Ensemble Learning

Presenter: Van Anh Le

What is ensemble learning?

- A group of predictors (classifiers or regressors) is called an ensemble; thus, this technique is called *Ensemble Learning*,

- *Ensemble Learning* algorithm is called an *Ensemble method*.

Stories of Success

- Million-dollar prize

$1 million prize for a 10% improvement over Netflix's current movie recommender

- Supervised learning task
  - Training data is a set of users and ratings (1,2,3,4,5 stars) those users have given to movies.
  - Construct a classifier that given a user and an unrated movie, correctly classifies that movie as either 1, 2, 3, 4, or 5 stars

- Competition
  - At first, single-model methods are developed, and performances are improved
  - However, improvements slowed down
  - Later, individuals and teams merged their results, and significant improvements are observed

Leaderboard

Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
Grand Prize - RMSE = 0.8567 - Winning Team: BellKor's Pragmatic Chaos				
1	BellKor's Pragmatic Chaos	0.8567	10.06	2009-07-26 18:18:28
2	The Ensemble	0.8567	10.06	2009-07-26 18:38:22
3	Grand Prize Team	0.8582	9.90	2009-07-10 21:24:40
4	Opera Solutions and Vandelay United	0.8588	9.84	2009-07-10 01:12:31
5	Vandelay Industries !	0.8591	9.81	2009-07-10 00:32:20
6	PragmaticTheory	0.8594	9.77	2009-06-24 12:06:56
7	BellKor in BigChaos	0.8601	9.70	2009-05-13 08:14:09
8	Dace	0.8612	9.59	2009-07-24 17:18:43
9	Feeds2	0.8622	9.48	2009-07-12 13:11:51
10	BigChaos	0.8623	9.47	2009-04-07 12:33:59
12	BellKor	0.8624	9.46	2009-07-26 17:19:11
Progress Prize 2008 - RMSE = 0.8627 - Winning Team: BellKor in BigChaos				
13	xiangliang	0.8642	9.27	2009-07-15 14:53:22
14	Gravity	0.8643	9.26	2009-04-22 18:31:32
15	Ces	0.8651	9.18	2009-06-21 19:24:53
20	acmehill	0.8668	9.00	2009-03-21 16:20:50
Progress Prize 2007 - RMSE = 0.8723 - Winning Team: KorBell				
Cinematch score - RMSE = 0.9525				

"Our final solution (RMSE=0.8712) consists of blending 107 individual results. "

"Predictive accuracy is substantially improved when blending multiple predictors. Our experience is that most efforts should be concentrated in deriving substantially different approaches, rather than refining a single technique. "

Overview Data Kernels Discussion Leaderboard Rules **Late Submission**

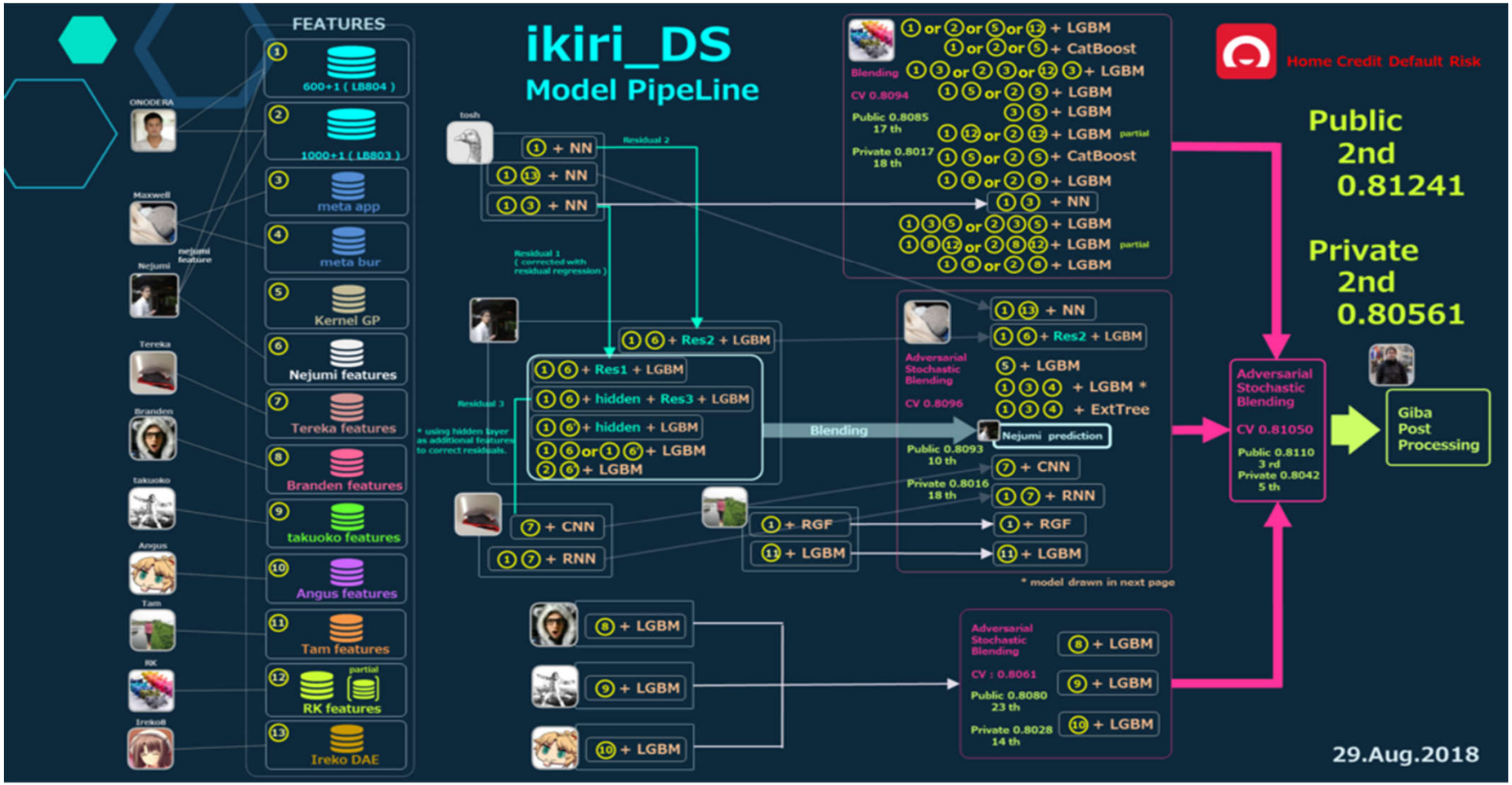
Public Leaderboard Private Leaderboard

This leaderboard is calculated with approximately 20% of the test data.
The final results will be based on the other 80%, so the final standings may be different.

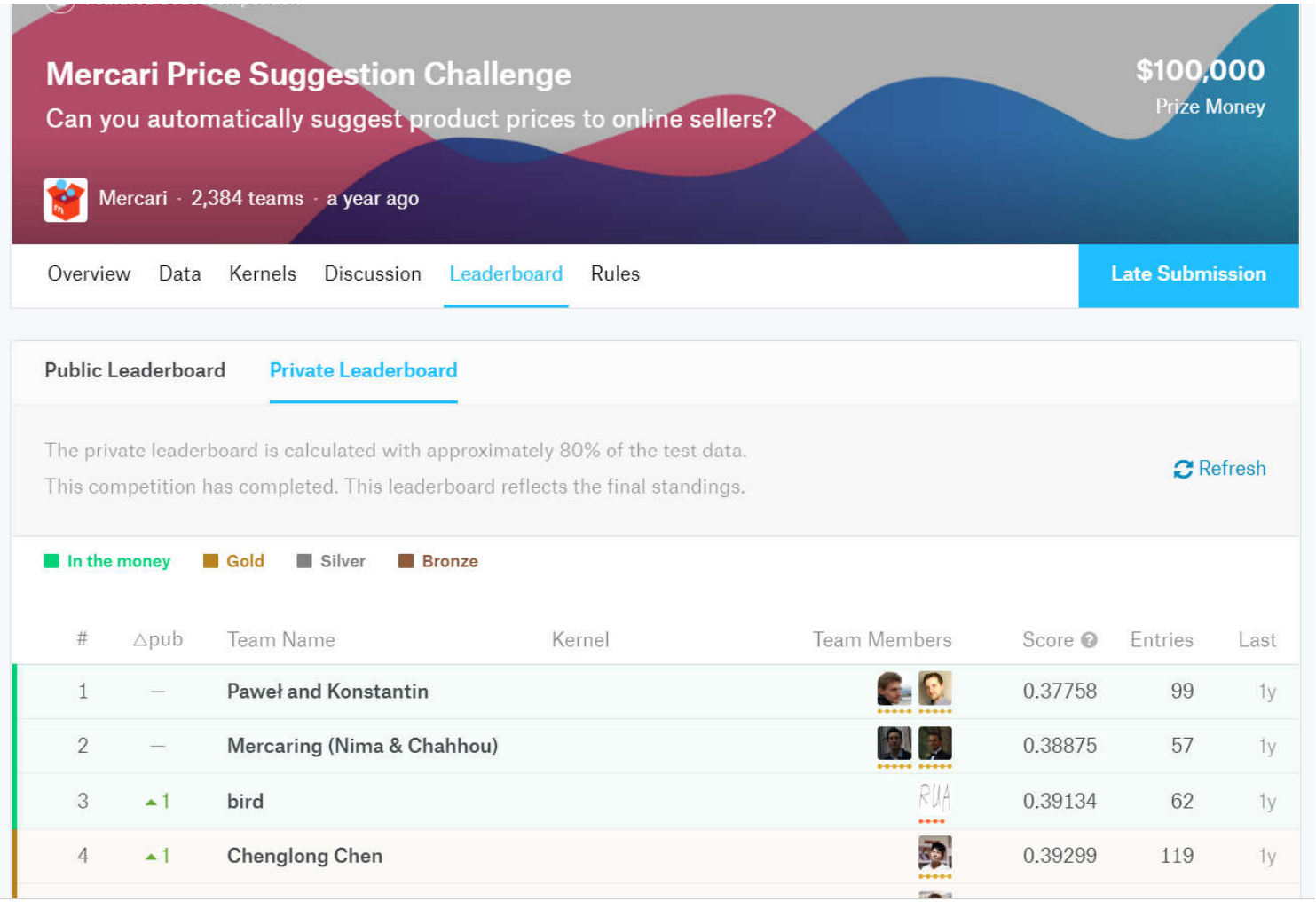
Raw Data Refresh

In the money Gold Silver Bronze

#	Team Name	Kernel	Team Members	Score	Entries	Last
1	**Kraków, Lublin i Zhabinka**			0.81724	329	10mo
2	**ikiri_DS**		+9	0.81241	477	10mo
3	**circlecircle**			0.81124	231	10mo
4	**alijs & Evgeny**			0.81086	143	10mo



https://www.kaggle.com/c/home-credit-default-risk/discussion/64722



https://www.kaggle.com/c/mercari-price-suggestion-challenge



NWT JUICY COUTURE JACKET XS

California • 03/23/2018 06:45 PM • Report item

$ 45.00 ♡ 0

Sign up now and buy at $ 40.00

Buy now

Condition	Like new
Size	XS (0-2)
Shipping	$7.00
Brand	Juicy Couture
Category	Women Athletic apparel Jackets

Description

Brand new!! JUICY COUTURE JACKET XS!!!

```
merge_predictions =
-0.0203
+0.0604 * data1_huber
+0.1051 * data1_huber
+0.0911 * data1_clf
+0.0760 * data1_clf
+0.0851 * data2_huber_bin
+0.0981 * data2_huber
+0.0819 * data2_clf_bin
+0.0717 * data2_clf
+0.0958 * data3_huber_bin
+0.1226 * data3_huber
+0.0578 * data3_clf_bin
+0.0642 * data3_clf
⇒ RMSLE 0.3733
```

Why Ensemble Works?

- Uncorrelated error reduction
  - Suppose we have 5 completely independent classifiers for majority voting
  - If accuracy is 70% for each
    - 10 (.7^3)(.3^2)+5(.7^4)(.3)+(.7^5)
    - **83.7% majority vote accuracy**
  - 101 such classifiers
    - **99.9% majority vote accuracy**

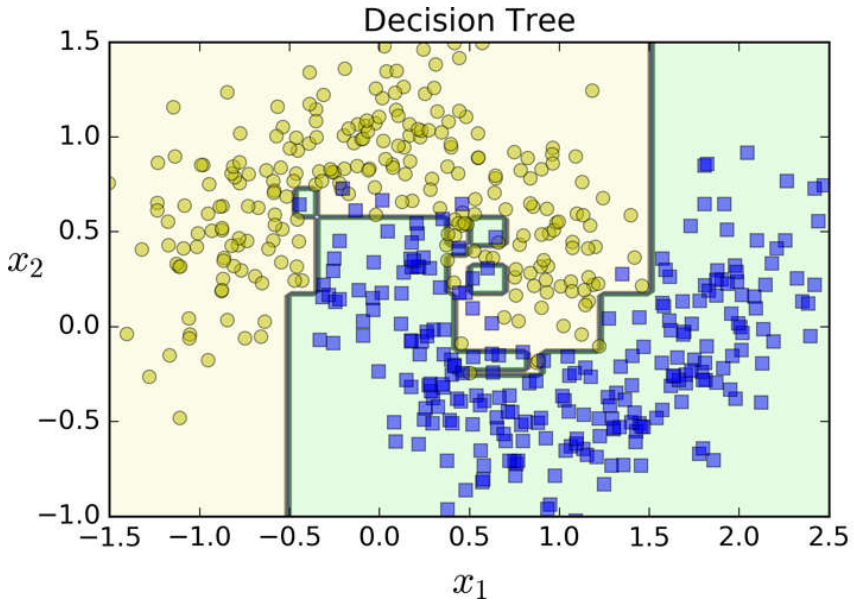
Note: Binomial Distribution: The probability of observing *x* heads in a sample of *n* independent coin tosses, where in each toss the probability of heads is *p*, is

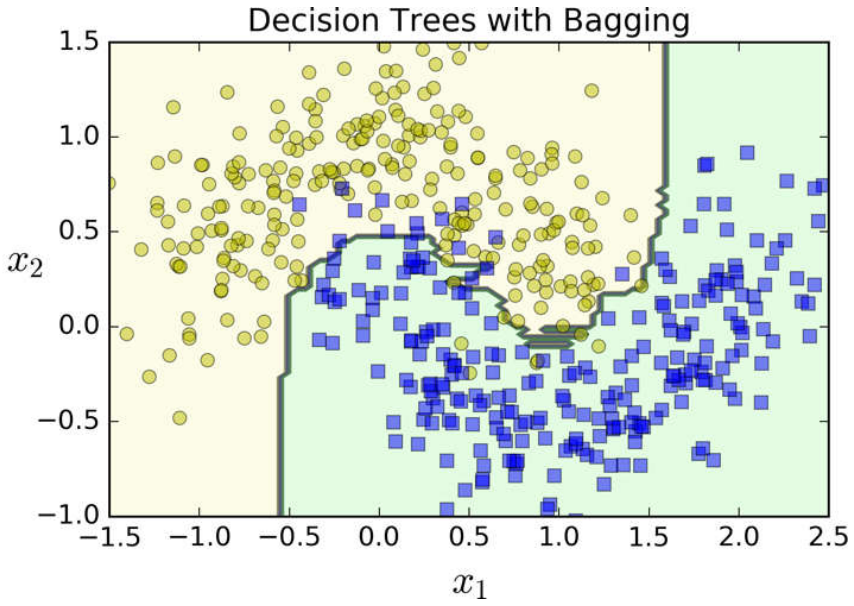
$$P(X = x|p, n) = \frac{n!}{r!(n-x)!}p^x(1-p)^{n-x}$$

from T. Holloway, Introduction to Ensemble Learning, 2007.

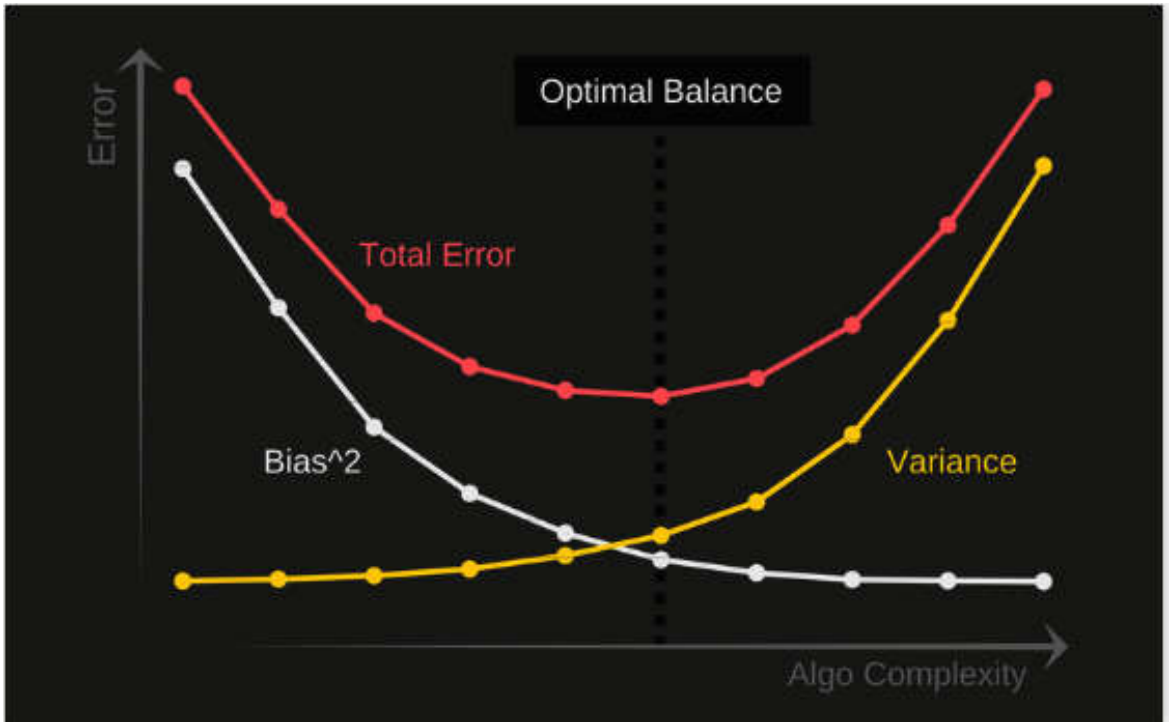
Why Ensemble Works?

- Overcome limitations of single hypothesis
  - The target function may not be implementable with individual classifiers, but may be approximated by model averaging



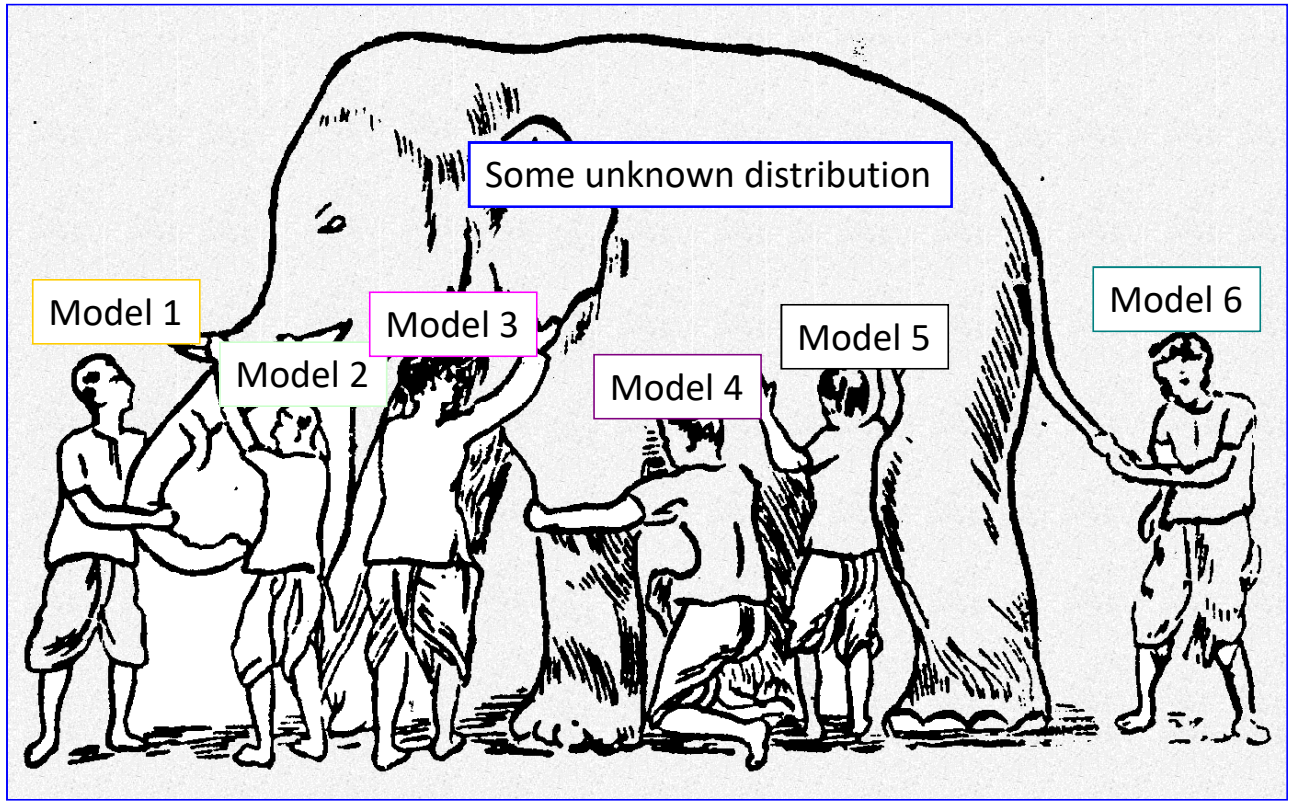


Why Ensemble Works?



Total Error = Bias^2 + Variance + Irreducible Error

Why Ensemble Works?



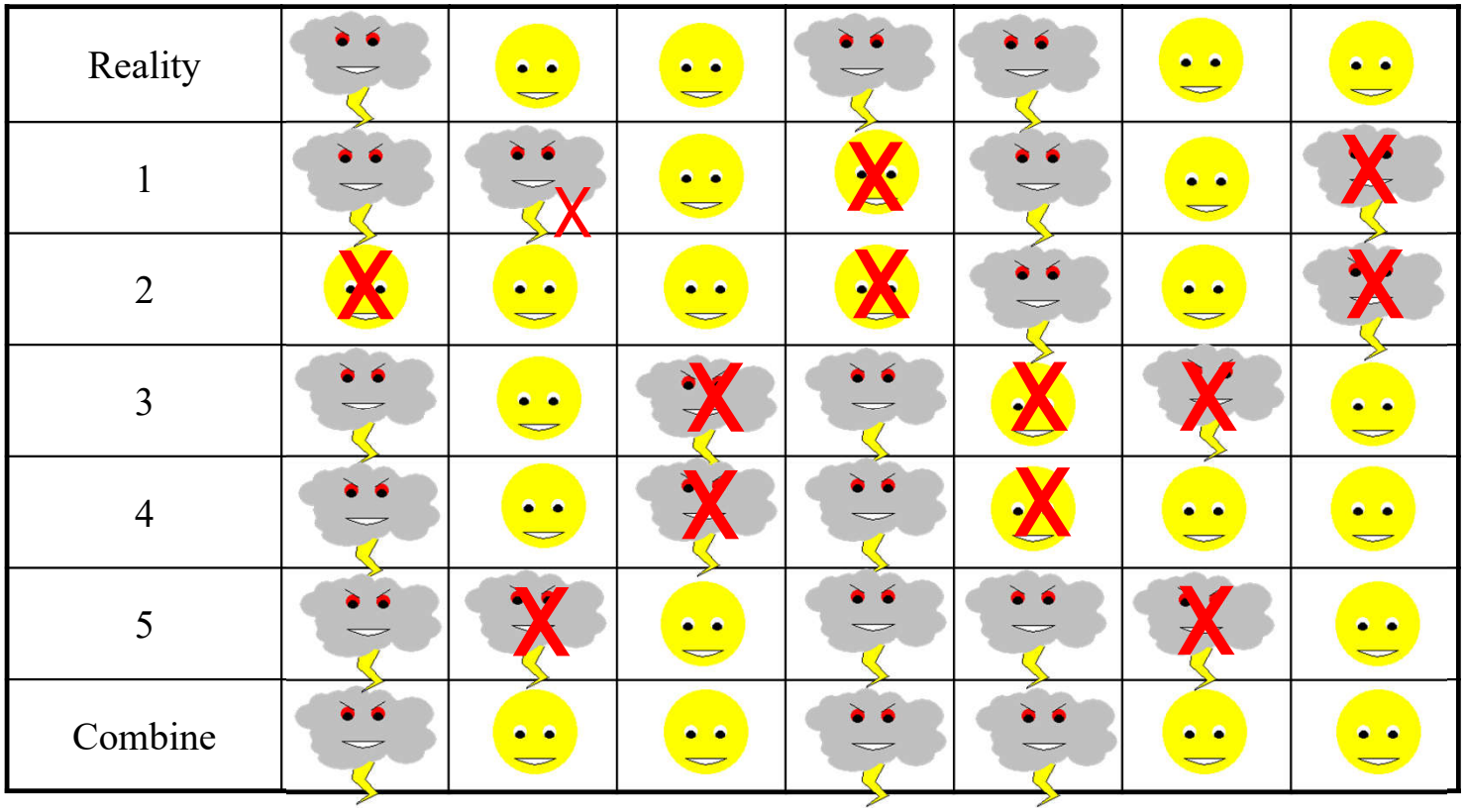
Ensemble gives the global picture!

http://ews.uiuc.edu/~jinggao3/sdm10ensemble.htm

Basic Ensemble Techniques

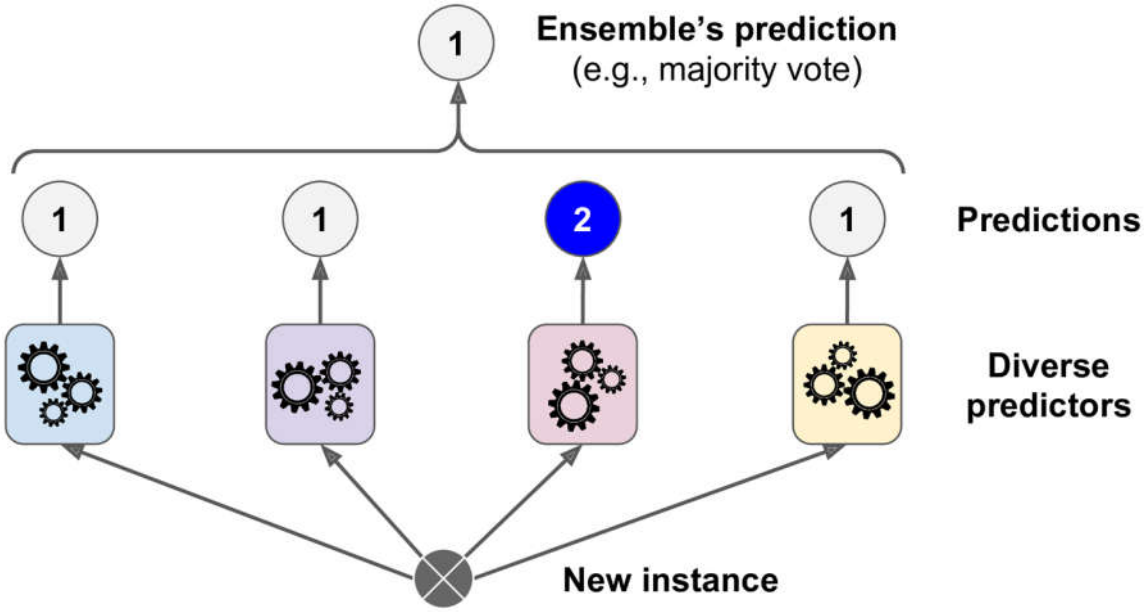
- **Max Voting**

Example: Weather Forecast



Basic Ensemble Techniques

- **Max Voting**



Hard voting classifier predictions

Basic Ensemble Techniques

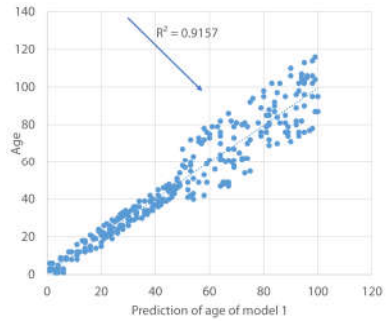
- **Averaging**

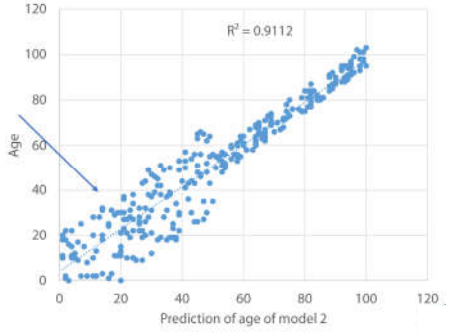
Colleague 1	Colleague 2	Colleague 3	Colleague 4	Colleague 5	**Final rating**
4	5	5	4	4	4.4

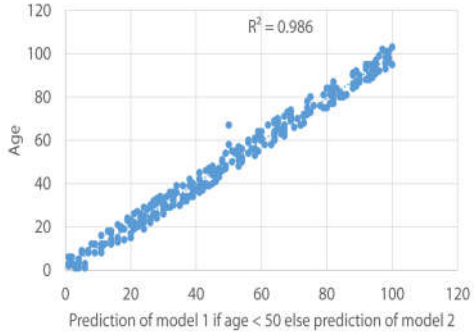
- **Weighted Averaging**

	Colleague 1	Colleague 2	Colleague 3	Colleague 4	Colleague 5	**Final rating**
Weight	0.23	0.23	0.18	0.18	0.18	
Rating	4	5	5	4	4	4.41

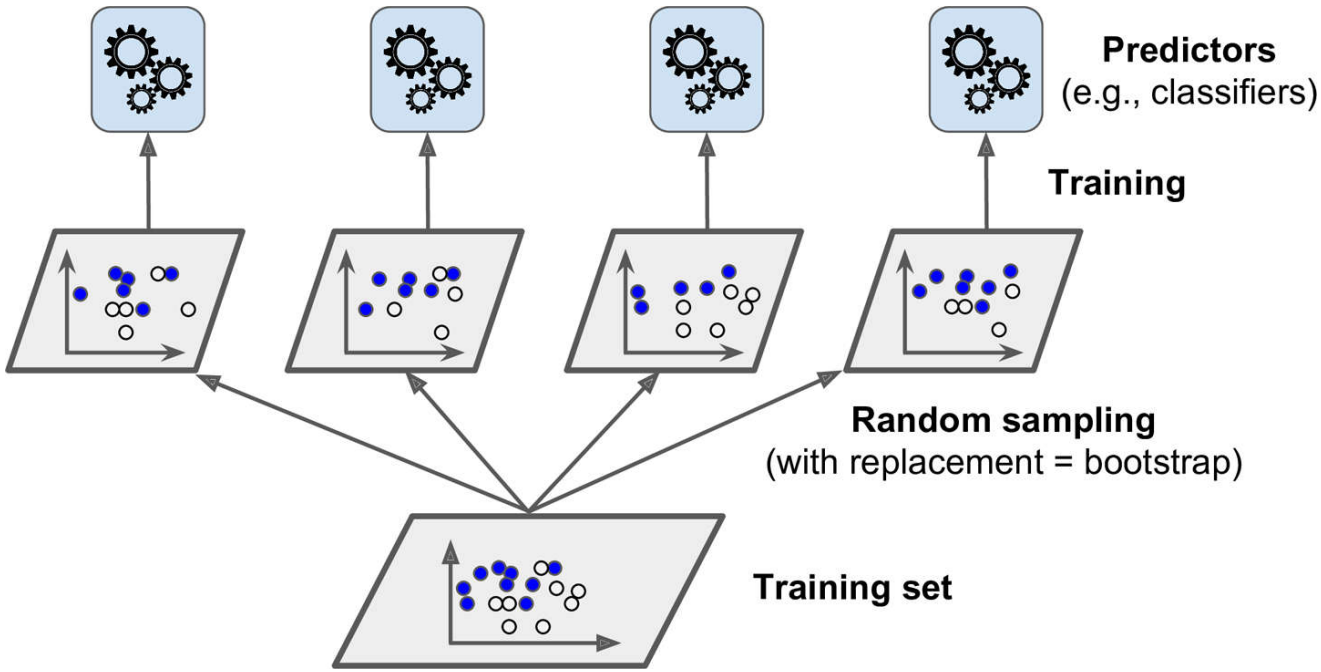
- **Conditional averaging**







Bagging and Pasting



Pasting/bagging training set sampling and training

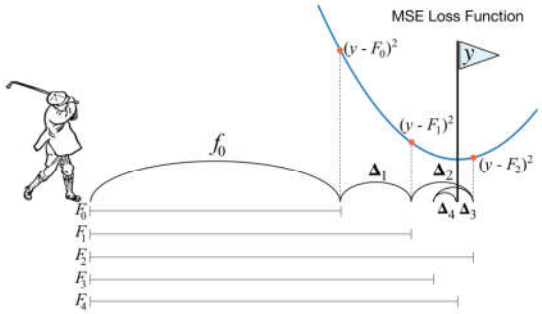
Parameters that control bagging

- Changing the seed
- Row (Sub) sampling or Bootstrapping
- Shuffling
- Column (Sub) sampling
- Model-specific parameters
- Number of models (or bags)
- (Optionally) parallelism

Random forest

Boosting

- A form of weighted averaging of models where each model is built sequentially via taking into account the past model performance. (Kazanova)



Boosting- weight based

- Dataset

Rownum	x0	x1	x2	x3	y
0	0.94	0.27	0.80	0.34	1
1	0.84	0.79	0.89	0.05	1
2	0.83	0.11	0.23	0.42	1
3	0.74	0.26	0.03	0.41	0
4	0.08	0.29	0.76	0.37	0
5	0.71	0.76	0.43	0.95	1
6	0.08	0.72	0.97	0.04	0

Boosting- weight based

- Step 1: Prediction

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

Boosting- weight based

- Step 2: get absolute error

Rownum	x0	x1	x2	x3	y	pred	abs.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

Boosting- weight based

- Step 3: Add weight

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

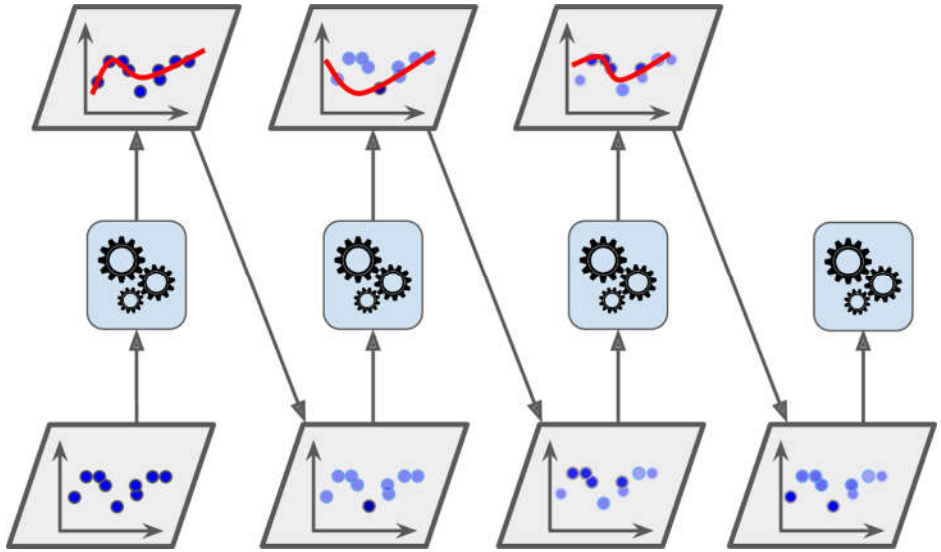
Boosting- weight based

- Step 4: Train the next model with new weights

Rownum	x0	x1	x2	x3	y	weight
0	0.94	0.27	0.80	0.34	1	1.20
1	0.84	0.79	0.89	0.05	1	1.25
2	0.83	0.11	0.23	0.42	1	1.35
3	0.74	0.26	0.03	0.41	0	1.40
4	0.08	0.29	0.76	0.37	0	1.55
5	0.71	0.76	0.43	0.95	1	1.66
6	0.08	0.72	0.97	0.04	0	1.02

Boosting- weight based

$$h(x) = a_1h_1(x) + a_2h_2(x) \quad + \ldots + a_nh_n(x)$$



AdaBoost sequential training with instance weight updates

Given: $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in \mathcal{X}$, $y_i \in \{-1, +1\}$.

Initialize $D_1(i) = 1/m$ for $i = 1, \ldots, m$. ← Initial Distribution of Data

For $t = 1, \ldots, T$:

- Train weak learner using distribution D_t.
- Get weak hypothesis $h_t : \mathcal{X} \to \{-1, +1\}$. ← Train model
- Aim: select h_t with low weighted error:

$$\varepsilon_t = \Pr_{i \sim D_t}[h_t(x_i) \neq y_i].$$ ← Error of model

- Choose $\alpha_t = \frac{1}{2} \ln\left(\frac{1 - \varepsilon_t}{\varepsilon_t}\right)$. ← Coefficient of model
- Update, for $i = 1, \ldots, m$:

$$D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$$ ← Update Distribution

where Z_t is a normalization factor (chosen so that D_{t+1} will be a distribution).

Output the final hypothesis:

$$H(x) = \operatorname{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x)\right).$$ ← Final average

Boosting – Residual based

Rownum	x0	x1	x2	x3	y
0	0.94	0.27	0.80	0.34	1
1	0.84	0.79	0.89	0.05	1
2	0.83	0.11	0.23	0.42	1
3	0.74	0.26	0.03	0.41	0
4	0.08	0.29	0.76	0.37	0
5	0.71	0.76	0.43	0.95	1
6	0.08	0.72	0.97	0.04	0

Boosting – Residual based

- Step 1: Get model 1 predictions

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

Boosting – Residual based

- Step 2: Get residual

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

Boosting – Residual based

- Step 3: Update new y for next model

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

Boosting – Residual based

- Step 4: New predictions

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

Boosting – Residual based

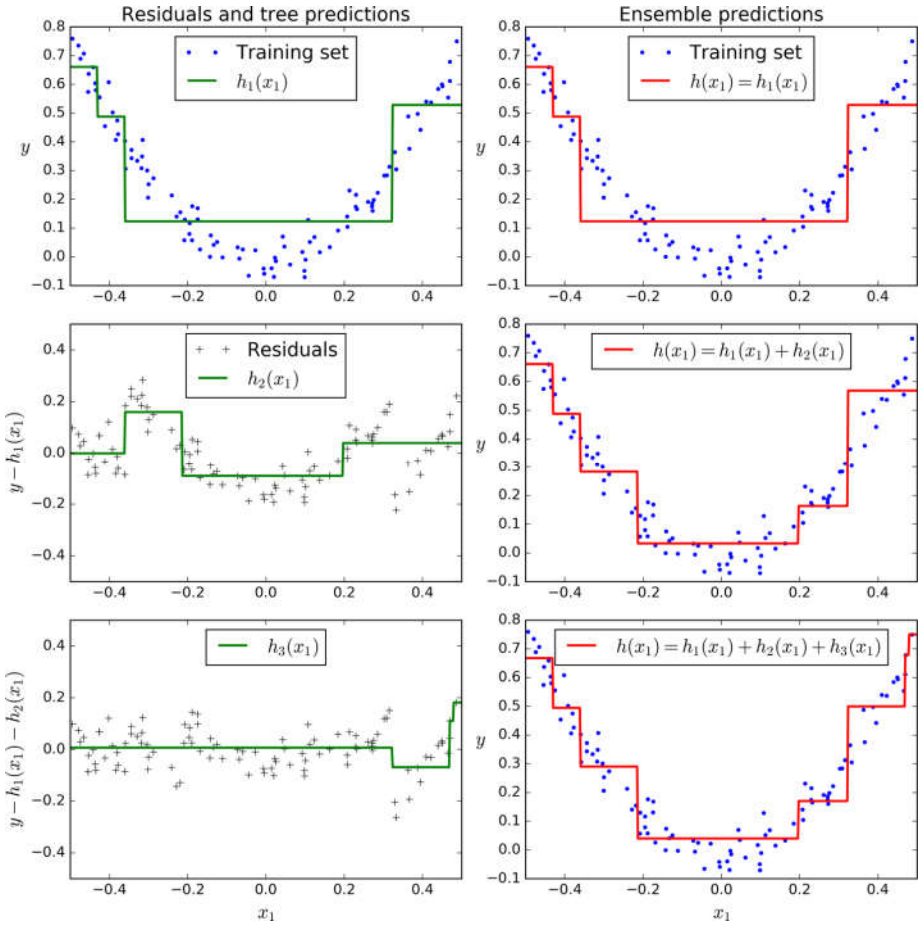
- Reviewed

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 row number 1: final prediction = 0.75 + 0.20 = 0.95

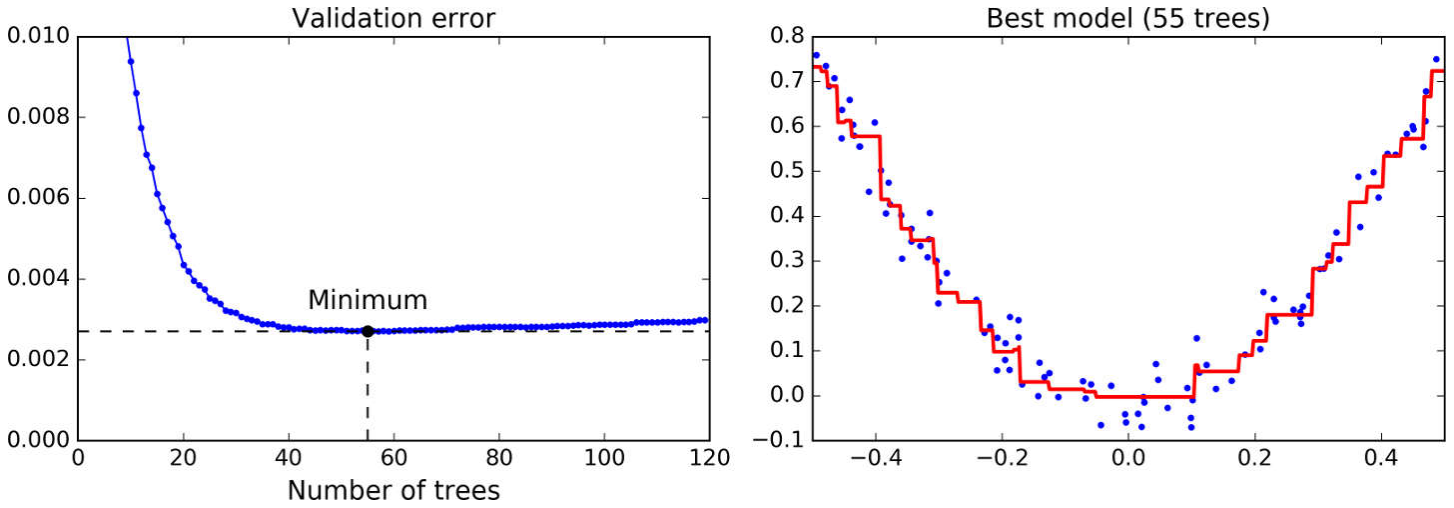
Boosting – Residual based

- Gradient boosting



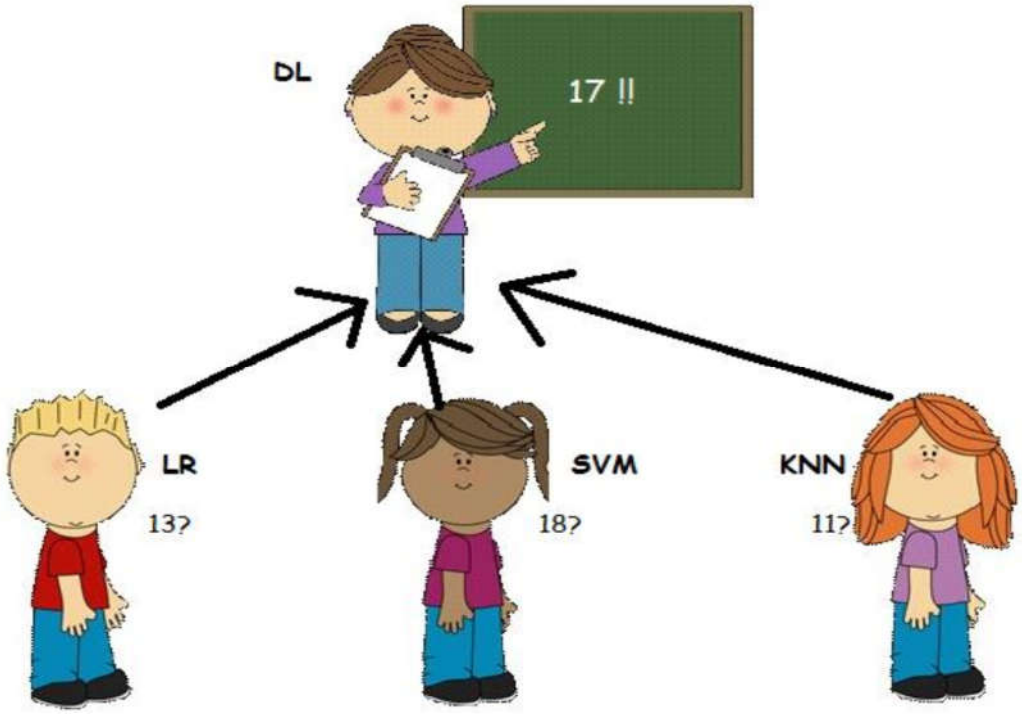
Boosting – Residual based

- Gradient boosting

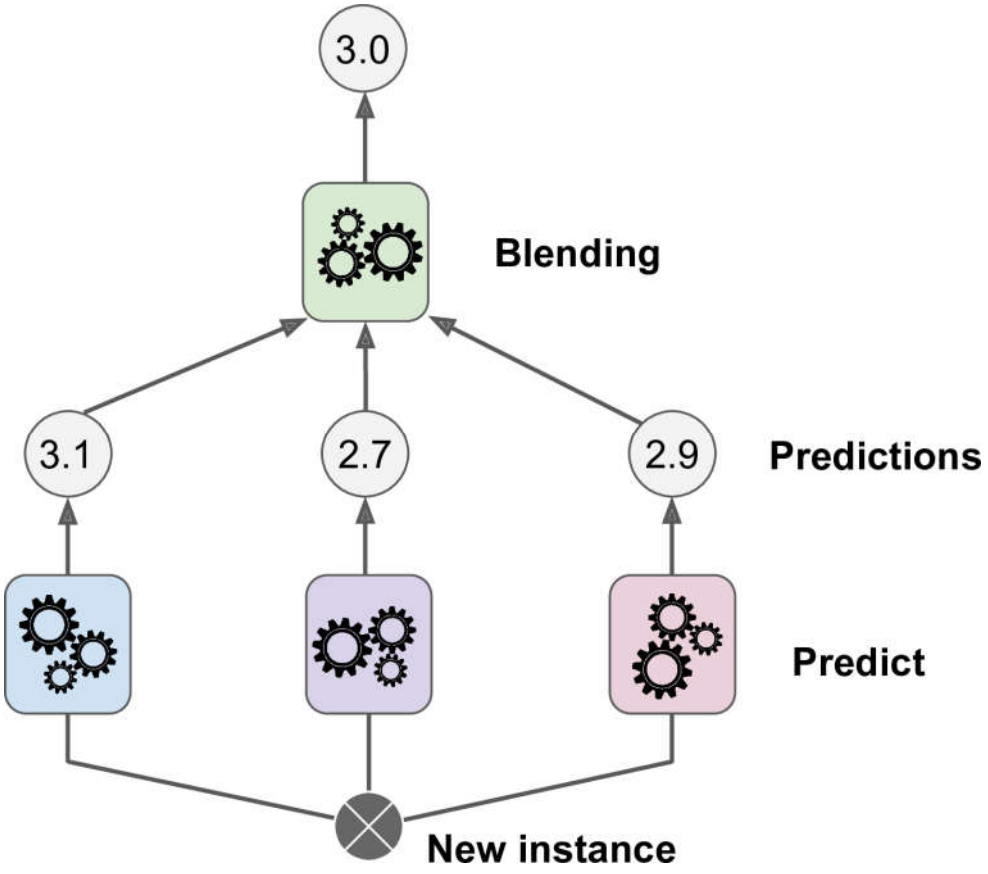


Tuning the number of trees

Stacking

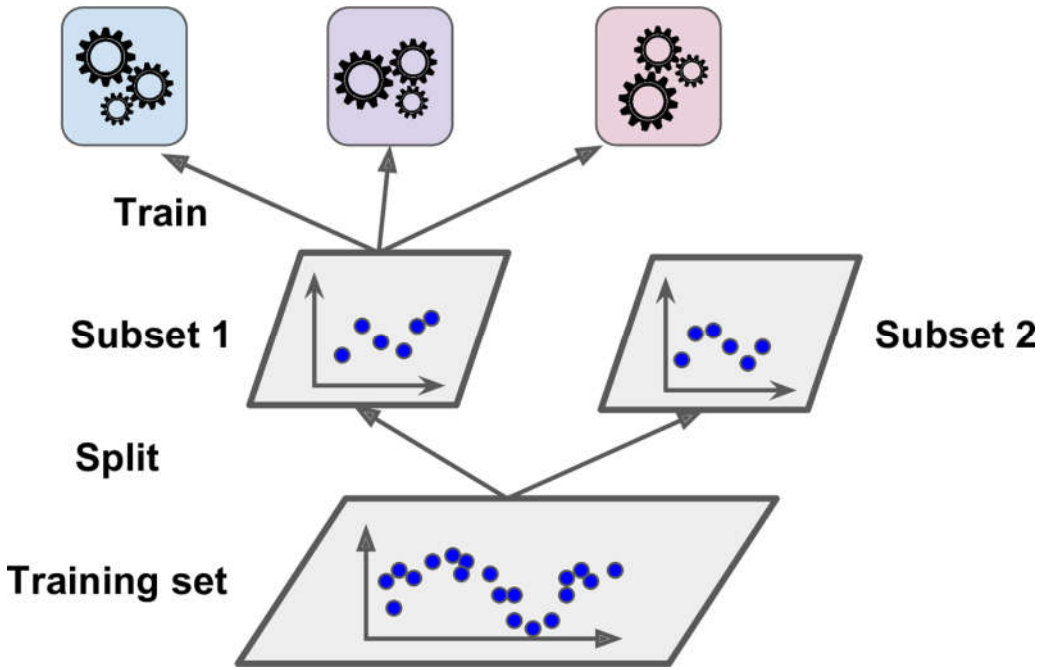


Stacking



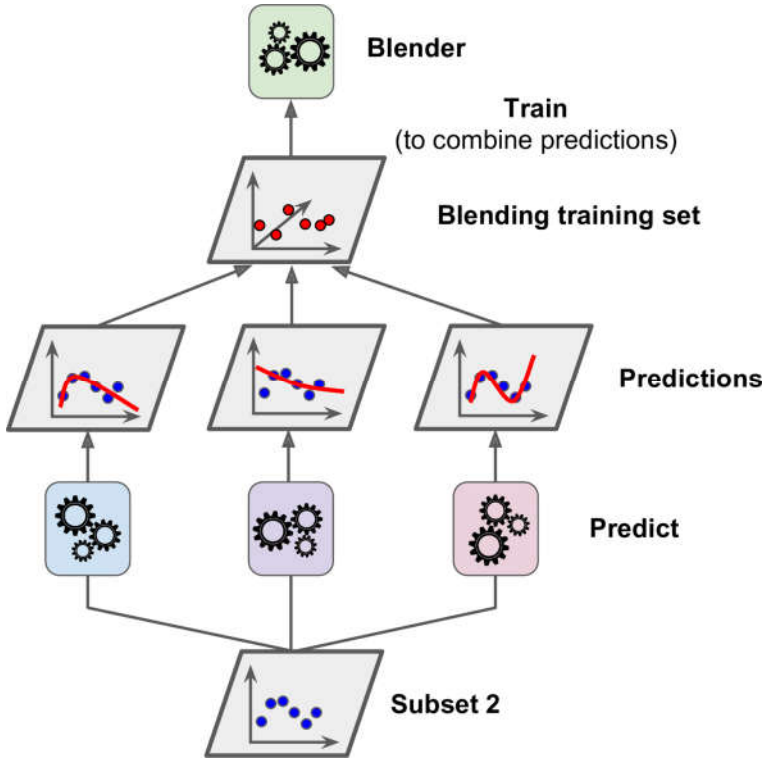
ggregating predictions using a blending predictor

Stacking



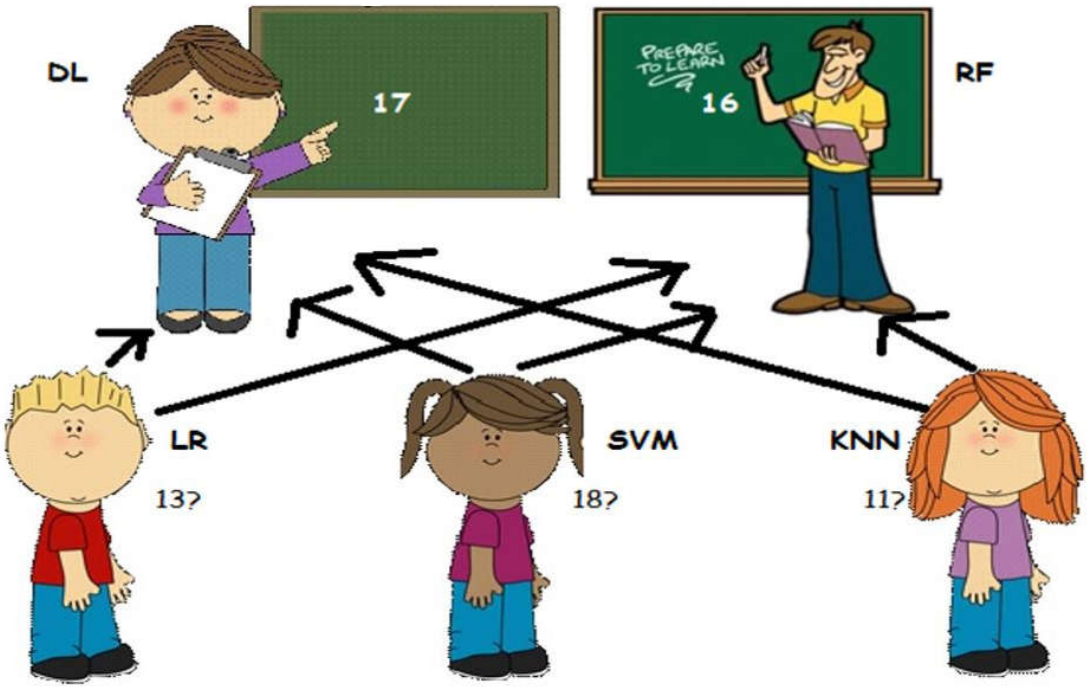
Training the first layer

Stacking

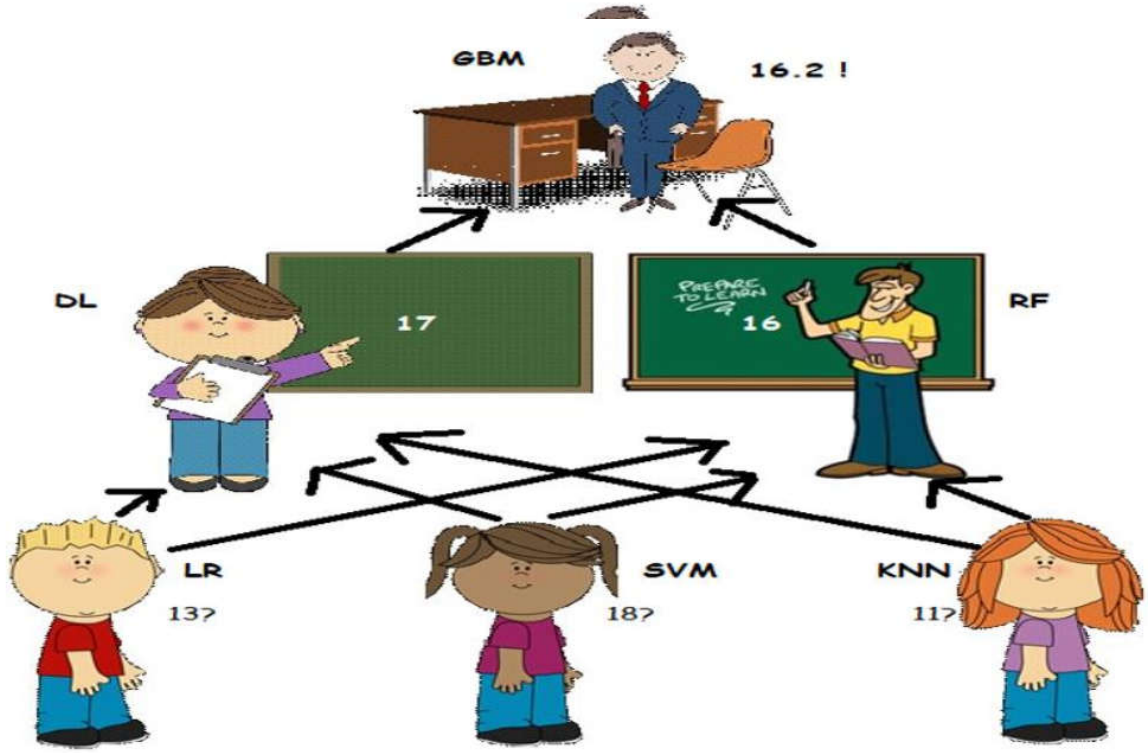


Training the blender

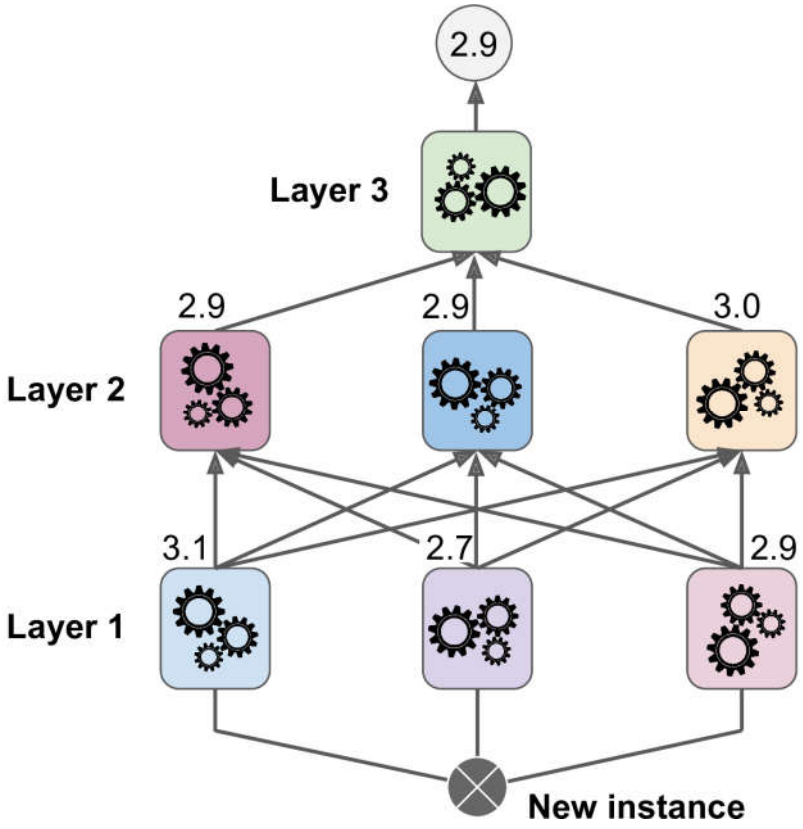
Stacking



Stacking



Stacking



Predictions in a multilayer stacking ensemble

How come?

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**

B1

pred0
0.24
0.95
0.64
0.89
0.11

C1

pred0
0.50
0.62
0.22
0.90
0.20

Step 2: for next model

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**

B1

pred0	pred1
0.24	0.72
0.95	0.25
0.64	0.80
0.89	0.58
0.11	0.20

C1

pred0	pred1
0.50	0.50
0.62	0.59
0.22	0.31
0.90	0.47
0.20	0.09

Step 3: for model 2

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
0.50	0.50	0.39	?
0.62	0.59	0.46	?
0.22	0.31	0.54	?
0.90	0.47	0.09	?
0.20	0.09	0.61	?

Step 4: train meta model

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

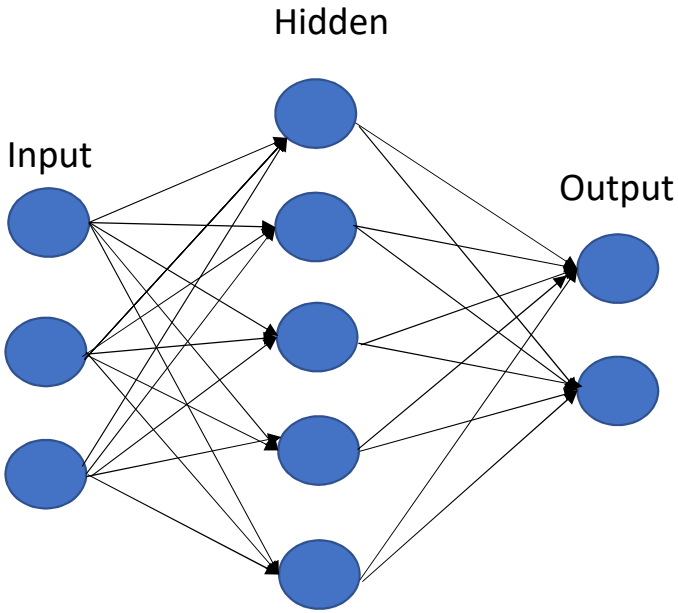
C1				
pred0	pred1	pred2	y	Preds3
0.50	0.50	0.39	?	0.45
0.62	0.59	0.46	?	0.23
0.22	0.31	0.54	?	0.99
0.90	0.47	0.09	?	0.34
0.20	0.09	0.61	?	0.05

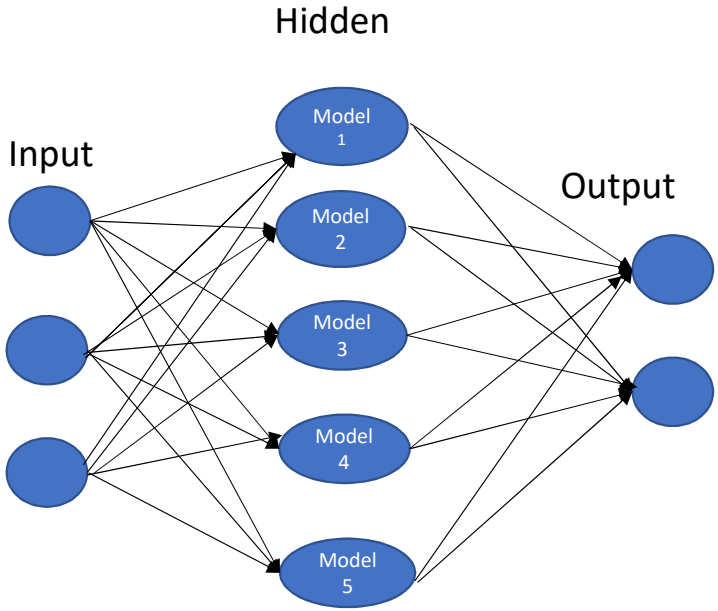
Train algorithm **3** on B1 and make predictions for C1

StackNET

- **StackNET** is a scalable meta modelling methodology that utilizes stacking to combine multiple models in a neural network architecture of multiple levels.

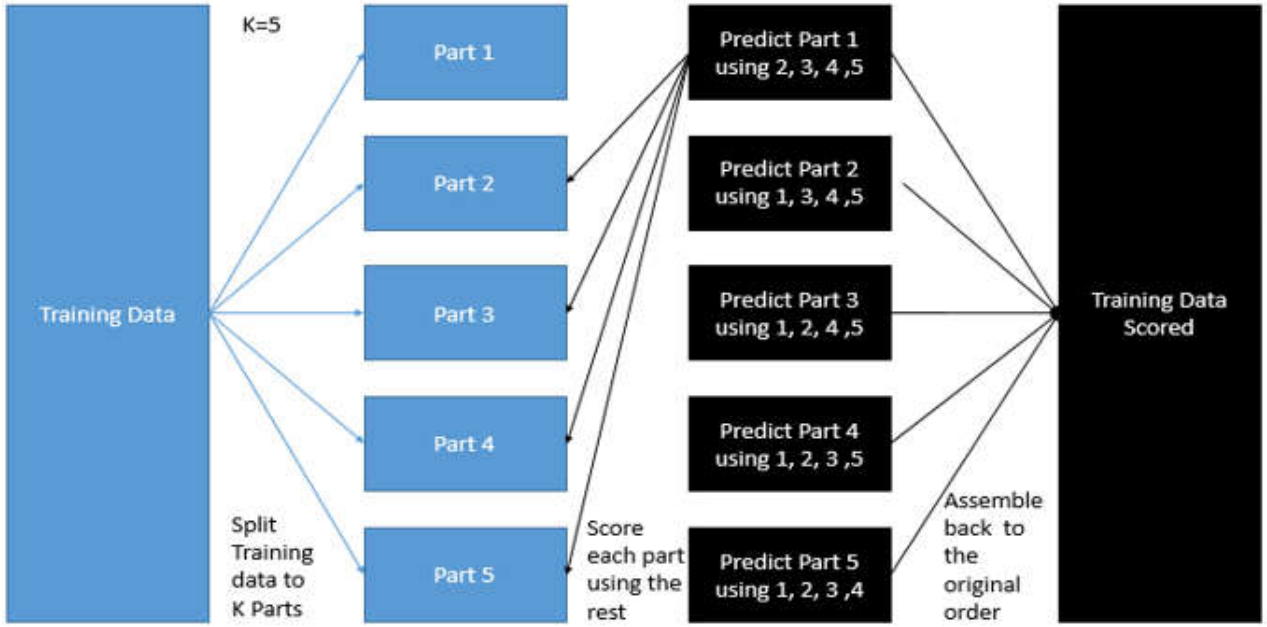
Neural network VS StackNet





How to train

- Cannot use **BP** (not all model differentiable)
- Use **Kfold** paradigm



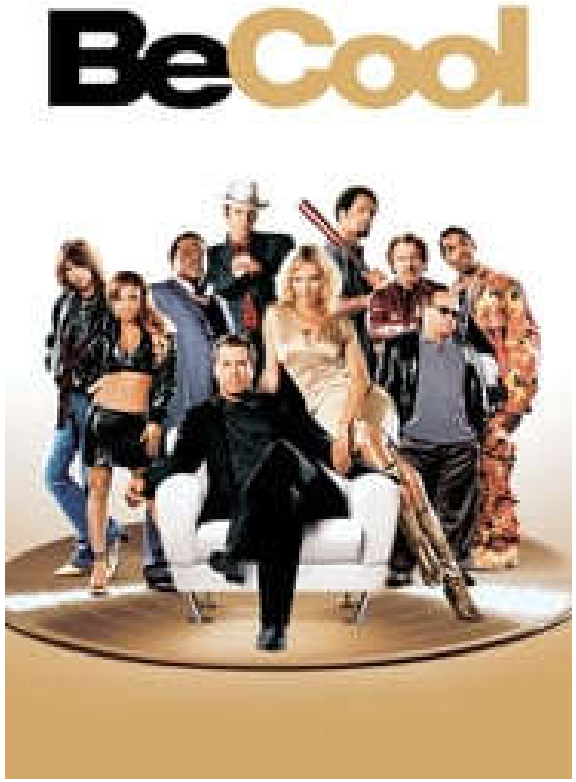
TMDB Box Office Prediction

Can you predict a movie's worldwide box office revenue?

7398 movies and a variety of metadata obtained from The Movie Database (TMDB)

- belongs_to_collection
- budget
- genres
- Homepage
- imdb_id
- original_language
- original_title
- overview
- popularity
- poster_path
- production_companies
- production_countries
- release_date
- runtime
- spoken_languages
- Status
- tagline
- Title
- Keywords
- Cast
- Crew
- revenue

- Overview: “Disenchanted with the movie industry, Chili Palmer tries the music industry, meeting and romancing a widow of a music exec on the way..”
- Tagline: “Everyone is looking for the next big hit”
- [{'cast_id': 4, 'character': 'Chili Palmer', 'credit_id': '52fe43cbc3a36847f8070361', 'gender': 2, 'id': 8891, 'name': 'John Travolta', 'order': 0, 'profile_path': '/ns8uZHEHzV18ifqA9secv8c2Ard.jpg'}, {'cast_id': 5, 'character': 'Edie Athens', 'credit_id': '52fe43cbc3a36847f8070365', 'gender': 1, 'id': 139, 'name': 'Uma Thurman', 'order': 1, 'profile_path': '/6SuOc2R7kXjq3Em24KTNDW9qblJ.jpg’}, ……
- [{'credit_id': '52fe43cbc3a36847f807039f', 'department': 'Production', 'gender': 2, 'id': 518, 'job': 'Producer', 'name': 'Danny DeVito', 'profile_path': '/zKuyzmKzPLG7RJo7lbbHjx6CCZc.jpg'}, {'credit_id': '55f51e369251415ca000081c', 'department': 'Camera', 'gender': 0, 'id': 904, 'job': 'Director of Photography', 'name': 'Jeffrey L. Kimball', 'profile_path': None},

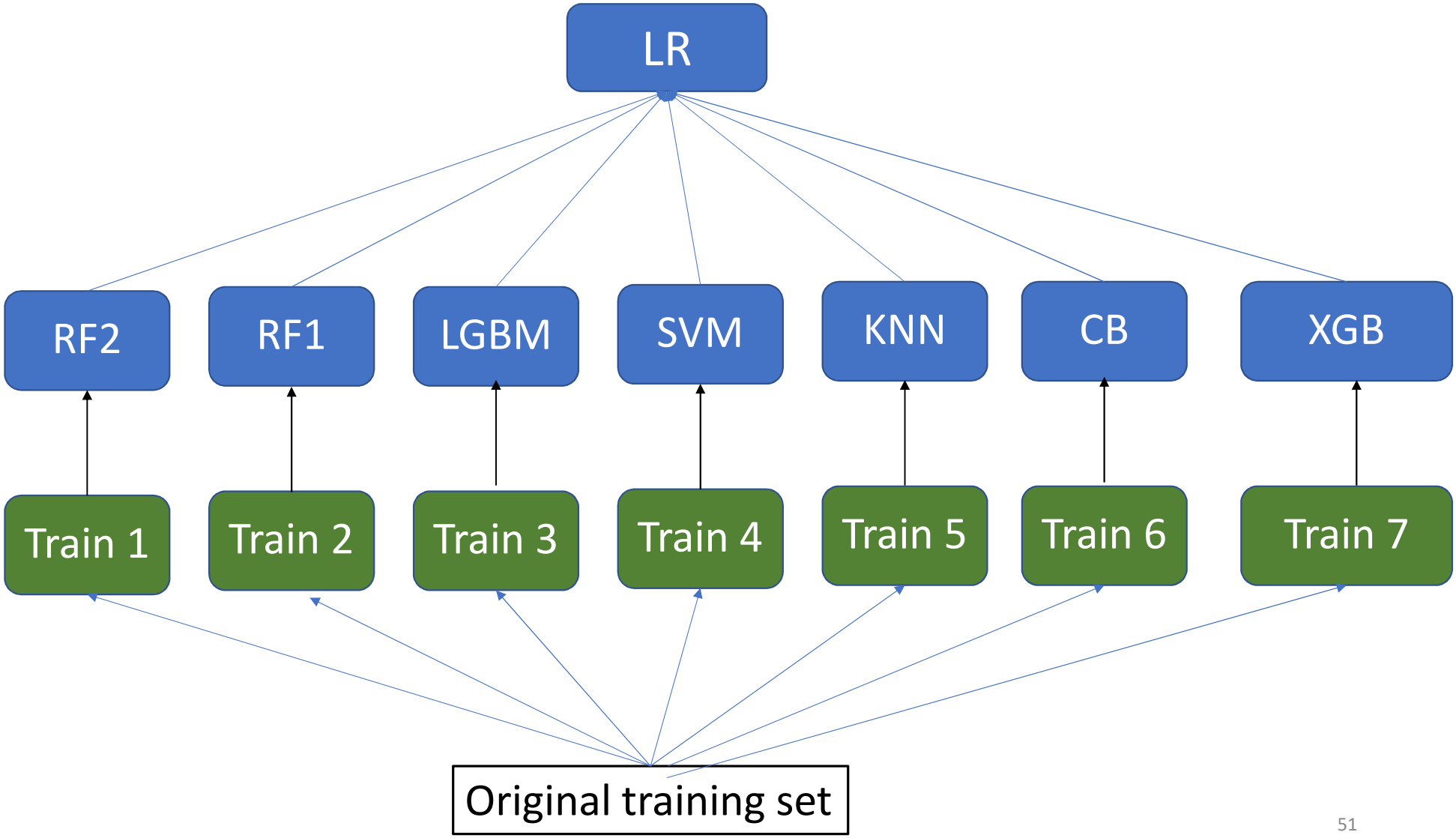


TMDB Box Office Prediction

- Evaluation

Root-Mean-Squared-Logarithmic-Error (RMSLE)

$$\epsilon = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\log(p_i + 1) - \log(a_i + 1))^2}$$


LR
RF2
RF1
LGBM
SVM
KNN
CB
XGB
Train 1
Train 2
Train 3
Train 4
Train 5
Train 6
Train 7
Original training set

References

- https://www.analyticsvidhya.com/blog/2018/06/comprehensive-guide-for-ensemble-models/
- https://www.coursera.org/learn/competitive-data-science/
- https://github.com/kaz-Anova/StackNet
- http://ews.uiuc.edu/~jinggao3/sdm10ensemble.htm
- https://www.cs.princeton.edu/~schapire/papers/explaining-adaboost.pdf
- Géron, Aurélien. *Hands-On Machine Learning with Scikit-Learn and TensorFlow : Concepts, Tools, and Techniques to Build Intelligent Systems*. Sebastopol: O'Reilly Media, Incorporated, 2017. Web.

Thank you for listening