Lecture 4: Ensemble Learning

Pilsung Kang

School of Industrial Management Engineering

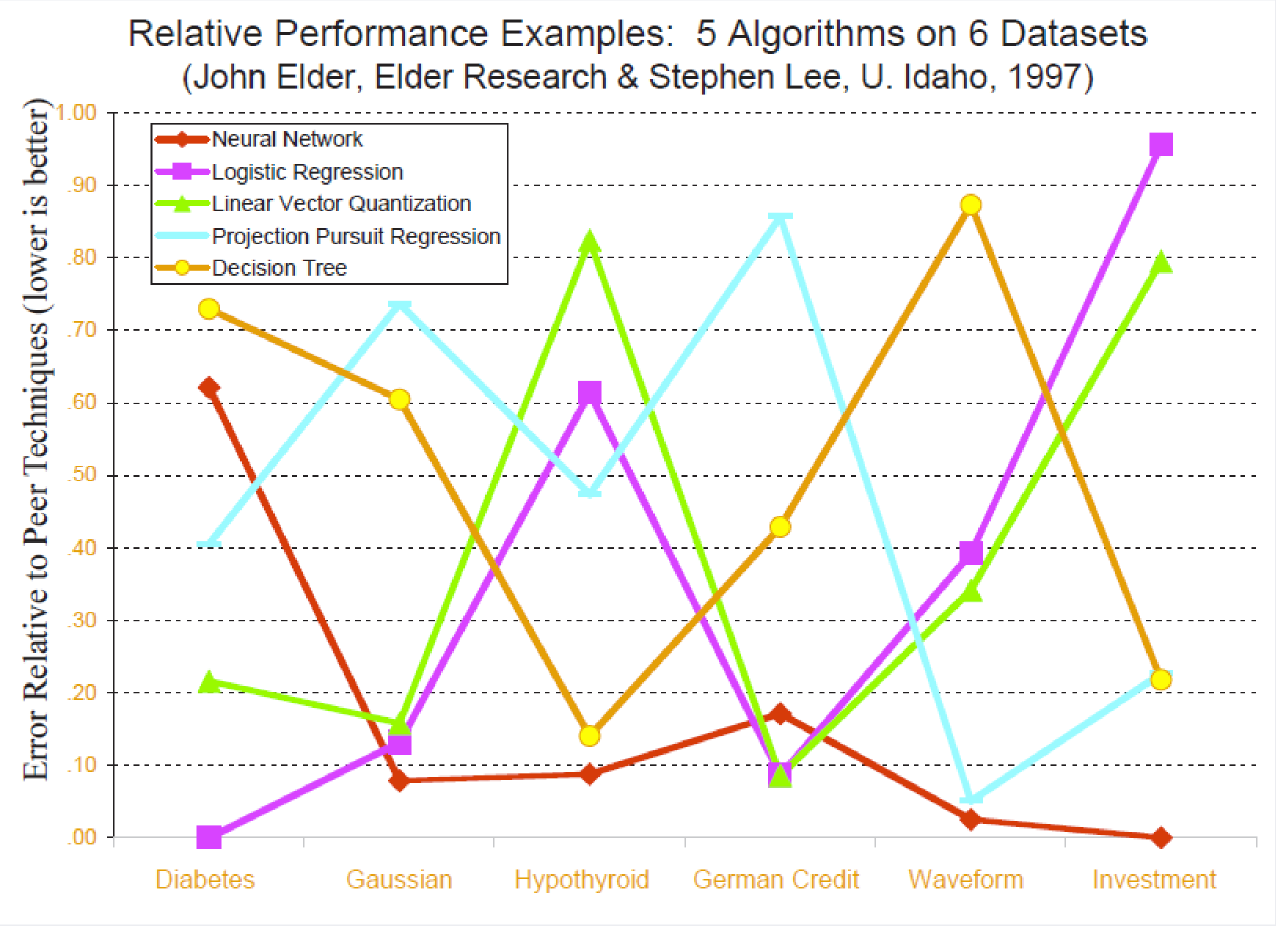
Korea University

AGENDA

Backgrounds

Seni and Elder (2010)

- Can we have a superior algorithm for all datasets?
  - ✓ Every algorithm scored best or next-to-best on at least two of the six data sets.

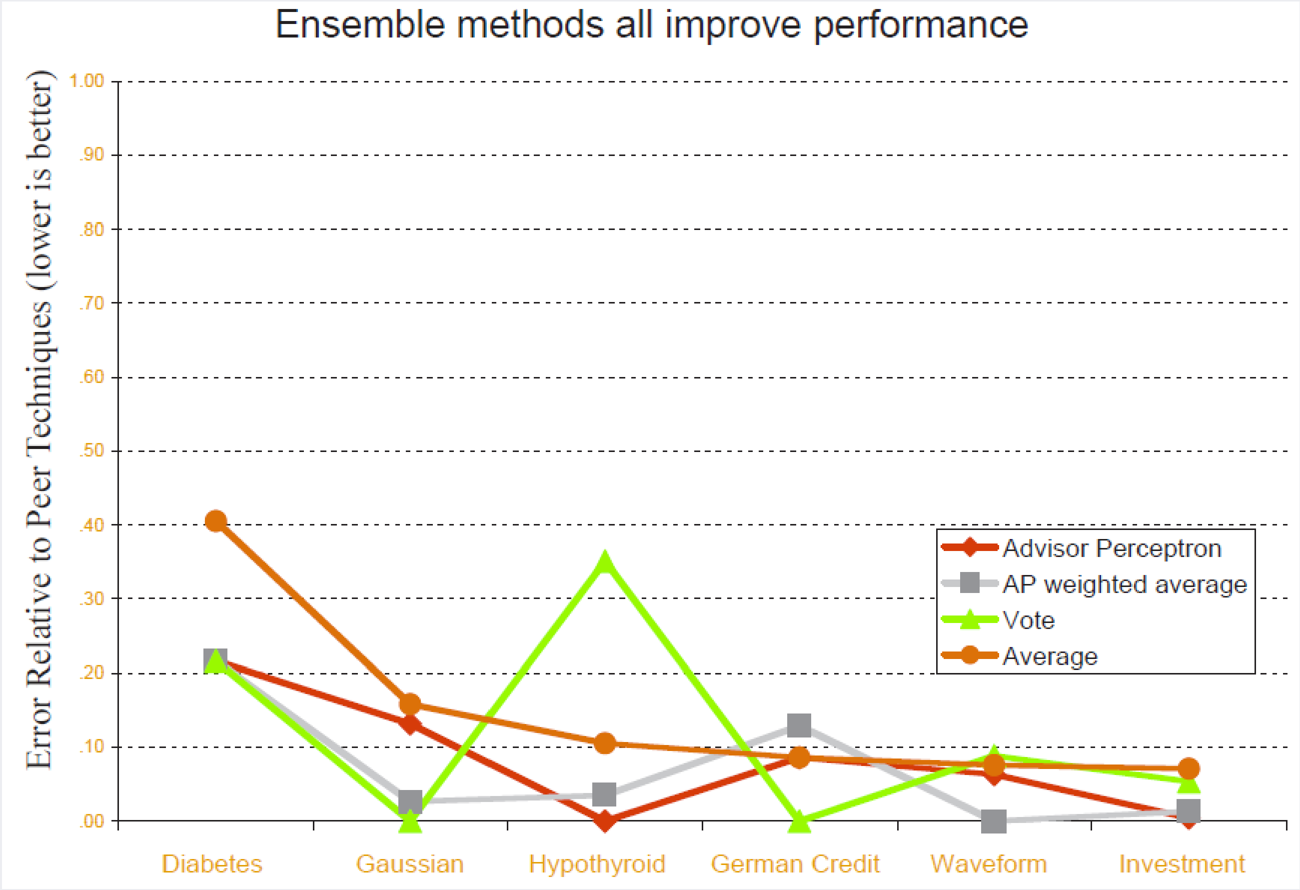


Backgrounds

- No Free Lunch Theorem
  - ✓ Can we expect any classification method to be superior or inferior overall?
  - ✓ No Free Lunch Theorem: No
  - ✓ If the goal is to obtain good generalization performance, there is no context-independent or usage-independent reasons to favor one algorithm over others
  - ✓ If one algorithm seems to outperform another in a particular situation, it is a consequence of its fit to a particular pattern recognition problem
  - ✓ In practice, experience with a broad range of techniques is the best insurance for solving arbitrary new classification problems

Motivation

- However, if they are properly combined…
  - ✓ Every ensemble method competes well against the best of the individual algorithms



Empirical Evidence

Opitz and Maclin (1999)

- Empirical study 1: Single vs. Ensemble algorithms for 23 datasets

Data Set	Cases	Class	Features Cont	Features Disc	Neural Network Inputs	Neural Network Outputs	Neural Network Hiddens	Neural Network Epochs
breast-cancer-w	699	2	9	-	9	1	5	20
credit-a	690	2	6	9	47	1	10	35
credit-g	1000	2	7	13	63	1	10	30
diabetes	768	2	9	-	8	1	5	30
glass	214	6	9	-	9	6	10	80
heart-cleveland	303	2	8	5	13	1	5	40
hepatitis	155	2	6	13	32	1	10	60
house-votes-84	435	2	-	16	16	1	5	40
hypo	3772	5	7	22	55	5	15	40
ionosphere	351	2	34	-	34	1	10	40
iris	159	3	4	-	4	3	5	80
kr-vs-kp	3196	2	-	36	74	1	15	20
labor	57	2	8	8	29	1	10	80
letter	20000	26	16	-	16	26	40	30
promoters-936	936	2	-	57	228	1	20	30
ribosome-bind	1877	2	-	49	196	1	20	35
satellite	6435	6	36	-	36	6	15	30
segmentation	2310	7	19	-	19	7	15	20
sick	3772	2	7	22	55	1	10	40
sonar	208	2	60	-	60	1	10	60
soybean	683	19	-	35	134	19	25	40
splice	3190	3	-	60	240	2	25	30
vehicle	846	4	18	-	18	4	10	40

Empirical Evidence

- Empirical study 1: Single vs. Ensemble algorithms for 23 datasets
  - ✓ Error rate: the lower, the better

	Neural Network					C4.5			
				Boosting				Boosting	
Data Set	Stan	Simp	Bag	Arc	Ada	Stan	Bag	Arc	Ada
breast-cancer-w	3.4	3.5	3.4	3.8	4.0	5.0	3.7	3.5	3.5
credit-a	14.8	13.7	13.8	15.8	15.7	14.9	13.4	14.0	13.7
credit-g	27.9	24.7	24.2	25.2	25.3	29.6	25.2	25.9	26.7
diabetes	23.9	23.0	22.8	24.4	23.3	27.8	24.4	26.0	25.7
glass	38.6	35.2	33.1	32.0	31.1	31.3	25.8	25.5	23.3
heart-cleveland	18.6	17.4	17.0	20.7	21.1	24.3	19.5	21.5	20.8
hepatitis	20.1	19.5	17.8	19.0	19.7	21.2	17.3	16.9	17.2
house-votes-84	4.9	4.8	4.1	5.1	5.3	3.6	3.6	5.0	4.8
hypo	6.4	6.2	6.2	6.2	6.2	0.5	0.4	0.4	0.4
ionosphere	9.7	7.5	9.2	7.6	8.3	8.1	6.4	6.0	6.1
iris	4.3	3.9	4.0	3.7	3.9	5.2	4.9	5.1	5.6
kr-vs-kp	2.3	0.8	0.8	0.4	0.3	0.6	0.6	0.3	0.4
labor	6.1	3.2	4.2	3.2	3.2	16.5	13.7	13.0	11.6
letter	18.0	12.8	10.5	5.7	4.6	14.0	7.0	4.1	3.9
promoters-936	5.3	4.8	4.0	4.5	4.6	12.8	10.6	6.8	6.4
ribosome-bind	9.3	8.5	8.4	8.1	8.2	11.2	10.2	9.3	9.6
satellite	13.0	10.9	10.6	9.9	10.0	13.8	9.9	8.6	8.4
segmentation	6.6	5.3	5.4	3.5	3.3	3.7	3.0	1.7	1.5
sick	5.9	5.7	5.7	4.7	4.5	1.3	1.2	1.1	1.0
sonar	16.6	15.9	16.8	12.9	13.0	29.7	25.3	21.5	21.7
soybean	9.2	6.7	6.9	6.7	6.3	8.0	7.9	7.2	6.7
splice	4.7	4.0	3.9	4.0	4.2	5.9	5.4	5.1	5.3
vehicle	24.9	21.2	20.7	19.1	19.7	29.4	27.1	22.5	22.9

Empirical Evidence

Caruana and Niculescu-Mizil (2006)

- Empirical study 2: 8 algorithms on 11 datasets
  - ✓ Algorithms
    - SVM, ANN, Logistic regression (LOGREG), Naïve Bayes (NB), KNN, Random Forests (RF), Decision Trees (DT), Bagged trees (BAG-DT), Boosted trees (BST-DT), Boosted stumps (BST-STMP)
  - ✓ Data sets

PROBLEM	#ATTR	TRAIN SIZE	TEST SIZE	%POZ
ADULT	14/104	5000	35222	25%
BACT	11/170	5000	34262	69%
COD	15/60	5000	14000	50%
CALHOUS	9	5000	14640	52%
COV_TYPE	54	5000	25000	36%
HS	200	5000	4366	24%
LETTER.P1	16	5000	14000	3%
LETTER.P2	16	5000	14000	53%
MEDIS	63	5000	8199	11%
MG	124	5000	12807	17%
SLAC	59	5000	25000	50%

Empirical Evidence

- Empirical study 2: 8 algorithms on 11 datasets
  - ✓ Normalized score by datasets

MODEL	CAL	COVT	ADULT	LTR.P1	LTR.P2	MEDIS	SLAC	HS	MG	CALHOUS	COD	BACT	MEAN
BST-DT	PLT	**.938**	.857	**.959**	**.976**	.700	.869	**.933**	.855	**.974**	**.915**	.878*	**.896***
RF	PLT	.876	.930	.897	.941	**.810**	.907*	.884	.883	.937	.903*	.847	.892
BAG-DT	–	.878	.944*	.883	.911	.762	.898*	.856	**.898**	.948	.856	**.926**	.887*
BST-DT	ISO	.922*	.865	.901*	.969	.692*	.878	.927	.845	.965	.912*	.861	.885*
RF	–	.876	.946*	.883	.922	.785	.912*	.871	.891*	.941	.874	.824	.884
BAG-DT	PLT	.873	.931	.877	.920	.752	.885	.863	.884	.944	.865	.912*	.882
RF	ISO	.865	.934	.851	.935	.767*	**.920**	.877	.876	.933	.897*	.821	.880
BAG-DT	ISO	.867	.933	.840	.915	.749	.897	.856	.884	.940	.859	.907*	.877
SVM	PLT	.765	.886	.936	.962	.733	.866	.913*	.816	.897	.900*	.807	.862
ANN	–	.764	.884	.913	.901	.791*	.881	.932*	.859	.923	.667	.882	.854
SVM	ISO	.758	.882	.899	.954	.693*	.878	.907	.827	.897	.900*	.778	.852
ANN	PLT	.766	.872	.898	.894	.775	.871	.929*	.846	.919	.665	.871	.846
ANN	ISO	.767	.882	.821	.891	.785*	.895	.926*	.841	.915	.672	.862	.842
BST-DT	–	.874	.842	.875	.913	.523	.807	.860	.785	.933	.835	.858	.828
KNN	PLT	.819	.785	.920	.937	.626	.777	.803	.844	.827	.774	.855	.815
KNN	–	.807	.780	.912	.936	.598	.800	.801	.853	.827	.748	.852	.810
KNN	ISO	.814	.784	.879	.935	.633	.791	.794	.832	.824	.777	.833	.809
BST-STMP	PLT	.644	**.949**	.767	.688	.723	.806	.800	.862	.923	.622	.915*	.791
SVM	–	.696	.819	.731	.860	.600	.859	.788	.776	.833	.864	.763	.781
BST-STMP	ISO	.639	.941	.700	.681	.711	.807	.793	.862	.912	.632	.902*	.780
BST-STMP	–	.605	.865	.540	.615	.624	.779	.683	.799	.817	.581	.906*	.710
DT	ISO	.671	.869	.729	.760	.424	.777	.622	.815	.832	.415	.884	.709
DT	–	.652	.872	.723	.763	.449	.769	.609	.829	.831	.389	.899*	.708
DT	PLT	.661	.863	.734	.756	.416	.779	.607	.822	.826	.407	.890*	.706
LR	–	.625	.886	.195	.448	.777*	.852	.675	.849	.838	.647	.905*	.700
LR	ISO	.616	.881	.229	.440	.763*	.834	.659	.827	.833	.636	.889*	.692
LR	PLT	.610	.870	.185	.446	.738	.835	.667	.823	.832	.633	.895	.685
NB	ISO	.574	.904	.674	.557	.709	.724	.205	.687	.758	.633	.770	.654
NB	PLT	.572	.892	.648	.561	.694	.732	.213	.690	.755	.632	.756	.650
NB	–	.552	.843	.534	.556	.011	.714	-.654	.655	.759	.636	.688	.481

Empirical Evidence

- Empirical study 2: 8 algorithms on 11 datasets
  - ✓ Normalized score by various metrics

MODEL	CAL	ACC	FSC	LFT	ROC	APR	BEP	RMS	MXE	MEAN	OPT-SEL
BST-DT	PLT	.843*	.779	**.939**	**.963**	**.938**	.929*	**.880**	**.896**	**.896**	**.917**
RF	PLT	.872*	.805	.934*	.957	.931	**.930**	.851	.858	.892	.898
BAG-DT	–	.846	.781	.938*	.962*	.937*	.918	.845	.872	.887*	.899
BST-DT	ISO	.826*	.860*	.929*	.952	.921	.925*	.854	.815	.885	.917*
RF	–	**.872**	.790	.934*	.957	.931	**.930**	.829	.830	.884	.890
BAG-DT	PLT	.841	.774	.938*	.962*	.937*	.918	.836	.852	.882	.895
RF	ISO	.861*	**.861**	.923	.946	.910	.925	.836	.776	.880	.895
BAG-DT	ISO	.826	.843*	.933*	.954	.921	.915	.832	.791	.877	.894
SVM	PLT	.824	.760	.895	.938	.898	.913	.831	.836	.862	.880
ANN	–	.803	.762	.910	.936	.892	.899	.811	.821	.854	.885
SVM	ISO	.813	.836*	.892	.925	.882	.911	.814	.744	.852	.882
ANN	PLT	.815	.748	.910	.936	.892	.899	.783	.785	.846	.875
ANN	ISO	.803	.836	.908	.924	.876	.891	.777	.718	.842	.884
BST-DT	–	.834*	.816	**.939**	**.963**	**.938**	.929*	.598	.605	.828	.851
KNN	PLT	.757	.707	.889	.918	.872	.872	.742	.764	.815	.837
KNN	–	.756	.728	.889	.918	.872	.872	.729	.718	.810	.830
KNN	ISO	.755	.758	.882	.907	.854	.869	.738	.706	.809	.844
BST-STMP	PLT	.724	.651	.876	.908	.853	.845	.716	.754	.791	.808
SVM	–	.817	.804	.895	.938	.899	.913	.514	.467	.781	.810
BST-STMP	ISO	.709	.744	.873	.899	.835	.840	.695	.646	.780	.810
BST-STMP	–	.741	.684	.876	.908	.853	.845	.394	.382	.710	.726
DT	ISO	.648	.654	.818	.838	.756	.778	.590	.589	.709	.774
DT	–	.647	.639	.824	.843	.762	.777	.562	.607	.708	.763
DT	PLT	.651	.618	.824	.843	.762	.777	.575	.594	.706	.761
LR	–	.636	.545	.823	.852	.743	.734	.620	.645	.700	.710
LR	ISO	.627	.567	.818	.847	.735	.742	.608	.589	.692	.703
LR	PLT	.630	.500	.823	.852	.743	.734	.593	.604	.685	.695
NB	ISO	.579	.468	.779	.820	.727	.733	.572	.555	.654	.661
NB	PLT	.576	.448	.780	.824	.738	.735	.537	.559	.650	.654
NB	–	.496	.562	.781	.825	.738	.735	.347	-.633	.481	.489

Empirical Evidence

Fernández-Delgado et al. (2014)

- Empirical study 3: 179 algorithms on 121 datasets

Data set	#pat.	#inp.	#cl.	%Maj.	Data set	#pat.	#inp.	#cl.	%Maj.
abalone	4177	8	3	34.6	energy-y1	768	8	3	46.9
ac-inflam	120	6	2	50.8	energy-y2	768	8	3	49.9
acute-nephritis	120	6	2	58.3	fertility	100	9	2	88.0
adult	48842	14	2	75.9	flags	194	28	8	30.9
annealing	798	38	6	76.2	glass	214	9	6	35.5
arrhythmia	452	262	13	54.2	haberman-survival	306	3	2	73.5
audiology-std	226	59	18	26.3	hayes-roth	132	3	3	38.6
balance-scale	625	4	3	46.1	heart-cleveland	303	13	5	54.1
balloons	16	4	2	56.2	heart-hungarian	294	12	2	63.9
bank	45211	17	2	88.5	heart-switzerland	123	12	2	39.0
blood	748	4	2	76.2	heart-va	200	12	5	28.0
breast-cancer	286	9	2	70.3	hepatitis	155	19	2	79.3
bc-wisc	699	9	2	65.5	hill-valley	606	100	2	50.7
bc-wisc-diag	569	30	2	62.7	horse-colic	300	25	2	63.7
bc-wisc-prog	198	33	2	76.3	ilpd-indian-liver	583	9	2	71.4
breast-tissue	106	9	6	20.7	image-segmentation	210	19	7	14.3
car	1728	6	4	70.0	ionosphere	351	33	2	64.1
ctg-10classes	2126	21	10	27.2	iris	150	4	3	33.3
ctg-3classes	2126	21	3	77.8	led-display	1000	7	10	11.1
chess-krvk	28056	6	18	16.2	lenses	24	4	3	62.5
chess-krvkp	3196	36	2	52.2	letter	20000	16	26	4.1
congress-voting	435	16	2	61.4	libras	360	90	15	6.7
conn-bench-sonar	208	60	2	53.4	low-res-spect	531	100	9	51.9
conn-bench-vowel	528	11	11	9.1	lung-cancer	32	56	3	40.6
connect-4	67557	42	2	75.4	lymphography	148	18	4	54.7
contrac	1473	9	3	42.7	magic	19020	10	2	64.8
credit-approval	690	15	2	55.5	mammographic	961	5	2	53.7
cylinder-bands	512	35	2	60.9	miniboone	130064	50	2	71.9
dermatology	366	34	6	30.6	molec-biol-promoter	106	57	2	50.0
echocardiogram	131	10	2	67.2	molec-biol-splice	3190	60	3	51.9
ecoli	336	7	8	42.6	monks-1	124	6	2	50.0

Data set	#pat.	#inp.	#cl.	%Maj.	Data set	#pat.	#inp.	#cl.	%Maj.
monks-2	169	6	2	62.1	soybean	307	35	18	13.0
monks-3	3190	6	2	50.8	spambase	4601	57	2	60.6
mushroom	8124	21	2	51.8	spect	80	22	2	67.1
musk-1	476	166	2	56.5	spectf	80	44	2	50.0
musk-2	6598	166	2	84.6	st-australian-credit	690	14	2	67.8
nursery	12960	8	5	33.3	st-german-credit	1000	24	2	70.0
oocMerl2F	1022	25	3	67.0	st-heart	270	13	2	55.6
oocMerl4D	1022	41	2	68.7	st-image	2310	18	7	14.3
oocTris2F	912	25	2	57.8	st-landsat	4435	36	6	24.2
oocTris5B	912	32	3	57.6	st-shuttle	43500	9	7	78.4
optical	3823	62	10	10.2	st-vehicle	846	18	4	25.8
ozone	2536	72	2	97.1	steel-plates	1941	27	7	34.7
page-blocks	5473	10	5	89.8	synthetic-control	600	60	6	16.7
parkinsons	195	22	2	75.4	teaching	151	5	3	34.4
pendigits	7494	16	10	10.4	thyroid	3772	21	3	92.5
pima	768	8	2	65.1	tic-tac-toe	958	9	2	65.3
pb-MATERIAL	106	4	3	74.5	titanic	2201	3	2	67.7
pb-REL-L	103	4	3	51.5	trains	10	28	2	50.0
pb-SPAN	92	4	3	52.2	twonorm	7400	20	2	50.0
pb-T-OR-D	102	4	2	86.3	vc-2classes	310	6	2	67.7
pb-TYPE	105	4	6	41.9	vc-3classes	310	6	3	48.4
planning	182	12	2	71.4	wall-following	5456	24	4	40.4
plant-margin	1600	64	100	1.0	waveform	5000	21	3	33.9
plant-shape	1600	64	100	1.0	waveform-noise	5000	40	3	33.8
plant-texture	1600	64	100	1.0	wine	179	13	3	39.9
post-operative	90	8	3	71.1	wine-quality-red	1599	11	6	42.6
primary-tumor	330	17	15	25.4	wine-quality-white	4898	11	7	44.9
ringnorm	7400	20	2	50.5	yeast	1484	8	10	31.2
seeds	210	7	3	33.3	zoo	101	16	7	40.6
semeion	1593	256	10	10.2					

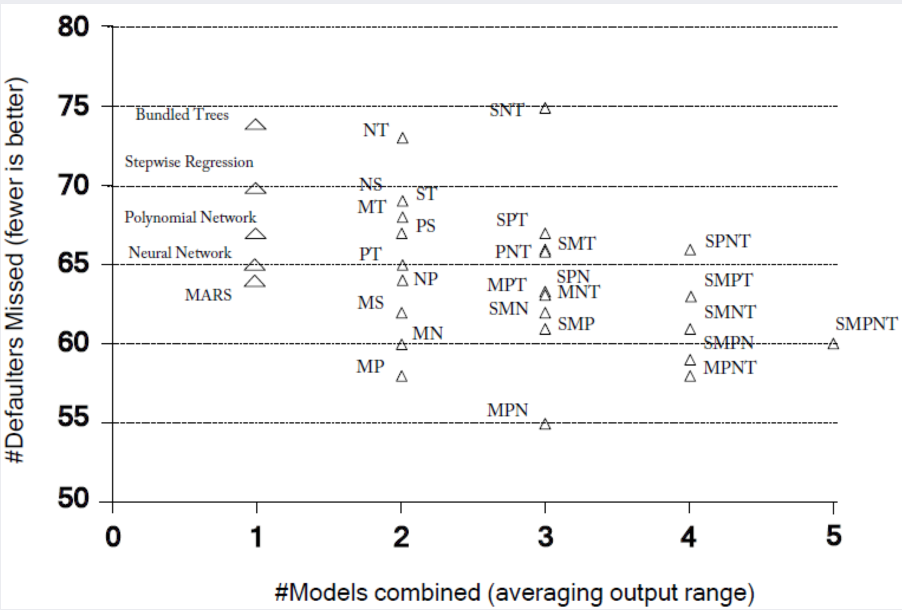
Empirical Evidence

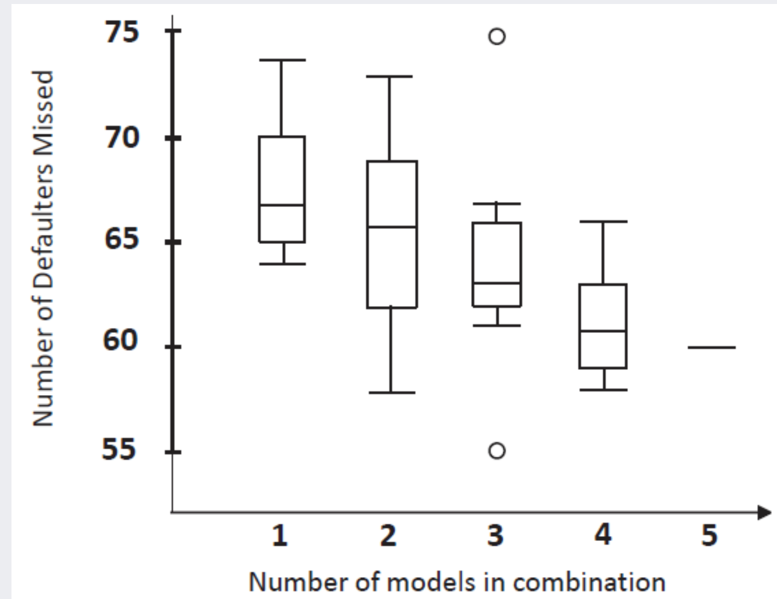
- Empirical study 3: 179 algorithms on 121 datasets

Rank	Acc.	κ	Classifier	Rank	Acc.	κ	Classifier
32.9	82.0	63.5	parRF_t (RF)	67.3	77.7	55.6	pda_t (DA)
33.1	**82.3**	**63.6**	rf_t (RF)	67.6	78.7	55.2	elm_m (NNET)
36.8	81.8	62.2	svm_C (SVM)	67.6	77.8	54.2	SimpleLogistic_w (LMR)
38.0	81.2	60.1	svmPoly_t (SVM)	69.2	78.3	57.4	MAB_J48_w (BST)
39.4	81.9	62.5	rforest_R (RF)	69.8	78.8	56.7	BG_REPTree_w (BAG)
39.6	82.0	62.0	elm_kernel_m (NNET)	69.8	78.1	55.4	SMO_w (SVM)
40.3	81.4	61.1	svmRadialCost_t (SVM)	70.6	78.3	58.0	MLP_w (NNET)
42.5	81.0	60.0	svmRadial_t (SVM)	71.0	78.8	58.23	BG_RandomTree_w (BAG)
42.9	80.6	61.0	C5.0_t (BST)	71.0	77.1	55.1	mlm_R (GLM)
44.1	79.4	60.5	avNNet_t (NNET)	71.0	77.8	56.2	BG_J48_w (BAG)
45.5	79.5	61.0	nnet_t (NNET)	72.0	75.7	52.6	rbf_t (NNET)
47.0	78.7	59.4	pcaNNet_t (NNET)	72.1	77.1	54.8	fda_R (DA)
47.1	80.8	53.0	BG_LibSVM_w (BAG)	72.4	77.0	54.7	lda_R (DA)
47.3	80.3	62.0	mlp_t (NNET)	72.4	79.1	55.6	svmlight_C (NNET)
47.6	80.6	60.0	RotationForest_w (RF)	72.6	78.4	57.9	AdaBoostM1_J48_w (BST)
50.1	80.9	61.6	RRF_t (RF)	72.7	78.4	56.2	BG_IBk_w (BAG)
51.6	80.7	61.4	RRFglobal_t (RF)	72.9	77.1	54.6	ldaBag_R (BAG)
52.5	80.6	58.0	MAB_LibSVM_w (BST)	73.2	78.3	56.2	BG_LWL_w (BAG)
52.6	79.9	56.9	LibSVM_w (SVM)	73.7	77.9	56.0	MAB_REPTree_w (BST)
57.6	79.1	59.3	adaboost_R (BST)	74.0	77.4	52.6	RandomSubSpace_w (DT)
58.5	79.7	57.2	pnn_m (NNET)	74.4	76.9	54.2	lda2_t (DA)
58.9	78.5	54.7	cforest_t (RF)	74.6	74.1	51.8	svmBag_R (BAG)
59.9	79.7	42.6	dkp_C (NNET)	74.6	77.5	55.2	LibLINEAR_w (SVM)
60.4	80.1	55.8	gaussprRadial_R (OM)	75.9	77.2	55.6	rbfDDA_t (NNET)
60.5	80.0	57.4	RandomForest_w (RF)	76.5	76.9	53.8	sda_t (DA)
62.1	78.7	56.0	svmLinear_t (SVM)	76.6	78.1	56.5	END_w (OEN)
62.5	78.4	57.5	fda_t (DA)	76.6	77.3	54.8	LogitBoost_w (BST)
62.6	78.6	56.0	knn_t (NN)	76.6	78.2	57.3	MAB_RandomTree_w (BST)
62.8	78.5	58.1	mlp_C (NNET)	77.1	78.4	54.0	BG_RandomForest_w (BAG)
63.0	79.9	59.4	RandomCommittee_w (OEN)	78.5	76.5	53.7	Logistic_w (LMR)
63.4	78.7	58.4	Decorate_w (OEN)	78.7	76.6	50.5	ctreeBag_R (BAG)
63.6	76.9	56.0	mlpWeightDecay_t (NNET)	79.0	76.8	53.5	BG_Logistic_w (BAG)
63.8	78.7	56.7	rda_R (DA)	79.1	77.4	53.0	lvq_t (NNET)
64.0	79.0	58.6	MAB_MLP_w (BST)	79.1	74.4	50.7	pls_t (PLSR)
64.1	79.9	56.9	MAB_RandomForest_w (BST)	79.8	76.9	54.7	hdda_R (DA)
65.0	79.0	56.8	knn_R (NN)	80.6	75.9	53.3	MCC_w (OEN)
65.2	77.9	56.2	multinom_t (LMR)	80.9	76.9	54.5	mda_R (DA)
65.5	77.4	56.6	gcvEarth_t (MARS)	81.4	76.7	55.2	C5.0Rules_t (RL)
65.5	77.8	55.7	glmnet_R (GLM)	81.6	78.3	55.8	lssvmRadial_t (SVM)
65.6	78.6	58.4	MAB_PART_w (BST)	81.7	75.6	50.9	JRip_t (RL)
66.0	78.5	56.5	CVR_w (OM)	82.0	76.1	53.3	MAB_Logistic_w (BST)
66.4	79.2	58.9	treebag_t (BAG)	84.2	75.8	53.9	C5.0Tree_t (DT)
66.6	78.2	56.8	BG_PART_w (BAG)	84.6	75.7	50.8	BG_DecisionTable_w (BAG)
66.7	75.5	55.2	mda_t (DA)	84.9	76.5	53.4	NBTree_w (DT)

Real-world Examples

- Credit card scoring
  - ✓ Mean error reduces with increasing degree of combination





- Netflix competition
  - ✓ The final edge was obtained by weighting contributions from the models of up to 30 competitors

Real-world Examples

- The 10 main takeaways from MLConf SF (2016)
  - ✓ It's (still) not all about Deep Learning
  - ✓ Choose the right problem to solve, with the right metric
  - ✓ Fine tuning your models is 5% of a project
  - ✓ **Ensembles almost always work better**
  - ✓ The trend towards personalization
  - ✓ Manual curation of content is still used in practice
  - ✓ Avoid the curse of complexity
  - ✓ Learn the best practices from established players
  - ✓ Everybody is using open source
  - ✓ Make sure you have support from the executives

https://tryolabs.com/blog/2016/11/18/10-main-takeaways-from-mlconf/

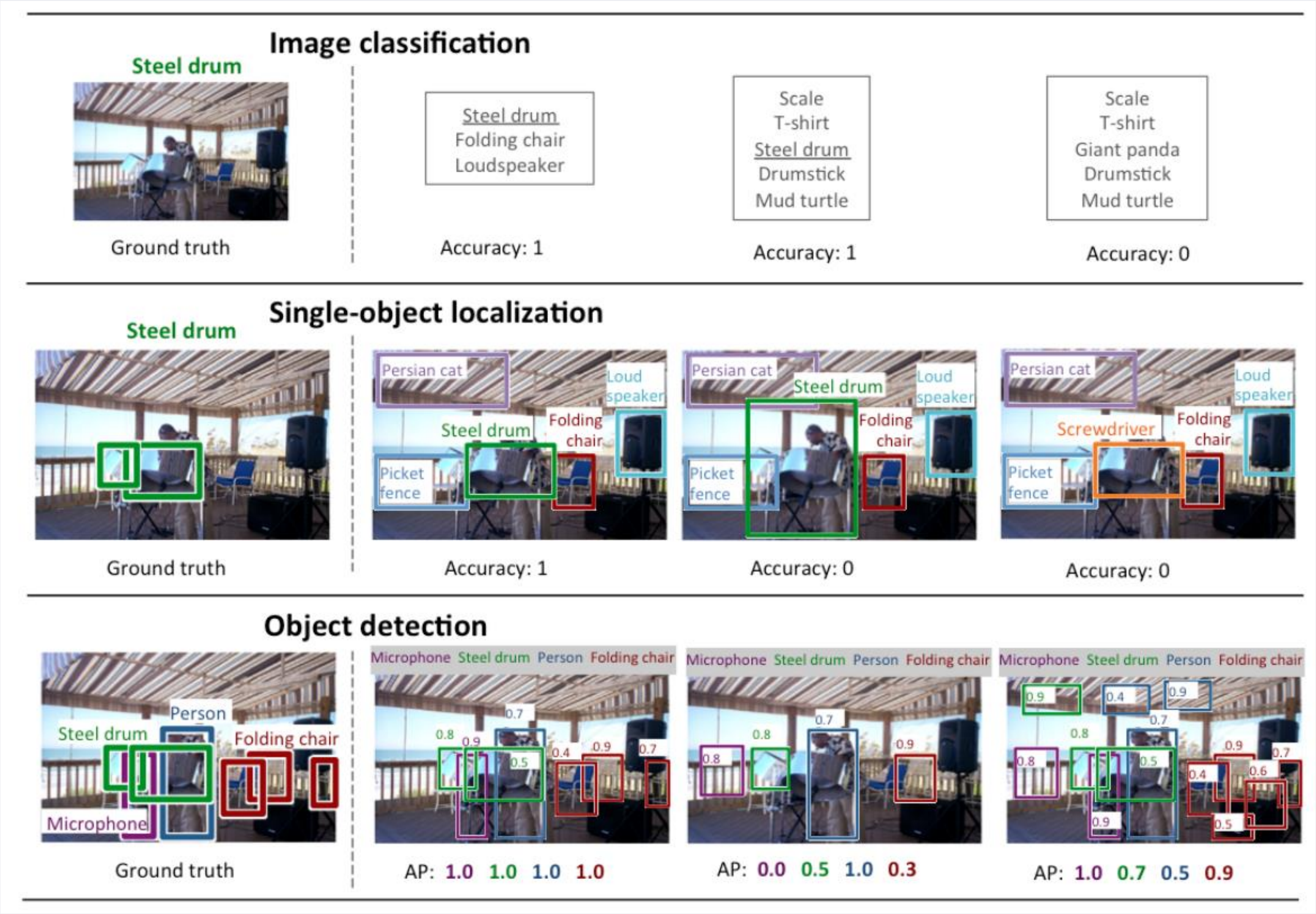
Real-world Examples

- Large Scale Visual Recognition Challenge
  - ✓ With given these images…

Real-world Examples

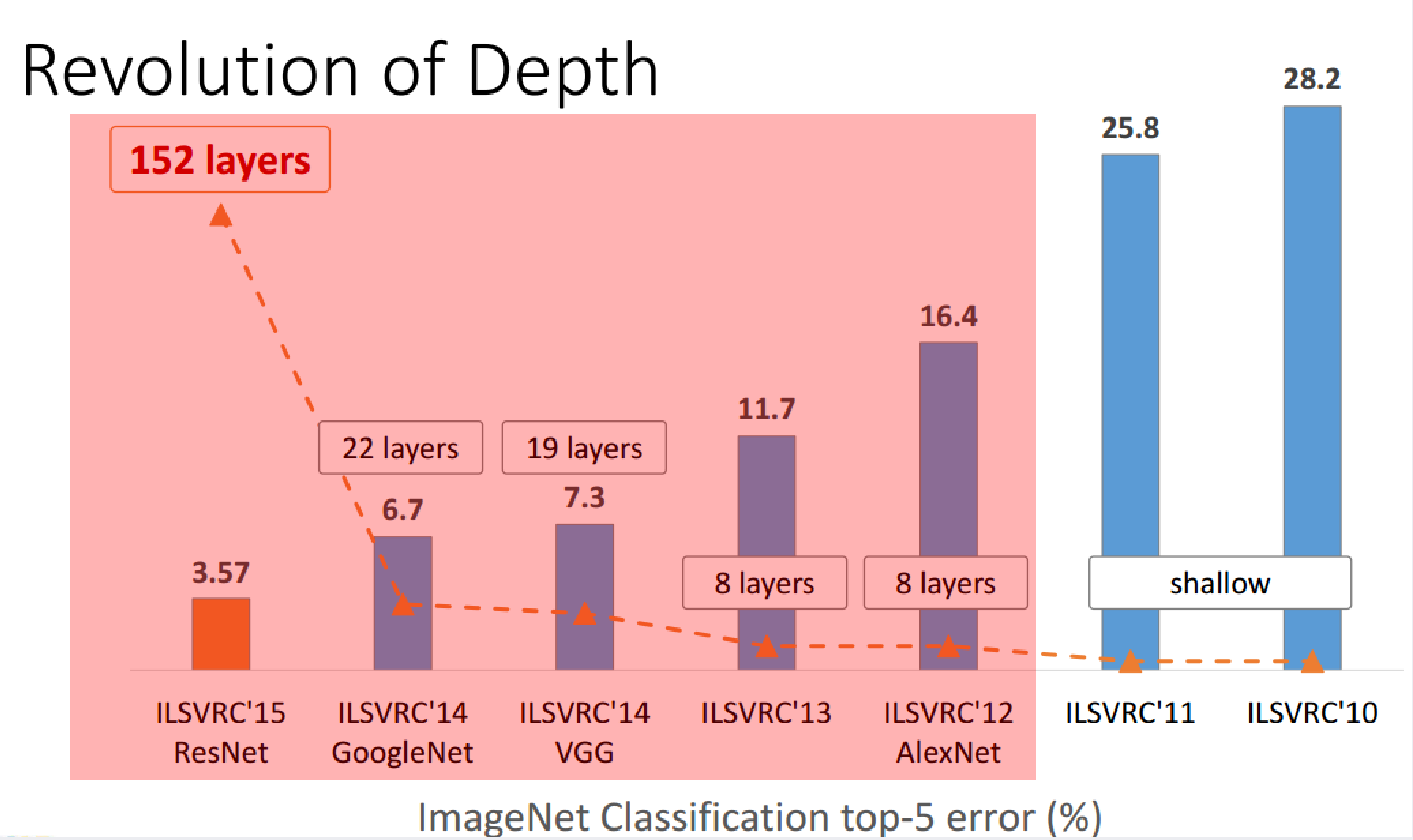
Russakovsky et al. (2015)

- Large Scale Visual Recognition Challenge
  - ✓ Tasks



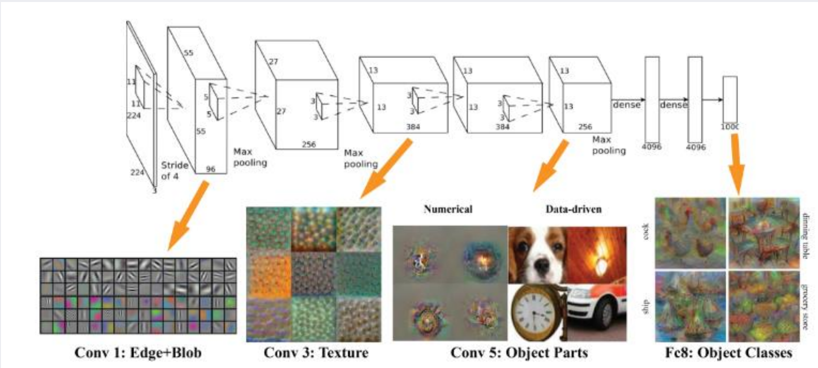
Real-world Examples

- Large Scale Visual Recognition Challenge (~ ILSVRC2015)

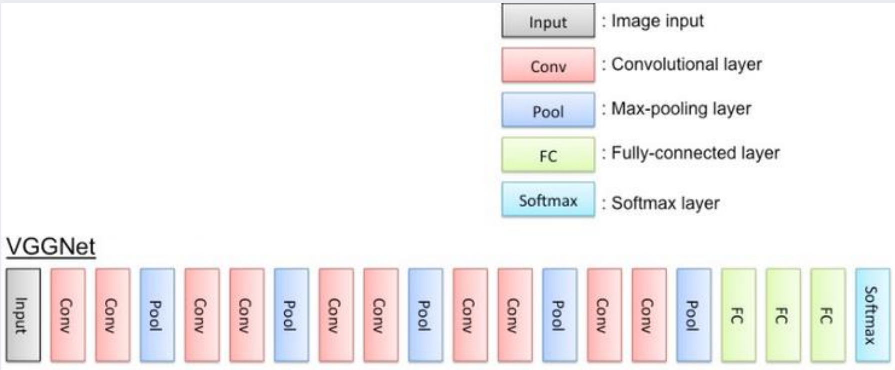


Real-world Examples

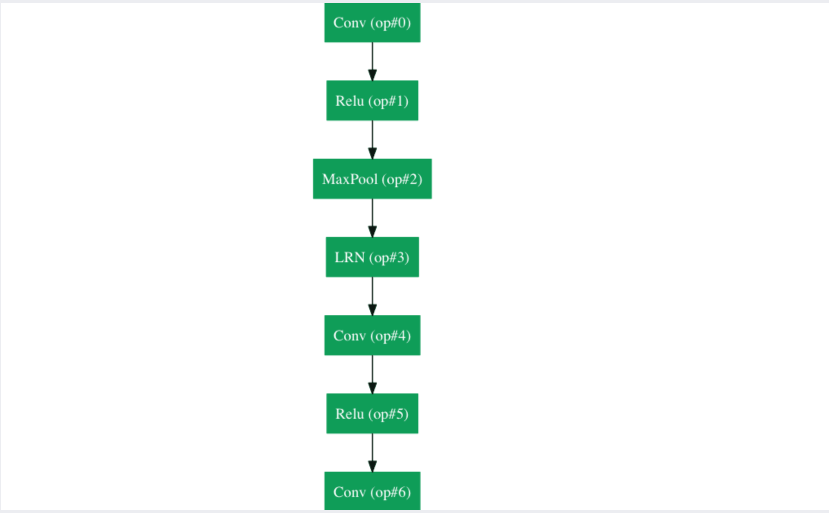
- Large Scale Visual Recognition Challenge (~ ILSVRC2015)



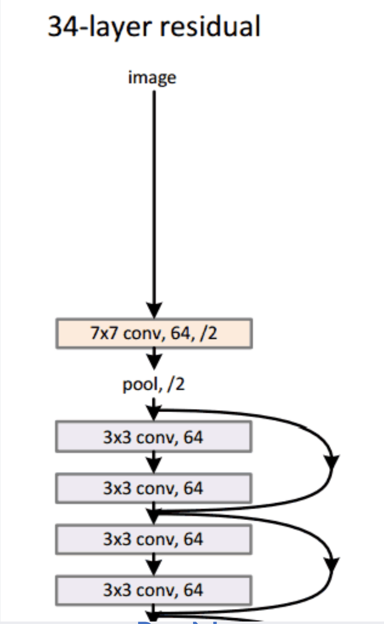
Alexnet



VGGNet



GoogLeNet



ResNet

Real-world Examples

- Large Scale Visual Recognition Challenge (ILSVRC2016 ~)
  - ✓ 2016

Object detection (DET)[top]

Task 1a: Object detection with provided training data

Ordered by number of categories won

Team name	Entry description	**Number of object categories won**	mean AP
CUImage	Ensemble of 6 models using provided data	109	0.662751
Hikvision	Ensemble A of 3 RPN and 6 FRCN models, mAP is 67 on val2	30	0.652704
Hikvision	Ensemble B of 3 RPN and 5 FRCN models, mean AP is 66.9, median AP is 69.3 on val2	18	0.652003

Object localization (LOC)[top]

Task 2a: Classification+localization with provided training data

Ordered by localization error

Team name	Entry description	**Localization error**	Classification error
Trimps-Soushen	Ensemble 3	0.077087	0.02991
Trimps-Soushen	Ensemble 4	0.077429	0.02991
Trimps-Soushen	Ensemble 2	0.077668	0.02991
Trimps-Soushen	Ensemble 1	0.079068	0.03144

  - ✓ 2017

Object detection (DET)[top]

Task 1a: Object detection with provided training data

Ordered by number of categories won

Team name	Entry description	**Number of object categories won**	mean AP
BDAT	submission4	85	0.731392
BDAT	submission3	65	0.732227
BDAT	submission2	30	0.723712
DeepView(ETRI)	Ensemble_A	10	0.593084
NUS-Qihoo_DPNs (DET)	Ensemble of DPN models	9	0.656932
KAISTNIA_ETRI	Ensemble Model5	1	0.61022
KAISTNIA_ETRI	Ensemble Model4	0	0.609402
KAISTNIA_ETRI	Ensemble Model2	0	0.608299
KAISTNIA_ETRI	Ensemble Model1	0	0.608278
KAISTNIA_ETRI	Ensemble Model3	0	0.60631

Object localization (LOC)[top]

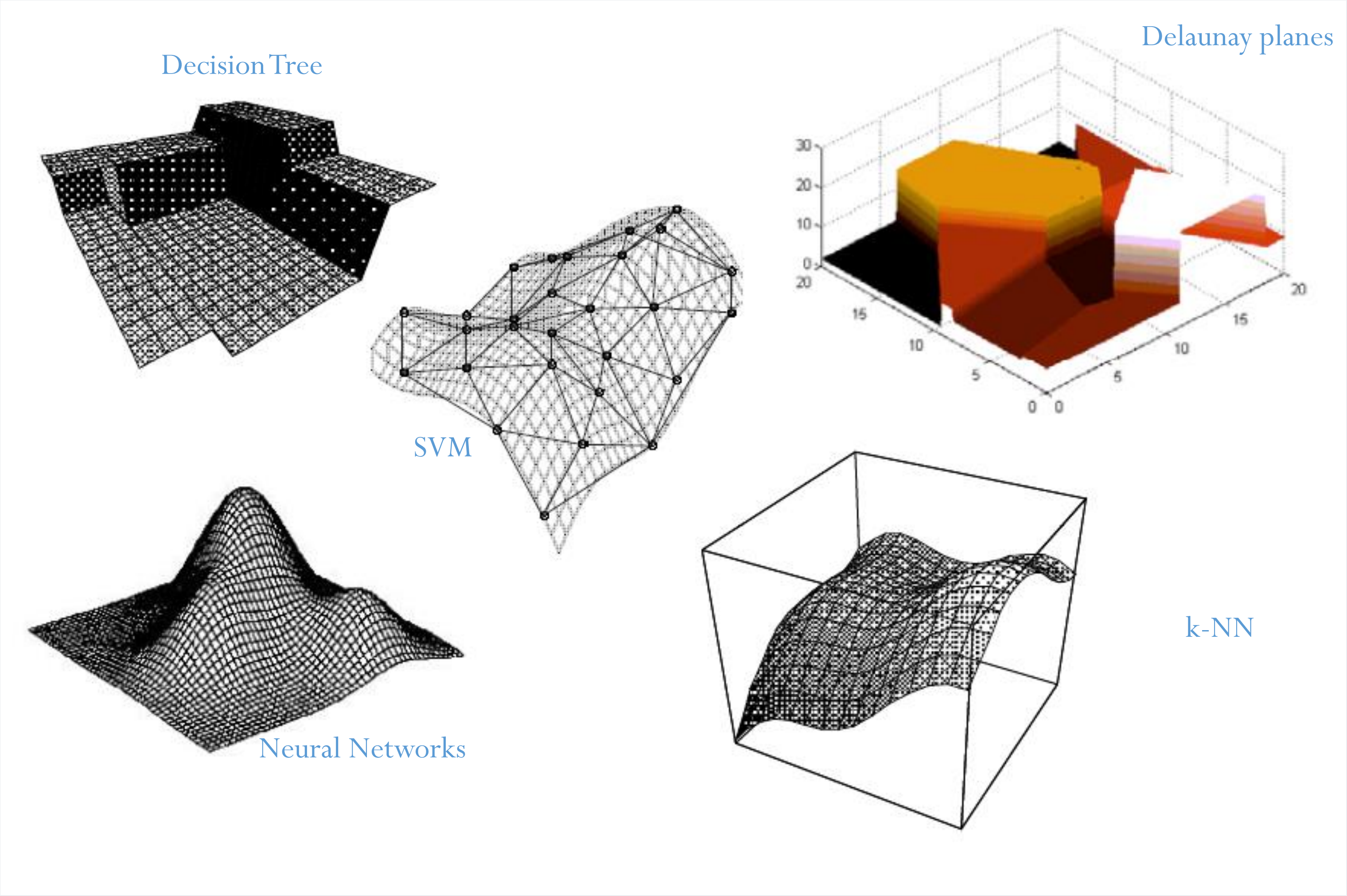
Task 2a: Classification+localization with provided training data

Ordered by localization error

Team name	Entry description	**Localization error**	Classification error
NUS-Qihoo_DPNs (CLS-LOC)	[E3] LOC:: Dual Path Networks + Basic Ensemble	0.062263	0.03413
Trimps-Soushen	Result-3	0.064991	0.02481
Trimps-Soushen	Result-2	0.06525	0.02481
Trimps-Soushen	Result-4	0.065261	0.02481
Trimps-Soushen	Result-5	0.065302	0.02481
Trimps-Soushen	Result-1	0.067698	0.02481

Theoretical Backgrounds: Model Space

- Different model produce different class boundaries or fitted functions



Theoretical Backgrounds: Bias-Variance Decomposition

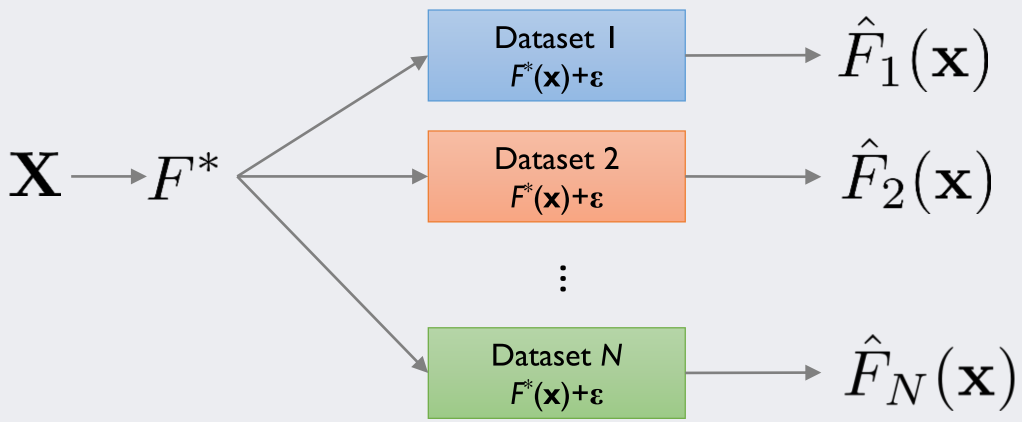
- Suppose the data comes from the "additive error" model

$$y = F^*(\mathbf{x}) + \epsilon, \quad \epsilon \sim N(0, \sigma^2)$$

✓ $F^*(\mathbf{x})$ is the target function that we are trying to learn, but do not really know

✓ The errors are independent and identically distributed

- Consider the estimation process



✓ The average fit over all possible datasets:

$$\bar{F}(\mathbf{x}) = E[\hat{F}_D(\mathbf{x})]$$

Theoretical Backgrounds: Bias-Variance Decomposition

- The MSE for a particular data point

$$Err(\mathbf{x}_0) = E\Big[y - \hat{F}(\mathbf{x})|\mathbf{x} = \mathbf{x}_0\Big]^2 \qquad (y = F^*(\mathbf{x}) + \epsilon)$$

$$= E\Big[\hat{F}^*(\mathbf{x}_0) + \epsilon - \hat{F}(\mathbf{x}_0)\Big]^2$$

$$= E\Big[\hat{F}^*(\mathbf{x}_0) - \hat{F}(\mathbf{x}_0)\Big]^2 + \sigma^2$$

$$= E\Big[\hat{F}^*(\mathbf{x}_0) - \bar{F}(\mathbf{x}_0) + \bar{F}(\mathbf{x}_0) - \hat{F}(\mathbf{x}_0)\Big]^2 + \sigma^2$$

Theoretical Backgrounds: Bias-Variance Decomposition

- The MSE for a particular data point

$$= E\left[\hat{F}^*(\mathbf{x}_0) - \bar{F}(\mathbf{x}_0) + \bar{F}(\mathbf{x}_0) - \hat{F}(\mathbf{x}_0)\right]^2 + \sigma^2$$

✓ By the properties of the expectation operator

$$= E\left[\hat{F}^*(\mathbf{x}_0) - \bar{F}(\mathbf{x}_0)\right]^2 + E\left[\bar{F}(\mathbf{x}_0) - \hat{F}(\mathbf{x}_0)\right]^2 + \sigma^2$$

$$= \left[\hat{F}^*(\mathbf{x}_0) - \bar{F}(\mathbf{x}_0)\right]^2 + E\left[\bar{F}(\mathbf{x}_0) - \hat{F}(\mathbf{x}_0)\right]^2 + \sigma^2$$

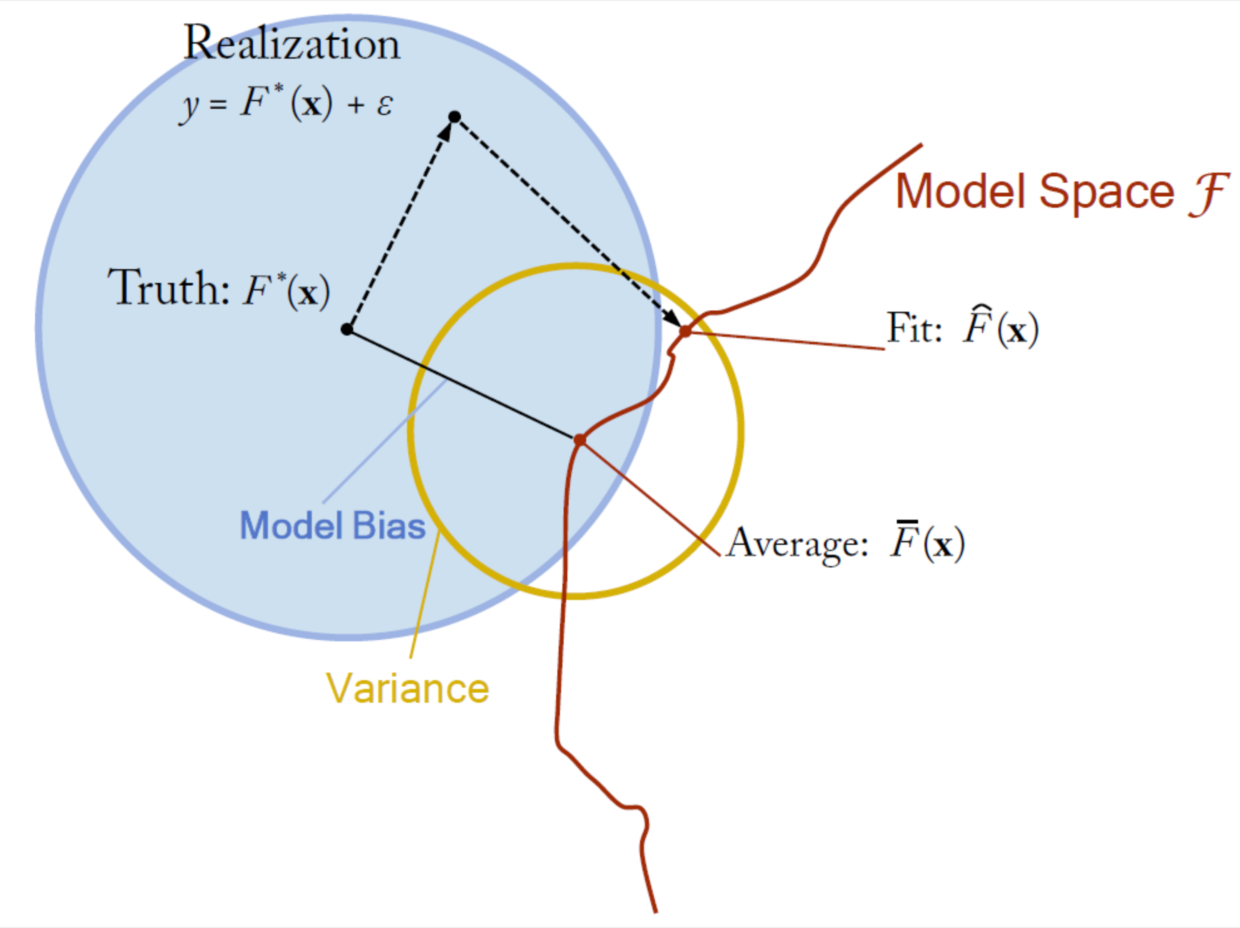
$$= Bias^2\left(\hat{F}(\mathbf{x}_0)\right) + Var\left(\hat{F}(\mathbf{x}_0)\right) + \sigma^2$$

Theoretical Backgrounds: Bias-Variance Decomposition

- Properties of Bias and Variance
  - ✓ Bias^2: the amount by which the average estimator differs from the truth
    - Low bias: on average, we will accurately estimate the function from the dataset
    - High bias implies a **poor** match
  - ✓ Variance: spread of the individual estimations around their mean
    - Low variance: estimated function does not change much with different datasets
    - High variance implies a **weak** match
  - ✓ Irreducible error: the error that was present in the original data
  - ✓ Bias and variance are not independent of each other

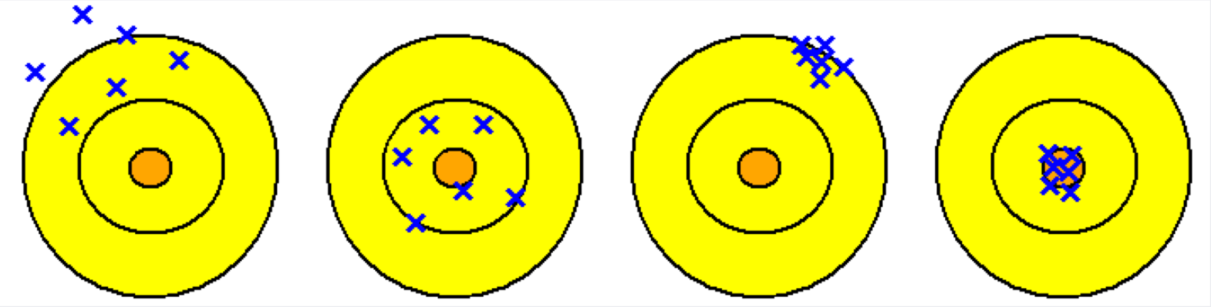
Theoretical Backgrounds: Bias-Variance Decomposition

- Graphical representation of Bias-Variance decomposition



Theoretical Backgrounds: Bias-Variance Decomposition

- Graphical representation of Bias-Variance decomposition



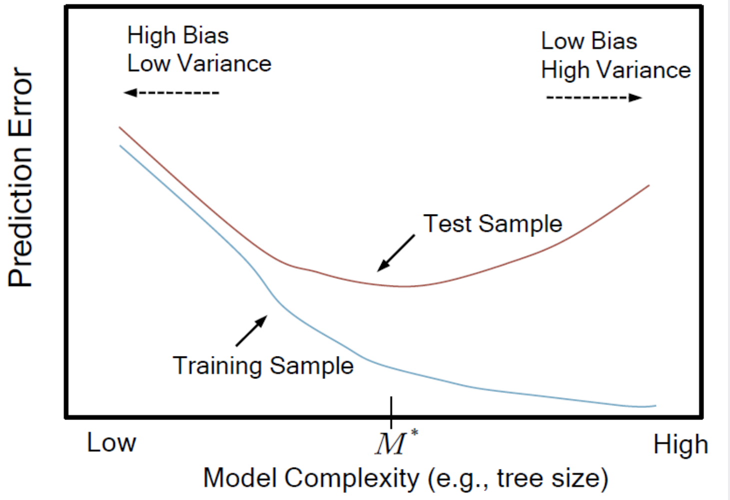
Bias	High	Low	High	Low
Variance	High	High	Low	Low

✓ Lower model complexity: high bias & low variance

- Logistic regression, LDA, k-NN with large k, etc.

✓ Higher model complexity: low bias & high variance

- DT, ANN, SVM, k-NN with small k, etc.



Bias-Variance Dilemma

The more complex (flexible) we make the model,
the lower the bias but the higher the variance it is subjected to.

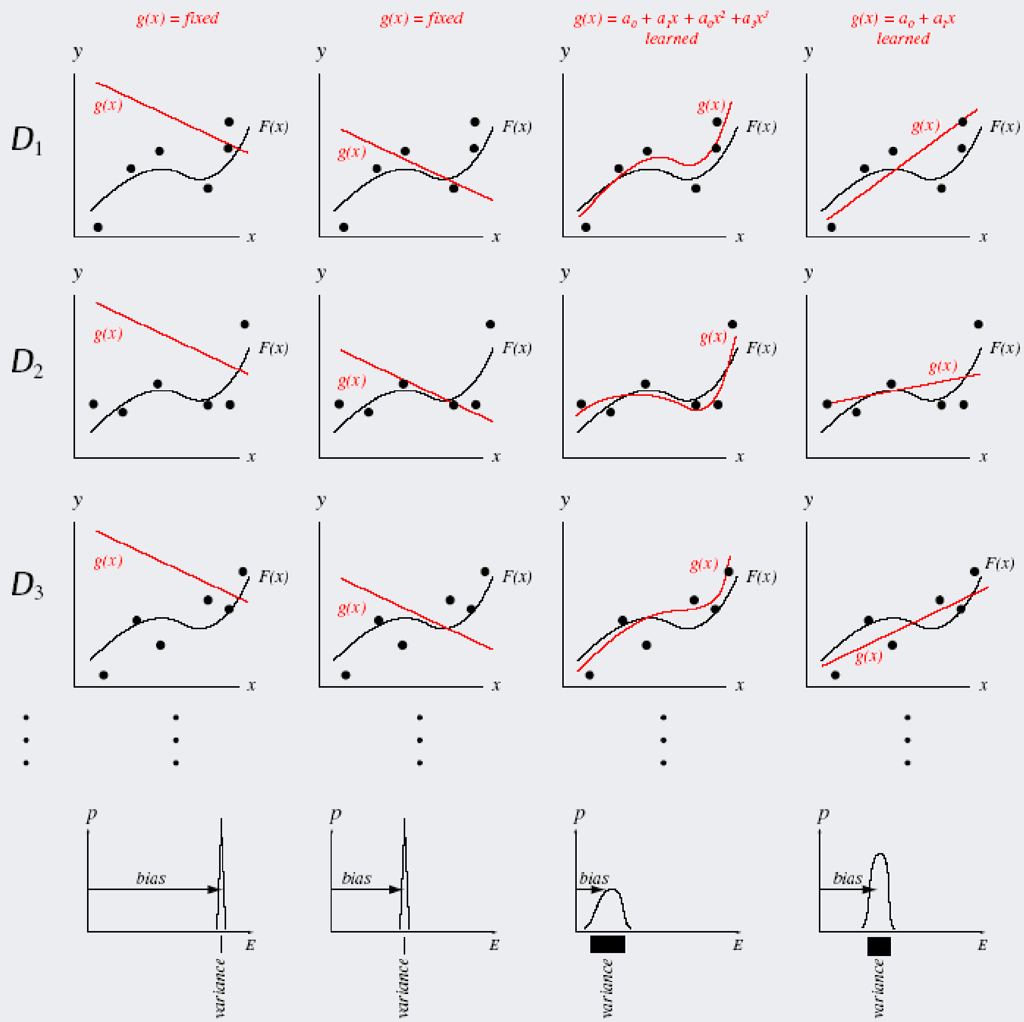
Theoretical Backgrounds: Bias-Variance Decomposition

- Bias-Variance example

Each column is a different model.

Each row is a different dataset of 6 points.

Histograms of mean-squared error of the fit.



Col 1:
Poor fixed linear model;
High bias, zero variance

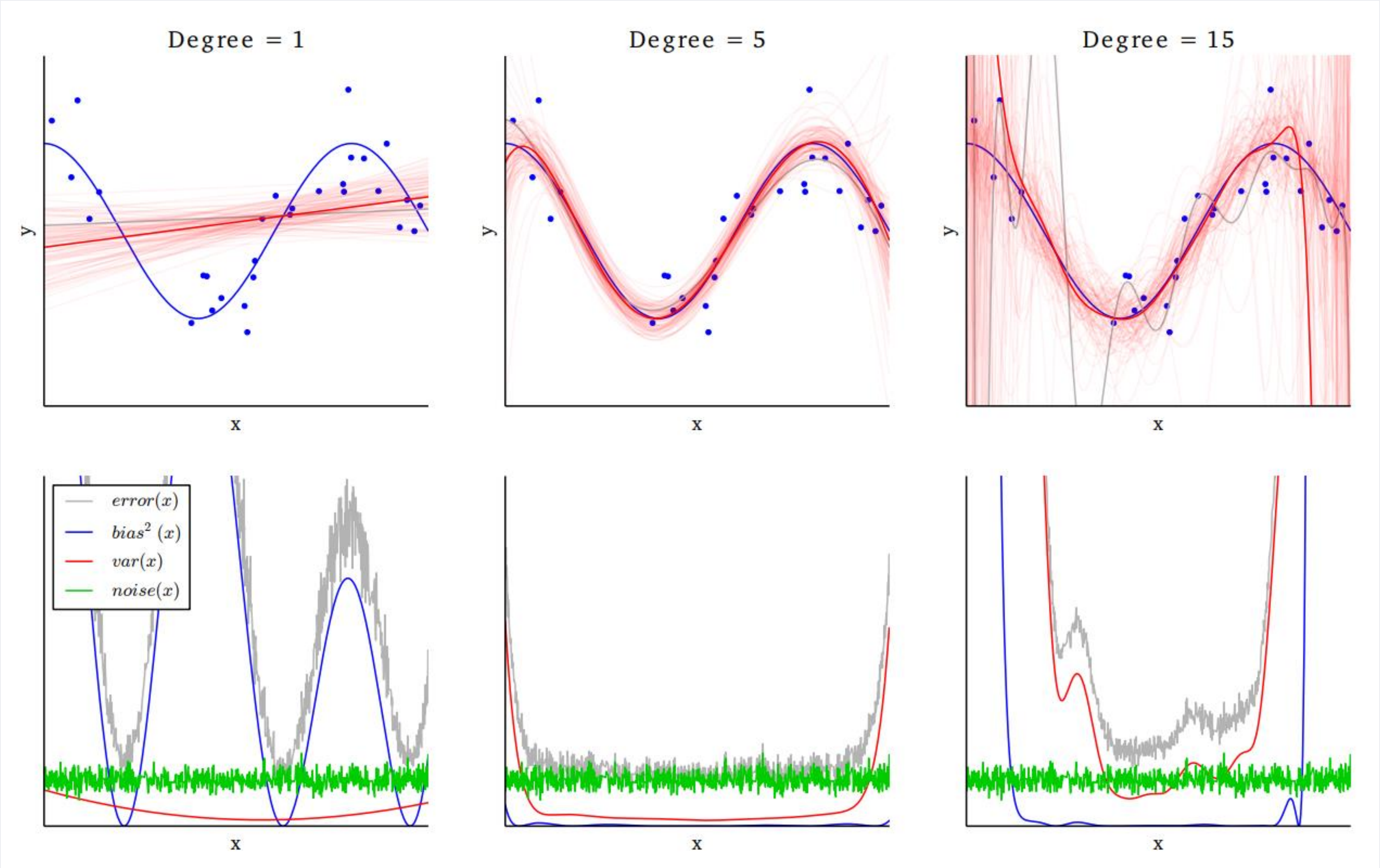
Col 2:
Slightly better fixed linear model;
Lower (but high) bias, zero variance.

Col 3:
Learned cubic model;
Low bias, moderate variance.

Col 4:
Learned linear model;
Intermediate bias and variance.

Theoretical Backgrounds: Bias-Variance Decomposition

- Bias-Variance example



Purpose of Ensemble

- Goal: Reduce the error through constructing multiple learners to
  - ✓ Reduce the variance: Bagging, Random Forests
  - ✓ Reduce the bias: AdaBoost
  - ✓ Both: Mixture of experts

- Two key questions on the ensemble construction
  - ✓ Q1: How to generate individual components of the ensemble systems (base classifiers) to achieve sufficient degree of diversity?
  - ✓ Q2: How to combine the outputs of individual classifiers?

Ensemble Diversity

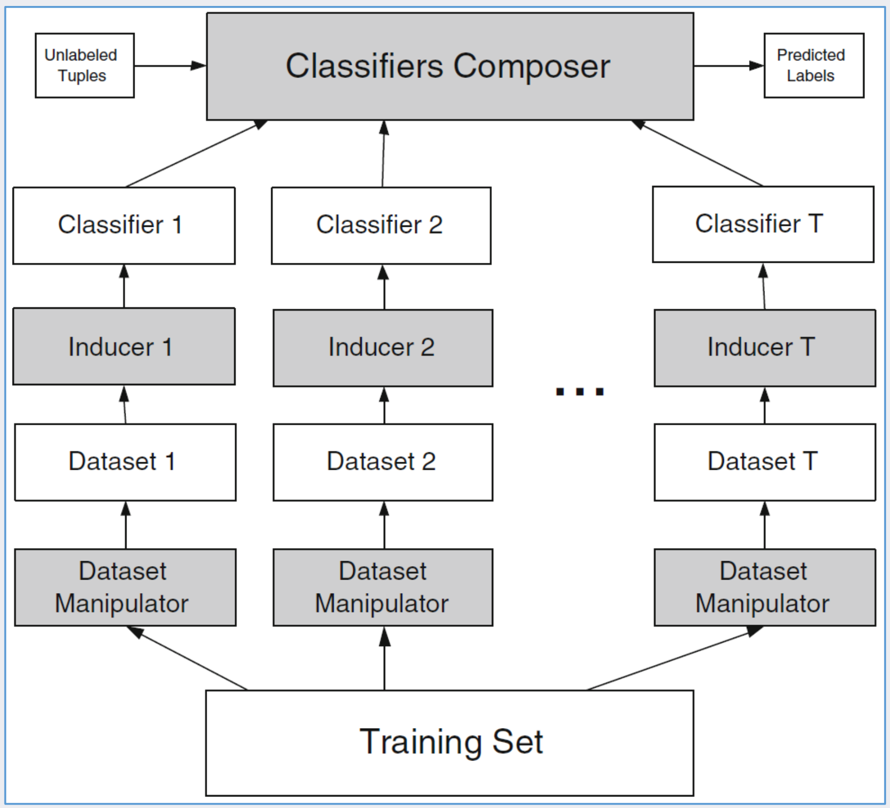
- Ensemble will have no gain from combining a set of identical models
  - ✓ Need base learners whose fitted functions are adequately different from those of others
  - ✓ Wish models to exhibit a certain element of diversity in their group behavior, though still retaining good performance individually.

Diversity	Implicit	Explicit
Description	Provide different random subset of the training data to each learner	Use some measurement ensuring it is substantially different from the other members
Ensemble Algorithms	Instance: Bagging Variables: Random Subspaces, Rotation Forests Both: Random Forests	Boosting, Negative Correlation Learning

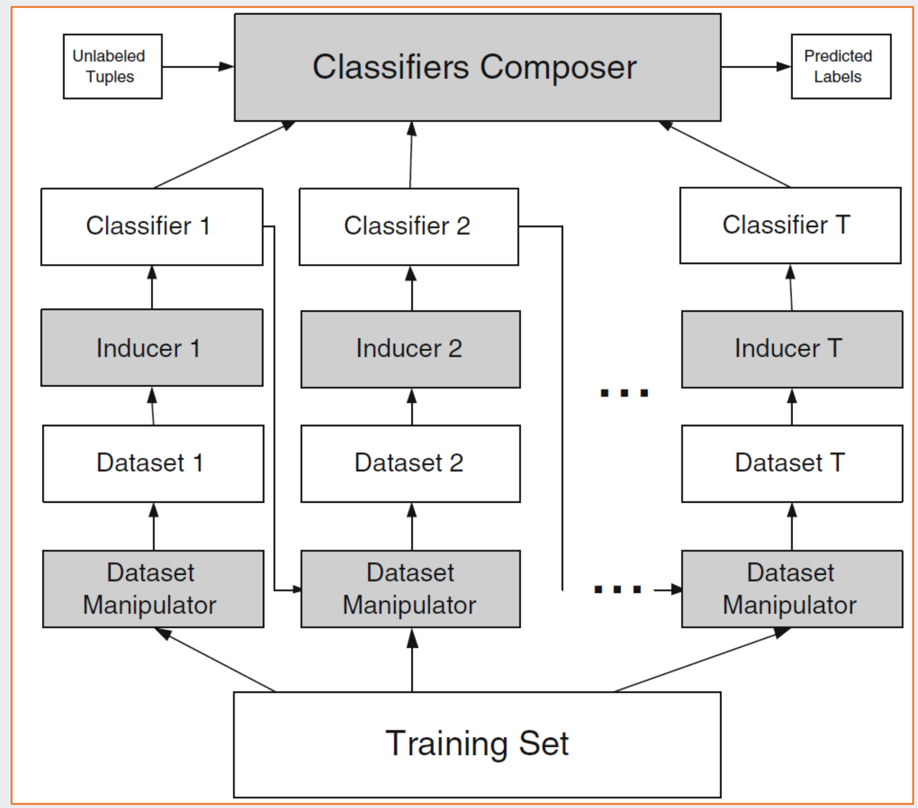
Ensemble Diversity

- Independent (implicit) vs. Model guided (explicit) instance selection

Independent instance selection



Model guided instance selection



Why Ensemble?

- Why Ensemble works?
  - ✓ True functions, estimations, and the expected error

$$y_m(\mathbf{x}) = f(\mathbf{x}) + \epsilon_m(\mathbf{x}). \quad \mathbb{E}_{\mathbf{x}}\left[\{y_m(\mathbf{x}) - f(\mathbf{x})\}^2\right] = \mathbb{E}_{\mathbf{x}}\left[\epsilon_m(\mathbf{x})^2\right]$$

  - ✓ The average error made by M individual models vs. Expected error of the ensemble

$$E_{Avg} = \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_{\mathbf{x}}\left[\epsilon_m(\mathbf{x})^2\right]$$

$$E_{Ensemble} = \mathbb{E}_{\mathbf{x}}\left[\left\{\frac{1}{M}\sum_{m=1}^{M} y_m(\mathbf{x}) - f(\mathbf{x})\right\}^2\right]$$

$$= \mathbb{E}_{\mathbf{x}}\left[\left\{\frac{1}{M}\sum_{m=1}^{M} \epsilon_m(\mathbf{x})\right\}^2\right]$$

Why Ensemble?

- Why Ensemble works?
  - ✓ Assume that the errors have zero mean and are uncorrelated,

$$\mathbb{E}_{\mathbf{x}}[\epsilon_m(\mathbf{x})] = 0, \qquad \mathbb{E}_{\mathbf{x}}[\epsilon_m(\mathbf{x})\epsilon_l(\mathbf{x})] = 0 \ (m \neq l)$$

  - ✓ The average error made by M individual models vs. Expected error of the ensemble

$$E_{Ensemble} = \frac{1}{M} E_{Avg}$$

  - ✓ In reality (errors are correlated), by the Cauchy's inequality

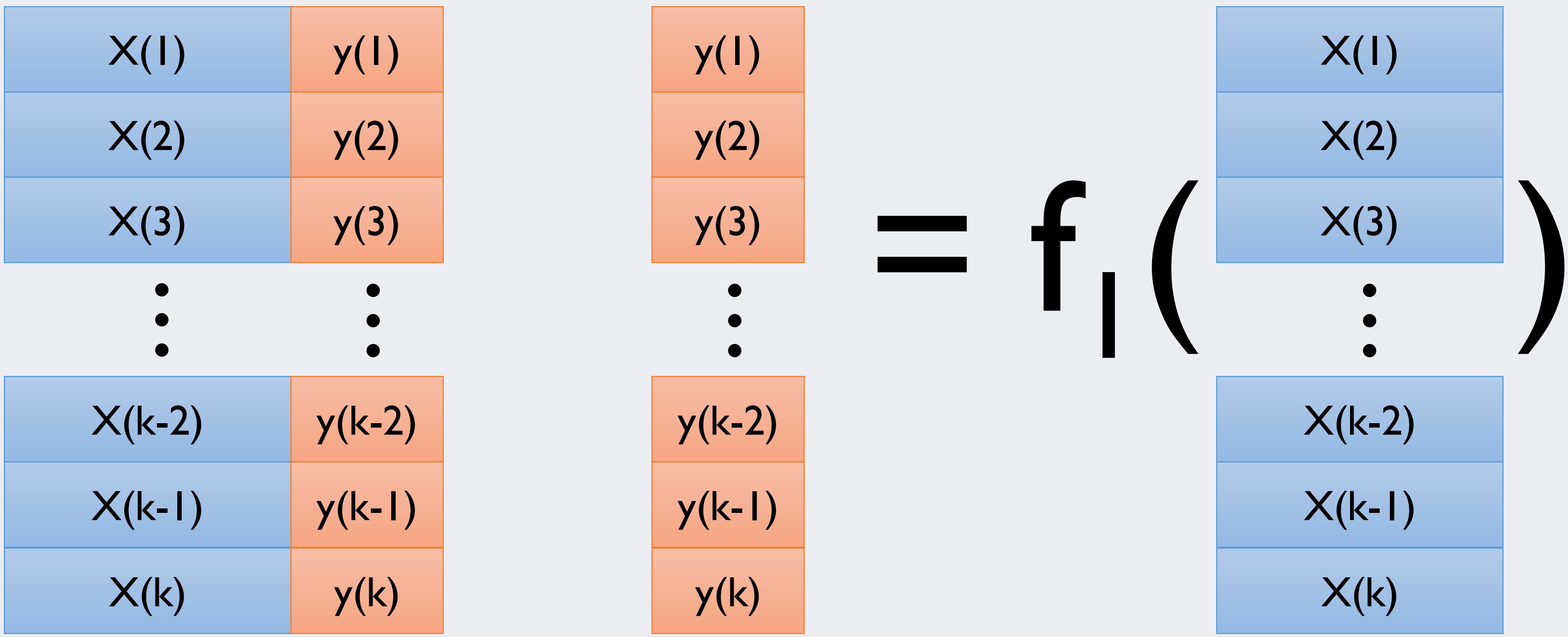
$$\Big[\sum_{m=1}^{M} \epsilon_m(\mathbf{x})\Big]^2 \leq M \sum_{m=1}^{M} \epsilon_m(\mathbf{x})^2 \Rightarrow \Big[\frac{1}{M}\sum_{m=1}^{M} \epsilon_m(\mathbf{x})\Big]^2 \leq \frac{1}{M}\sum_{m=1}^{M} \epsilon_m(\mathbf{x})^2$$

$$E_{Ensemble} \leq E_{Avg}$$

AGENDA

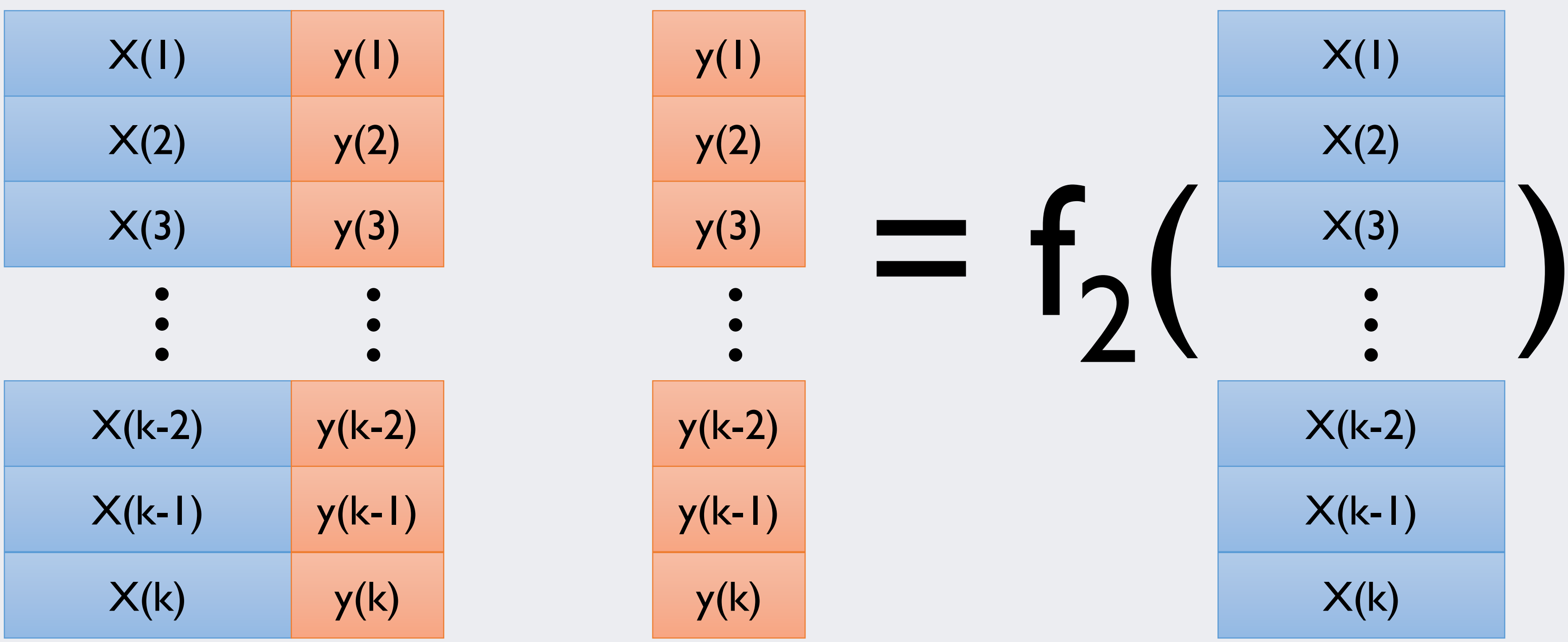
Sampling without Replacement

- K-fold data split
  - ✓ Entire data is split into k blocks; each classifier is trained only on different subset of (k-1) blocks



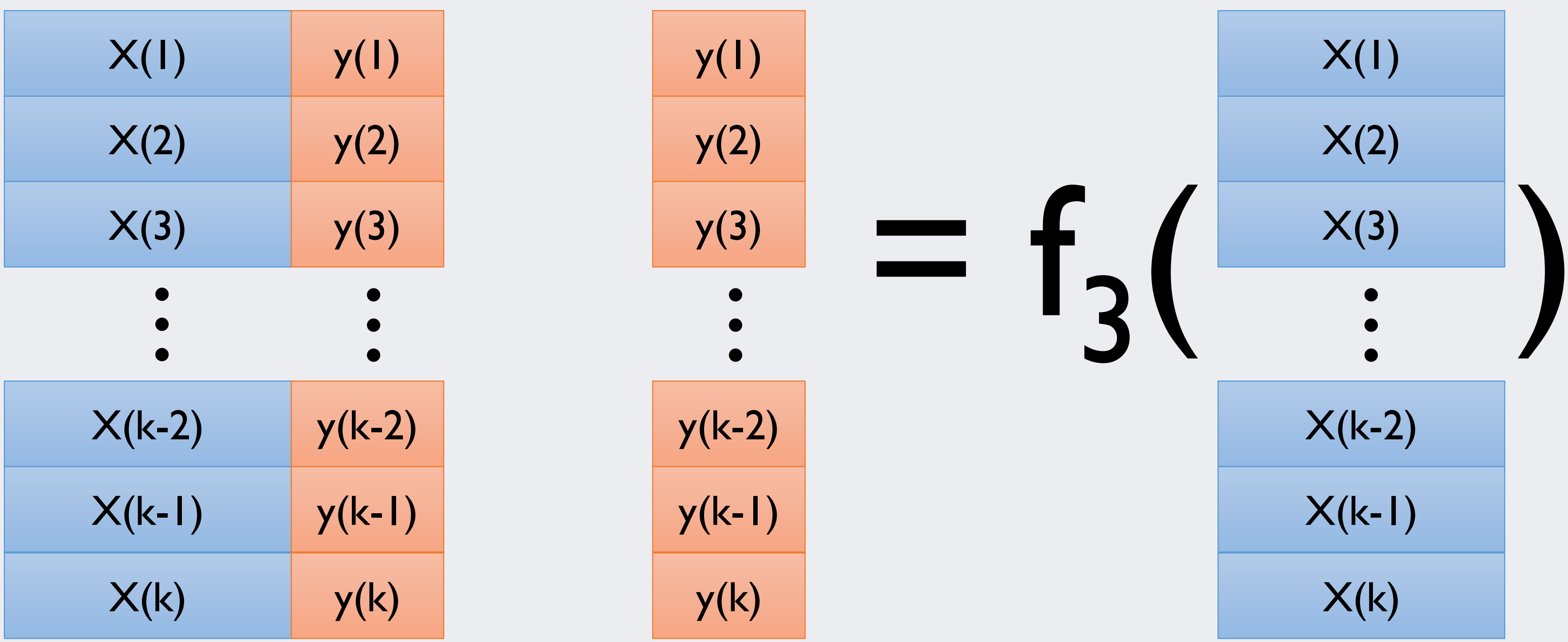
Sampling without Replacement

- K-fold data split
  - ✓ Entire data is split into k blocks; each classifier is trained only on different subset of (k-1) blocks



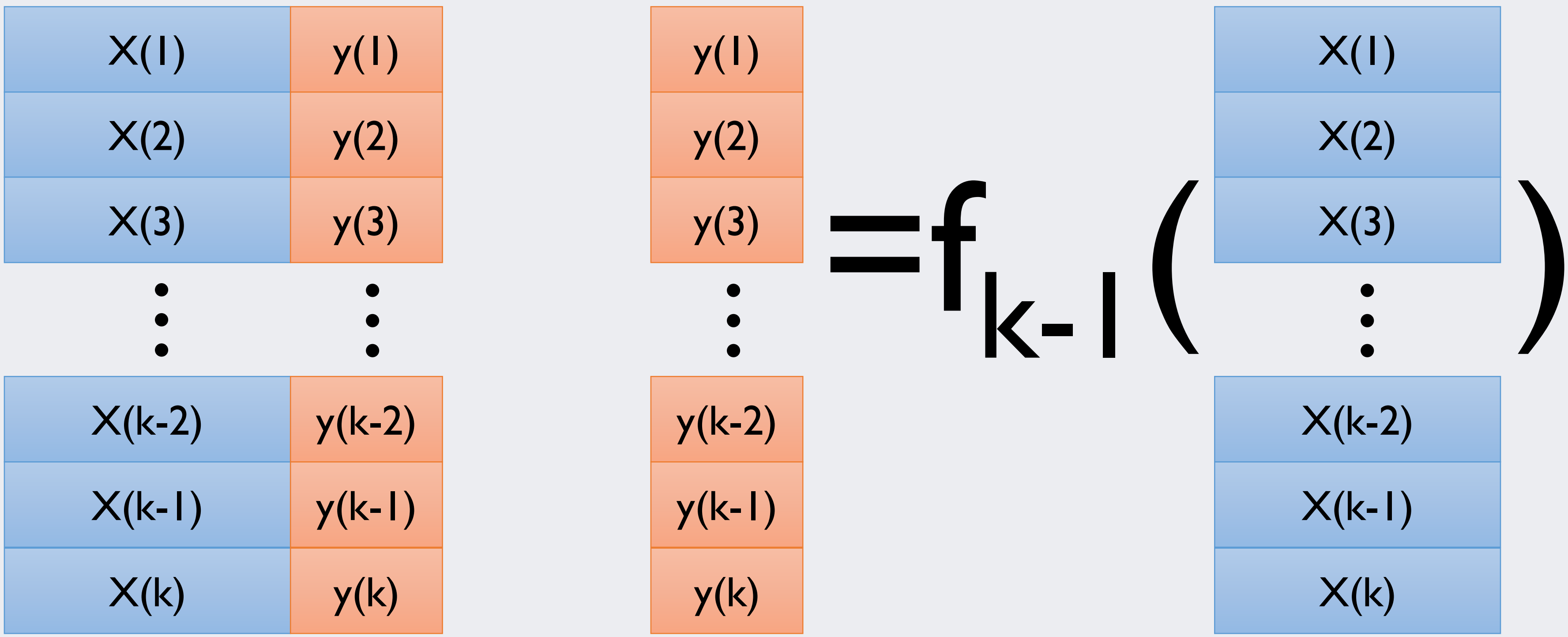
Sampling without Replacement

- K-fold data split
  - ✓ Entire data is split into k blocks; each classifier is trained only on different subset of (k-1) blocks



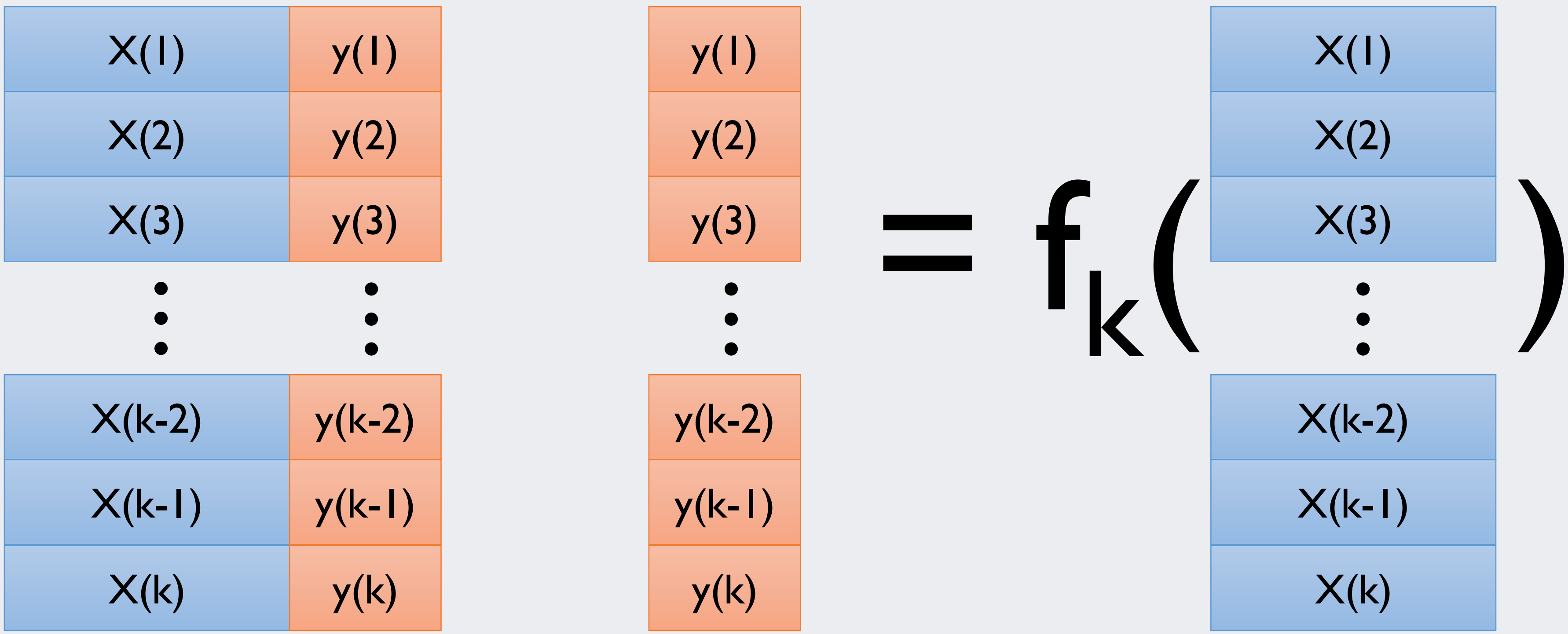
Sampling without Replacement

- K-fold data split
  - ✓ Entire data is split into k blocks; each classifier is trained only on different subset of (k-1) blocks



Sampling without Replacement

- K-fold data split
  - ✓ Entire data is split into k blocks; each classifier is trained only on different subset of (k-1) blocks



Sampling without Replacement

- K-fold data split
  - ✓ Entire data is split into k blocks; each classifier is trained only on different subset of (k-1) blocks
- Final output

$$\hat{y} = \delta\Big(f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_{k-1}(\mathbf{x}), f_k(\mathbf{x})\Big)$$

  - ✓ $\delta(\cdot)$: An aggregation function of individual outputs (ex: simple average)

Bootstrap Aggregating: Bagging

Breiman (1996)

- Main Idea
  - ✓ Each member of the ensemble is constructed from a different training dataset
  - ✓ Each dataset is generated by sampling from the total *N* data examples, choosing *N* items uniformly at random with replacement
  - ✓ Each dataset sample is known as a bootstrap

Original Dataset

x^1	y^1
x^2	y^2
x^3	y^3
x^4	y^4
x^5	y^5
x^6	y^6
x^7	y^7
x^8	y^8
x^9	y^9
x^{10}	y^{10}

Bootstrap 1

x^3	y^3
x^6	y^6
x^2	y^2
x^{10}	y^{10}
x^8	y^8
x^7	y^7
x^7	y^7
x^3	y^3
x^2	y^2
x^7	y^7

Bootstrap 2

x^7	y^7
x^1	y^1
x^{10}	y^{10}
x^1	y^1
x^8	y^8
x^6	y^6
x^2	y^2
x^6	y^6
x^4	y^4
x^9	y^9

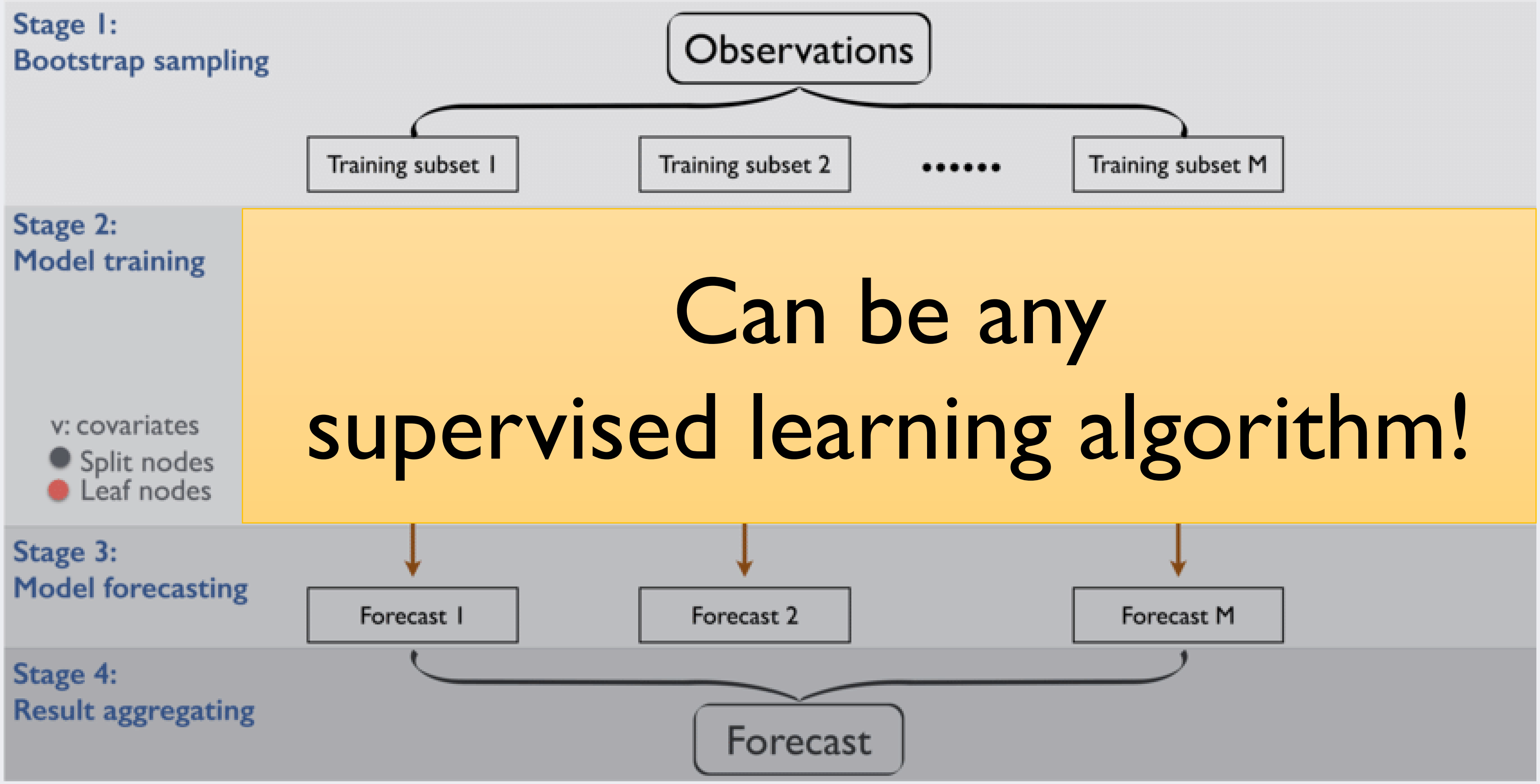
. . .

Bootstrap B

x^9	y^9
x^5	y^5
x^2	y^2
x^4	y^4
x^7	y^7
x^2	y^2
x^5	y^5
x^{10}	y^{10}
x^8	y^8
x^2	y^2

Bootstrap Aggregating: Bagging

- Bagging with Decision Tree



Bootstrap Aggregating: Bagging

- Result Aggregating
  - ✓ For classification problem
    - Majority voting

$$\hat{y}_{Ensemble} = arg\max_{i}\left(\sum_{j=1}^{n}\delta(\hat{y}_j = i), \quad i \in \{0,1\}\right)$$

Training Accuracy	Ensemble population	P(y=1) for a test instance	Predicted class label
0.80	Model 1	0.90	1
0.75	Model 2	0.92	1
0.88	Model 3	0.87	1
0.91	Model 4	0.34	0
0.77	Model 5	0.41	0
0.65	Model 6	0.84	1
0.95	Model 7	0.14	0
0.82	Model 8	0.32	0
0.78	Model 9	0.98	1
0.83	Model 10	0.57	1

$$\sum_{j=1}^{n}\delta(\hat{y}_j = 0) = 4$$

$$\sum_{j=1}^{n}\delta(\hat{y}_j = 1) = 6$$

$$\hat{y}_{Ensemble} = 1$$

Bootstrap Aggregating: Bagging

- Result Aggregating
  - ✓ For classification problem
    - Weighted voting (weight = training accuracy of individual models)

$$\hat{y}_{Ensemble} = arg\max_i \left(\frac{\sum_{j=1}^{n}(TrnAcc_j)\cdot\delta(\hat{y}_j = i)}{\sum_{j=1}^{n}(TrnAcc_j)}, \quad i \in \{0,1\} \right)$$

Training Accuracy	Ensemble population	P(y=1) for a test instance	Predicted class label
0.80	Model 1	0.90	1
0.75	Model 2	0.92	1
0.88	Model 3	0.87	1
0.91	Model 4	0.34	0
0.77	Model 5	0.41	0
0.65	Model 6	0.84	1
0.95	Model 7	0.14	0
0.82	Model 8	0.32	0
0.78	Model 9	0.98	1
0.83	Model 10	0.57	1

$$\frac{\sum_{j=1}^{n}(TrnAcc_j)\cdot\delta(\hat{y}_j = 0)}{\sum_{j=1}^{n}(TrnAcc_j)} = 0.424$$

$$\frac{\sum_{j=1}^{n}(TrnAcc_j)\cdot\delta(\hat{y}_j = 1)}{\sum_{j=1}^{n}(TrnAcc_j)} = 0.576$$

$$\hat{y}_{Ensemble} = 1$$

Bootstrap Aggregating: Bagging

- Result Aggregating
  - ✓ For classification problem
    - Weighted voting (weight = predicted probability for each class)

$$\hat{y}_{Ensemble} = arg\max_i \left(\frac{1}{n} \sum_{j=1}^{n} P(y=i), \quad i \in \{0,1\} \right)$$

Training Accuracy	Ensemble population	P(y=1) for a test instance	Predicted class label
0.80	Model 1	0.90	1
0.75	Model 2	0.92	1
0.88	Model 3	0.87	1
0.91	Model 4	0.34	0
0.77	Model 5	0.41	0
0.65	Model 6	0.84	1
0.95	Model 7	0.14	0
0.82	Model 8	0.32	0
0.78	Model 9	0.98	1
0.83	Model 10	0.57	1

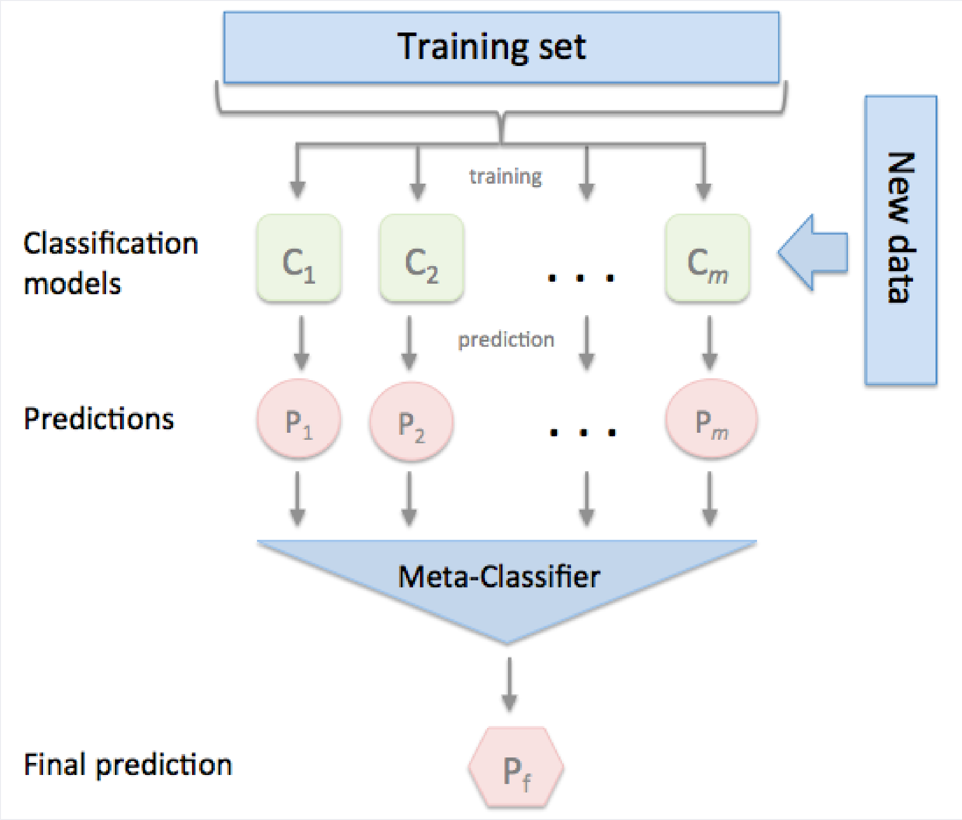
$$\frac{1}{n} \sum_{j=1}^{n} P(y=0) = 0.375$$

$$\frac{1}{n} \sum_{j=1}^{n} P(y=1) = 0.625$$

$$\hat{y}_{Ensemble} = 1$$

Bootstrap Aggregating: Bagging

- Result Aggregating: Stacking
  - ✓ Use another prediction model to aggregate the results
    - Input: Predictions made by ensemble members
    - Target: Actual true label



https://rasbt.github.io/mlxtend/user_guide/classifier/StackingClassifier/

Bootstrap Aggregating: Bagging

- Result Aggregating: Stacking
  - ✓ The winner of KDD-cup 2015
    - MOOC dropout prediction

*Jeong-Yoon Lee, Winning Data Science Competitions

Bootstrap Aggregating: Bagging

- Bagging: Algorithm

Algorithm 1 Bagging

Input: Required ensemble size T
Input: Training set $S = \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\}$
for $t = 1$ to T **do**
  Build a dataset S_t, by sampling N items, randomly *with replacement* from S.
  Train a model h_t using S_t, and add it to the ensemble.
end for
For a new testing point (x', y'),
If model outputs are continuous, combine them by averaging.
If model outputs are class labels, combine them by voting.

Bootstrap Aggregating: Bagging

- Bagging: Illustration

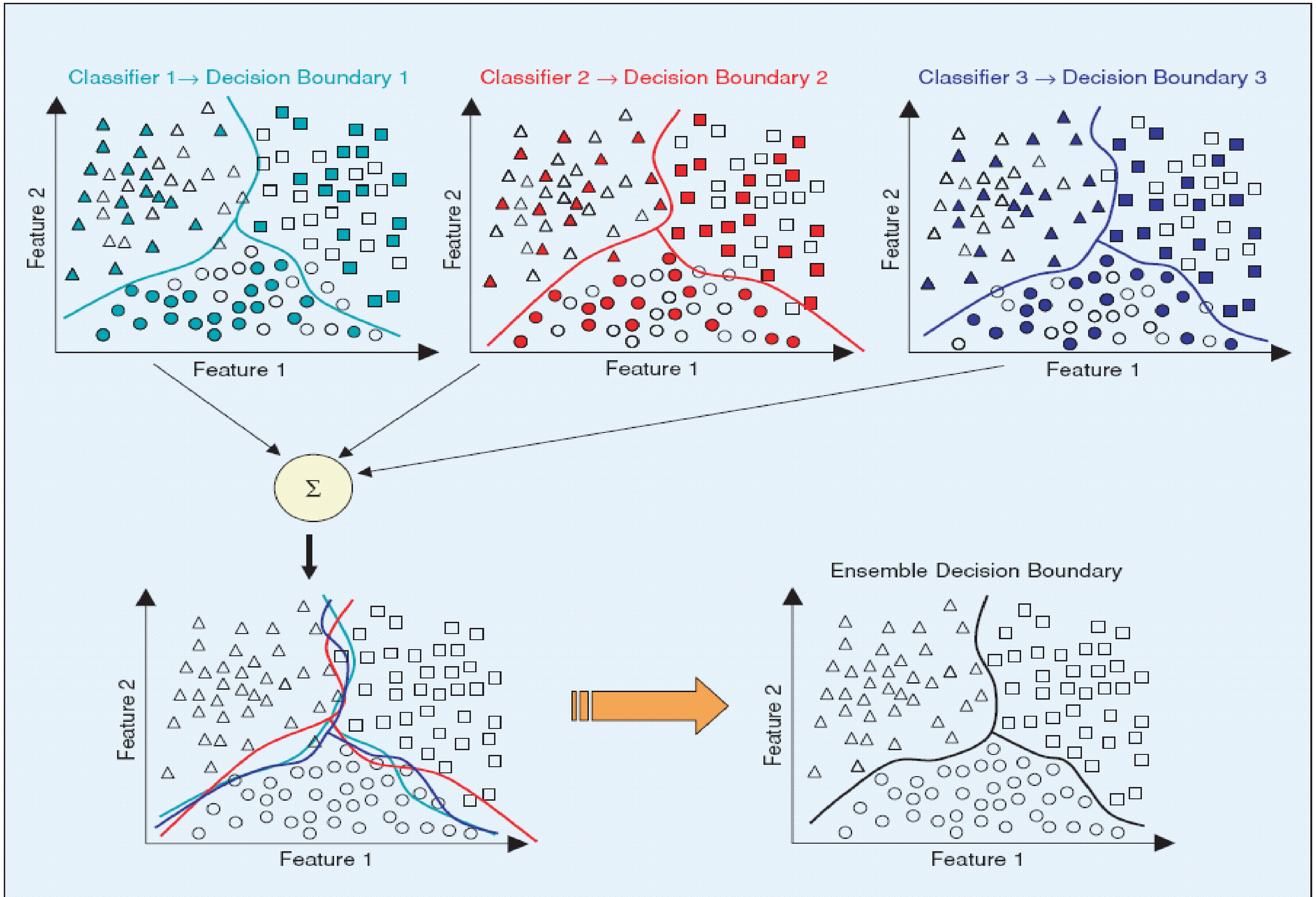
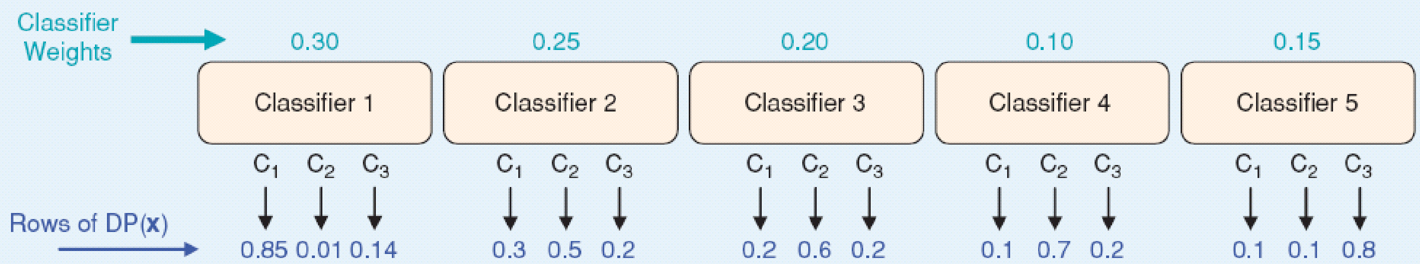


Figure 3. Combining classifiers that are trained on different subsets of the training data.

Bootstrap Aggregating: Bagging

- Aggregation examples

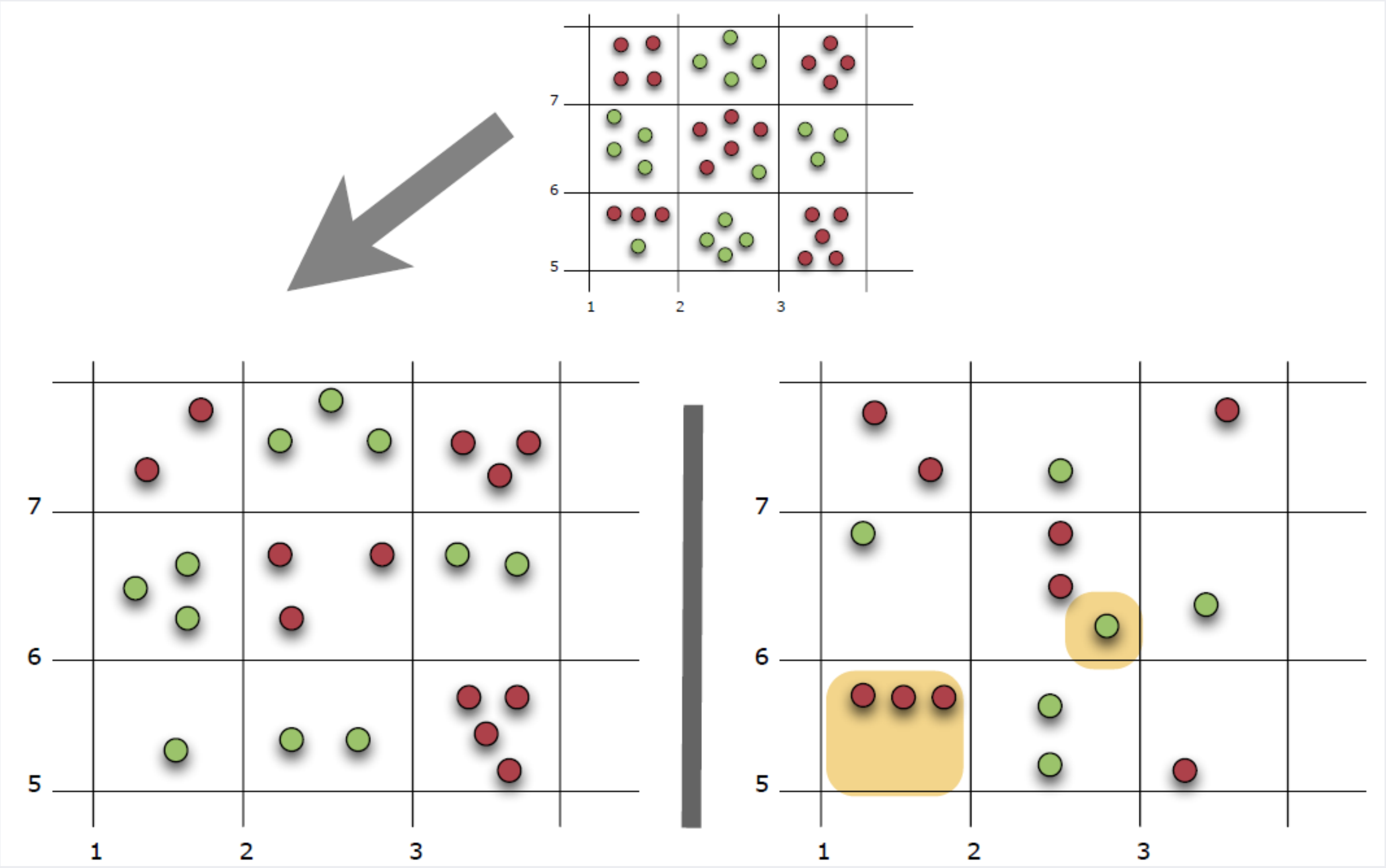


Product Rule:	**C_1: 0.0179,**	C_2: 0.00021,	C_3: 0.0009
Sum Rule:	C_1: 1.55,	**C_2: 1.91,**	C_3: 1.54
Mean Rule:	C_1: 0.310,	**C_2: 0.382,**	C_3: 0.308
Max Rule:	**C_1: 0.85,**	C_2: 0.7,	C_3: 0.8
Median Rule:	C_1: 0.2,	**C_2: 0.5,**	C_3: 0.2
Minimum Rule:	C_1: 0.1,	C_2: 0.01,	**C_3: 0.14**
Weighted Average	**C_1: 0.395,**	C_2: 0.333,	C_3: 0.272
Majority Voting:	C_1: 1,	**C_2: 3,**	C_3: 1 Vote
Weighted Majority Voting:	C_1: 0.3,	**C_2: 0.55,**	C_3: 0.15 Votes
Borda Count:	C_1: 5	**C_2: 6,**	C_3: 4 Votes

Figure 15. Example on various combination rules.

Bootstrap Aggregating: Bagging

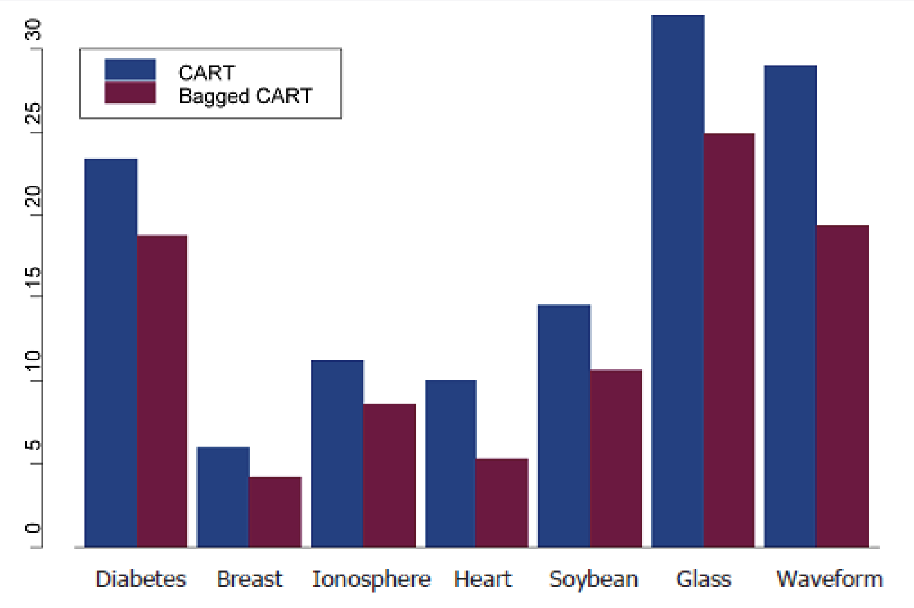
- Out of bag error (OOB Error)
  - ✓ Use the training instances that are not sampled for validation



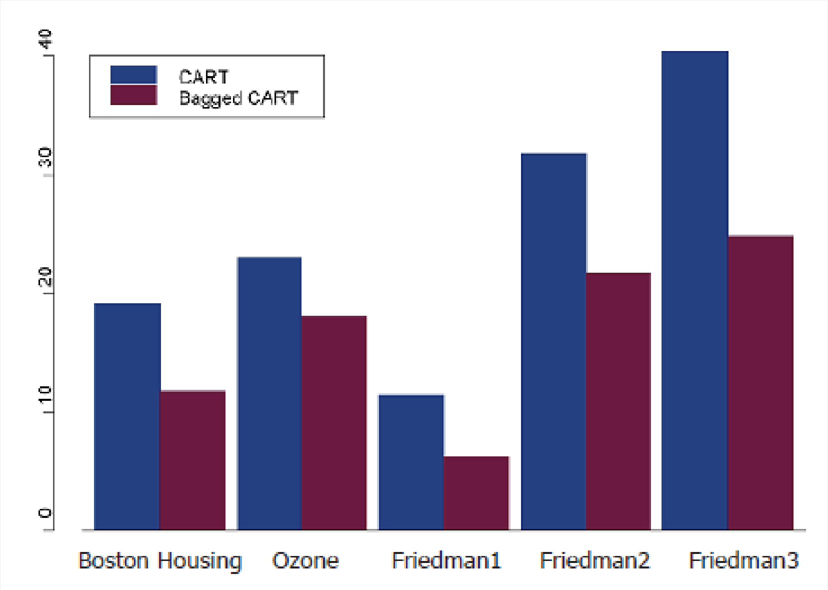
Bootstrap Aggregating: Bagging

- Bagged Trees vs. Single Tree

Classification



Regression



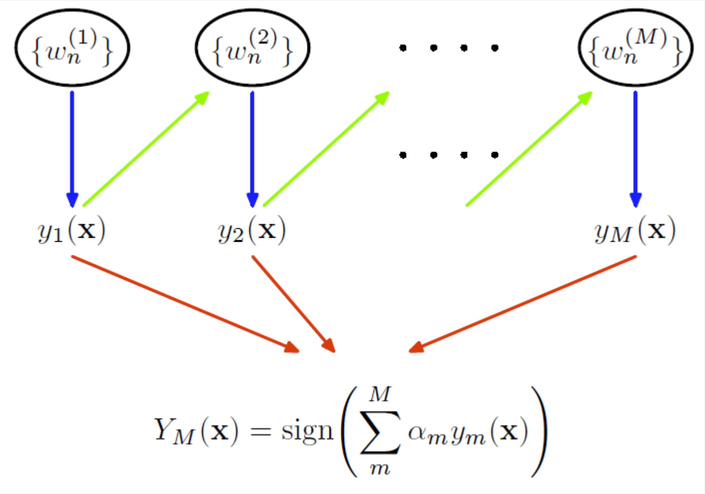
AGENDA

Boosting: AdaBoost

- AdaBoosting: Idea
  - ✓ Strong model vs. Weak model
    - A weak model, performing only slightly better than random guessing, could be boosted in to arbitrarily accurate strong model
  - ✓ New classifiers should focus on difficult cases
    - Examine the learning set
    - Get some rule of thumb
    - Reweight the examples of the training set, concentrate on hard cases for the previous rule
    - Derive the next rule of thumb
    - …
    - Build a single, accurate predictor by combining the rules of thumb

Boosting: AdaBoost

- AdaBoosting: Idea
  - ✓ Strong model vs. Weak model
    - A weak model, performing only slightly better than random guessing, could be **boosted** in to arbitrarily accurate strong model
  - ✓ Train models sequentially, with a new model training at each round
  - ✓ At the end of each round, misclassified examples are identified and have their emphasis increased in a new training set which is then fed back into the next round
  - ✓ Large errors made by earlier models can be compensated by the subsequent models



Boosting: AdaBoost

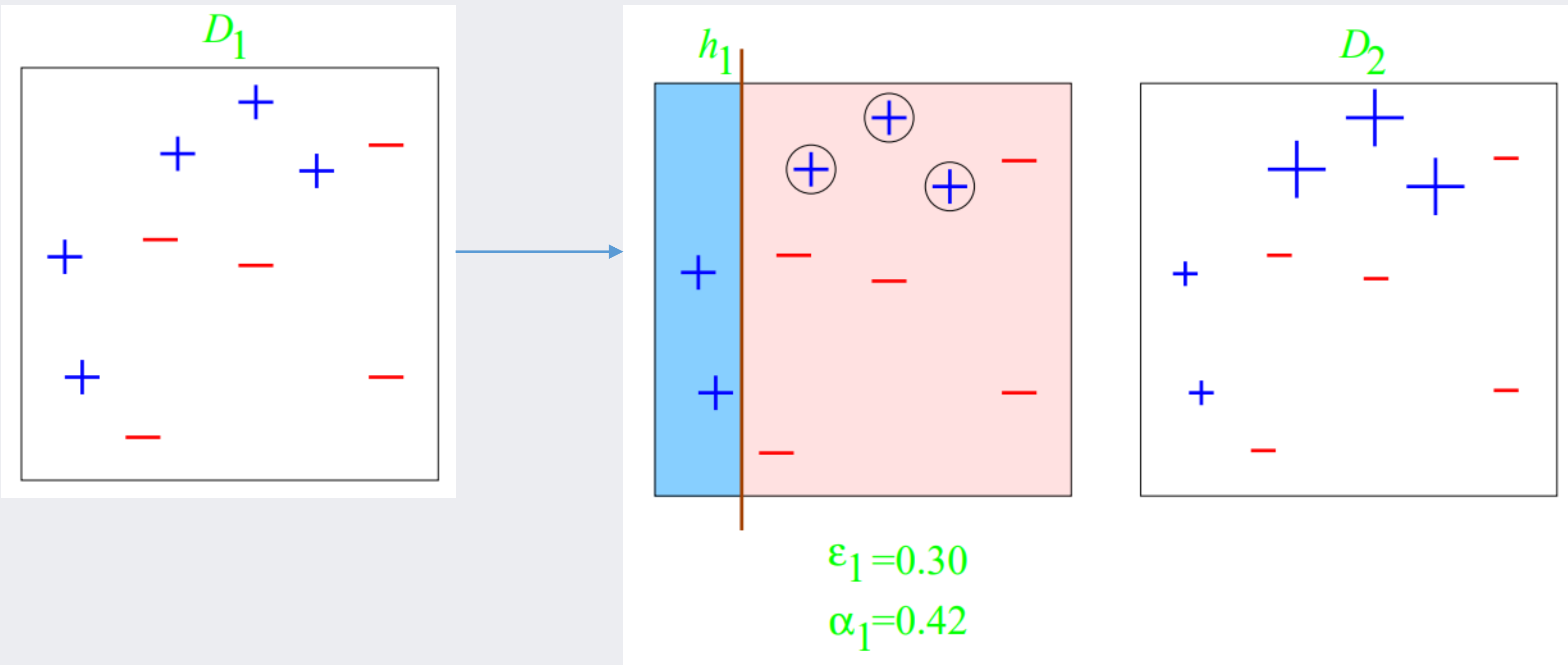
- AdaBoosting: Algorithm

Algorithm 2 Adaboost

Input: Required ensemble size T
Input: Training set $S = \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\}$, where $y_i \in \{-1, +1\}$
Define a uniform distribution $D_1(i)$ over elements of S.
for $t = 1$ to T **do**
  Train a model h_t using distribution D_t.
  Calculate $\epsilon_t = P_{D_t}(h_t(x) \neq y)$
  If $\epsilon_t \geq 0.5$ break
  Set $\alpha_t = \frac{1}{2} \ln\left(\frac{1-\epsilon_t}{\epsilon_t}\right)$
  Update $D_{t+1}(i) = \frac{D_t(i) \exp(-\alpha_t y_i h_t(x_i))}{Z_t}$
  where Z_t is a normalization factor so that D_{t+1} is a valid distribution.
end for
For a new testing point (x', y'),
$H(x') = sign\left(\sum_{t=1}^{T} \alpha_t h_t(x')\right)$

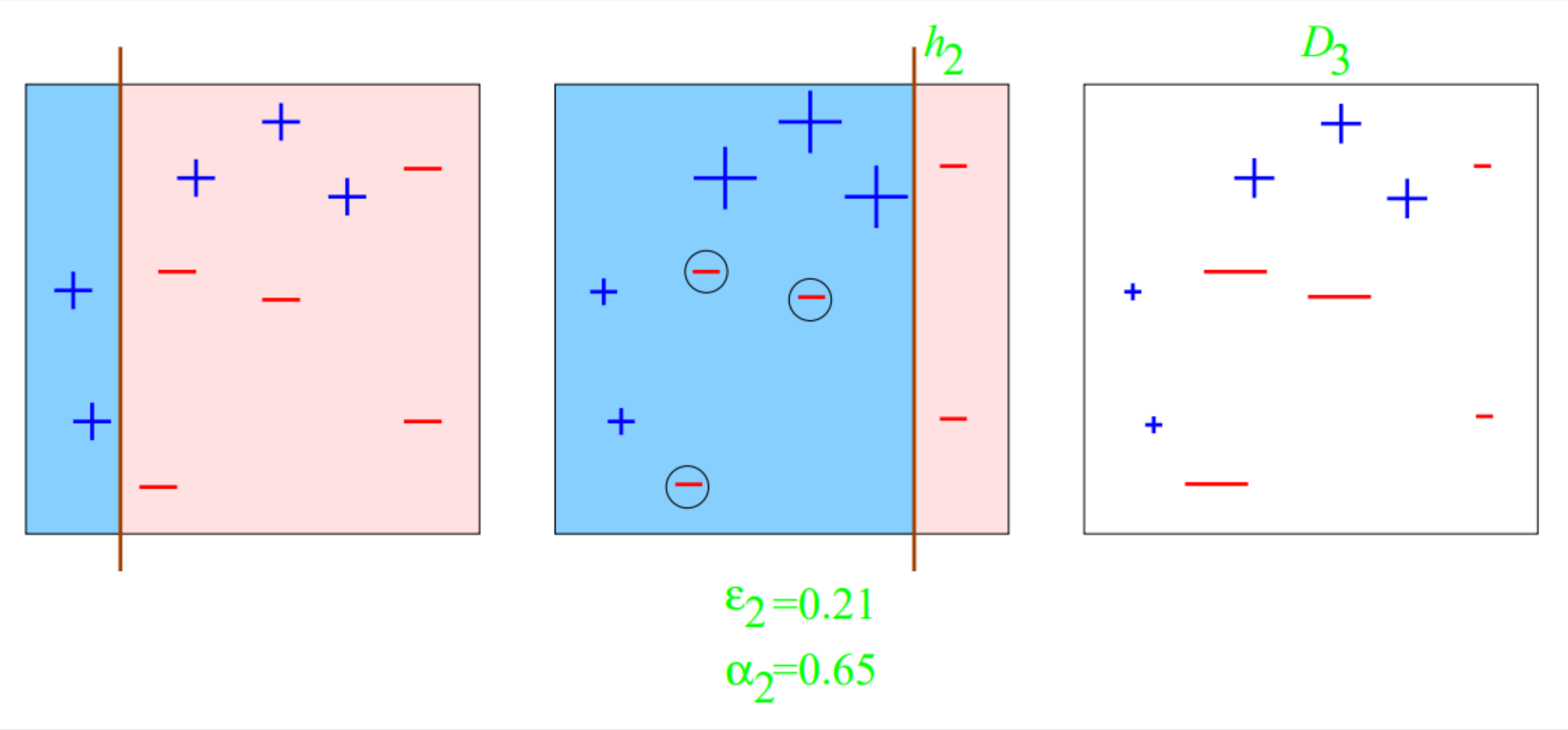
Boosting: AdaBoost

- Illustrative example 1
  - ✓ Round 1



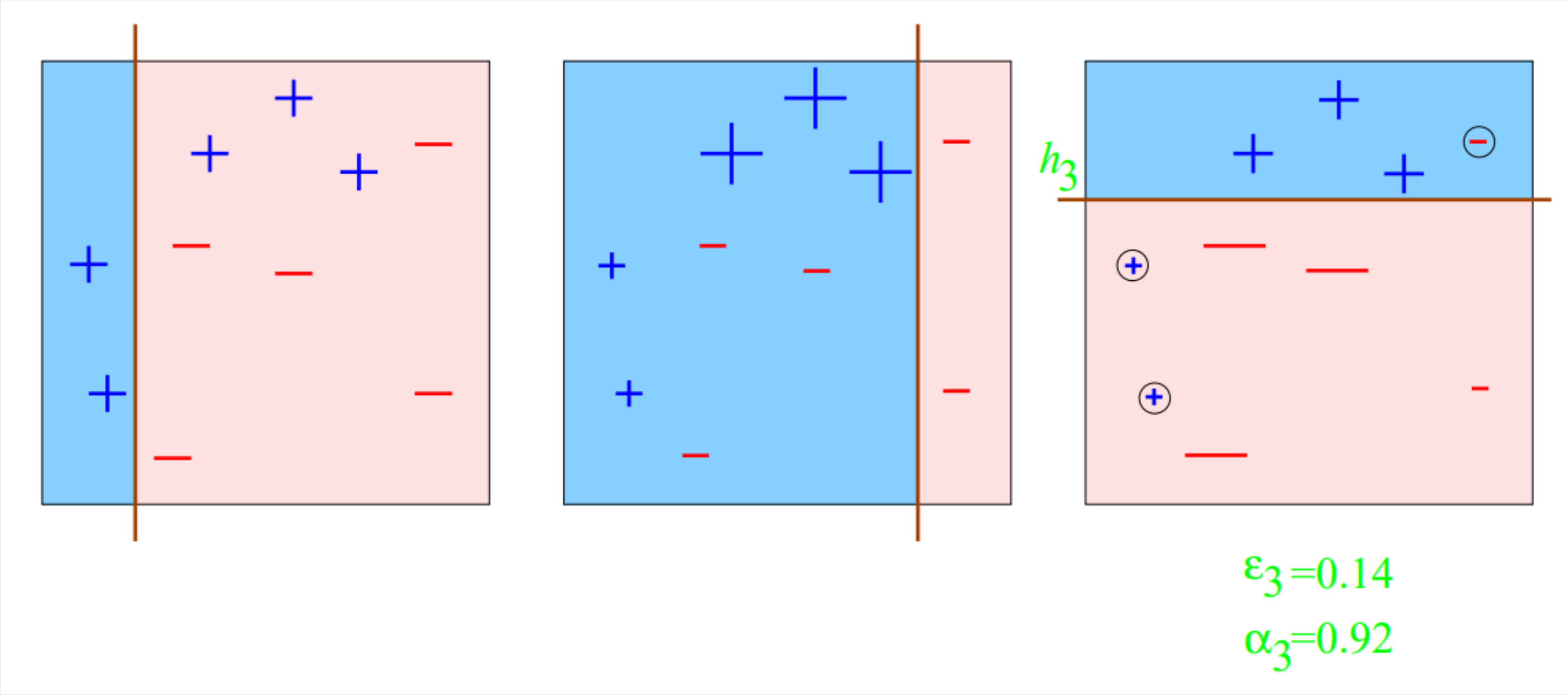
Boosting: AdaBoost

- Illustrative example 1
    - ✓ Round 2



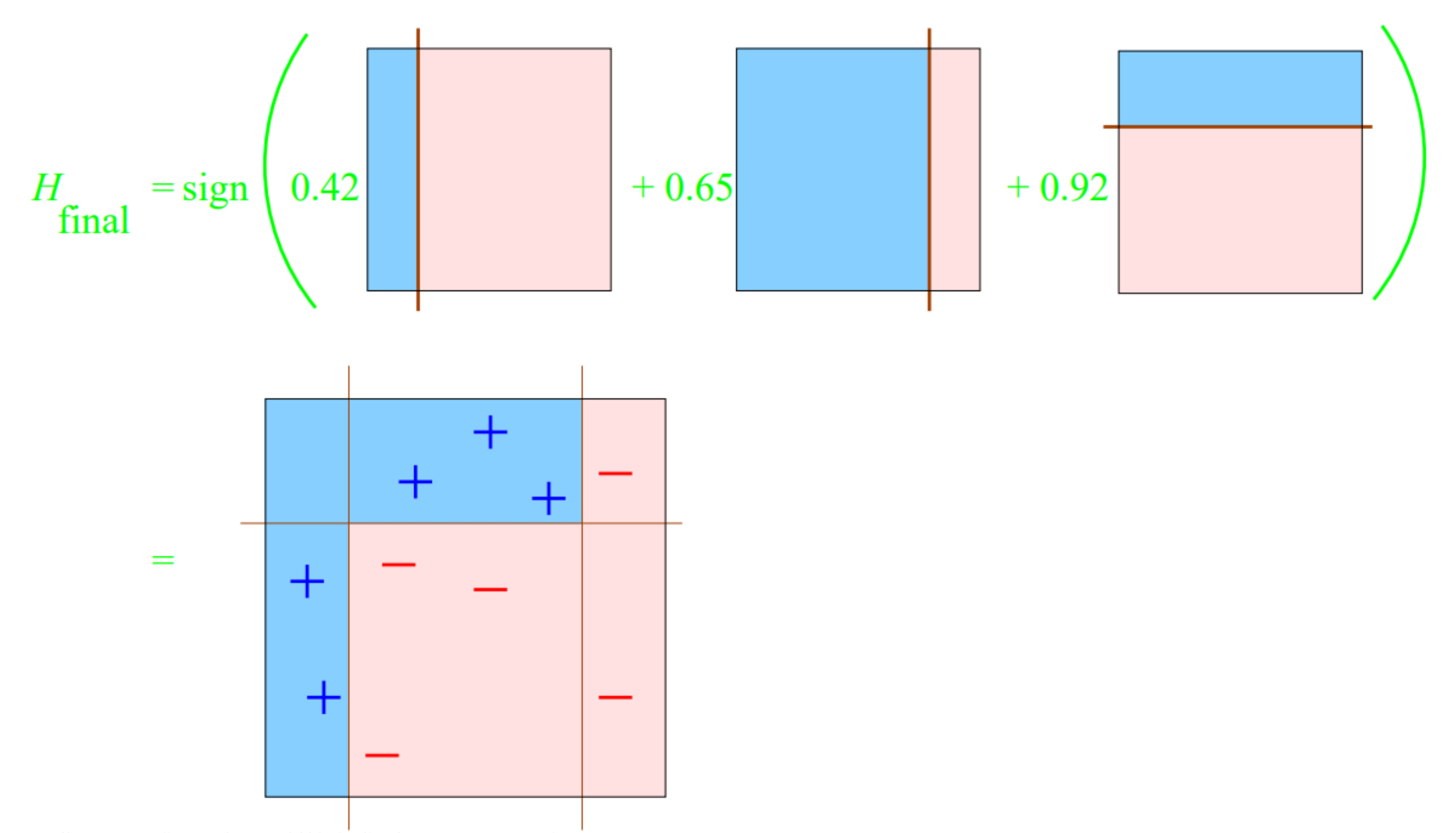
Boosting: AdaBoost

- Illustrative example 1
  - ✓ Round 3



Boosting: AdaBoost

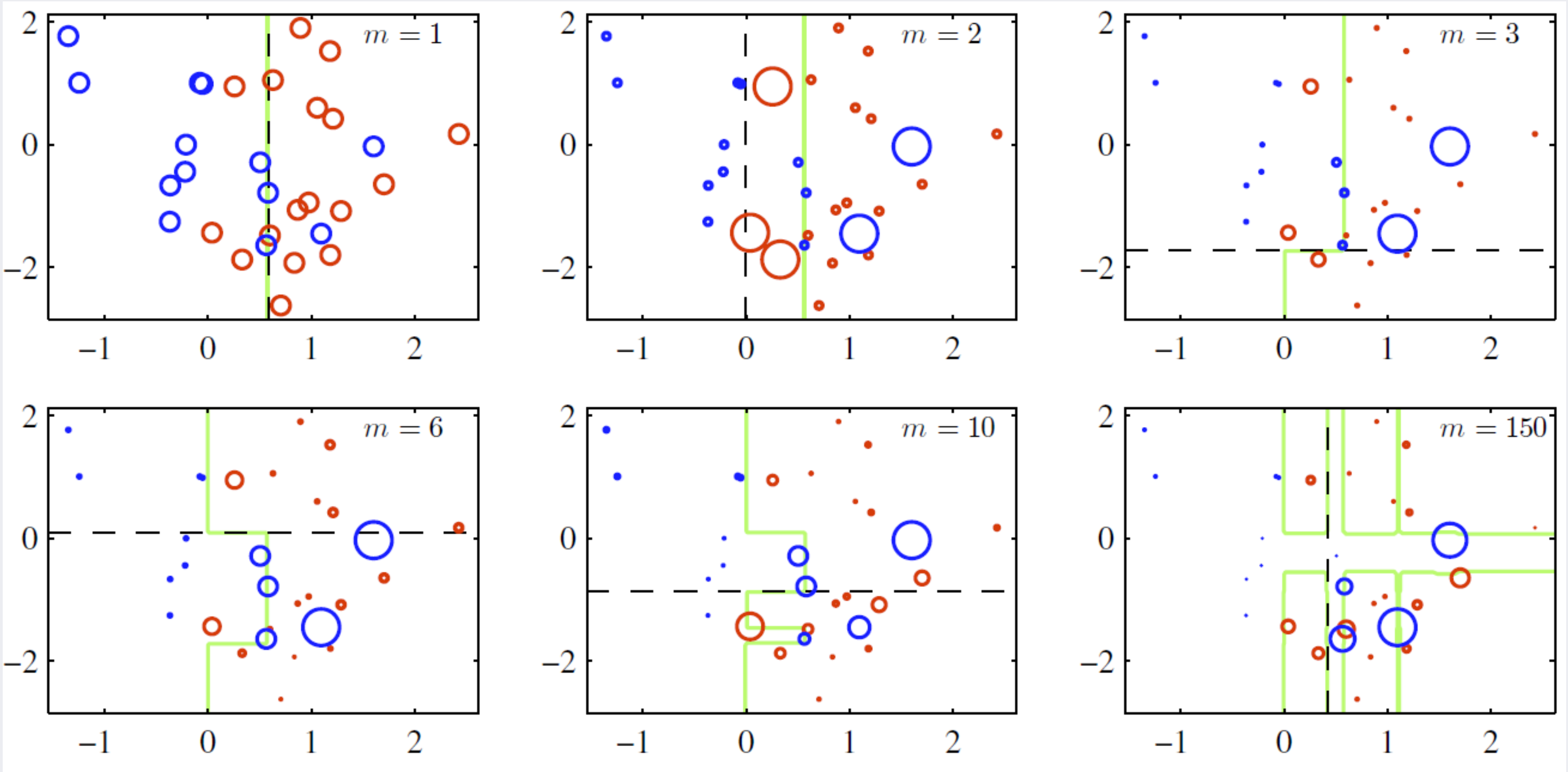
- Illustrative example 1
  - ✓ Final classifier



https://www.cse.buffalo.edu/~jcorso/t/CSE455/files/lecture_boosting.pdf

Boosting: AdaBoost

- Illustrative example 2



Boosting: AdaBoost

- AdaBoost in Action

https://www.youtube.com/watch?v=k4G2VCuOMMg

Boosting: AdaBoost

- Bagging vs. Boosting
  - ✓ Selected instances in each training dataset

A sample of a single classifier on an imaginary set of data.	
	(Original) Training Set
Training-set-1:	1, 2, 3, 4, 5, 6, 7, 8

A sample of Bagging on the same data.	
	(Resampled) Training Set
Training-set-1:	2, 7, 8, 3, 7, 6, 3, 1
Training-set-2:	7, 8, 5, 6, 4, 2, 7, 1
Training-set-3:	3, 6, 2, 7, 5, 6, 2, 2
Training-set-4:	4, 5, 1, 4, 6, 4, 3, 8

A sample of Boosting on the same data.	
	(Resampled) Training Set
Training-set-1:	2, 7, 8, 3, 7, 6, 3, 1
Training-set-2:	1, 4, 5, 4, 1, 5, 6, 4
Training-set-3:	7, 1, 5, 8, 1, 8, 1, 4
Training-set-4:	1, 1, 6, 1, 1, 3, 1, 5

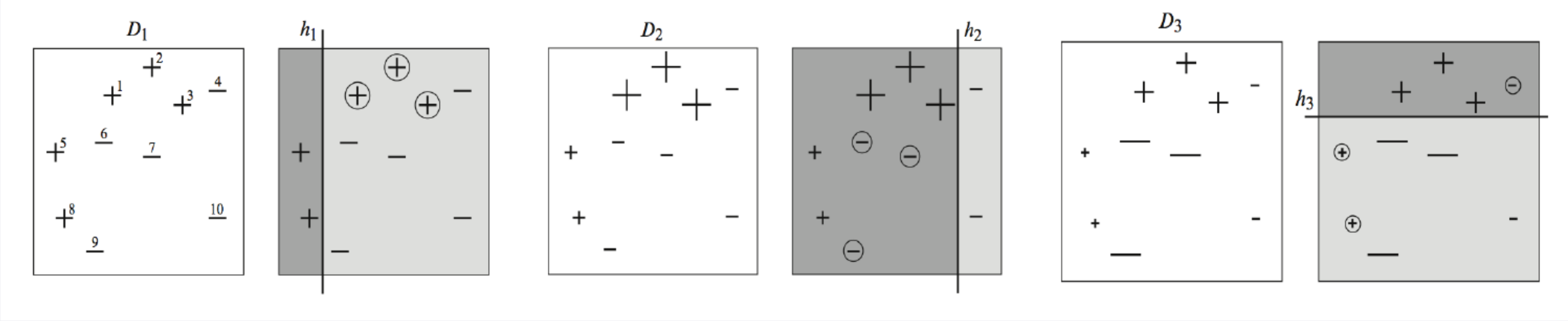
Boosting: AdaBoost

- Face detection with AdaBoost

Gradient Boosting Machine: GBM

Gradient Boosting = Gradient Descent + Boosting

- Adaboost



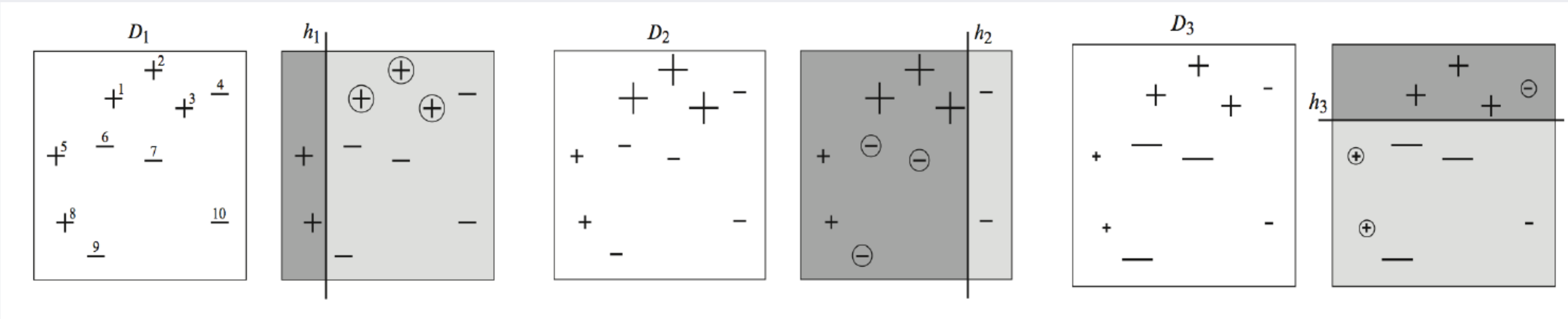
✓ Fit an additive model (ensemble) $\sum_t \rho_t h_t(x)$ in a forward stage-wise manner.

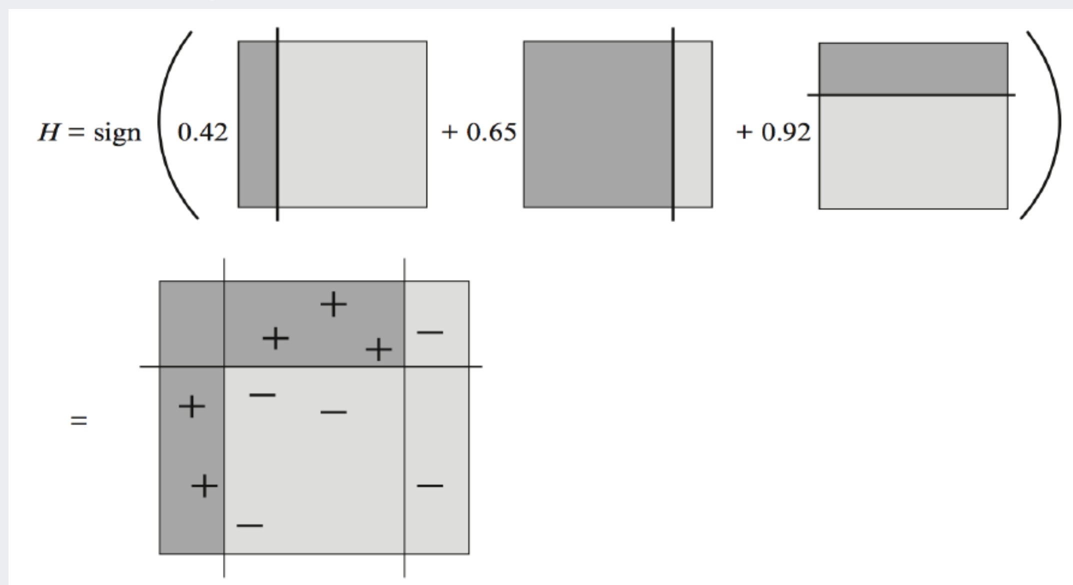
✓ In each stage, introduce a weak leaner to compensate the shortcomings of existing weak leaners.

✓ In Adaboost, "shortcomings" are identified by high-weight data points.

Gradient Boosting Machine: GBM

- Adaboost





$$H(x) = \sum_t \rho_t h_t(x)$$

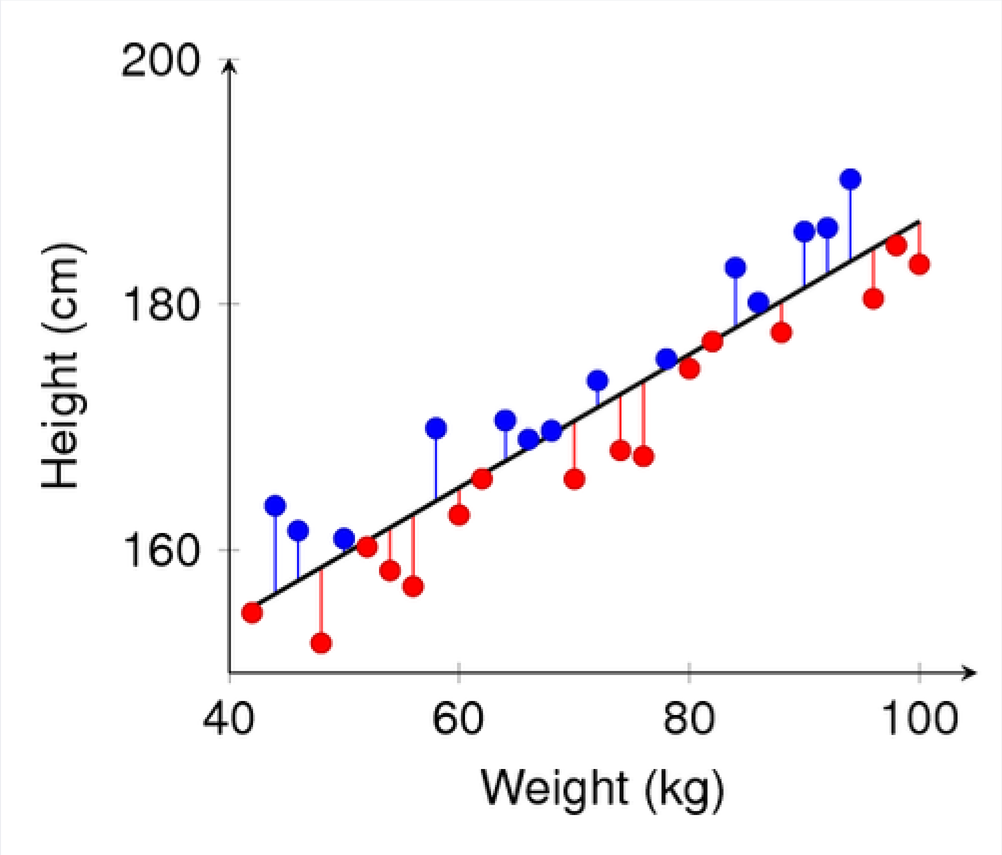
Gradient Boosting Machine: GBM

- Gradient Boosting
  - ✓ Fit an additive model (ensemble) $\sum_t \rho_t h_t(x)$ in a forward stage-wise manner.
  - ✓ In each stage, introduce a weak leaner to compensate the shortcomings of existing weak leaners.
  - ✓ In Gradient Boosting, "shortcomings" are identified by gradients.
  - ✓ Both high-weight data points and gradients tell us how to improve our model.

- Gradient Boosting for Different Problems
  - ✓ Difficulty: Regression < Classification < Ranking
    - Associated with the complexity of the derivative of a loss function

Gradient Boosting Machine: GBM

- Motivation (for regression problem)
  - ✓ What if we attempt to predict the residuals with the additional regression model?



Gradient Boosting Machine: GBM

- Main idea

Original Dataset

x^1	y^1
x^2	y^2
x^3	y^3
x^4	y^4
x^5	y^5
x^6	y^6
x^7	y^7
x^8	y^8
x^9	y^9
x^{10}	y^{10}

Modified Dataset 1

x^1	$y^1-f_1(x^1)$
x^2	$y^2-f_1(x^2)$
x^3	$y^3-f_1(x^3)$
x^4	$y^4-f_1(x^4)$
x^5	$y^5-f_1(x^5)$
x^6	$y^6-f_1(x^6)$
x^7	$y^7-f_1(x^7)$
x^8	$y^8-f_1(x^8)$
x^9	$y^9-f_1(x^9)$
x^{10}	$y^{10}-f_1(x^{10})$

Modified Dataset 2

x^1	$y^1-f_1(x^1)-f_2(x^1)$
x^2	$y^2-f_1(x^2)-f_2(x^2)$
x^3	$y^3-f_1(x^3)-f_2(x^3)$
x^4	$y^4-f_1(x^4)-f_2(x^4)$
x^5	$y^5-f_1(x^5)-f_2(x^5)$
x^6	$y^6-f_1(x^6)-f_2(x^6)$
x^7	$y^7-f_1(x^7)-f_2(x^7)$
x^8	$y^8-f_1(x^8)-f_2(x^8)$
x^9	$y^9-f_1(x^9)-f_2(x^9)$
x^{10}	$y^{10}-f_1(x^{10})-f_2(x^{10})$

• • •

$$y = f_1(\mathbf{x}) \qquad y - f_1(\mathbf{x}) = f_2(\mathbf{x}) \qquad y - f_1(\mathbf{x}) - f_2(\mathbf{x}) = f_3(\mathbf{x})$$

Gradient Boosting Machine: GBM

- How is this idea related to the gradient?
  - ✓ Loss function of the ordinary least square (OLS)

$$\min L = \frac{1}{2}\sum_{i=1}^{n}(y_i - f(\mathbf{x}_i))^2$$

  - ✓ Gradient of the Loss function

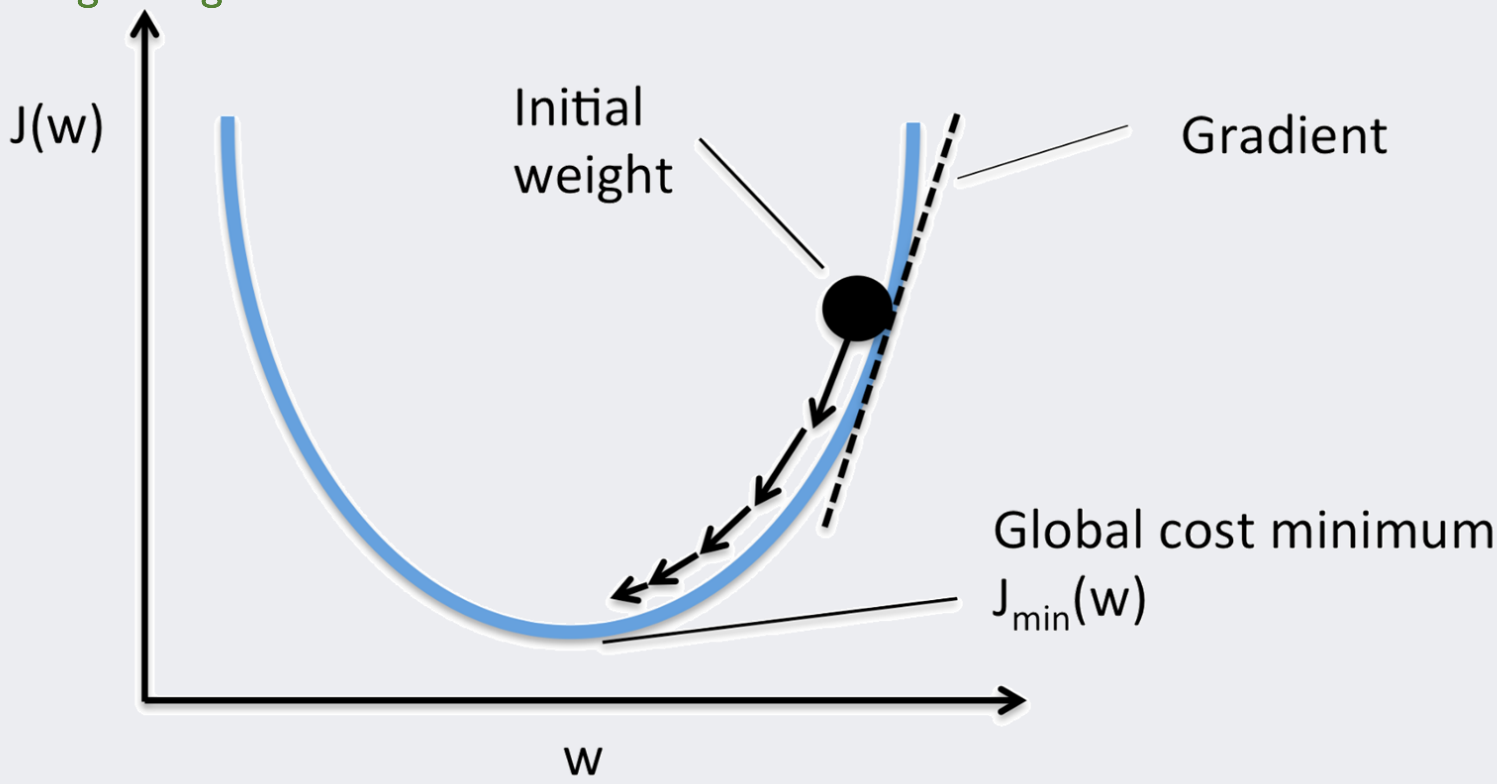
$$\frac{\partial L}{\partial f(\mathbf{x}_i)} = f(\mathbf{x}_i) - y_i$$

  - ✓ Residuals are the negative gradient of the loss function

$$y_i - f(\mathbf{x}_i) = -\frac{\partial L}{\partial f(\mathbf{x}_i)}$$

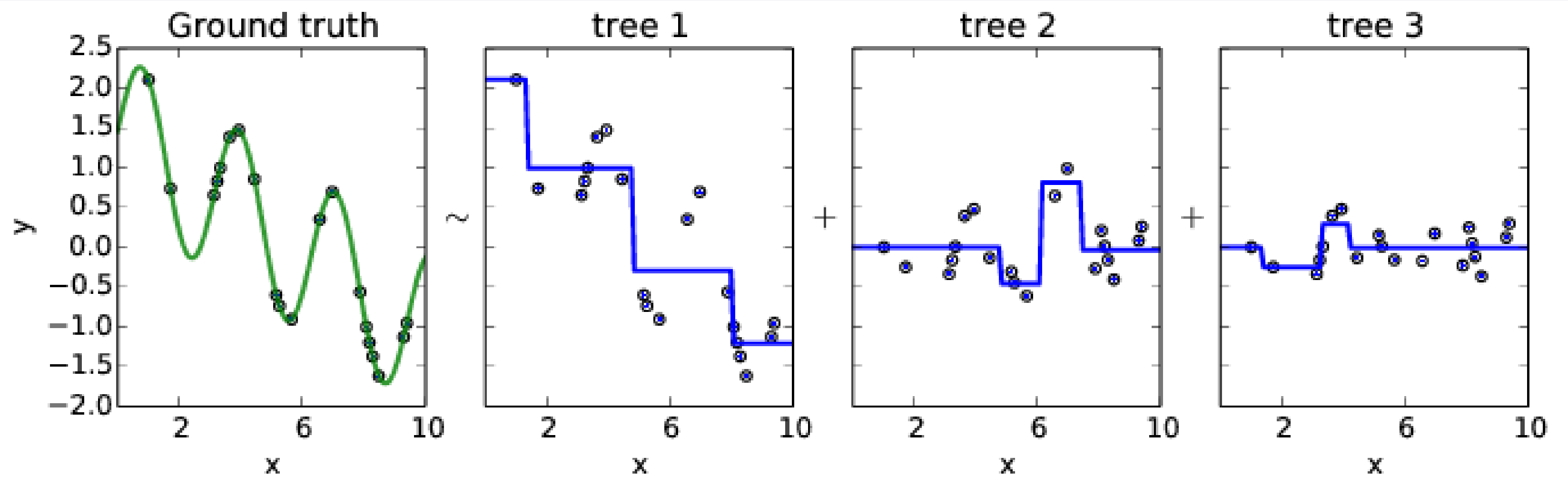
Gradient Boosting Machine: GBM

- Gradient Descent Algorithm
  - ✓ Blue line: value of loss function with a given parameter
  - ✓ Black point: current state
  - ✓ Arrows: the direction that the parameter should follow to minimize the loss function
    = negative gradient of the loss function



Gradient Boosting Machine: GBM

- GBM
  - ✓ Regression example



https://www.quora.com/How-would-you-explain-gradient-boosting-machine-learning-technique-in-no-more-than-300-words-to-non-science-major-college-students

Gradient Boosting Machine: GBM

Friedman (2001), Natekin and Knoll (2013)

- Gradient Boosting: Algorithm

1. Initialize $f_0(x) = \arg\min_\gamma \sum_{i=1}^{N} L(y_i, \gamma)$.
2. For $m = 1$ to M:

   2.1 For $i = 1, \ldots, N$ compute

   $$g_{im} = \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x_i)=f_{m-1}(x_i)}$$

   2.2 Fit a regression tree to the targets g_{im} giving terminal regions $R_{jm}, j = 1, \ldots, J_m$.

   2.3 For $j = 1, \ldots, J_m$ compute

   $$\gamma_{jm} = \arg\min_{\gamma} \sum_{x_i \in R_{jm}} L(y_i, f_{m-1}(x_i) + \gamma)$$

   2.4 Update $f_m(x) = f_{m-1}(x) + \sum_{j=1}^{J_m} \gamma_{jm} I(x \in R_{jm})$

3. Output $\hat{f}(x) = f_M(x)$.

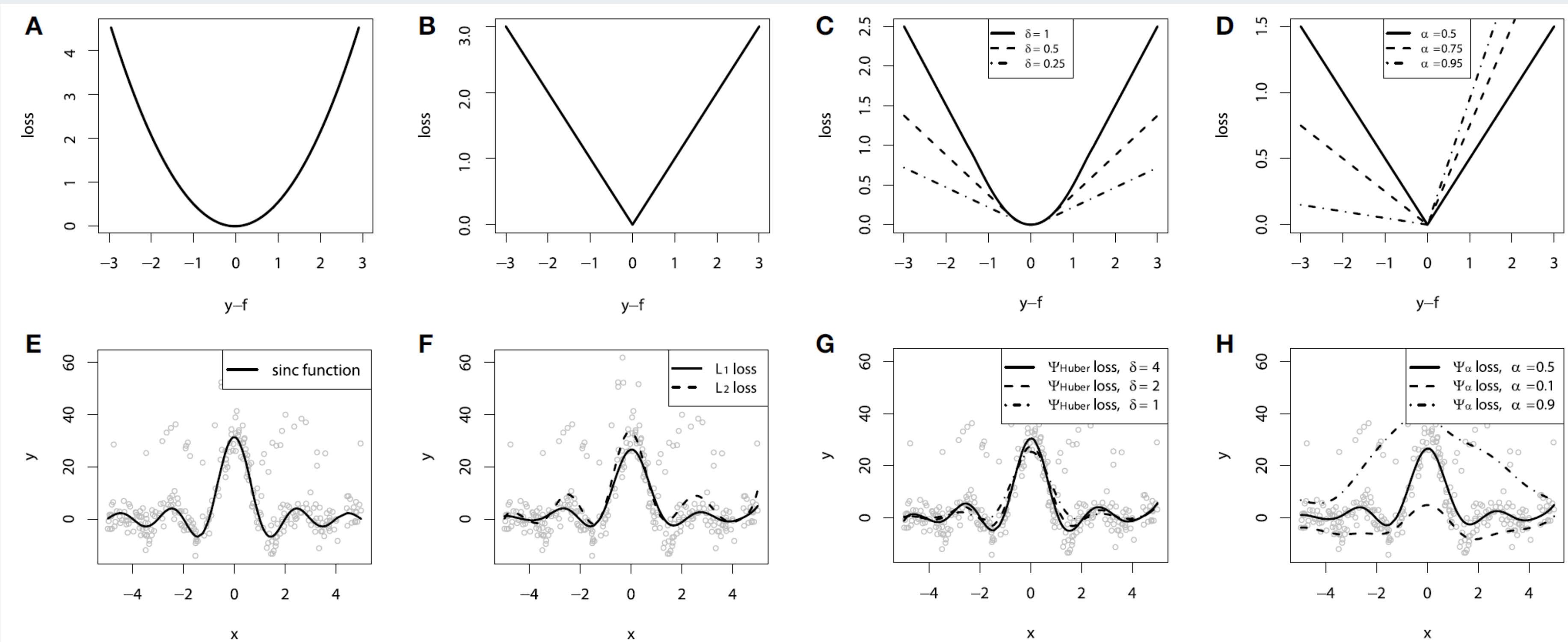
Gradient Boosting Machine: GBM

- Loss Functions for Regression

Loss Function	Formula
Squared loss (L_2)	$\Psi(y,f)_{L_2} = \frac{1}{2}(y-f)^2$
Absolute loss (L_1)	$\Psi(y,f)_{L_1} = \lvert y-f \rvert$
Huber loss	$\Psi(y,f)_{\text{Huber},\,\delta} = \begin{cases} \frac{1}{2}(y-f)^2 & \lvert y-f\rvert \le \delta \\ \delta(\lvert y-f\rvert - \delta/2) & \lvert y-f\rvert > \delta \end{cases}$
Quantile loss	$\Psi(y,f)_{\alpha} = \begin{cases} (1-\alpha)\lvert y-f\rvert & y-f \le 0 \\ \alpha\lvert y-f\rvert & y-f > 0 \end{cases}$

Gradient Boosting Machine: GBM

- Loss Functions for Regression



Gradient Boosting Machine: GBM

- Loss Functions for Classification

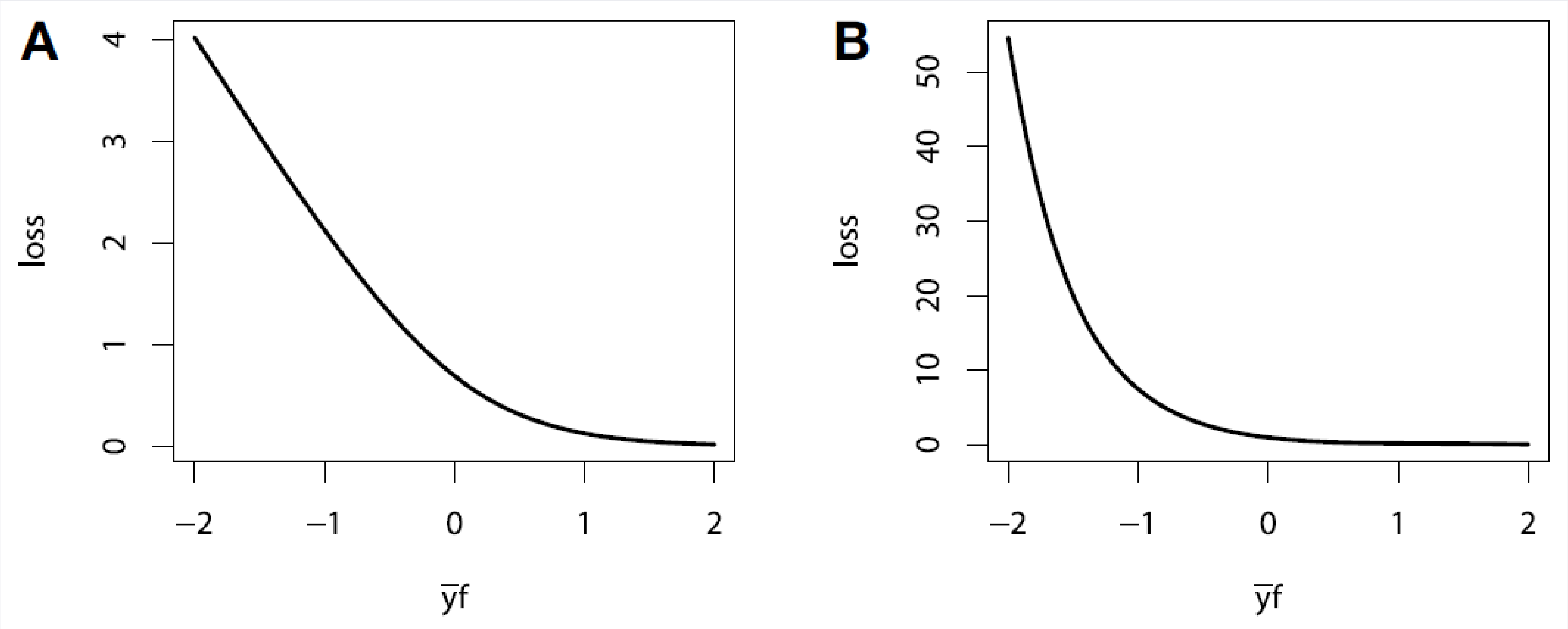
Loss Function	Formula
Bernoulli loss	$\Psi(y,f)_{\text{Bern}} = \log(1+\exp(-2\bar{y}f))$
Adaboost loss	$\Psi(y,f)_{\text{Ada}} = \exp(-\bar{y}f)$

(Note)

In binary classification, the target is usually defined by $y \in \{0,1\}$, but here we define $\bar{y} = 2y - 1$ so that $\bar{y} \in \{-1,1\}$

Gradient Boosting Machine: GBM

- Loss Functions for Classification



(A) Bernoulli loss function. **(B)** Adaboost loss function.

Gradient Boosting Machine: GBM

- Overfitting problem in GBM

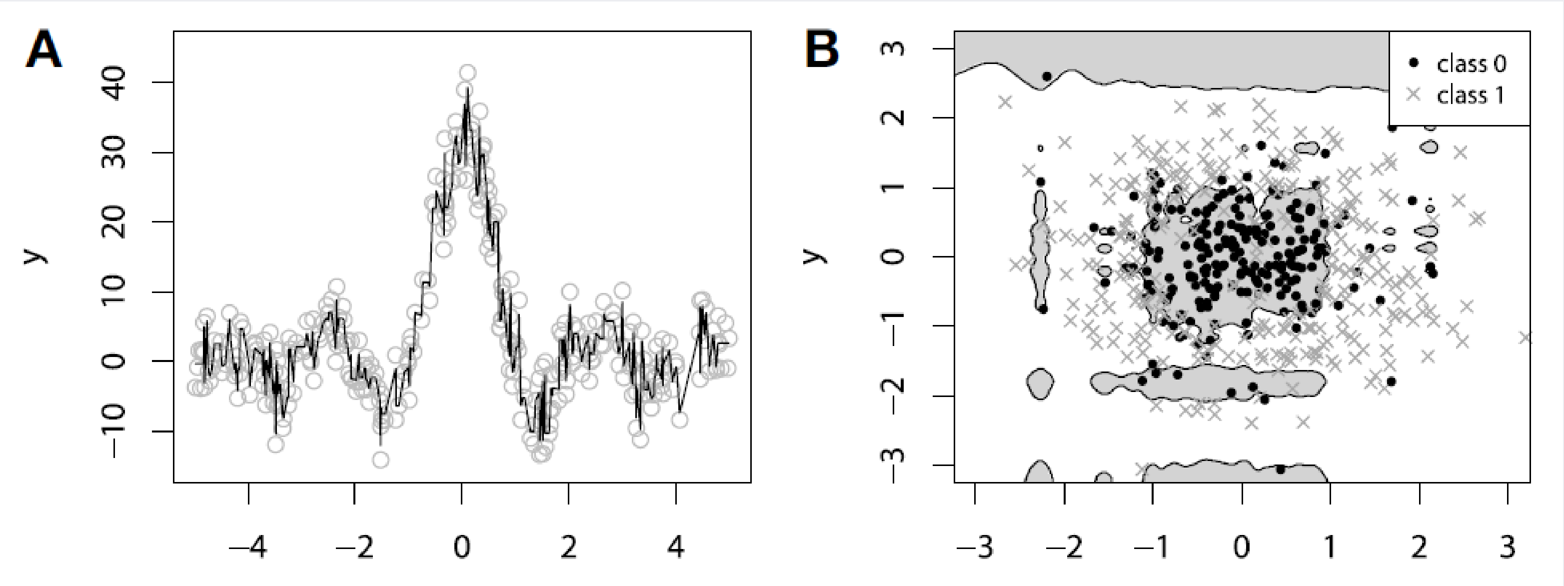


FIGURE 4 | Examples of overfitting in GBMs on: (A) regression task; (B) classification task. Demonstration of fitting a decision-tree GBM to a noisy sinc(x) data: **(C)** $M = 100$, $\lambda = 1$; **(D)** $M = 1000$, $\lambda = 1$; **(E)** $M = 100$, $\lambda = 0.1$; **(F)** $M = 1000$, $\lambda = 0.1$.

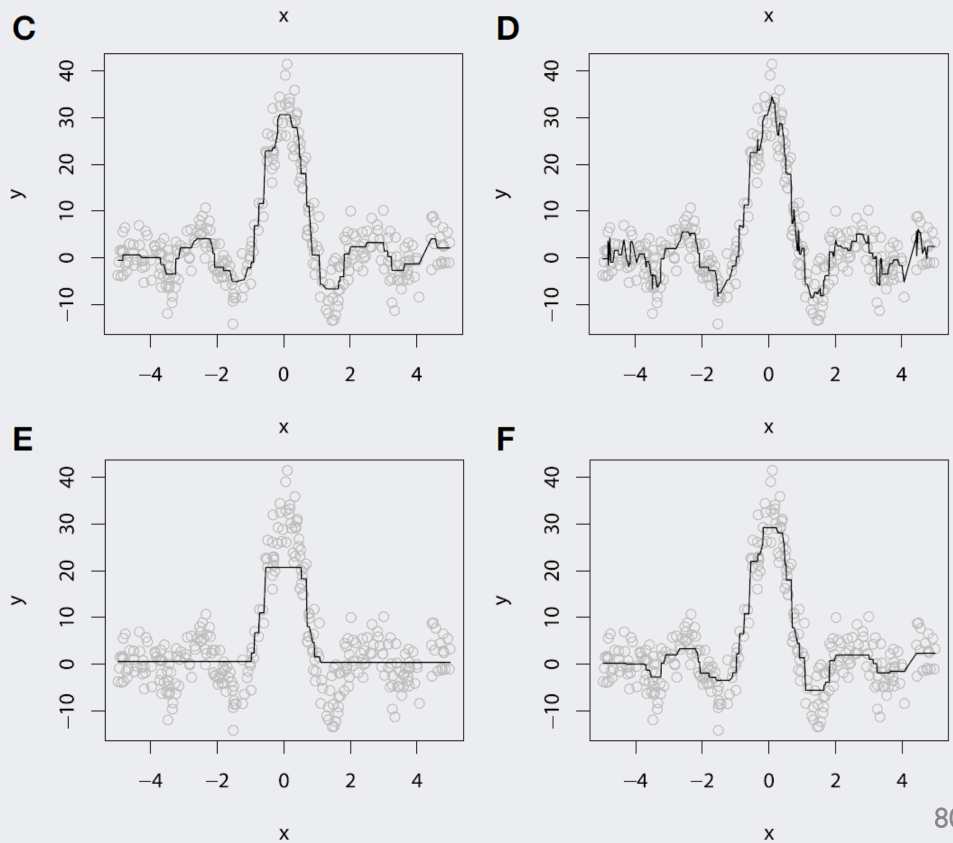
Gradient Boosting Machine: GBM

- Regularization
  - ✓ Subsampling
    - At each learning iteration, only a random part of the training data is used to fit a consecutive base-learner.
    - The training data is typically sampled without replacement, but bagging can be also acceptable.

Gradient Boosting Machine: GBM

- Regularization
  - ✓ Shrinkage
    - Used for reducing/shrinking the impact of each additional fitted base-leaners.
    - Better to improve a model by taking many small steps than by taking fewer large steps.

$$\widehat{f}_t \leftarrow \widehat{f}_{t-1} + \lambda \rho_t h(x, \theta_t)$$



Gradient Boosting Machine: GBM

- Regularization
  - ✓ Early Stopping
    - Use the validation error

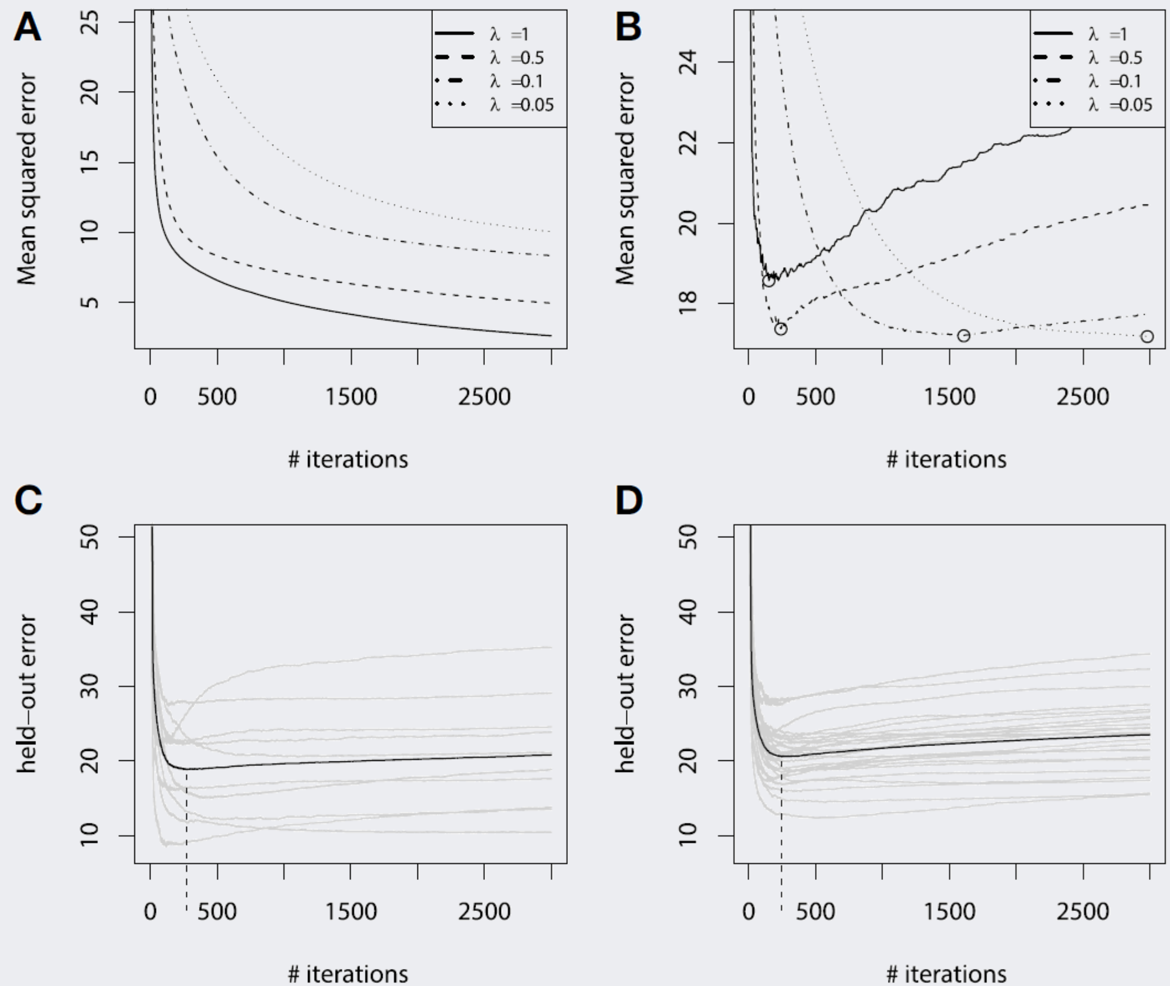


FIGURE 5 | Error curves for GBM fitting on sinc(x) data: (A) training set error; (B) validation set error. Error curves for learning simulations and number of base-learners *M* estimation: **(C)** error curves for cross-validation; **(D)** error curves for bootstrap estimates.

Gradient Boosting Machine: GBM

- Variable Importance in Tree-based Gradient Boosting
  - ✓ $Influence_j(T)$: importance of the variable j in a single tree T.
  - ✓ Assume that there are L terminal nodes → $L-1$ splits.

$$Influence_j(T) = \sum_{i=1}^{L-1} (IG_i \times \mathbf{1}(S_i = j))$$

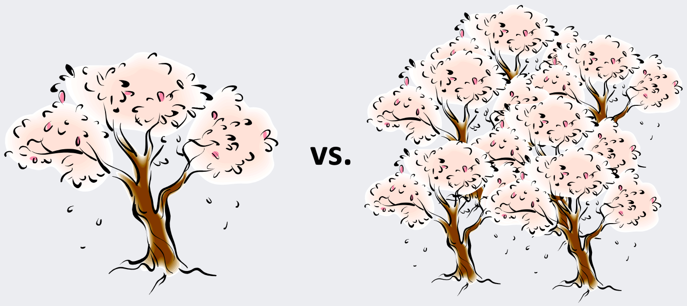
  - ✓ Variable importance of Gradient boosting

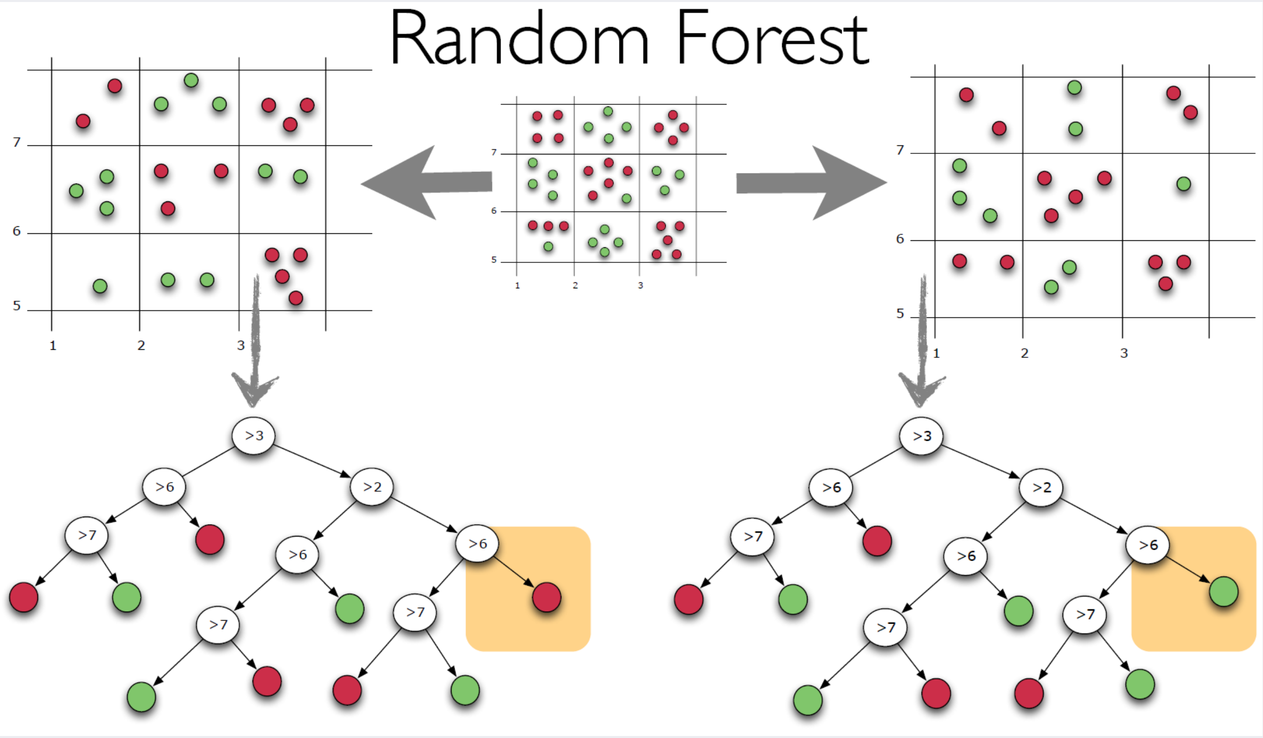
$$Influence_j = \frac{1}{M} \sum_{k=1}^{M} Influence_j(T_k)$$

AGENDA

Random Forests

- A specialized bagging for decision tree algorithms
- Two ways to increase the diversity of ensemble
  - ✓ Bagging
  - ✓ Randomly chosen predictor variables





Random Forests

- Random Forests: Algorithm

1. For $b = 1$ to B:

    (a) Draw a $\boxed{\text{bootstrap sample}}$ $\mathbf{Z}^*$ of size N from the training data.

    (b) Grow a random-forest tree T_b to the bootstrapped data, by recursively repeating the following steps for each terminal node of the tree, until the minimum node size n_{min} is reached.

        i. Select $\boxed{m \text{ variables at random}}$ from the p variables.

        ii. Pick the best variable/split-point among the m.

        iii. Split the node into two daughter nodes.

2. Output the ensemble of trees $\{T_b\}_1^B$.

To make a prediction at a new point x:

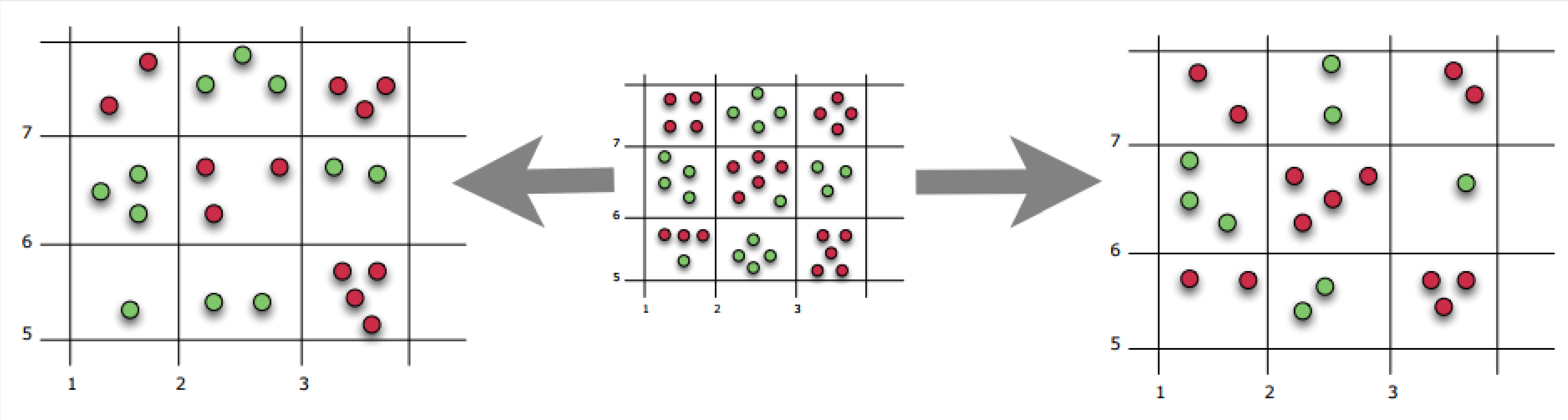
Regression: $\hat{f}_{\mathrm{rf}}^B(x) = \frac{1}{B}\sum_{b=1}^B T_b(x)$.

Classification: Let $\hat{C}_b(x)$ be the class prediction of the bth random-forest tree. Then $\hat{C}_{\mathrm{rf}}^B(x) = \textit{majority vote}\ \{\hat{C}_b(x)\}_1^B$.

https://stat.ethz.ch/education/semesters/ss2012/ams/slides/v10.2.pdf

Random Forests

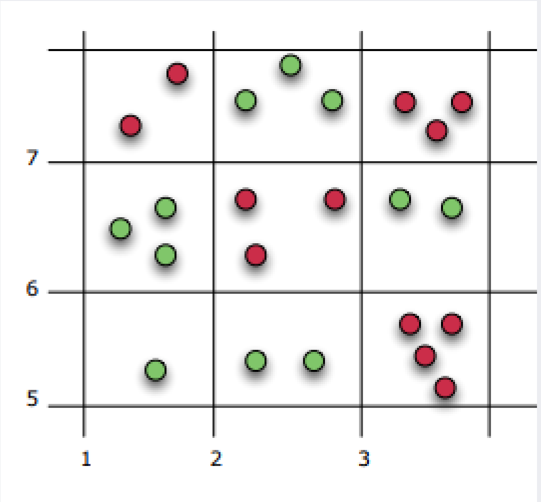
- Bagging
  - ✓ Sampling with replacement

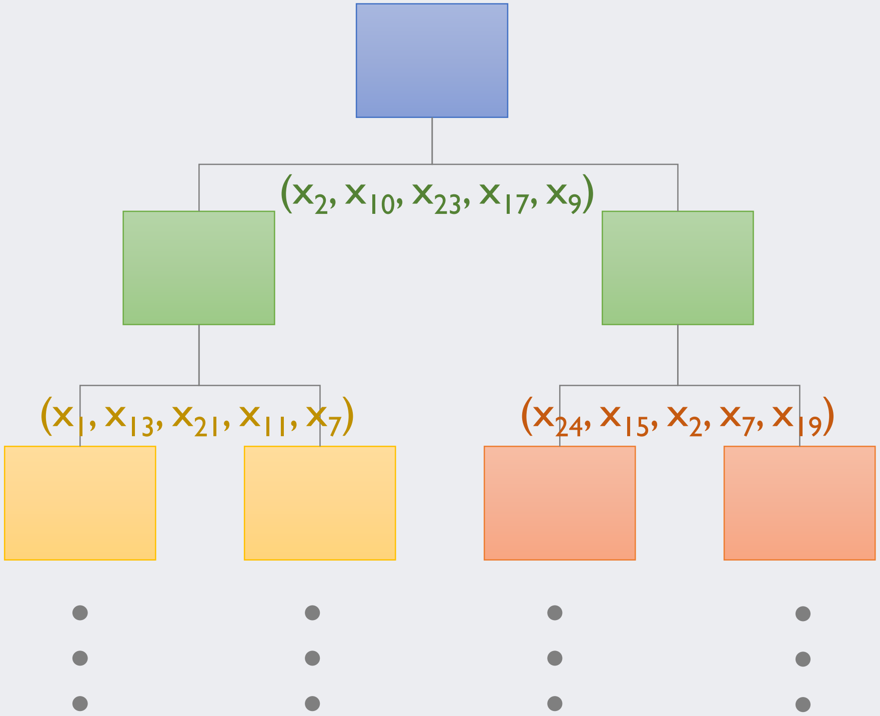


Random Forests

- Bagging
  - ✓ Randomly selected variable

Bootstrap i (X in R^{25})





Random Forests

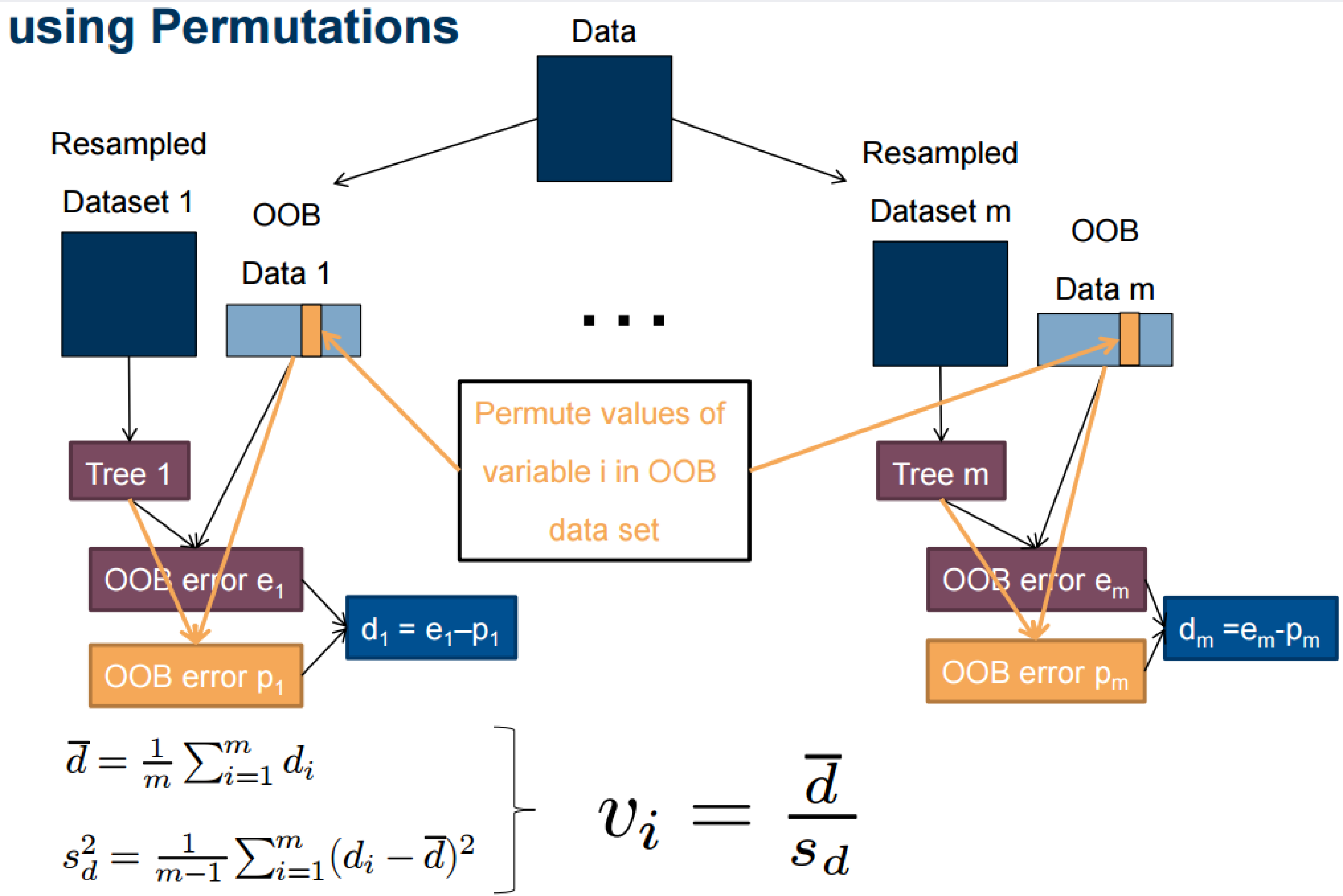
- Generalization Error
  - ✓ Each tree in random forests may over-fit the data because pruning is not conducted.
  - ✓ If the population size is large enough, then the generalization error of random forests bounded by

$$Generalization\ Error \leq \frac{\bar{\rho}(1-s^2)}{s^2}$$

    - $\bar{\rho}$ is the mean value of the correlation coefficients between individual trees
    - s^2 is the margin function (for binary classification, it is simply the average difference proportions between the correct and incorrect trees over all training data.
  - ✓ The more accurate the individual classifiers, the larger the s^2 and the lower the generalization error
  - ✓ The less correlated among the classifiers, the lower the generalization error.

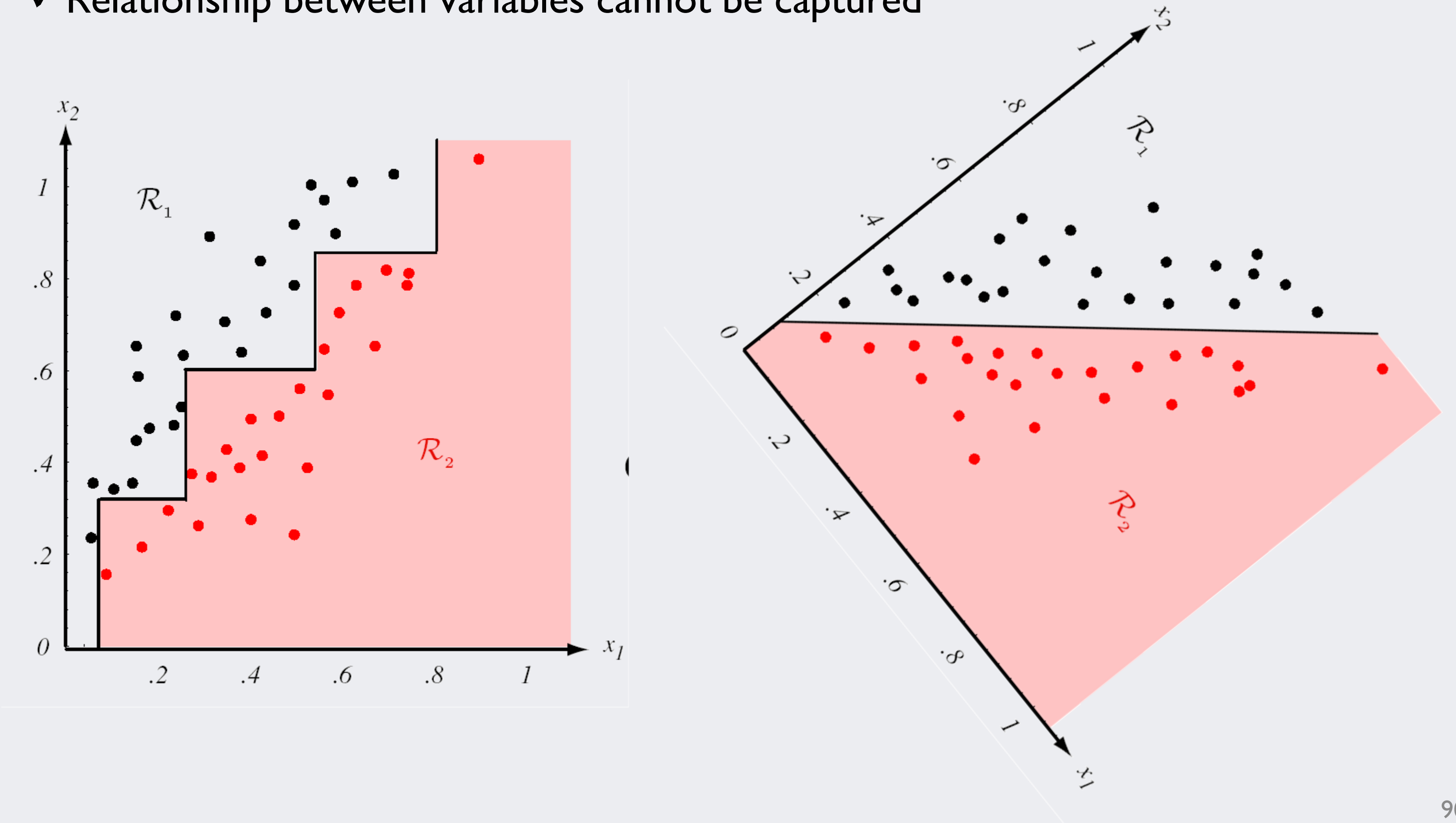
Random Forests

- Variable Importance



Rotation Forests

- Limitation of decision tree
  - ✓ Only one variable is used for a split
  - ✓ Relationship between variables cannot be captured



Rotation Forests

Rodriguez et al. (2006)

- Rotation Forest Algorithm

Training Phase
Given

- X: the objects in the training data set (an $N \times n$ matrix)
- Y: the labels of the training set (an $N \times 1$ matrix)
- L: the number of classifiers in the ensemble
- K: the number of subsets
- $\{\omega_1, \ldots, \omega_c\}$: the set of class labels

For $i = 1 \ldots L$

- Prepare the rotation matrix R_i^a:
  - Split $\mathbf{F}$ (the feature set) into K subsets: $\mathbf{F}_{i,j}$ (for $j = 1 \ldots K$)
  - For $j = 1 \ldots K$
    * Let $X_{i,j}$ be the data set X for the features in $\mathbf{F}_{i,j}$
    * Eliminate from $X_{i,j}$ a random subset of classes
    * Select a bootstrap sample from $X_{i,j}$ of size 75% of the number of objects in $X_{i,j}$. Denote the new set by $X'_{i,j}$
    * Apply PCA on $X'_{i,j}$ to obtain the coefficients in a matrix $C_{i,j}$
  - Arrange the $C_{i,j}$, for $j = 1 \ldots K$ in a rotation matrix R_i as in equation (1)
  - Construct R_i^a by rearranging the the columns of R_i so as to match the order of features in $\mathbf{F}$.
- Build classifier D_i using (XR_i^a, Y) as the training set

Classification Phase

- For a given $\mathbf{x}$, let $d_{i,j}(\mathbf{x}R_i^a)$ be the probability assigned by the classifier D_i to the hypothesis that $\mathbf{x}$ comes from class ω_j. Calculate the confidence for each class, ω_j, by the average combination method:

$$\mu_j(\mathbf{x}) = \frac{1}{L}\sum_{i=1}^{L} d_{i,j}(\mathbf{x}R_i^a), \quad j = 1, \ldots, c.$$

- Assign $\mathbf{x}$ to the class with the largest confidence.

Rotation Forests

Rodriguez et al. (2006)

- Rotation Forest Algorithm

$$R_i = \begin{bmatrix} \mathbf{a}_{i,1}^{(1)}, \mathbf{a}_{i,1}^{(2)}, \ldots, \mathbf{a}_{i,1}^{(M_1)}, & [\mathbf{0}] & \cdots & [\mathbf{0}] \\ [\mathbf{0}] & \mathbf{a}_{i,2}^{(1)}, \mathbf{a}_{i,2}^{(2)}, \ldots, \mathbf{a}_{i,2}^{(M_2)}, & \cdots & [\mathbf{0}] \\ \vdots & \vdots & \ddots & \vdots \\ [\mathbf{0}] & [\mathbf{0}] & \cdots & \mathbf{a}_{i,K}^{(1)}, \mathbf{a}_{i,K}^{(2)}, \ldots, \mathbf{a}_{i,K}^{(M_K)} \end{bmatrix}$$

Rotation Forests

Rodriguez et al. (2006)

- Rotation Forest Algorithm

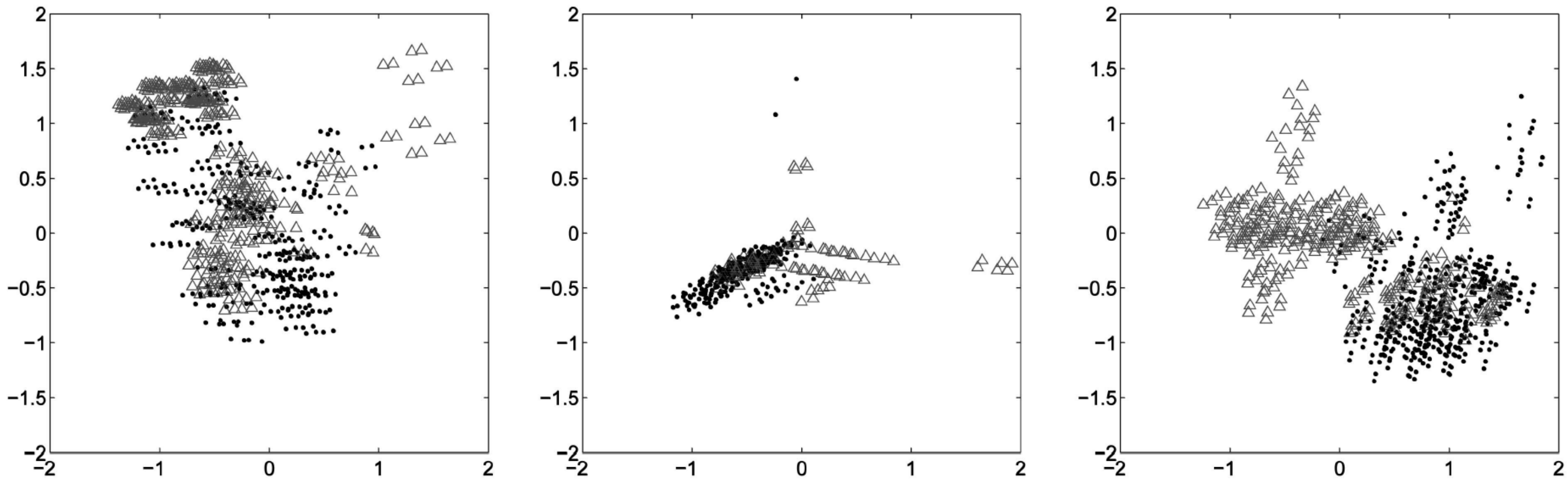


Fig. 2. Scatterplot of the “mushroom” data (22 nominal features). The binary representation of the features (121 in total) was randomly split into three subsets and PCA was applied on each subset. The scatterplots show the two classes in the space defined by the first two principal components.

Decision Jungle

- Forest vs. Jungle

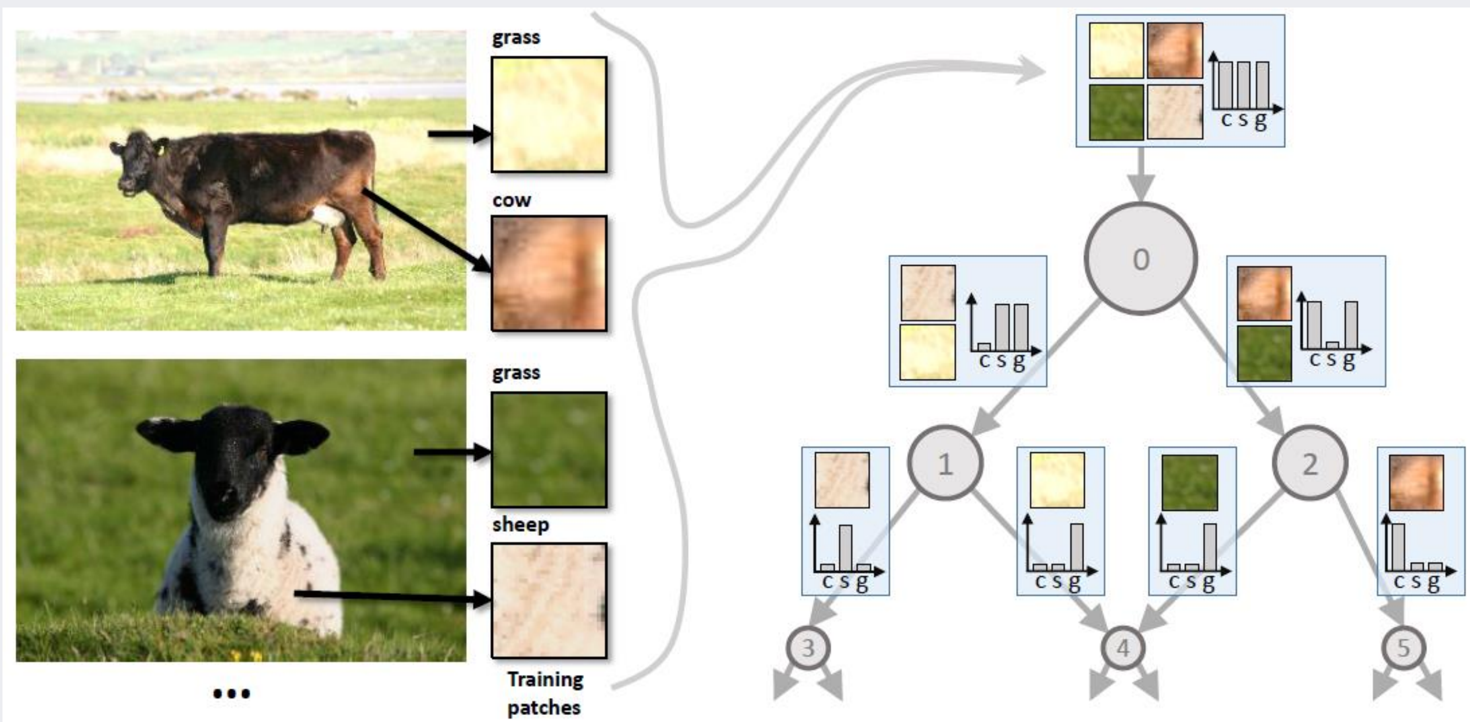
Forest

Jungle

Decision Jungle

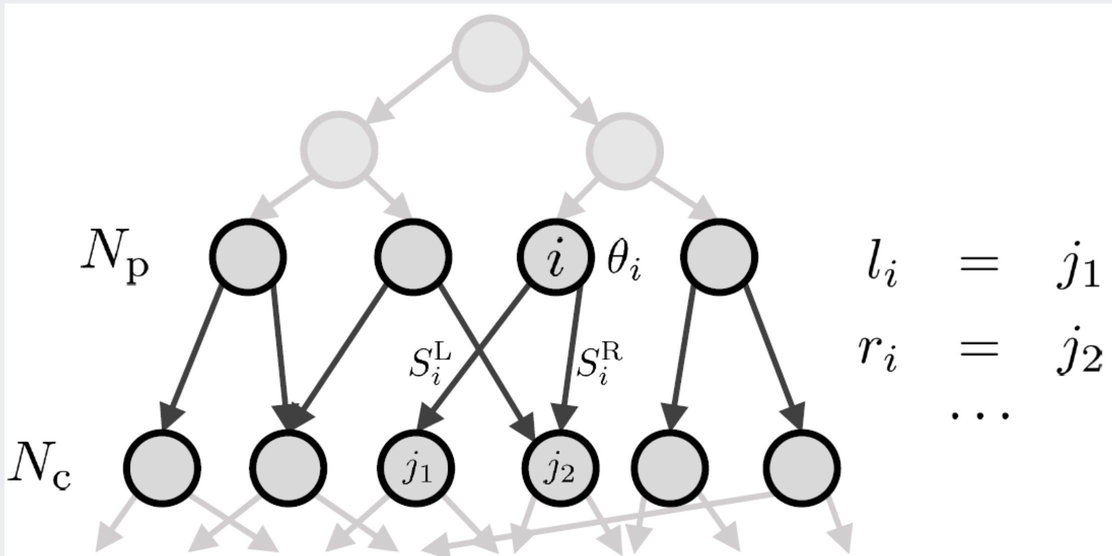
Shotton et al. (2013)

- Motivation
  - ✓ Limitation of decision tree (random forests): the number of nodes in decision trees will grow exponentially with depth given enough data.
- Main Contribution
  - ✓ Allows multiple paths from the root to each leaf.



Decision Jungle

- Notation



✓ N_p: a set of parent nodes

✓ N_c: a set of child nodes ($M = |N_c|$ is a parameter of decision jungle)

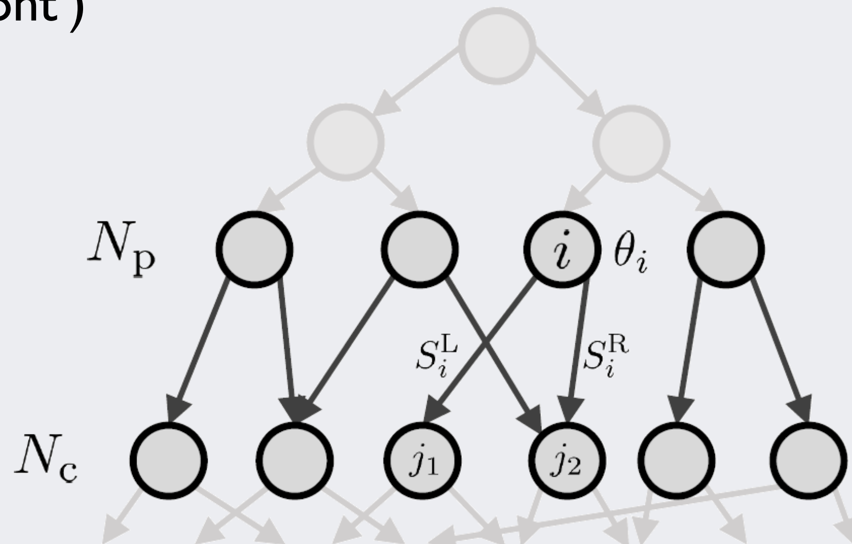
✓ θ_i: the parameters of the split feature function f for parent node i in N_p

✓ S_i: the set of labelled training instances (x, y) that reach node i

$$S_i^{\mathrm{L}}(\theta_i) = \{(x, y) \in S_i \mid f(\theta_i, x) \leq 0\} \qquad S_i^{\mathrm{R}}(\theta_i) = S_i \setminus S_i^{\mathrm{L}}(\theta_i)$$

Decision Jungle

- Notation (cont')



$$
\begin{aligned}
l_i &= j_1 \\
r_i &= j_2 \\
&\ldots
\end{aligned}
$$

✓ l_i in N_c: the current assignment of the left outwards edge from parent node i

✓ r_i in N_c: the current assignment of the right outwards edge from parent node i

✓ The set of instances that reach any child node j in N_c

$$S_j(\{\theta_i\}, \{l_i\}, \{r_i\}) = \left[\bigcup_{i \in N_\mathrm{p} \text{ s.t. } l_i = j} S_i^\mathrm{L}(\theta_i) \right] \cup \left[\bigcup_{i \in N_\mathrm{p} \text{ s.t. } r_i = j} S_i^\mathrm{R}(\theta_i) \right]$$

Decision Jungle

- Training
  - ✓ Goal: joint minimization of the objective over the split parameters $\{\theta_i\}$ and the child assignment $\{l_i\}$ and $\{r_i\}$.
  - ✓ Task of learning the current level of a DAG

$$\min_{\{\theta_i\},\{l_i\},\{r_i\}} E(\{\theta_i\},\{l_i\},\{r_i\})$$

  - ✓ Objective function: total weighted entropy of instances

$$E(\{\theta_i\},\{l_i\},\{r_i\}) = \sum_{j \in N_c} |S_j| \, H(S_j)$$

  - ✓ The minimization problem is hard to solve exactly

Decision Jungle

- Lsearch
  - ✓ Starts from a feasible assignment of the parameters, and alternate between two coordinate descent step
    - Split optimization step

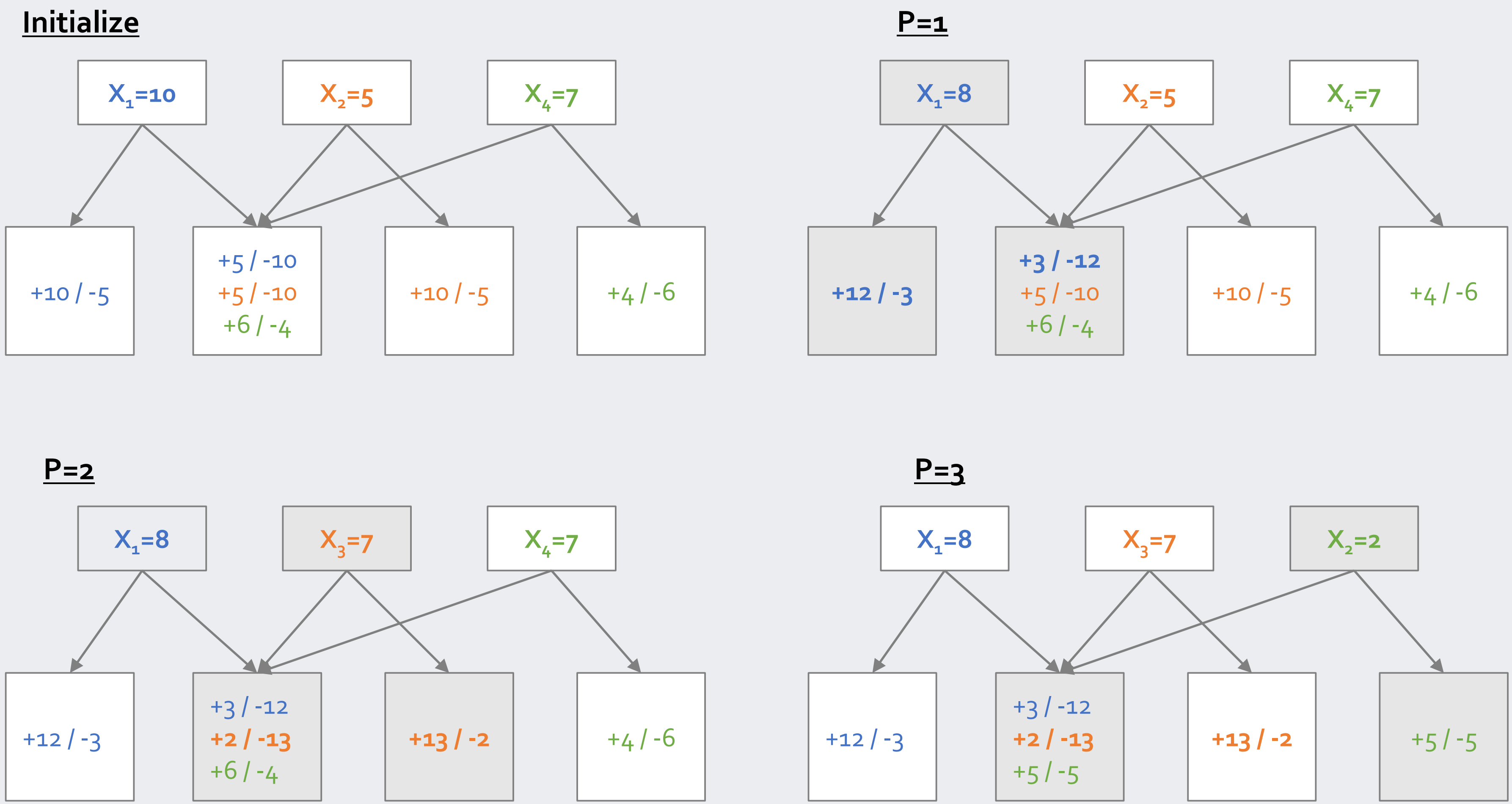
$$
\begin{aligned}
&\text{for } k \in N_{\mathrm{p}} \\
&\qquad \theta_k \leftarrow \underset{\theta'_k}{\operatorname{argmin}}\, E(\theta'_k \cup \{\theta_i\}_{i \in N_{\mathrm{p}} \setminus \{k\}}, \{l_i\}, \{r_i\})
\end{aligned}
$$

    - Branch optimization step

$$
\begin{aligned}
&\text{for } k \in N_{\mathrm{p}} \\
&\qquad l_k \leftarrow \underset{l'_k \in N_{\mathrm{c}}}{\operatorname{argmin}}\, E(\{\theta_i\}, l'_k \cup \{l_i\}_{i \in N_{\mathrm{p}} \setminus \{k\}}, \{r_i\}) \\
&\qquad r_k \leftarrow \underset{r'_k \in N_{\mathrm{c}}}{\operatorname{argmin}}\, E(\{\theta_i\}, \{l_i\}, r'_k \cup \{r_i\}_{i \in N_{\mathrm{p}} \setminus \{k\}})
\end{aligned}
$$

Decision Jungle

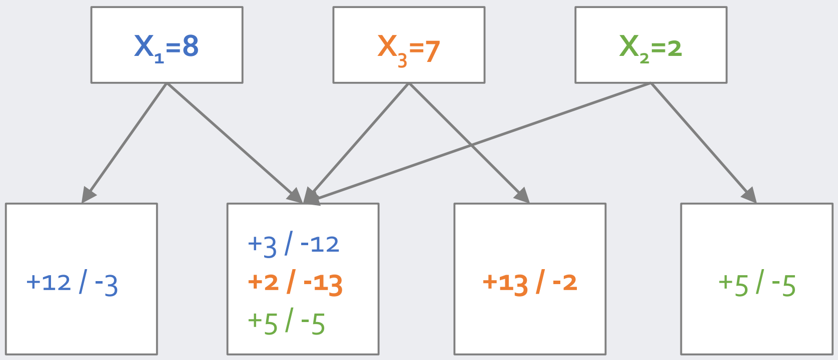
- Split Optimization Example



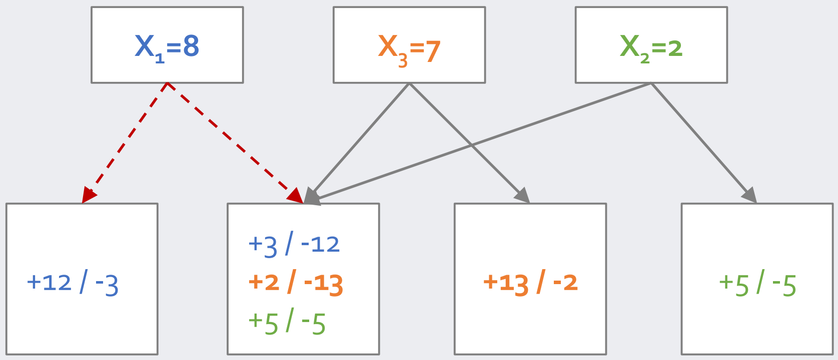
Decision Jungle

- Branch Optimization Example

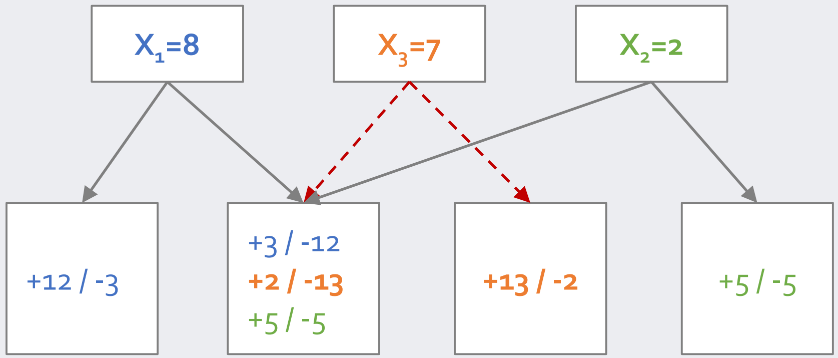
Current assignment



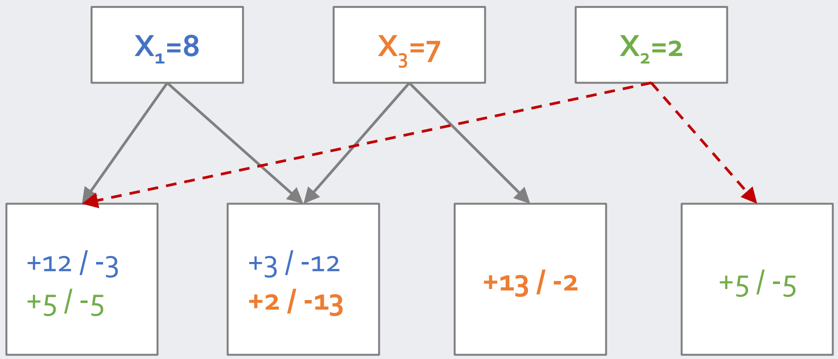
P=1



P=2

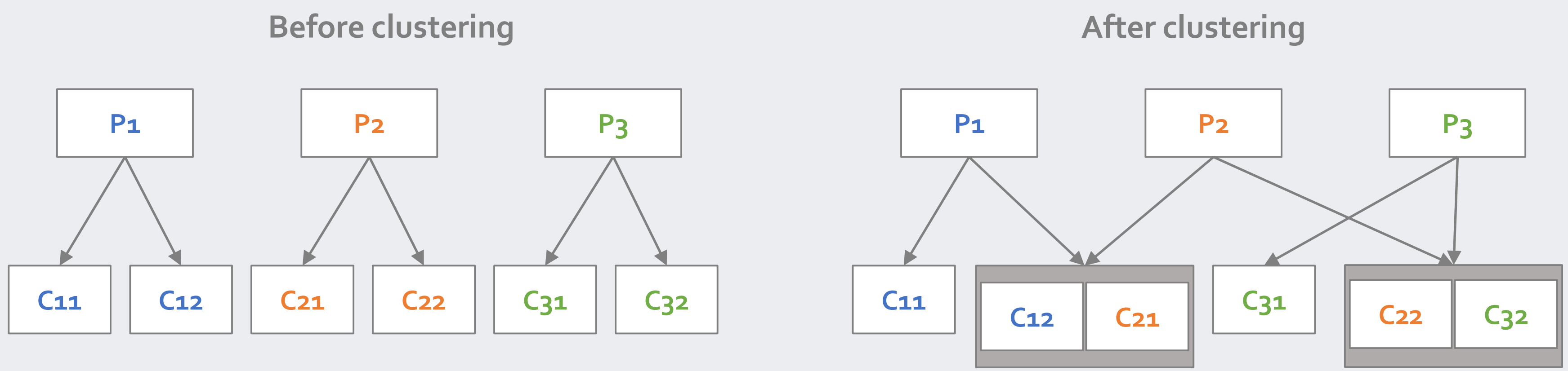


P=3



Decision Jungle

- Cluster Search
  - ✓ Branching variables more globally
    - 2|N_p| temporary child nodes are built via conventional tree-based, training-objective minimization procedures
    - The temporary nodes are clustered into M = |N_c| groups

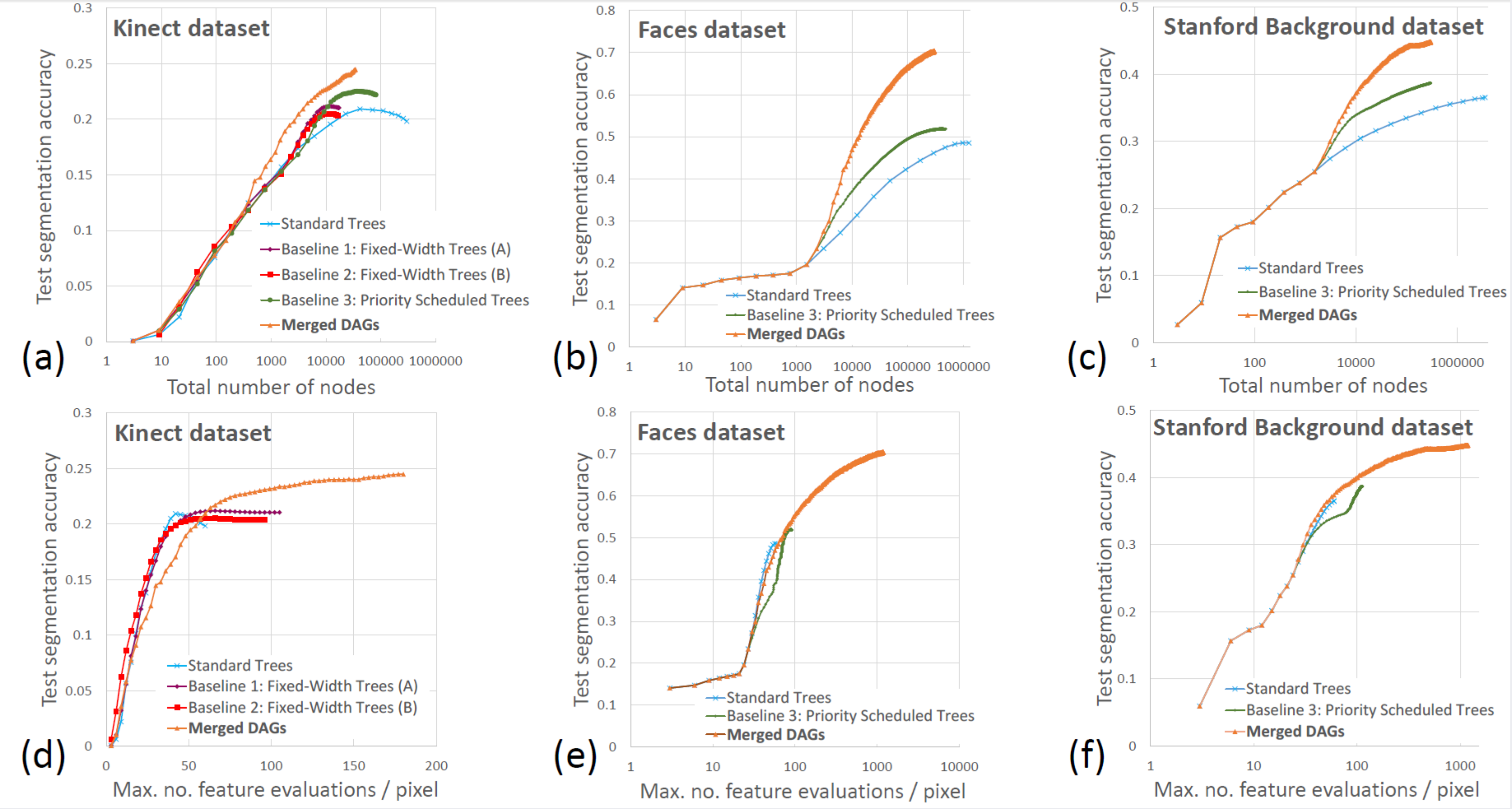


Decision Jungle

- Experiments
  - ✓ Datasets
    - **Kinect body part classification**: 31 classes, 1,000 training images with 250 example pixels randomly sampled per image, 1,000 test datasets
    - **Facial features segmentation**: 8 classes, 1,000 training images using every pixel
    - **Stanford background dataset**: 8 classes with 715 training images
    - **UCI data sets**: 28 classification datasets
  - ✓ Benchmark algorithms
    - Standard Forests
    - Fixed-width Tree: (A) split M/2 parent nodes with highest information gain, (B) select M child nodes that most reduce the objective
    - Priority scheduled trees

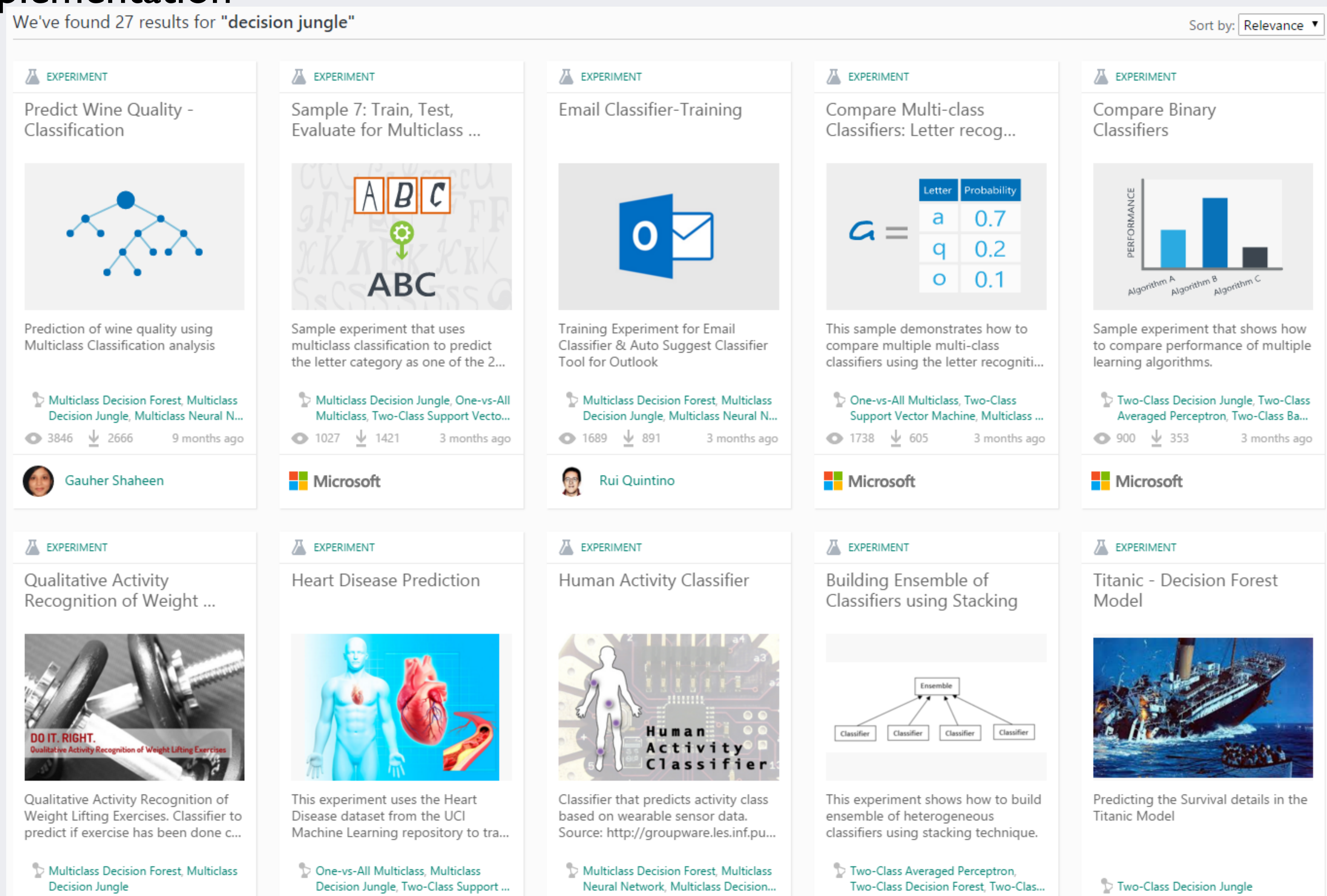
Decision Jungle

- Experimental Results: Image Classification



Decision Jungle

- Implementation



http://gallery.cortanaanalytics.com/browse?s=%22decision%20jungle%22


Q & A

References

Research Papers

- Breiman, L. (1996). Bagging predictors. Machine learning 24(2): 123-140.
- Breiman, L. (2001). Random forests. Machine learning 45(1): 5-32.
- Caruana, R. and Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms. In Proceedings of the 23rd international conference on Machine learning: 161-168.
- Fernández-Delgado, M., Cernadas, E., Barro, S., and Amorim, D. (2014). Do we need hundreds of classifiers to solve real world classification problems? The Journal of Machine Learning Research 15(1): 3133-3181.
- Freund, Y., Schapire, R., and Abe, N. (1999). A short introduction to boosting. Journal-Japanese Society For Artificial Intelligence 14:771-780.
- Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. Annals of statistics, 1189-1232.
- Kang, P., Cho, S., and MacLachlan, D. L. (2012). Improved response modeling based on clustering, under-sampling, and ensemble. Expert Systems with Applications 39(8): 6738-6753.
- Kang, S., Cho, S., and Kang, P. (2015). Multi-class classification via heterogeneous ensemble of one-class classifiers. Engineering Applications of Artificial Intelligence 43: 35-43.
- Liang, G., Zhu, X., and Zhang, C. (2011). An empirical study of bagging predictors for imbalanced data with different levels of class distribution. In Proceedings of the Advances in Artificial Intelligence: 213-222.
- Natekin, A., & Knoll, A. (2013). Gradient boosting machines, a tutorial. Frontiers in neurorobotics, 7:21

References

Research Papers

- Opitz, D. & Maclin, R. (1999). Popular ensemble methods: An empirical study. Journal of Artificial Intelligence Research 11: 169-198.
- Rodriguez, J. J., Kuncheva, L. I., and Alonso, C. J. (2006). Rotation forest: A new classifier ensemble method. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(10): 1619-1630.
- Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Berg, A. C. (2015). Imagenet large scale visual recognition challenge. International Journal of Computer Vision, 115(3), 211-252.
- Seni, G. and Elder, J. F. (2010). Ensemble methods in data mining: improving accuracy through combining predictions. Synthesis Lectures on Data Mining and Knowledge Discovery 2(1): 1-126.
- Shotton, J., Sharp, T., Kohli, P., Nowozin, S., Winn, J., and Criminisi, A. (2013). Decision jungles: Compact and rich models for classification. In Advances in Neural Information Processing Systems (NIPS'13): 234-242.
- Viola, P. and Jones, M. J. (2004). Robust real-time face detection. International journal of computer vision 57(2): 137-154.

References

Other materials

- Mixture of Experts: http://people.cs.pitt.edu/~milos/courses/cs2750-Spring2011/Lectures/class18.pdf
- Li, C. A gentle introduction to gradient boosting, http://www.ccs.neu.edu/home/vip/teach/MLcourse/4_boosting/slides/gradient_boosting.pdf
- Ihler, A. (2012). Machine Learning and Data Mining: Ensemble of Leaners, http://sli.ics.uci.edu/Classes/2011W-178?action=download&upname=CS178_L10.pdf
- Corso, J. Boosting and AdaBoost: https://www.cse.buffalo.edu/~jcorso/t/CSE455/files/lecture_boosting.pdf