Kaggle Competitions 101

The Airbnb of Data Scientists

AGENDA

Anatomy of a Kaggle Competition [10-15 mins]

- Competition Mechanics
- Register, download data, and access kaggle notebooks

Let’s Ensemble [60 mins]

- Algorithms, tools, and frameworks in perspective

Competition- House Prices Advanced Regression [30 mins]

GOALS

Get familiar with the mechanics of Data Science Competitions

Get comfortable with the Kaggle Kernel environment

Familiarise with algorithms behind most Kaggle winning solutions

At the end of this tutorial:

- Submitted at least 2 entries to a Kaggle competition
- Applied Algorithms on Kaggle Data
  - Gradient Boosting Machines, Xgboost, Model Averaging and Model Stacking

TUTORIAL MATERIAL

go.ncsu.edu/kaggle

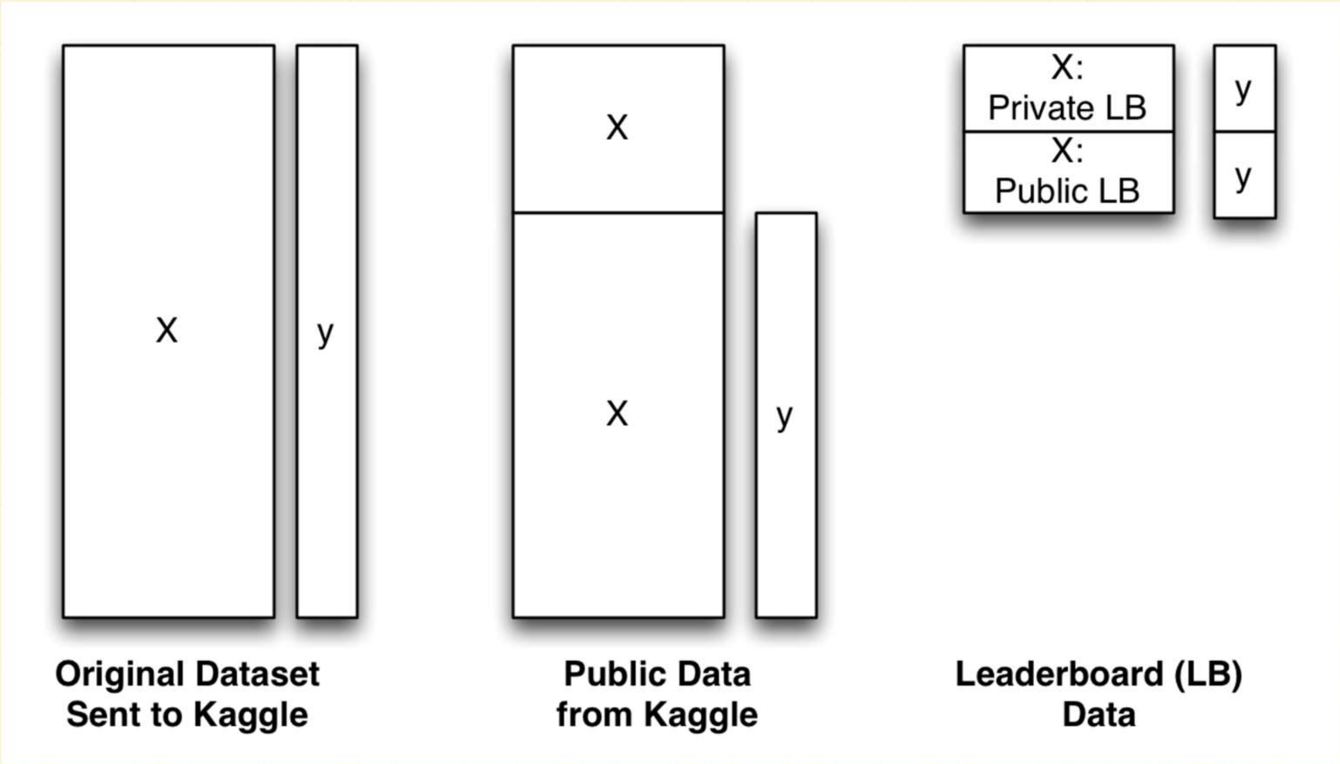
(slides+notebooks)

ANATOMY OF KAGGLE COMPETITIONS

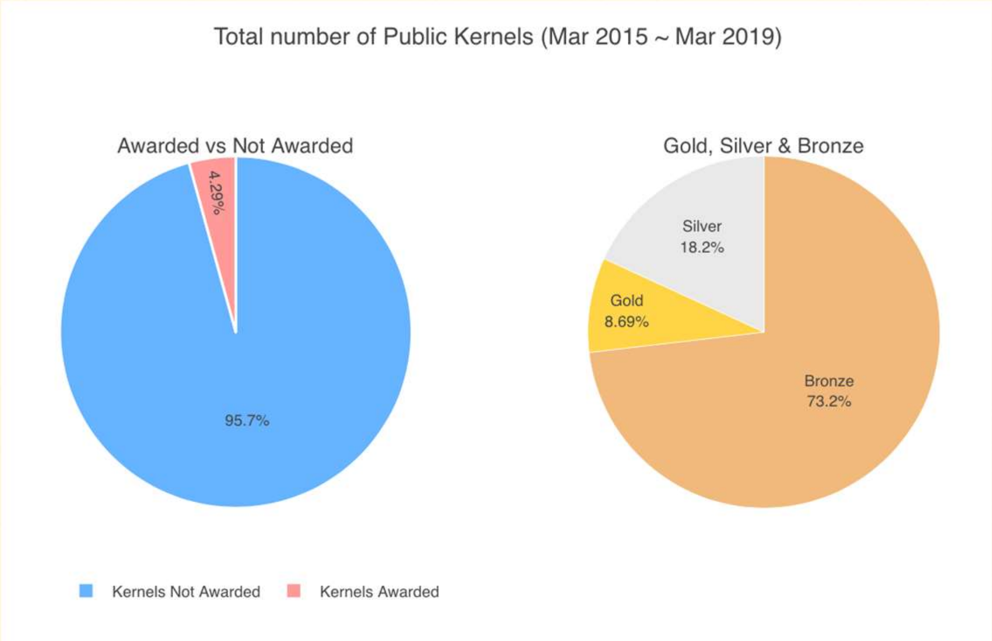
COMPETITION CONCEPTS

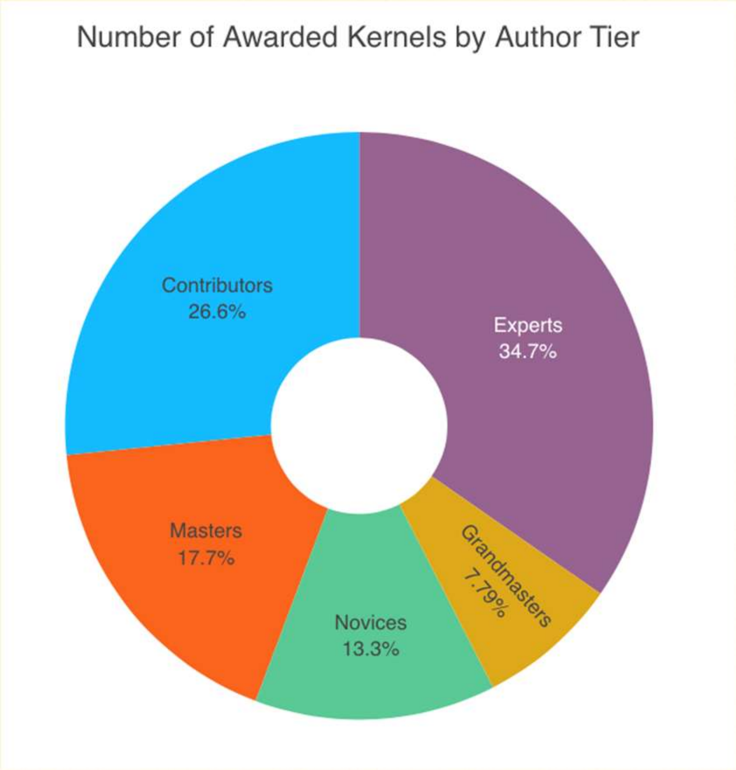
- Data
- Model
- Submission (only predictions)
- Evaluation function— RMSE, Logloss, Mean Absolute Error, ROCAUC, etc
- Leaderboard (Public/Private)
- Kernels

PUBLIC/PRIVATE TESTS

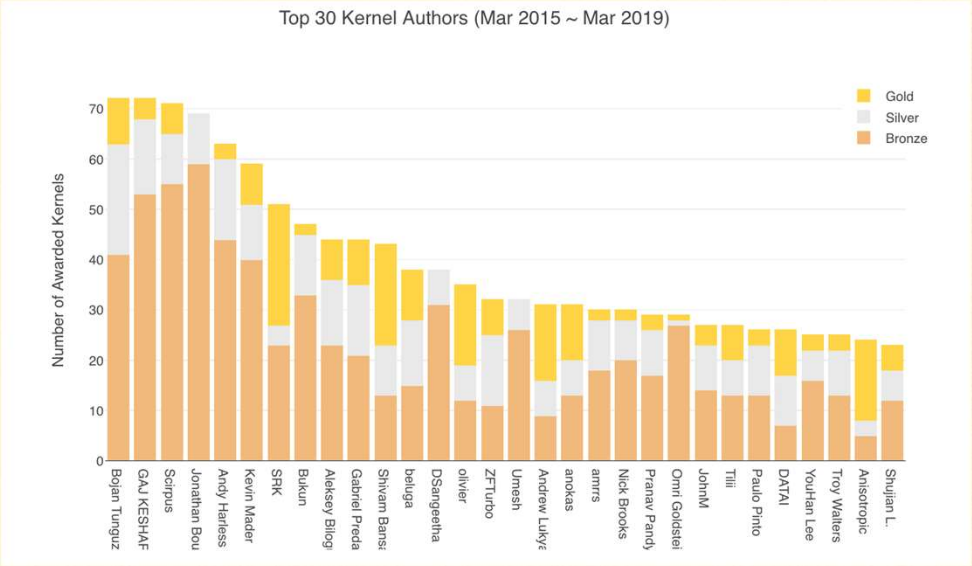


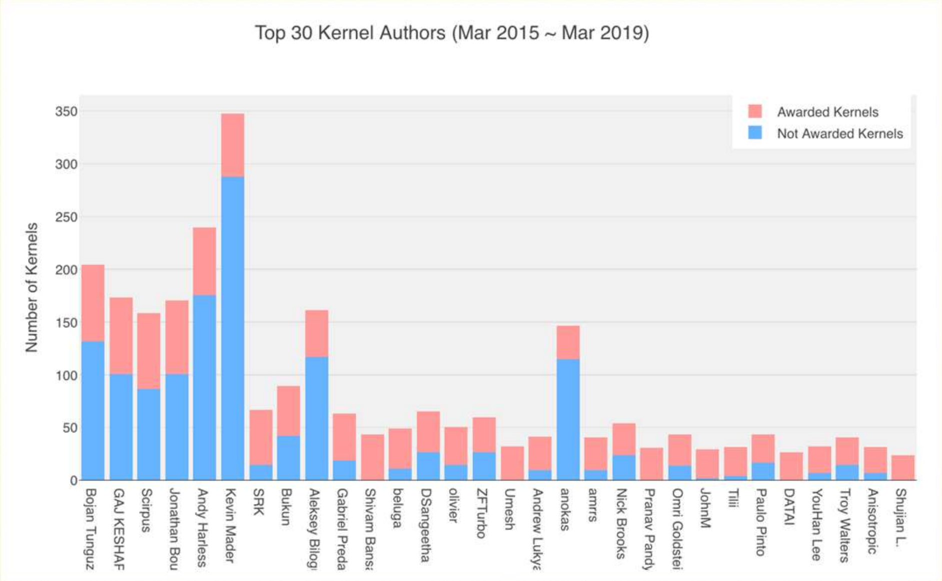
BEHIND THE MEDALS (1/2)





BEHIND THE MEDALS (2/2)





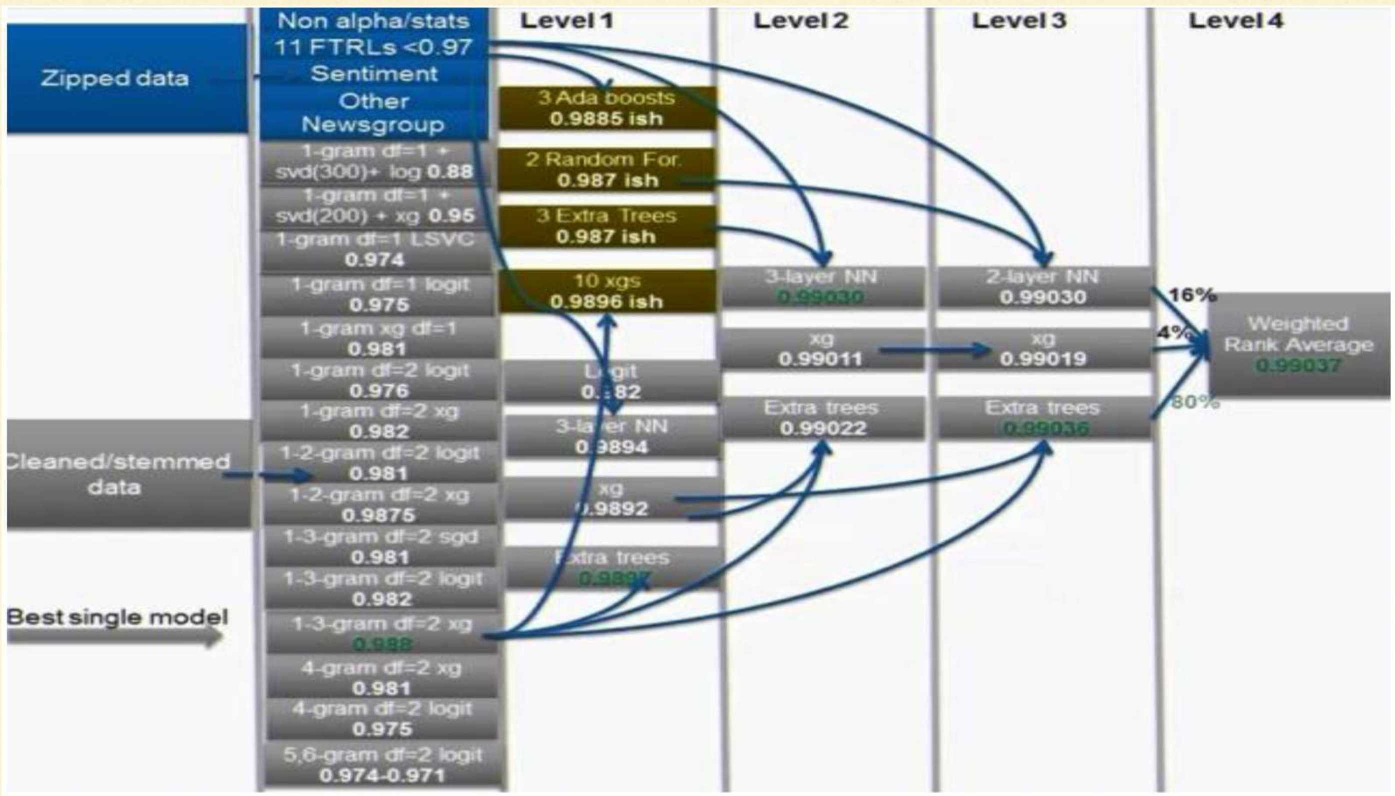
LET'S ENSEMBLE

An ode to Leo Brieman

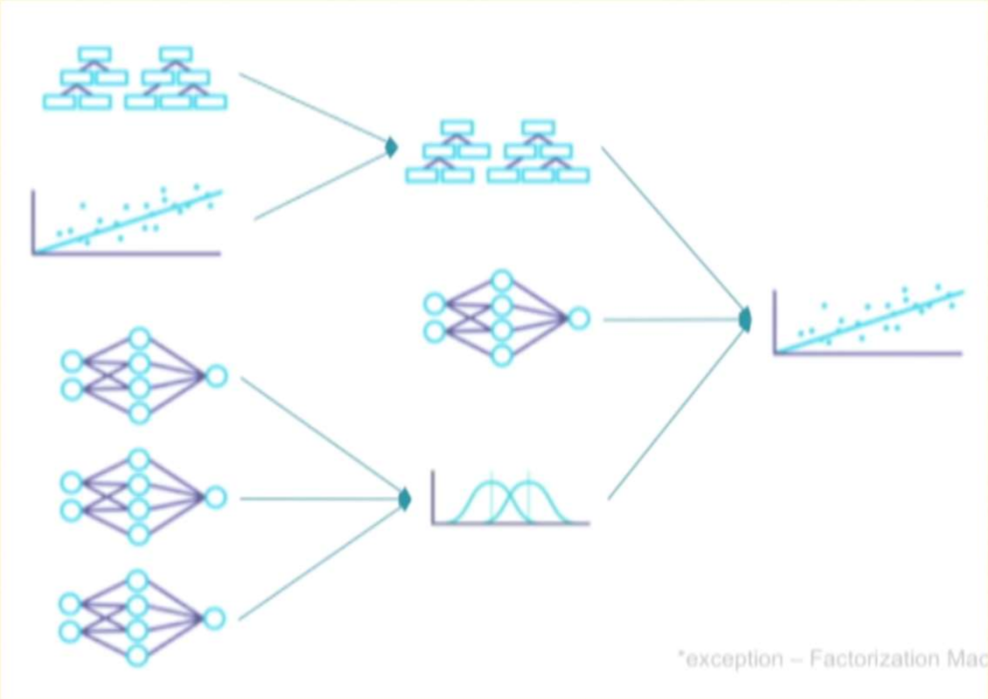
DATASET: UCI'S HEART DISEASE DATASET

Attribute Characteristics: Numeric Categorical	**Missing Values:** No	**Dataset Characteristic:** Univariate
No. of Instances: 303	**Evaluation Metric:** Accuracy	**Area:** Health and Medicine
No of Attributes: 13	**Associate Tasks:** Binary Classification	**Link:** https://www.kaggle.com/ronitf/heart-disease-uci

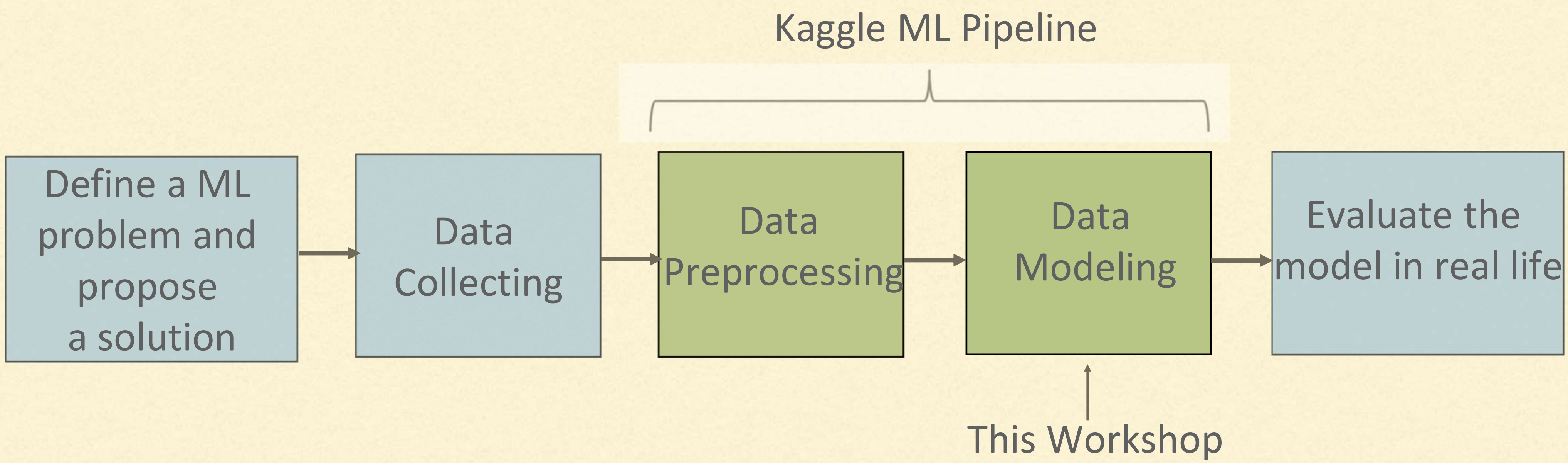
EXAMPLE- WINNING SOLUTIONS



STACKING EXAMPLE



REAL WORLD ML PIPELINE



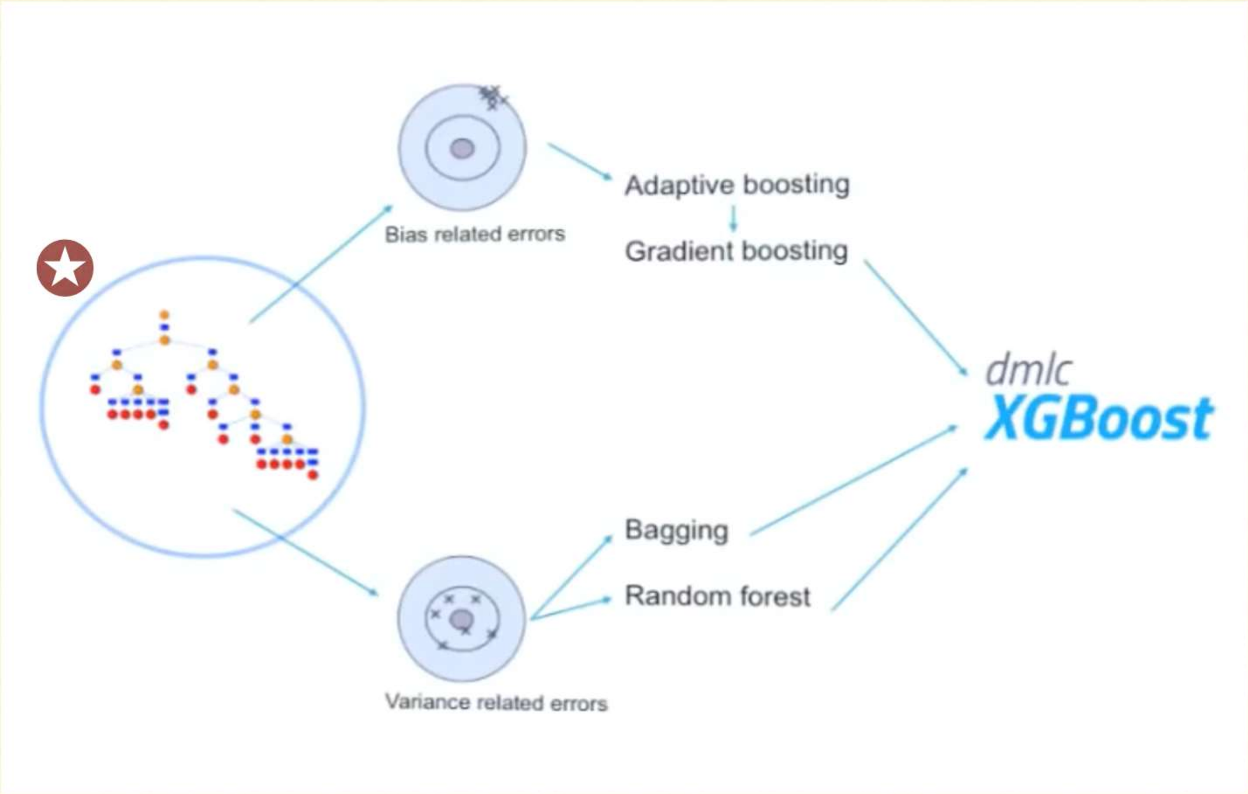
MODERN MACHINE LEARNING LANDSCAPE

Since 2016, Kaggle has been dominated by two approaches:

- Gradient Boosting Machines
- Deep Learning

(And Combined Models (eg. Voting, Averaging, and Stacking))

OUR ROADMAP



Bias related errors
Adaptive boosting
Gradient boosting
dmlc
XGBoost
Bagging
Random forest
Variance related errors

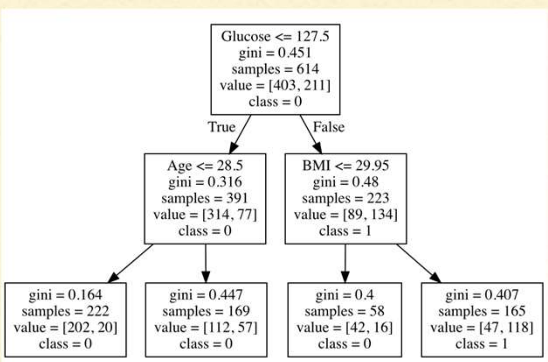
DECISION TREE

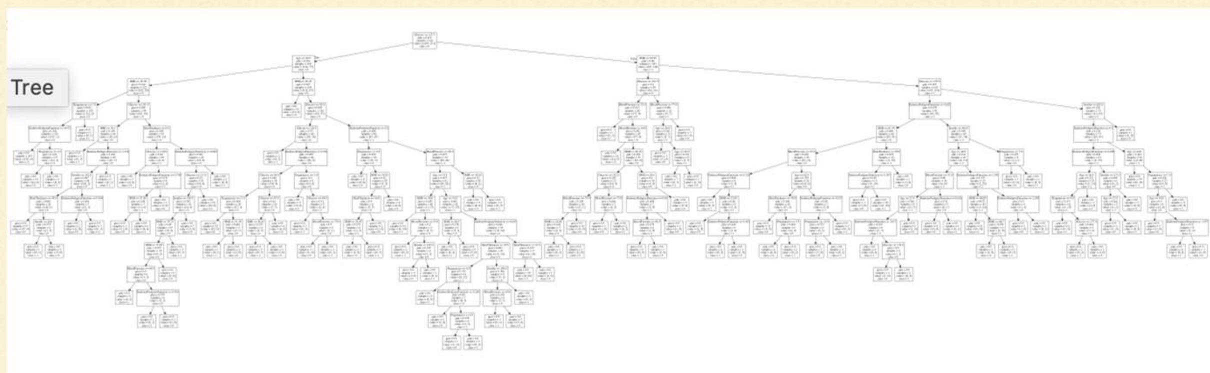
Nodes are split by finding the best split.

For classification, entropy or gini entropy are criteria for measuring impurity of a node.

For regression, we use mean squared error.

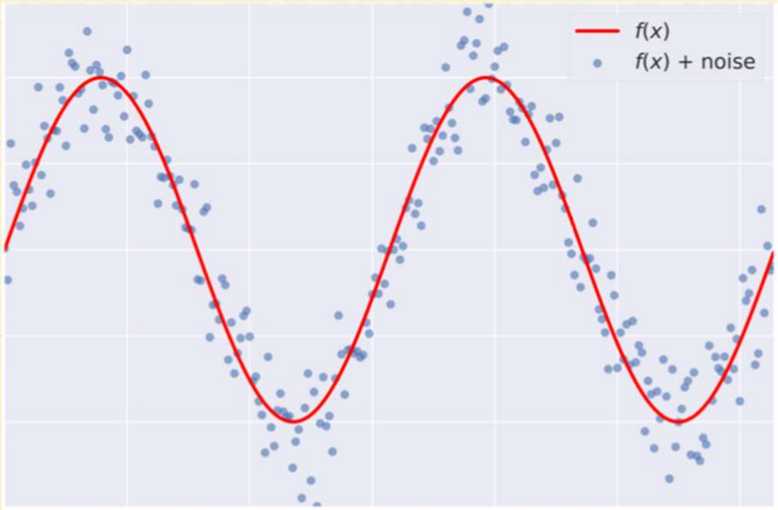
Pros and Cons





PREDICTIVE ANALYSIS

Supervised Learning: y= f(x), f is unknown.



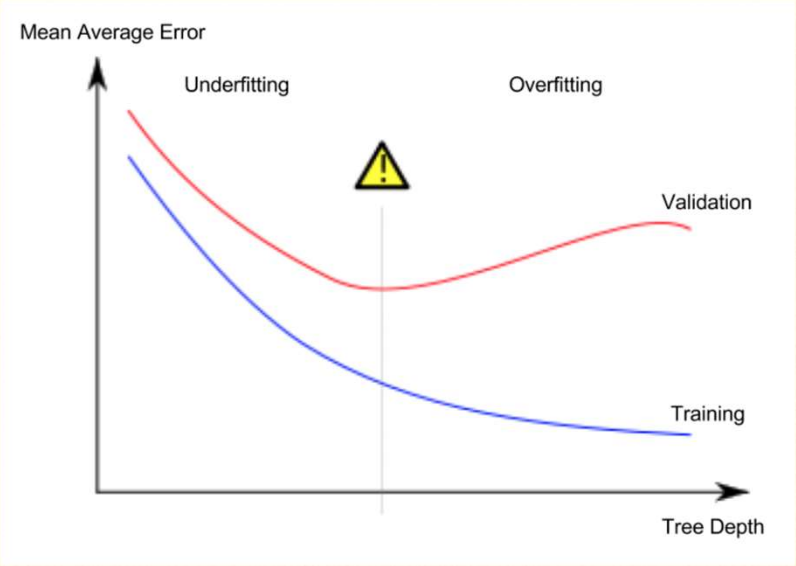
GOALS OF SUPERVISED LEARNING

Find a model $\hat{f}$ that best approximates f: $f \approx \hat{f}$

$\hat{f}$ can be Linear Regression, KNN, Decision Tree, Neural Networks, etc..

Discard noise as much as possible

End goal: $\hat{f}$ should receive a low predictive error on unseen data



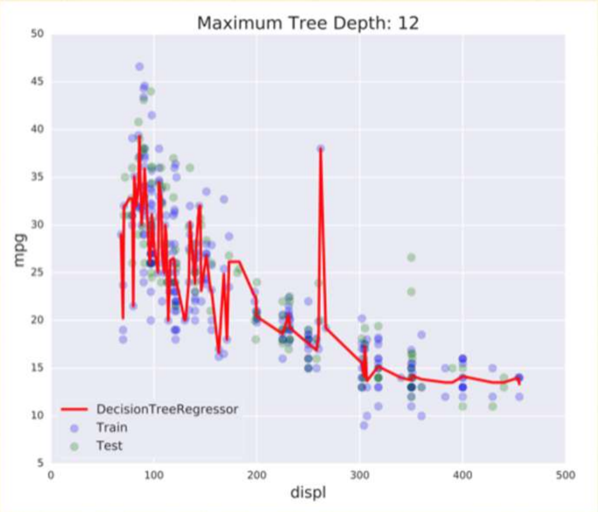
DIFFICULTIES IN APPROXIMATING F

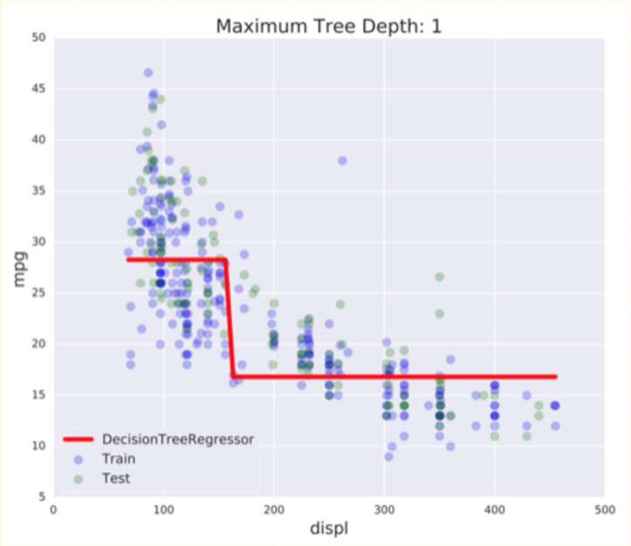
- **Overfitting**

$\hat{f}$(x) fits the training set noise

- **Underfitting**

$\hat{f}$ is not flexible enough to approximate



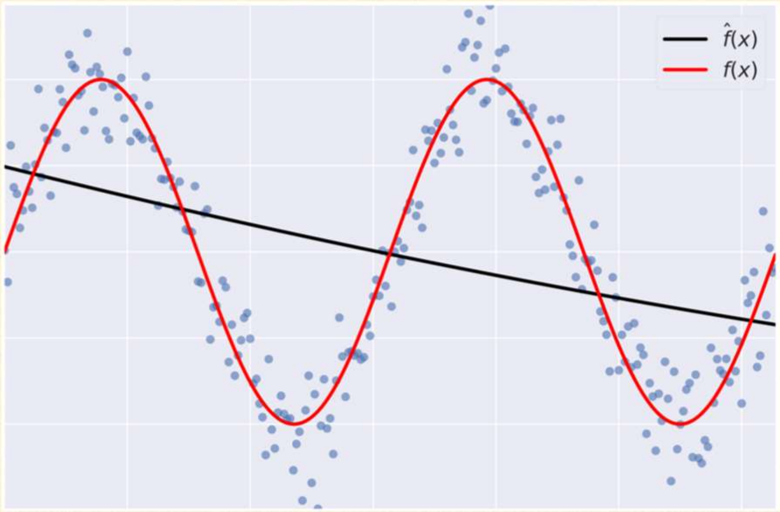


GENERALIZATION ERROR

- Generalization error of $\hat{f}$: Does $\hat{f}$ generalize well on unseen data?
- It can be decomposed as follows:
    - Generalization Error of $\hat{f}$: $Bias^2 + Variance + Irreducible\ Error$

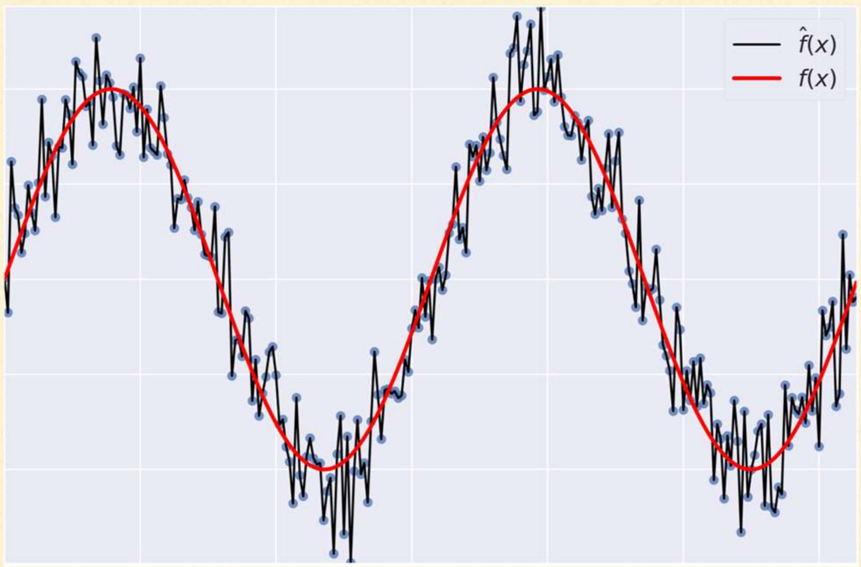
BIAS

Error term that tells you, on average how much $\hat{f} \neq f$

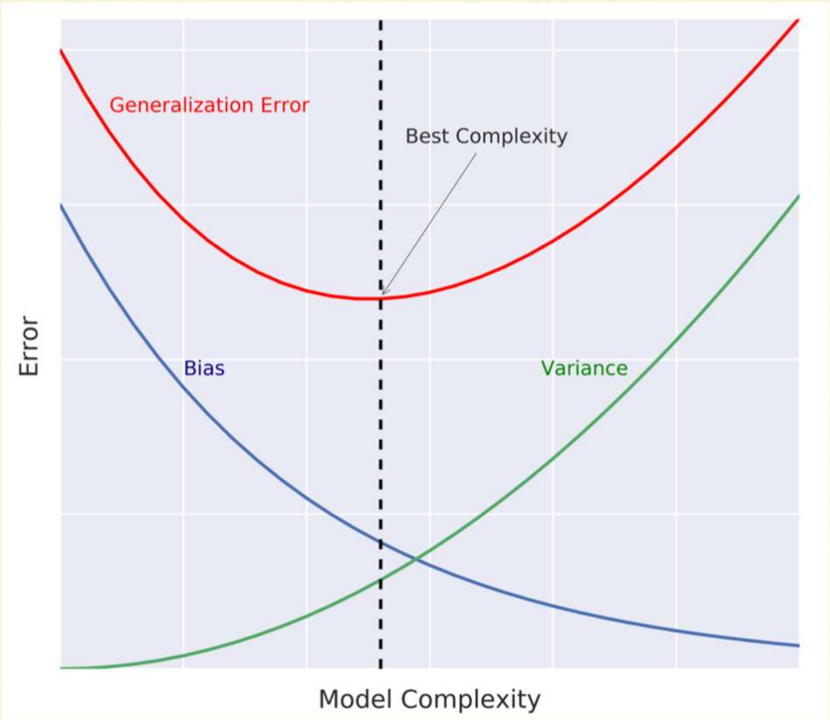


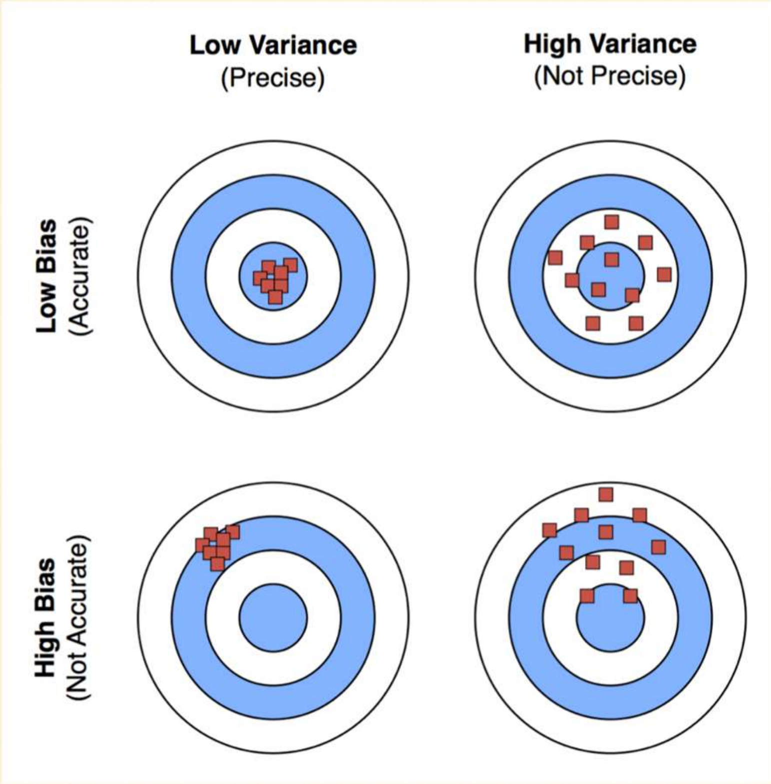
VARIANCE

Tells you how much $\hat{f}$ is inconsistent over different training sets.



BIAS-VARIANCE TRADEOFF





ESTIMATING THE GENERALIZATION ERROR

How do we estimate the generalization error of a model?

Cannot be done directly because:

- f is unknown
- usually you only have one dataset
- noise is unpredictable

Solution: train-test split, K-fold, CV fold

BAGGING (1/2)

Introduced by Leo Breiman in 1996 as a method to reduce model variance.

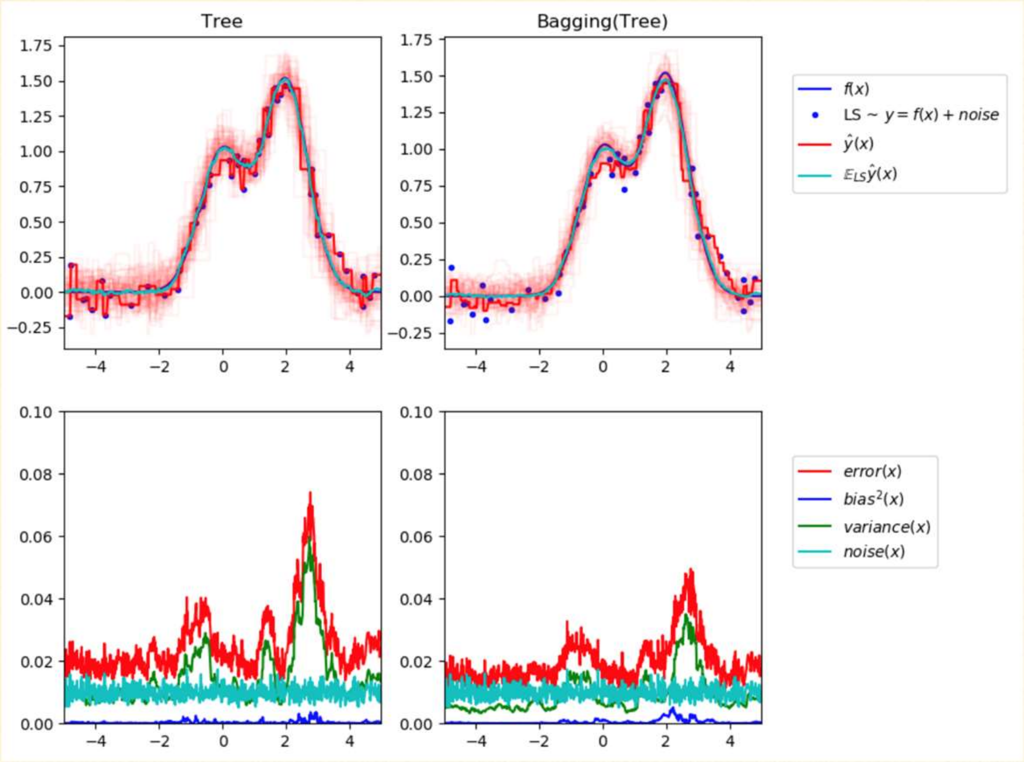
Also called Bootstrap Aggregation

Fit many large trees to bootstrap-resampled version of the training data, to classify by majority voting.

Bias of individual trees increases slightly

But that is compensated by averaging slightly different versions of the same model, which has the effect of reducing variance, thereby improve accuracy.

BAGGING (2/2)



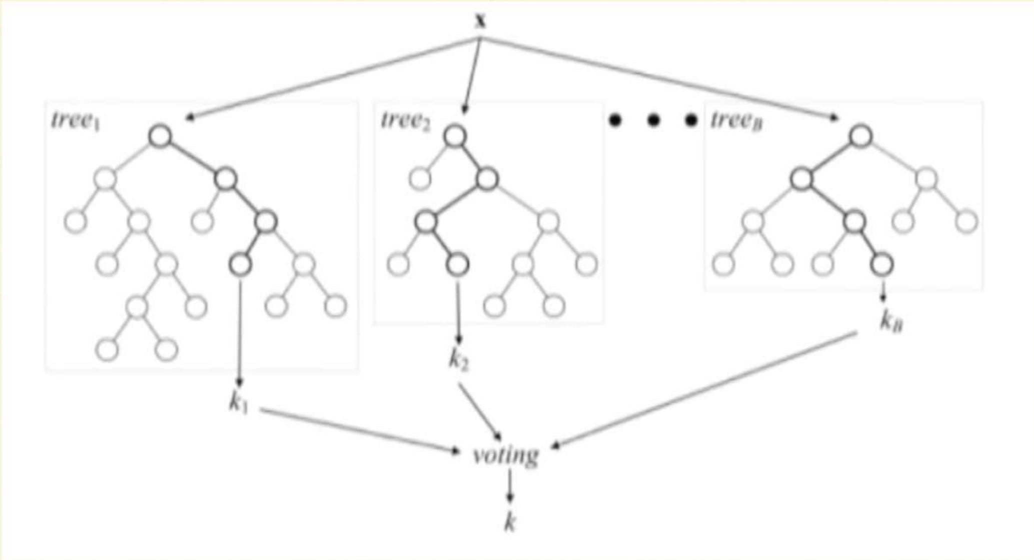
Tree
Bagging(Tree)
f(x)
LS ~ y = f(x) + noise
ŷ(x)
E_LS ŷ(x)
error(x)
bias²(x)
variance(x)
noise(x)

RANDOM FORESTS

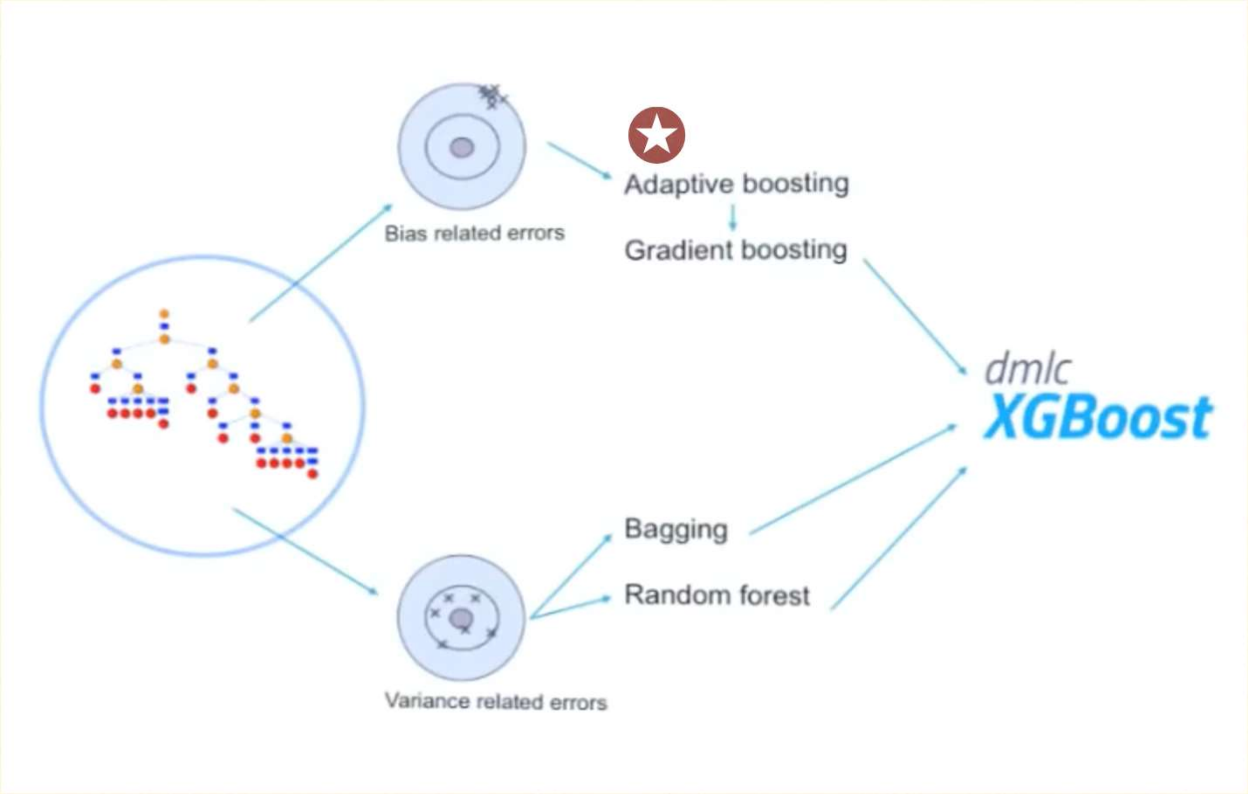
Each tree is built from samples drawn with replacement from the training set.

Nodes are split by finding the best split among a random subset of features.

As a result of this randomness, bias of the result slightly increases (w.r.t. the bias of a single non-random tree) but due to averaging its variance also decreases.



ROADMAP REVISITED

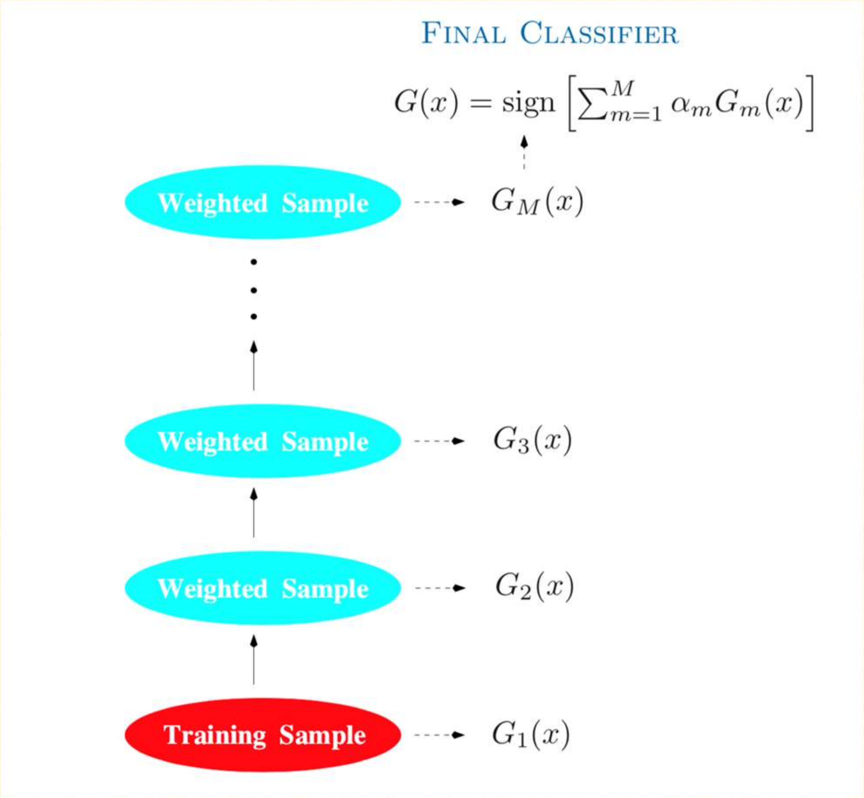


BOOSTING

A form of weighted averaging of models where each model is built sequential via taking into account the past model performance.

Main Boosting types:

- Weight based (Adaboost)
- Residual based (GBM)



ADABOOST

Rownum	x0	x1	x2	x3	y	pred	abs.error	weight
0	0.94	0.27	0.80	0.34	1	0.80	0.20	1.20
1	0.84	0.79	0.89	0.05	1	0.75	0.25	1.25
2	0.83	0.11	0.23	0.42	1	0.65	0.35	1.35
3	0.74	0.26	0.03	0.41	0	0.40	0.40	1.40
4	0.08	0.29	0.76	0.37	0	0.55	0.55	1.55
5	0.71	0.76	0.43	0.95	1	0.34	0.66	1.66
6	0.08	0.72	0.97	0.04	0	0.02	0.02	1.02

Picture Credits: Coursera Course “How to win data science competitions”

RESIDUAL BASED BOOSTING

Rownum	x0	x1	x2	x3	y	pred	error
0	0.94	0.27	0.80	0.34	1	0.80	0.20
1	0.84	0.79	0.89	0.05	1	0.75	0.25
2	0.83	0.11	0.23	0.42	1	0.65	0.35
3	0.74	0.26	0.03	0.41	0	0.40	-0.40
4	0.08	0.29	0.76	0.37	0	0.55	-0.55
5	0.71	0.76	0.43	0.95	1	0.34	0.66
6	0.08	0.72	0.97	0.04	0	0.02	-0.02

Picture Credits: Coursera Course “How to win data science competitions”

EXAMPLE (CONT.)

Rownum	x0	x1	x2	x3	y	new pred	old pred
0	0.94	0.27	0.80	0.34	0.2	0.15	0.80
1	0.84	0.79	0.89	0.05	0.25	0.20	0.75
2	0.83	0.11	0.23	0.42	0.35	0.40	0.65
3	0.74	0.26	0.03	0.41	-0.4	-0.30	0.40
4	0.08	0.29	0.76	0.37	-0.55	-0.20	0.55
5	0.71	0.76	0.43	0.95	0.66	0.24	0.34
6	0.08	0.72	0.97	0.04	-0.02	-0.01	0.02

To predict Rownum=1 we would say :

Final prediction = 0.75 + 0.20 = **0.95**

Picture Credits: Coursera Course “How to win data science competitions”

RESIDUAL BASED BOOSTING PARAMETERS

Learning rate (shrinkage)

Number of estimators

Row (sub) sampling

Column (sub) sampling

Input model- better be trees

RESIDUAL BASED IMPLEMENTATIONS

Xgboost

Lightgbm

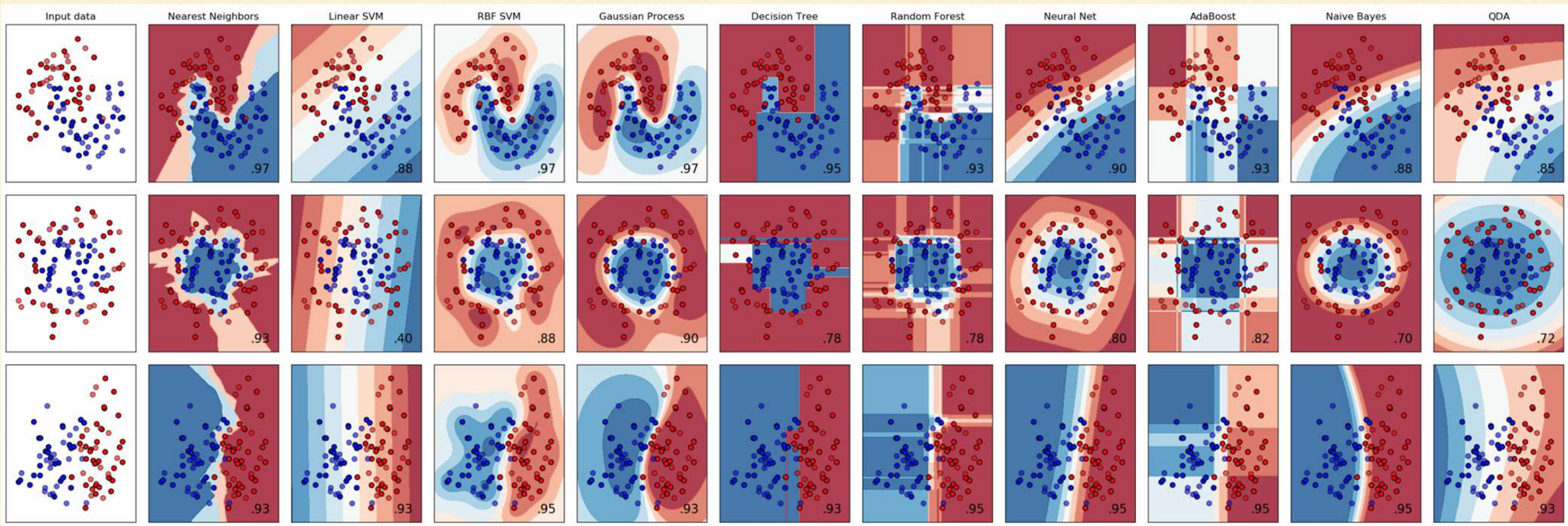
H2O’s GBM

CatBoost

Sklearn’s GBM

COMBINING MODELS

Get additional performance gains by combining strong models made with different algorithms.

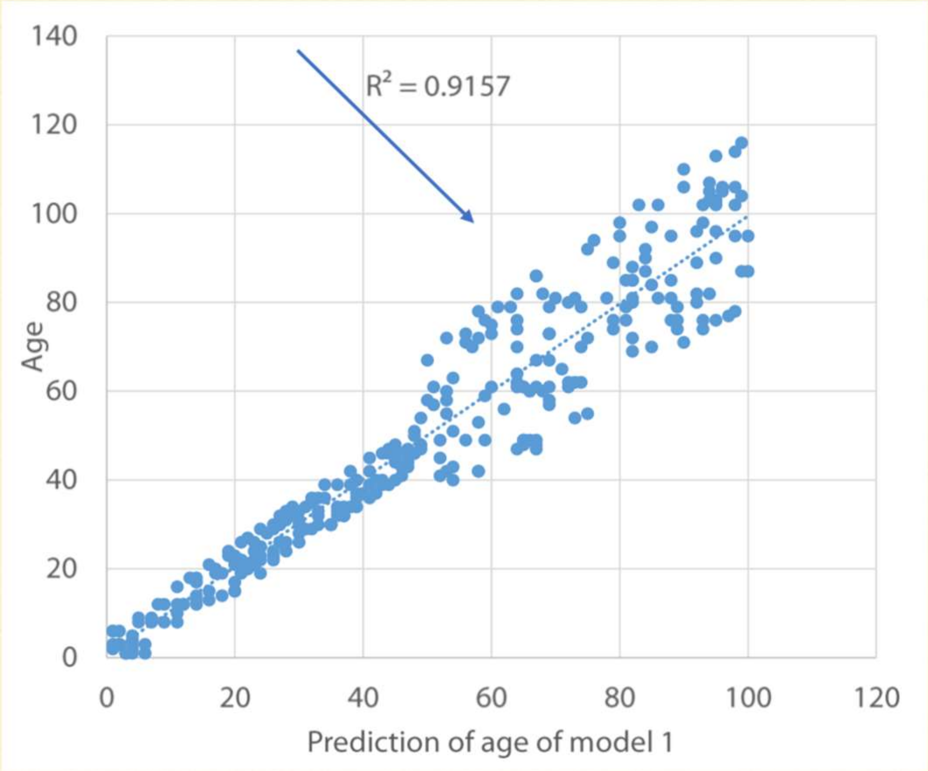


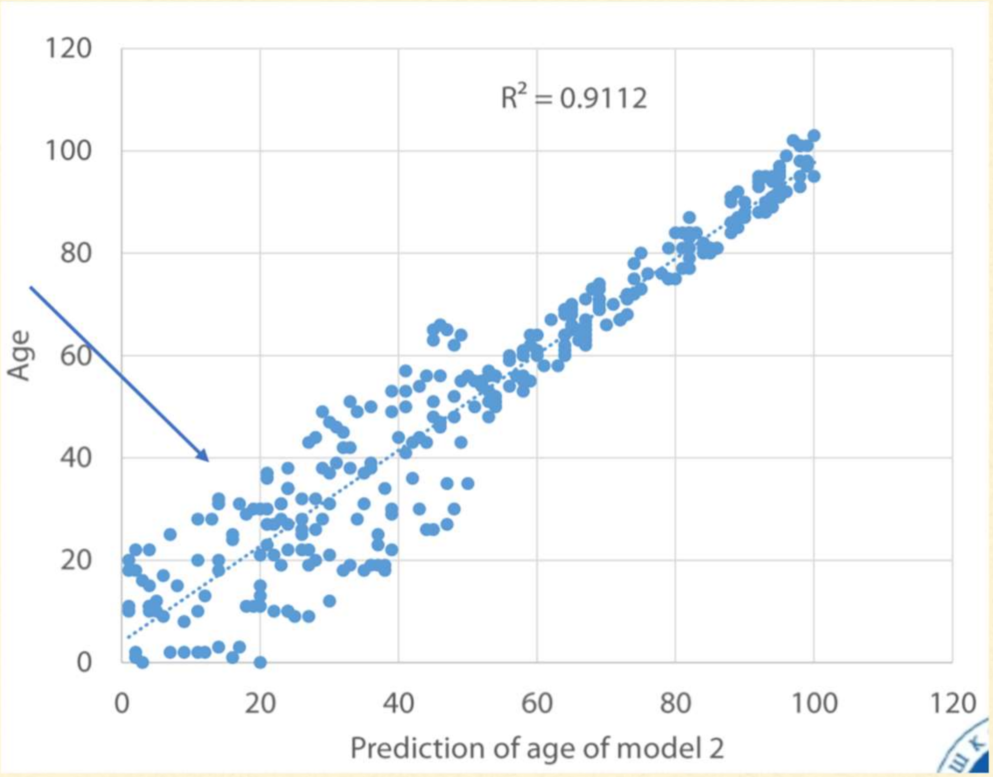
COMBINING MODELS

Three typical ways to combine models are:

- Averaging
  - Simple Averaging
  - Weighted Averaging
  - Conditional Averaging
- Majority Voting
- Stacking

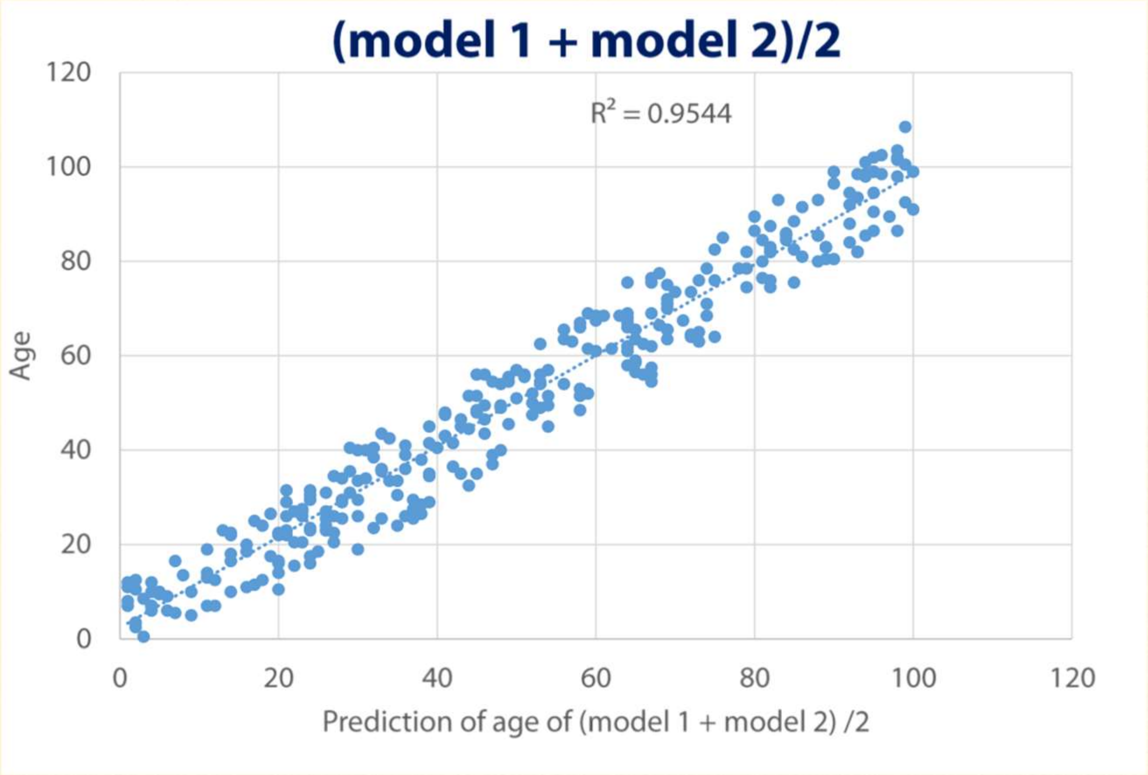
AVERAGING EXAMPLE





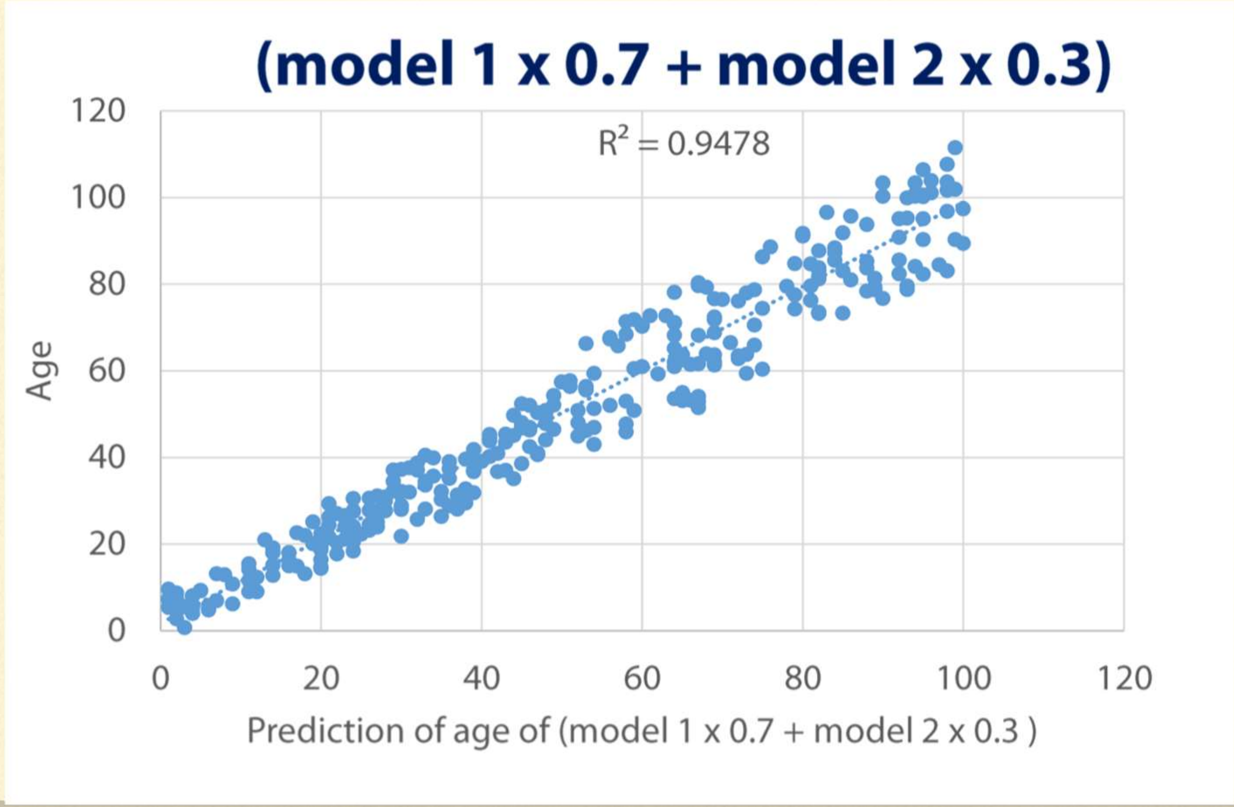
Picture Credits: Coursera Course “How to win data science competitions”

SIMPLE AVERAGING



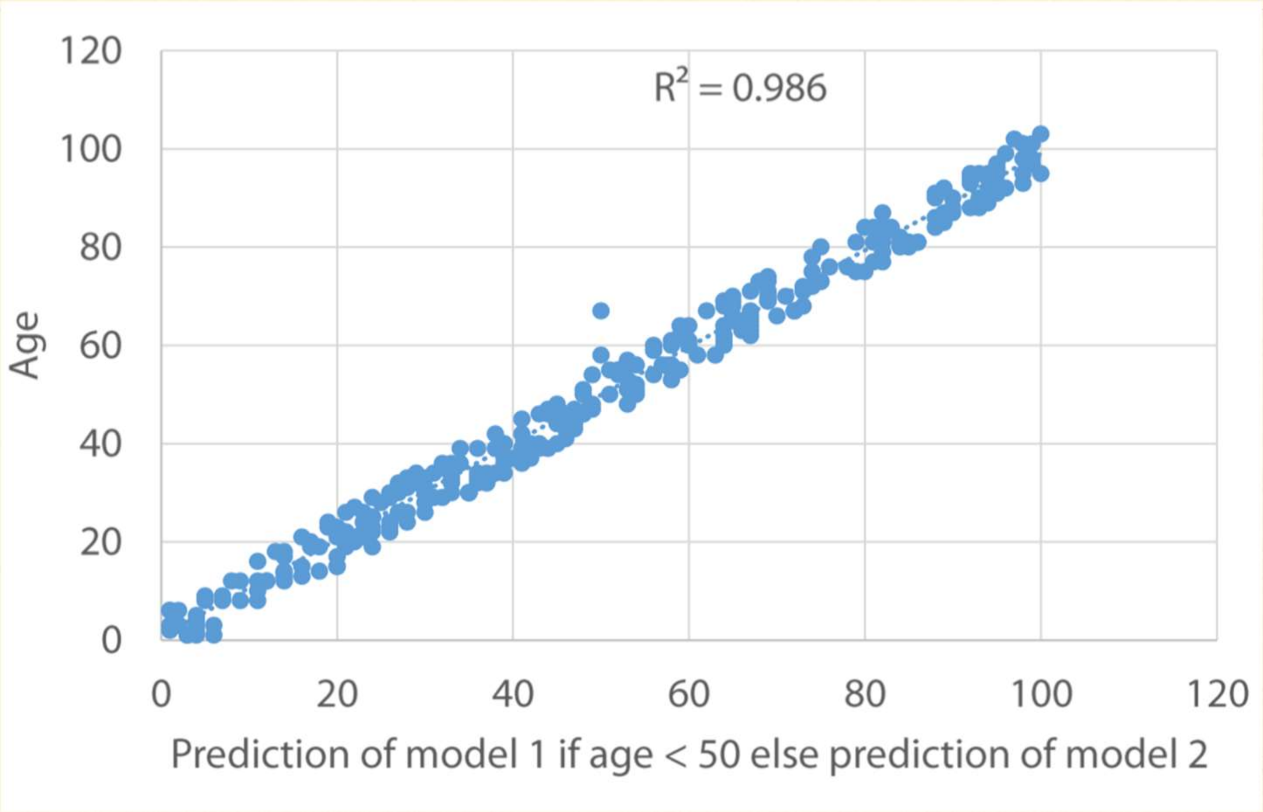
Picture Credits: Coursera Course "How to win data science competitions"

WEIGHTED AVERAGING



Picture Credits: Coursera Course “How to win data science competitions”

CONDITIONAL AVERAGING

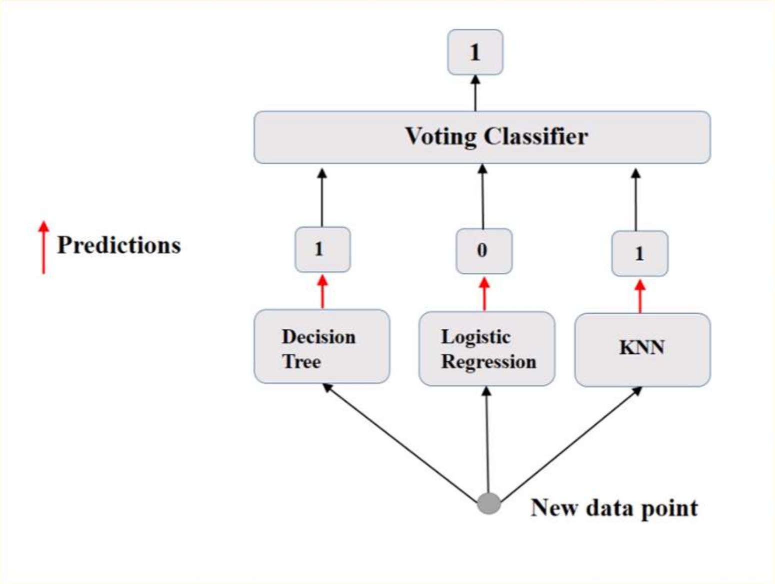


Picture Credits: Coursera Course “How to win data science competitions”

VOTING CLASSIFIERS (1/2)

Binary Classification task

Meta-Model prediction: **hard voting** (also known as majority voting).



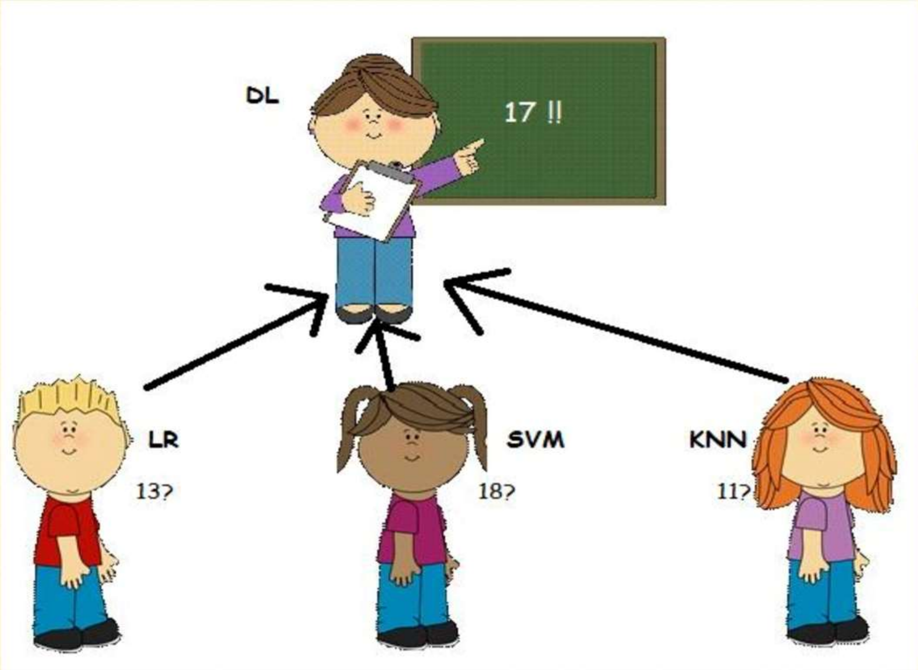
VOTING CLASSIFIERS (2/2)

In **soft voting**, every individual classifier provides a probability value that a specific data point belongs to a particular target class. The predictions are weighted by the classifier's importance and summed up. Then the target label with the greatest sum of weighted probabilities wins the vote.

classifier	class 0	class 1
classifier #1	0.3 w_1	0.7 w_1
classifier #2	0.5 w_2	0.5 w_2
classifier #3	0.4 w_3	0.6 w_3

STACKING

Making predictions of a number of models on a hold-out set and then using a different (meta) model to train on these predictions.



Picture Credits: Coursera Course “How to win data science competitions”

STACKING EXAMPLE

A				
X0	x1	x2	xn	y
0.17	0.25	0.93	0.79	1
0.35	0.61	0.93	0.57	0
0.44	0.59	0.56	0.46	0
0.37	0.43	0.74	0.28	1
0.96	0.07	0.57	0.01	1

B				
X0	x1	x2	xn	y
0.89	0.72	0.50	0.66	0
0.58	0.71	0.92	0.27	1
0.10	0.35	0.27	0.37	0
0.47	0.68	0.30	0.98	0
0.39	0.53	0.59	0.18	1

C				
X0	x1	x2	xn	y
0.29	0.77	0.05	0.09	?
0.38	0.66	0.42	0.91	?
0.72	0.66	0.92	0.11	?
0.70	0.37	0.91	0.17	?
0.59	0.98	0.93	0.65	?

Train algorithm **0** on A and make predictions for B and C and save to **B1**, **C1**
Train algorithm **1** on A and make predictions for B and C and save to **B1**, **C1**
Train algorithm **2** on A and make predictions for B and C and save to **B1**, **C1**

B1			
pred0	pred1	pred2	y
0.24	0.72	0.70	0
0.95	0.25	0.22	1
0.64	0.80	0.96	0
0.89	0.58	0.52	0
0.11	0.20	0.93	1

C1				
pred0	pred1	pred2	y	Preds3
0.50	0.50	0.39	?	0.45
0.62	0.59	0.46	?	0.23
0.22	0.31	0.54	?	0.99
0.90	0.47	0.09	?	0.34
0.20	0.09	0.61	?	0.05

Train algorithm **3** on B1 and make predictions for C1

Picture Credits: Coursera Course “How to win data science competitions”

METHODOLOGY

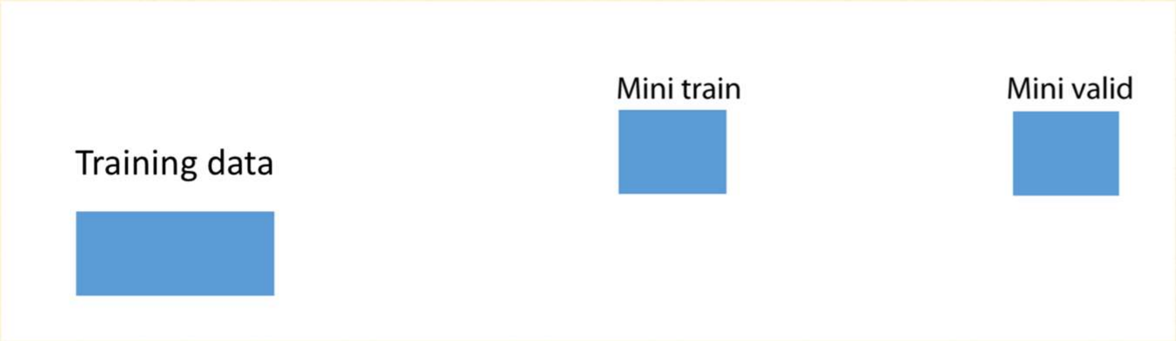
Wolpert introduced stacking in 1992. It involves:

1. Splitting the train set into two disjoint sets.
2. Train several base learners on the first part (train set).
3. Make predictions with base learners on the second part (validation set).
4. Using the predictions from the previous step as inputs to train a higher level learner.

```
#split train data in 2 parts, training and  valdiation.
training,valid,ytraining,yvalid = train_test_split(train,y,test_size=0.5)
#specify models
model1=RandomForestRegressor()
model2=LinearRegression()
#fit models
model1.fit(training,ytraining)
model2.fit(training,ytraining)
#make predictions for validation
preds1=model1.predict(valid)
preds2=model2.predict(valid)
#make predictions for test data
test_preds1=model1.predict(test)
test_preds2=model2.predict(test)
#Form a new dataset for valid and test via stacking the predictions
stacked_predictions=np.column_stack((preds1,preds2))
stacked_test_predictions=np.column_stack((test_preds1,test_preds2))
#specify meta model
meta_model=LinearRegression()
#fit meta model on stacked predictions
meta_model.fit(stacked_predictions,yvalid)
#make predictions on the stacked predictions of the test data
final_predictions=meta_model.predict(stacked_test_predictions)
```

HOW TO TRAIN (1/4)

For large datasets:



Picture Credits: Coursera Course “How to win data science competitions”

HOW TO TRAIN (2/4)

We use K-fold CV

Fold 1

x0	x1	x2	x3	y
0.94	0.27	0.80	0.34	1
0.02	0.22	0.17	0.84	0
0.83	0.11	0.23	0.42	1
0.74	0.26	0.03	0.41	0
0.08	0.29	0.76	0.37	0
0.71	0.76	0.43	0.95	1
0.08	0.72	0.97	0.04	0
0.84	0.79	0.89	0.05	1

Predict

0.94	0.27	0.80	0.34	1
0.02	0.22	0.17	0.84	0

Train

0.83	0.11	0.23	0.42	1
0.74	0.26	0.03	0.41	0
0.08	0.29	0.76	0.37	0
0.71	0.76	0.43	0.95	1
0.08	0.72	0.97	0.04	0
0.84	0.79	0.89	0.05	1

pred
0.96
0.03
0.00
0.00
0.00
0.00
0.00
0.00

Picture Credits: Coursera Course “How to win data science competitions”

HOW TO TRAIN (3/4)

Fold 1

x0	x1	x2	x3	y
0.94	0.27	0.80	0.34	1
0.02	0.22	0.17	0.84	0
0.83	0.11	0.23	0.42	1
0.74	0.26	0.03	0.41	0
0.08	0.29	0.76	0.37	0
0.71	0.76	0.43	0.95	1
0.08	0.72	0.97	0.04	0
0.84	0.79	0.89	0.05	1

Predict

0.83	0.11	0.23	0.42	1
0.74	0.26	0.03	0.41	0

Train

0.94	0.27	0.80	0.34	1
0.02	0.22	0.17	0.84	0
0.08	0.29	0.76	0.37	0
0.71	0.76	0.43	0.95	1
0.08	0.72	0.97	0.04	0
0.84	0.79	0.89	0.05	1

pred
0.96
0.03
0.90
0.12
0.00
0.00
0.00
0.00

Picture Credits: Coursera Course “How to win data science competitions”

HOW TO TRAIN (4/4)

x0	x1	x2	x3	y
0.94	0.27	0.80	0.34	1
0.02	0.22	0.17	0.84	0
0.83	0.11	0.23	0.42	1
0.74	0.26	0.03	0.41	0
0.08	0.29	0.76	0.37	0
0.71	0.76	0.43	0.95	1
0.08	0.72	0.97	0.04	0
0.84	0.79	0.89	0.05	1

Predict

0.08	0.72	0.97	0.04	0
0.84	0.79	0.89	0.05	1

Train

0.94	0.27	0.80	0.34	1
0.02	0.22	0.17	0.84	0
0.83	0.11	0.23	0.42	1
0.74	0.26	0.03	0.41	0
0.08	0.29	0.76	0.37	0
0.71	0.76	0.43	0.95	1

test
0.43
0.03
0.90
0.12
0.03
0.77
0.18
0.91

pred	pred
0.96	0.00
0.03	0.00
0.90	0.00
0.12	0.00
0.03	0.00
0.77	0.00
0.18	0.00
0.91	0.00

Picture Credits: Coursera Course “How to win data science competitions”

THINGS TO BE MINDFUL OF

With time sensitive data—respect time

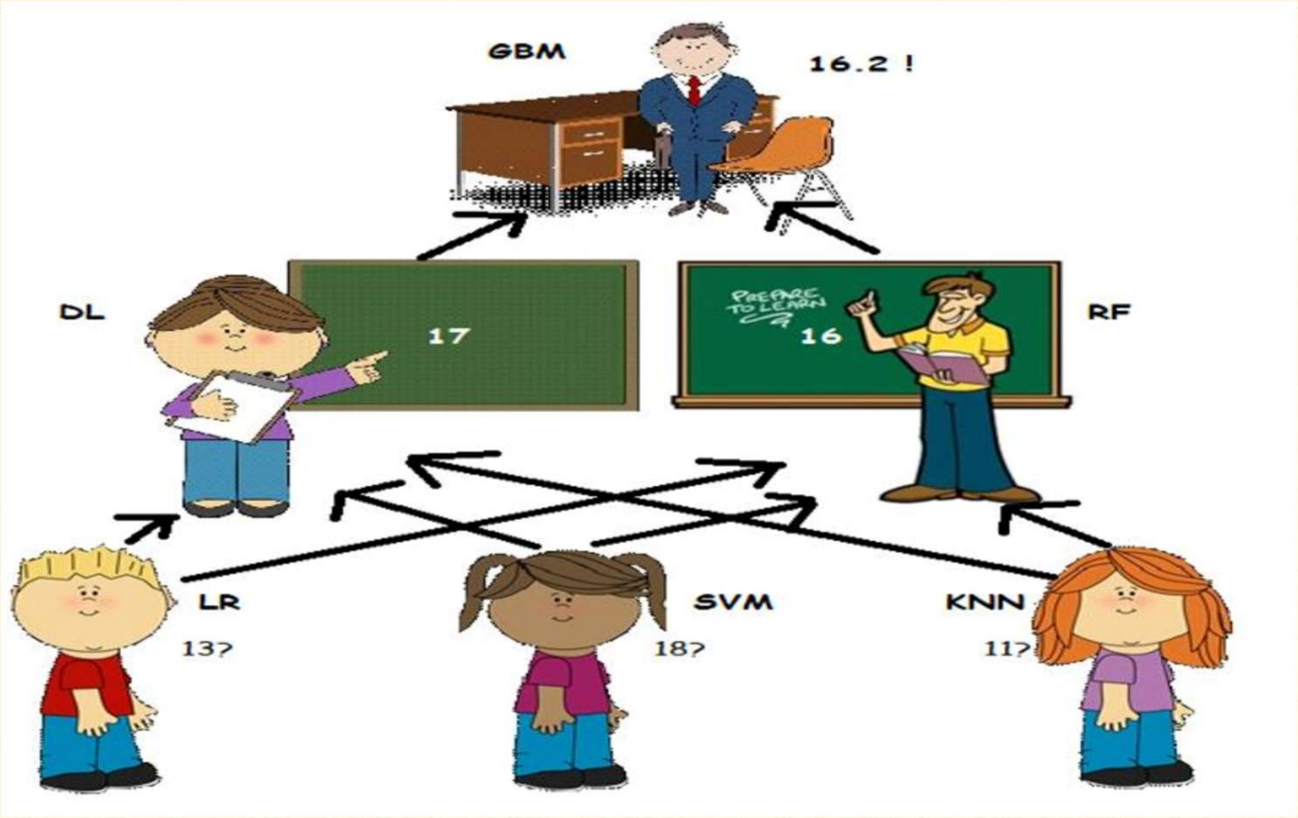
Diversity is as important as performance

Diversity may come from

- Different algorithms
- Different input features

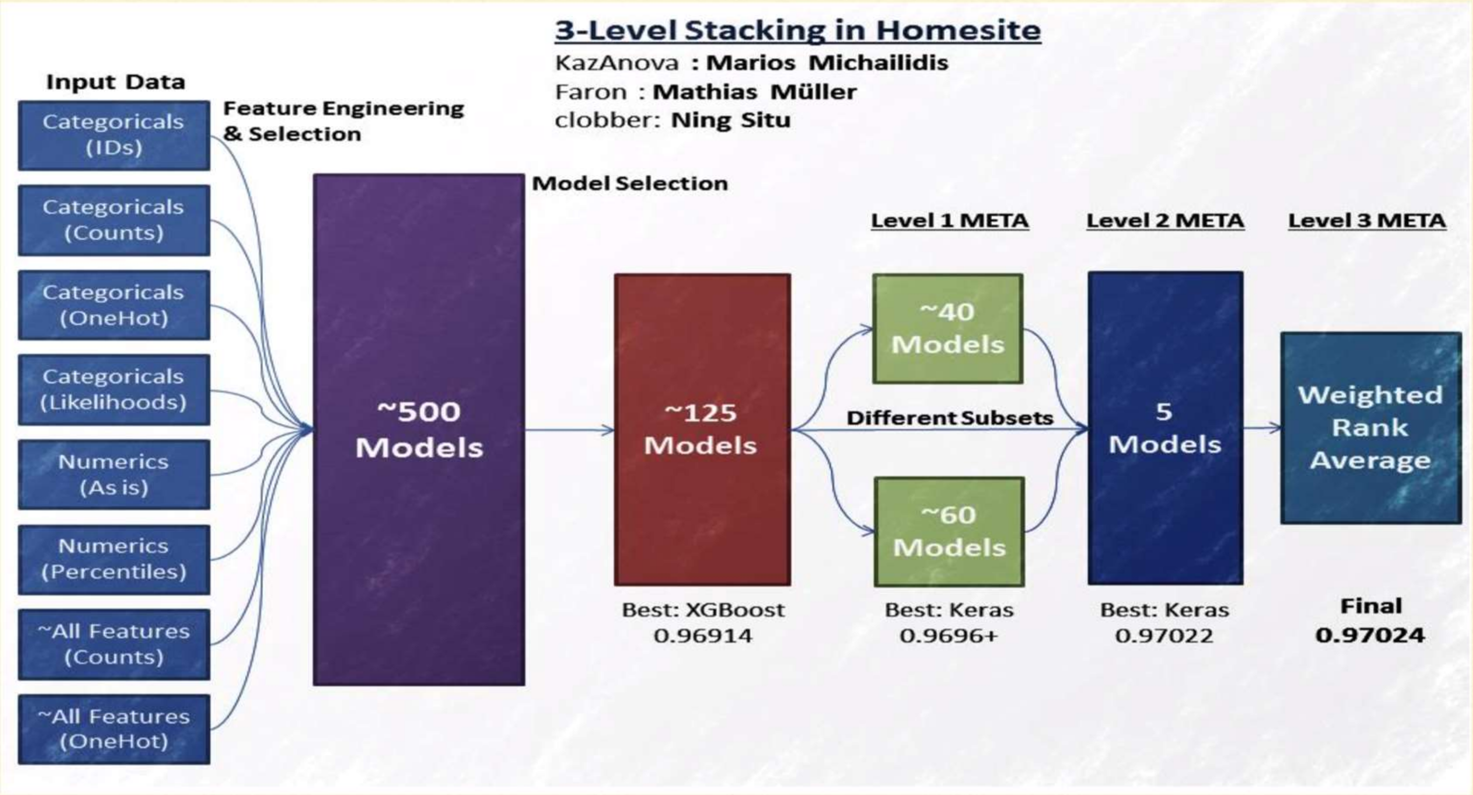
Performance plateauing after N models

NAIVE EXAMPLE



Picture Credits: Coursera Course “How to win data science competitions”

HOMESITE-WINNING SOLUTION



Picture Credits: Coursera Course “How to win data science competitions”

ON TO COMPETITIONS

HOUSE PRICES: ADVANCED REGRESSION TECHNIQUES

DATASET: HOUSE PRICES COMPETITION

DataSet Characteristics: Multivariate	**Attribute Characteristics:** Integer	**Associate Tasks:** Regression
No. of Instances: 1460	**Missing Values:** Yes	**Area:** Real-Estate
No of Attributes: 79	**Evaluation Metric:** RMSLE	**Link:** House Prices: Advanced Regression Techniques

Questions?

Thank You!