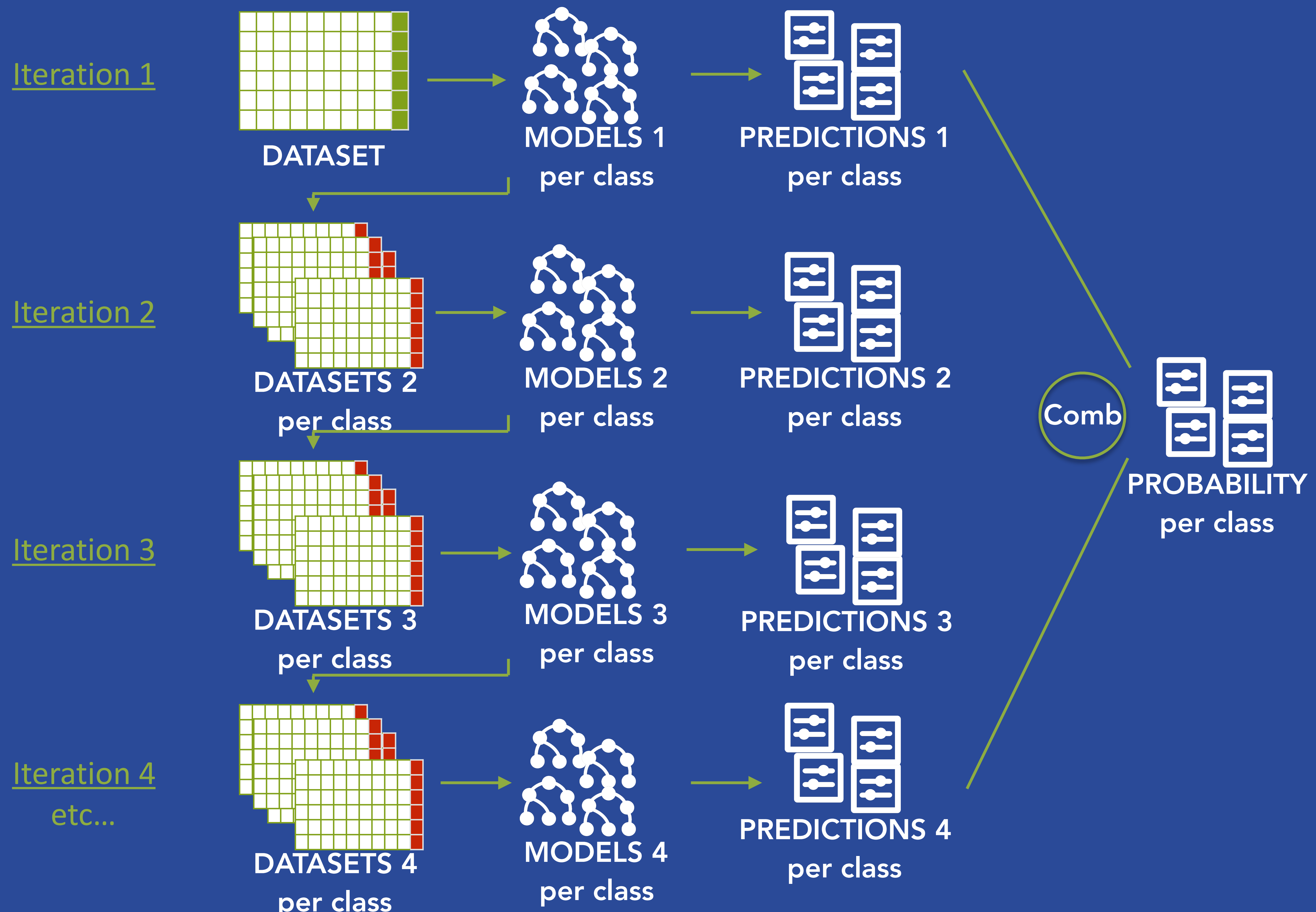
pregnancies	plasma glucose	blood pressure	triceps skin thickness	insulin	bmi	diabetes pedigree	age	ERROR
6	148	72	35	0	33.6	0.627	50	0.1
1	85	66	29	0	26.6	0.351	31	-0.7
8	183	64	0	0	23.3	0.672	32	0.54
1	89	66	23	94	28.1	0.167	21	-0.12

MODEL 2
RED/NOT RED ERR

...and repeat for each class at each iteration

PREDICTED ERROR
0.05
-0.54
0.32
-0.22

Boosting Classification



Boosting Config

Things are getting harder to configure…

- Number of iterations - similar to number of models for DF/RDF
- Iterations can be limited with Early Stopping:
    - Early out of bag: tests with the out-of-bag samples
    - Early holdout: tests with a portion of the dataset
    - None: performs all iterations. Note: In general, it is better to use a high number of iterations and let the early stopping work.
- Learning Rate: Controls how aggressively boosting will fit the data:
    - Larger values ~ maybe quicker fit, but risk of overfitting
- You can combine sampling with Boosting!
    - Samples with Replacement
    - Add Randomize
- Individual tree parameters are still available
    - Balanced objective, Missing splits, Node Depth, etc.

Ensembles Demo

Which Ensemble Method

- For "large" / "complex" datasets
  - Use DF/RDF with deeper node threshold
  - Even better, use Boosting with more iterations
- For "noisy" data
  - Boosting may overfit
  - RDF preferred
- For "wide" data
  - Randomize features (RDF) will be quicker
- For "easy" data
  - A single model may be fine
  - Bonus: also has the best interpretability!
- For classification with "large" number of classes
  - Boosting will be slower
- For "general" data
  - DF/RDF likely better than a single model or Boosting.
  - Boosting will be slower since the models are processed serially
- **Real Answer: Use the one that works best!**
  - Ok, but seriously. Did you evaluate?

Your Turn!

- Pick ONE of the three Ensemble methods
- Create an ensemble for the Diabetes 80% Training dataset
- Evaluate the Ensemble with the 20% Test set.
- Compare the Evaluation with the Model you made earlier
- Which performs *better*?