Machine Learning: Part II: Combining Features, Kernels, and Algorithms

Ethem Alpaydın
alpaydin@boun.edu.tr

Ref: E. Alpaydın (2010). *Introduction to Machine Learning,* 2e, The MIT Press.

Rationale

- No Free Lunch Theorem: There is no algorithm that is always the most accurate
- Generate a group of base-learners which when combined has higher accuracy
- The need to generate models that are complementary/uncorrelated/diverse

How to generate complementary learners?

- Algorithms
- Hyperparameters
- Representations/Modalities/Views
- Training sets
- Subproblems

Voting

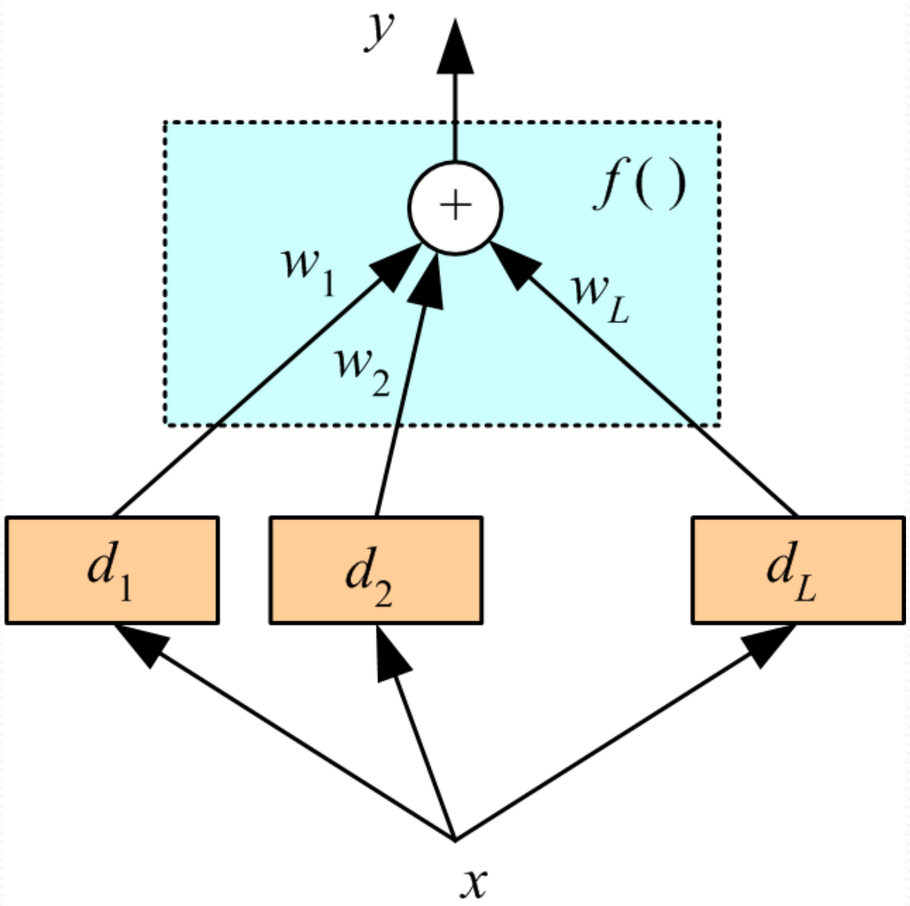
- Linear combination

$$y = \sum_{j=1}^{L} w_j d_j$$

$$w_j \geq 0 \text{ and } \sum_{j=1}^{L} w_j = 1$$

- Classification

$$y_i = \sum_{j=1}^{L} w_j d_{ji}$$



- Bayesian perspective:

$$P(C_i \mid x) = \sum_{\text{all models } \mathcal{M}_j} P(C_i \mid x, \mathcal{M}_j) P(\mathcal{M}_j)$$

If d_j are iid

$$E[y] = E\left[\sum_j \frac{1}{L} d_j\right] = \frac{1}{L} L \cdot E[d_j] = E[d_j]$$

$$\mathrm{Var}(y) = \mathrm{Var}\left(\sum_j \frac{1}{L} d_j\right) = \frac{1}{L^2} \mathrm{Var}\left(\sum_j d_j\right) = \frac{1}{L^2} L \cdot \mathrm{Var}(d_j) = \frac{1}{L} \mathrm{Var}(d_j)$$

Bias does not change, variance decreases by L

- If dependent, error increases with positive correlation

$$\mathrm{Var}(y) = \frac{1}{L^2} \mathrm{Var}\left(\sum_j d_j\right) = \frac{1}{L^2}\left[\sum_j \mathrm{Var}(d_j) + 2 \sum_j \sum_{i<j} Cov(d_i, d_j)\right]$$

Fixed Combination Rules

Rule	Fusion function $f(\cdot)$
Sum	$y_i = \frac{1}{L}\sum_{j=1}^{L} d_{ji}$
Weighted sum	$y_i = \sum_j w_j d_{ji}, w_j \geq 0, \sum_j w_j = 1$
Median	$y_i = \text{median}_j d_{ji}$
Minimum	$y_i = \min_j d_{ji}$
Maximum	$y_i = \max_j d_{ji}$
Product	$y_i = \prod_j d_{ji}$

	C_1	C_2	C_3
d_1	0.2	0.5	0.3
d_2	0.0	0.6	0.4
d_3	0.4	0.4	0.2
Sum	0.2	**0.5**	0.3
Median	0.2	**0.5**	0.4
Minimum	0.0	**0.4**	0.2
Maximum	0.4	**0.6**	0.4
Product	0.0	**0.12**	0.032

Error-Correcting Output Codes

- *K* classes; *L* problems (Dietterich and Bakiri, 1995)
- Code matrix **W** codes classes in terms of learners
- One per class
  L=*K*

$$\mathbf{W} = \begin{bmatrix} +1 & -1 & -1 & -1 \\ -1 & +1 & -1 & -1 \\ -1 & -1 & +1 & -1 \\ -1 & -1 & -1 & +1 \end{bmatrix}$$

- Pairwise
  L=*K*(*K*-1)/2

$$\mathbf{W} = \begin{bmatrix} +1 & +1 & +1 & 0 & 0 & 0 \\ -1 & 0 & 0 & +1 & +1 & 0 \\ 0 & -1 & 0 & -1 & 0 & +1 \\ 0 & 0 & -1 & 0 & -1 & -1 \end{bmatrix}$$

- Full code $L=2^{(K-1)}-1$

$$\mathbf{W}=\begin{bmatrix} -1 & -1 & -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & +1 & +1 & +1 & +1 \\ -1 & +1 & +1 & -1 & -1 & +1 & +1 \\ +1 & -1 & +1 & -1 & +1 & -1 & +1 \end{bmatrix}$$

- With reasonable L, find **W** such that the Hamming distance btw rows and columns are maximized.
- Voting scheme

$$y_i=\sum_{j=1}^{L} w_j d_{ji}$$

- Subproblems may be more difficult than one-per-K

Bagging

- Use bootstrapping to generate *L* training sets and train one base-learner with each (Breiman, 1996)
- Use voting (Average or median with regression)
- Unstable algorithms profit from bagging

AdaBoost

Generate a sequence of base-learners each focusing on previous one's errors

(Freund and Schapire, 1996)

Training:

For all $\{x^t, r^t\}_{t=1}^N \in \mathcal{X}$, initialize $p_1^t = 1/N$

For all base-learners $j = 1, \ldots, L$

Randomly draw $\mathcal{X}_j$ from $\mathcal{X}$ with probabilities p_j^t

Train d_j using $\mathcal{X}_j$

For each (x^t, r^t), calculate $y_j^t \leftarrow d_j(x^t)$

Calculate error rate: $\epsilon_j \leftarrow \sum_t p_j^t \cdot 1(y_j^t \neq r^t)$

If $\epsilon_j > 1/2$, then $L \leftarrow j - 1$; stop

$\beta_j \leftarrow \epsilon_j / (1 - \epsilon_j)$

For each (x^t, r^t), decrease probabilities if correct:

If $y_j^t = r^t$ $p_{j+1}^t \leftarrow \beta_j p_j^t$ Else $p_{j+1}^t \leftarrow p_j^t$

Normalize probabilities:

$Z_j \leftarrow \sum_t p_{j+1}^t; \quad p_{j+1}^t \leftarrow p_{j+1}^t / Z_j$

Testing:

Given x, calculate $d_j(x), j = 1, \ldots, L$

Calculate class outputs, $i = 1, \ldots, K$:

$$y_i = \sum_{j=1}^L \left(\log \frac{1}{\beta_j}\right) d_{ji}(x)$$

Mixture of Experts

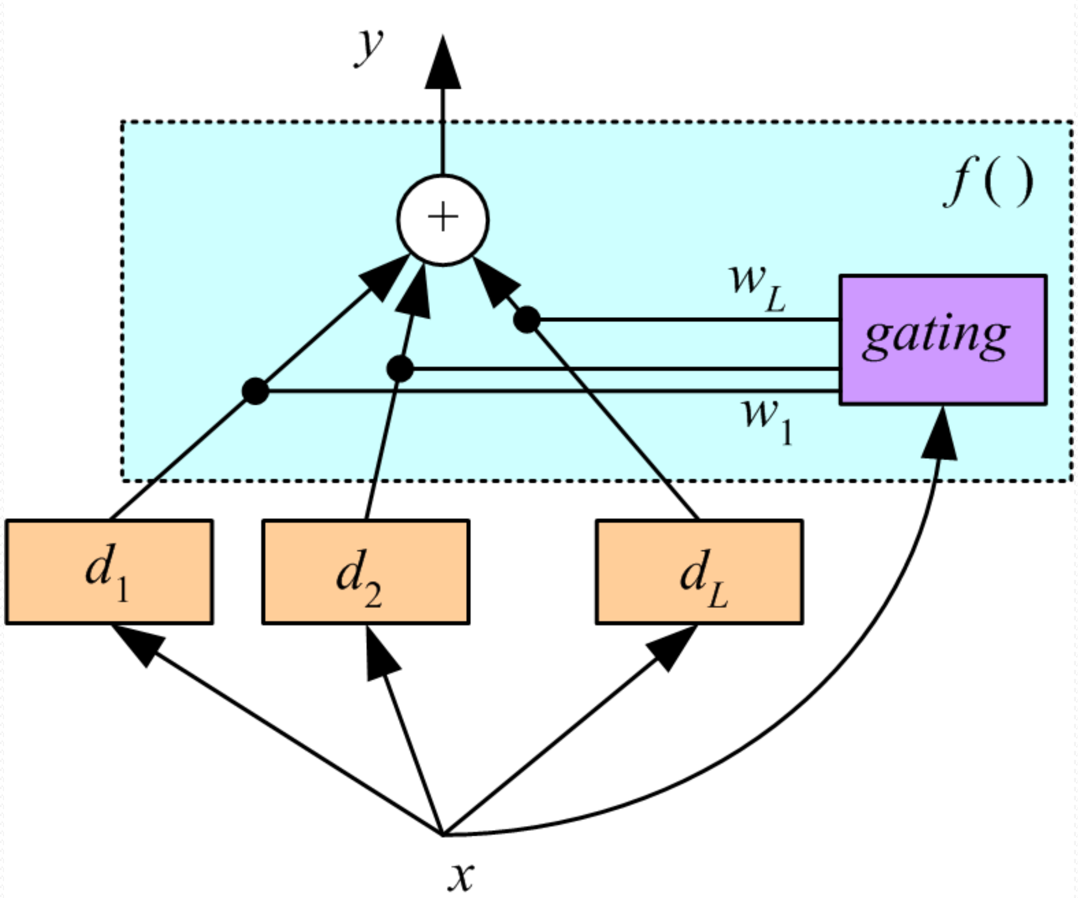
Voting where weights are input-dependent (gating)

$$y = \sum_{j=1}^{L} w_j d_j$$

(Jacobs et al., 1991)

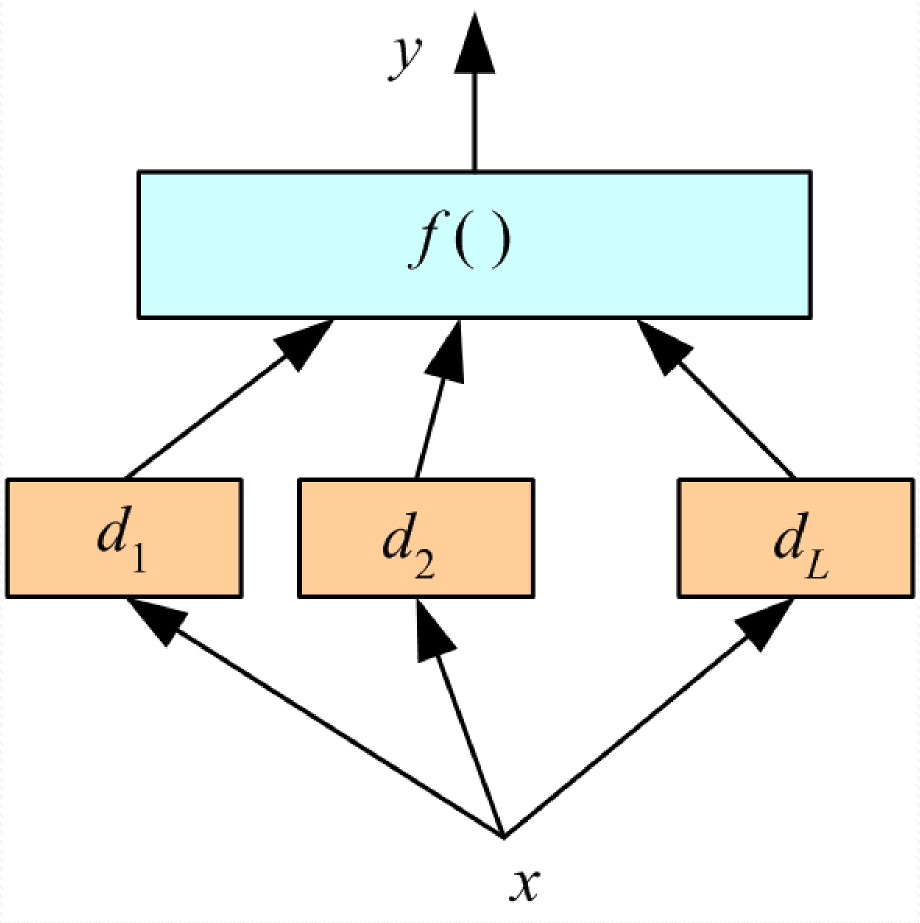
Experts or gating

can be nonlinear



Stacking

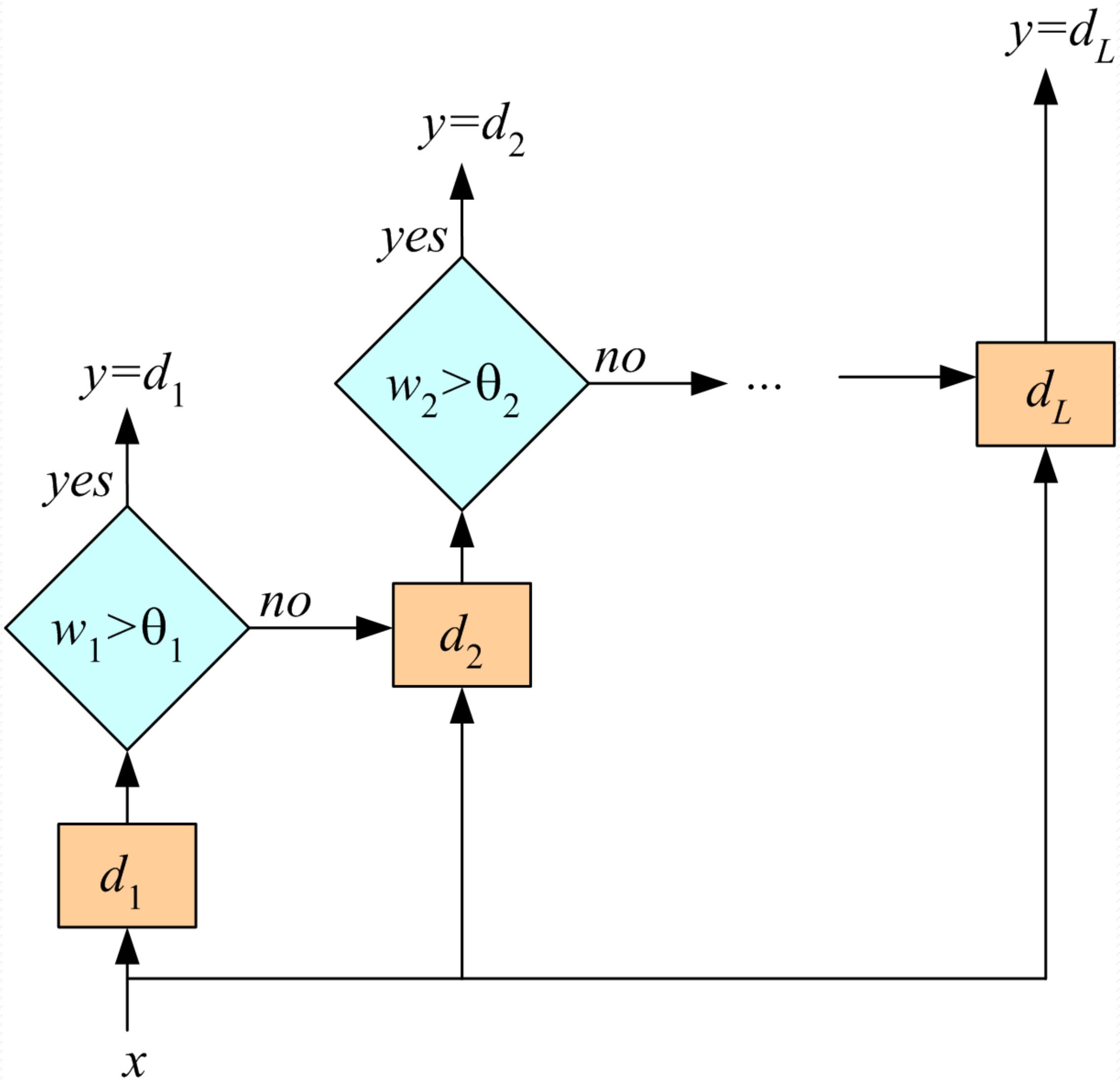
- Combiner *f* () is another learner (Wolpert, 1992)



Cascading

Use d_j only if preceding ones are not confident

Cascade learners in order of complexity



Combining Multiple Sources

- Early integration: Concat all features and train a single learner
- Late integration: With each feature set, train one learner, then train a combiner
- Intermediate integration: With each feature set, calculate a kernel, then use a single SVM with multiple kernels
- Combining features vs decisions vs kernels

Fine-Tuning an Ensemble

- Given an ensemble of dependent classifiers, do not use it as is, try to get independence

1. Subset selection: Forward (growing)/Backward (pruning) approaches to improve accuracy/diversity/independence
2. Train metaclassifiers: From the output of correlated classifiers, extract new combinations that are uncorrelated. Using PCA, we get "eigenlearners."

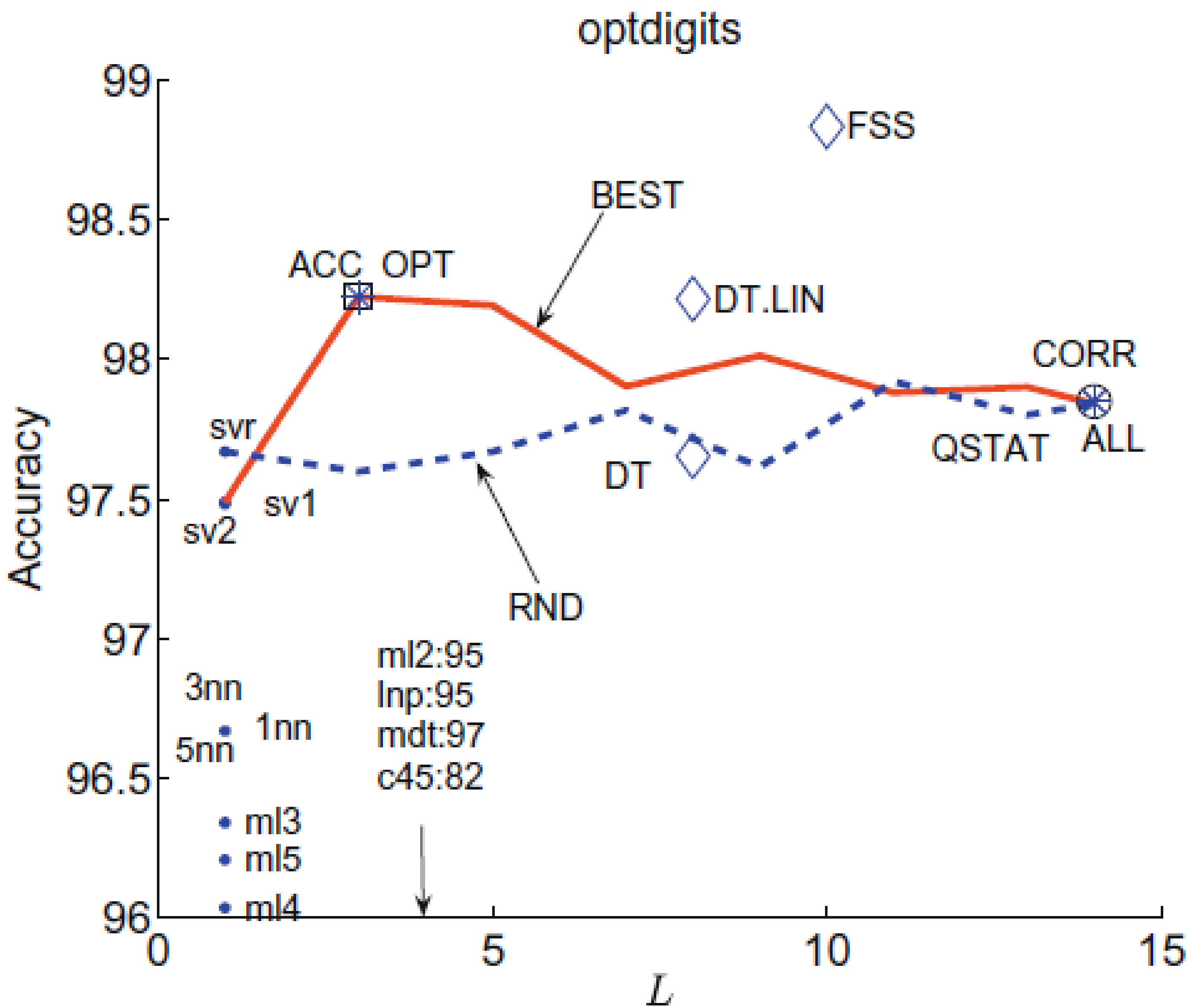
- Similar to feature selection vs feature extraction

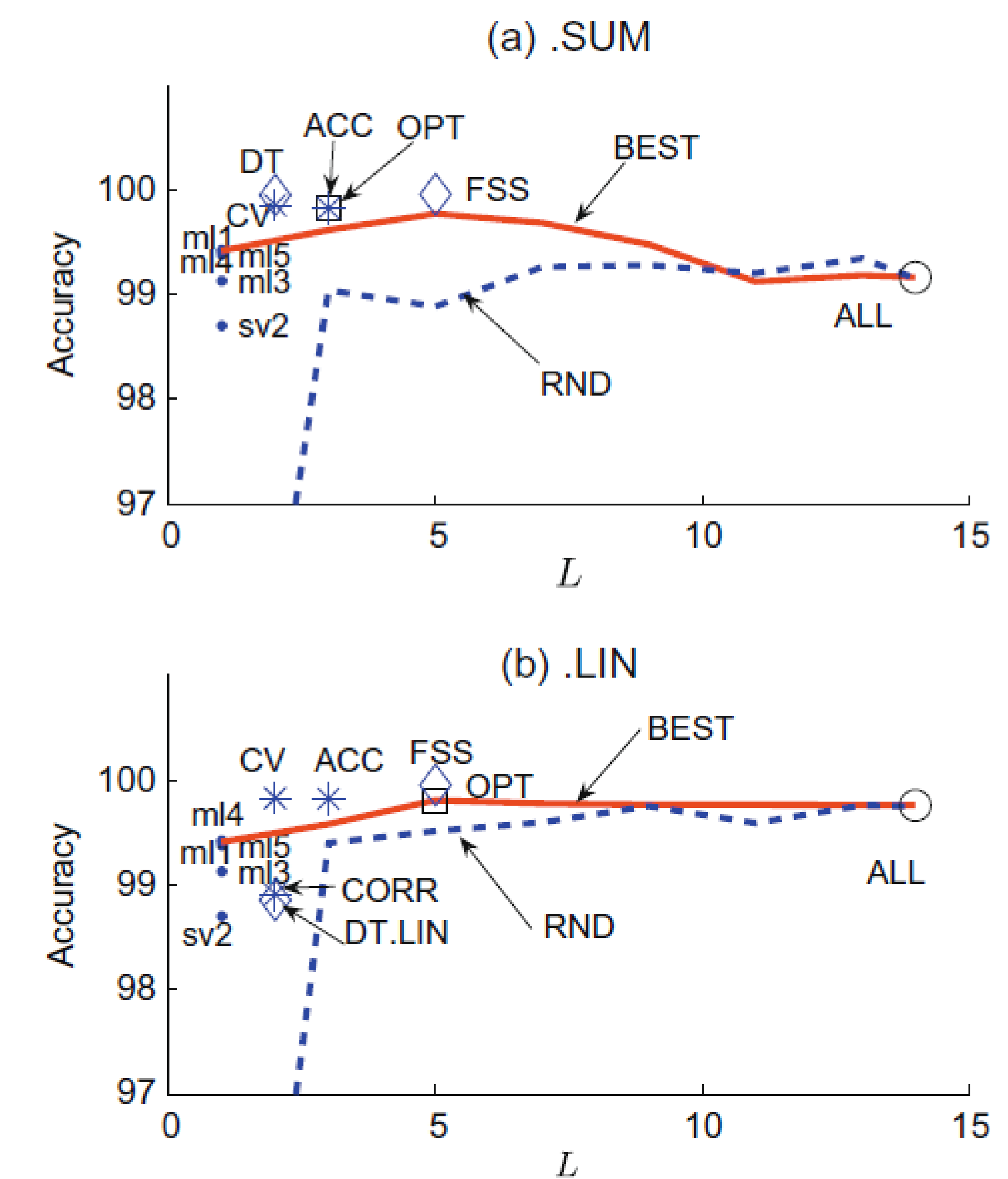
Incremental Construction of Ensembles

- Ref: A. Ulaş, M. Semerci, O. T. Yıldız, E. Alpaydın (2009) "Incremental Construction of Classifier and Discriminant Ensembles," *Information Sciences*, **179**, 1298-1318.
- Given an ensemble of dependent classifiers, do not use it as is, try to get independence by
- Classifier Ensembles by Subset selection: Forward (growing)/Backward (pruning) approaches to improve accuracy/diversity/independence
- Discriminant Ensembles by Decision Tree: Learn the final output from the L k dimensional discriminant values

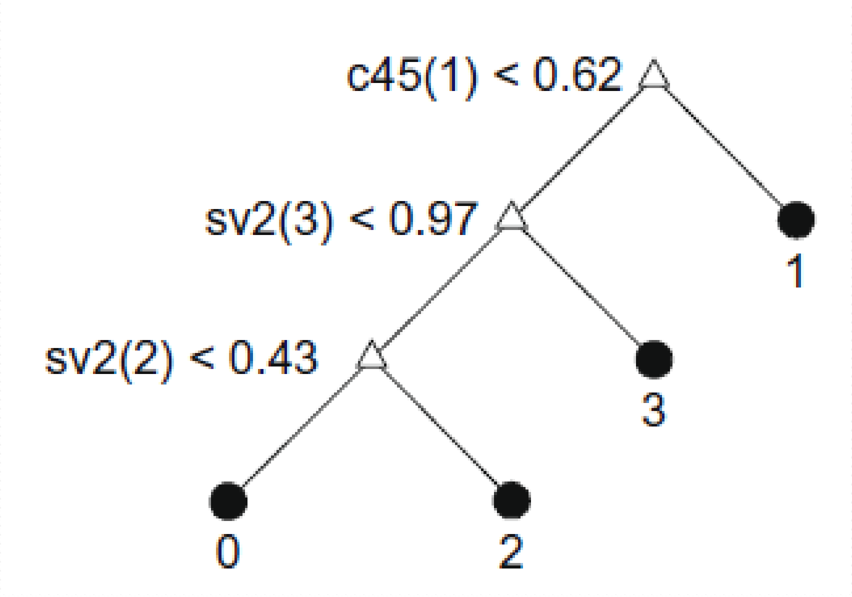
Incremental Construction by Forward Search

```
function icon(P)
E^0 ← ∅
for t = 0 to L − 1
    S_k^(t+1) ← E^(t) ∪ M_k, ∀M_k ∈ P where M_k ∉ E^(t)
    if ∃S_j^(t+1) such that S_j^(t+1) ≺ S_k^(t+1), ∀k ≠ j
            and S_j^(t+1) ≺ E^(t)
        then E^(t+1) ← S_j^(t+1), t ← t + 1
        else break
end for
return E^(t)
```


optdigits
99
98.5
98
97.5
97
96.5
96
0
5
10
15
L
Accuracy
FSS
BEST
ACC OPT
DT.LIN
CORR
svr
QSTAT
ALL
DT
sv1
sv2
RND
ml2:95
lnp:95
mdt:97
c45:82
3nn
1nn
5nn
ml3
ml5
ml4


(a) .SUM
Accuracy
L
ACC
OPT
DT
CV
FSS
BEST
ml1
ml4
ml5
ml3
sv2
RND
ALL
(b) .LIN
CV
ACC
FSS
OPT
BEST
ml4
ml1
ml5
ml3
CORR
DT.LIN
sv2
RND
ALL

Decision Tree Combiner



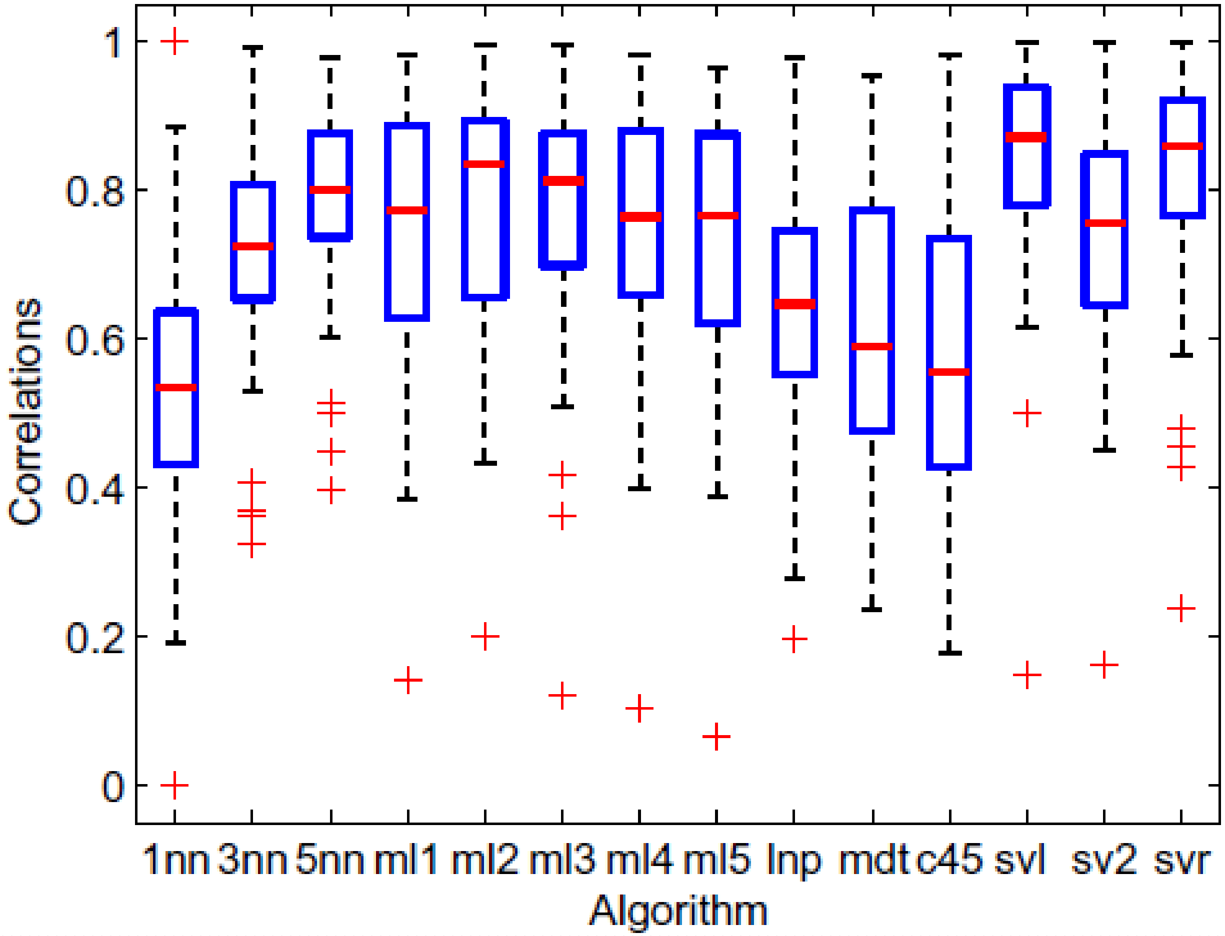
Eigenclassifiers for Combining Classifiers

- Ref: A. Ulaş, O. T. Yıldız, E. Alpaydın (2012) “Cost-Conscious Comparison of Supervised Learning Algorithms over Multiple Data Sets,” *Pattern Recognition*, **45**(4), 1772-1781.
- Train metaclassifiers: From the output of correlated classifiers, extract new combinations that are uncorrelated. Using PCA, we get “eigenlearners.”

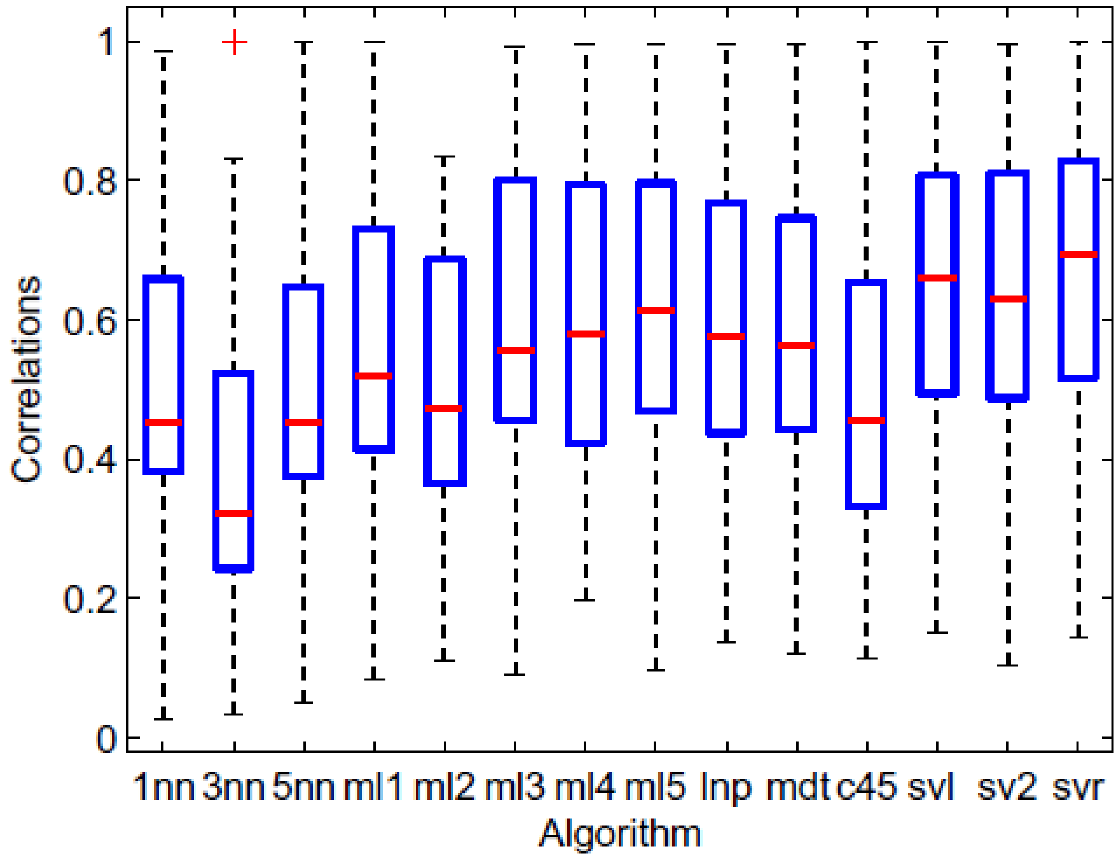
Correlation due to hyperparameters

	kn1	*kn3*	*kn5*	*ml1*	*ml2*	*ml3*	*ml4*	*ml5*	*lnp*	*mdt*	*c45*	*svl*	*sv2*	*svr*
kn1	1.00	**0.71**	**0.64**	0.37	0.37	0.37	0.38	0.37	0.38	0.35	0.30	0.39	0.34	0.44
kn3	**0.71**	1.00	**0.88**	0.51	0.50	0.51	0.51	0.51	0.51	0.45	0.41	0.53	0.45	0.58
kn5	**0.64**	**0.88**	1.00	0.57	0.56	0.57	0.57	0.57	0.55	0.49	0.45	0.59	0.49	**0.64**
ml1	0.37	0.51	0.57	1.00	**0.79**	**0.81**	**0.81**	**0.79**	**0.67**	0.59	0.52	**0.75**	0.53	**0.69**
ml2	0.37	0.50	0.56	**0.79**	1.00	**0.81**	**0.79**	**0.77**	**0.66**	**0.62**	0.54	**0.76**	0.52	**0.69**
ml3	0.37	0.51	0.57	**0.81**	**0.81**	1.00	**0.81**	**0.81**	**0.67**	**0.61**	0.53	**0.75**	0.53	**0.70**
ml4	0.38	0.51	0.57	**0.81**	**0.79**	**0.81**	1.00	**0.80**	**0.67**	**0.61**	0.53	**0.75**	0.52	**0.69**
ml5	0.37	0.51	0.57	**0.79**	**0.77**	**0.81**	**0.80**	1.00	**0.67**	**0.60**	0.52	**0.75**	0.53	**0.69**
lnp	0.38	0.51	0.55	**0.67**	**0.66**	**0.67**	**0.67**	**0.67**	1.00	0.57	0.48	**0.71**	0.45	**0.63**
mdt	0.35	0.45	0.49	0.59	**0.62**	**0.61**	**0.61**	**0.60**	0.57	1.00	0.50	**0.64**	0.45	**0.60**
c45	0.30	0.41	0.45	0.52	0.54	0.53	0.53	0.52	0.48	0.50	1.00	0.54	0.43	0.54
svl	0.39	0.53	0.59	**0.75**	**0.76**	**0.75**	**0.75**	**0.75**	**0.71**	**0.64**	0.54	1.00	0.57	**0.74**
sv2	0.34	0.45	0.49	0.53	0.52	0.53	0.52	0.53	0.45	0.45	0.43	0.57	1.00	**0.65**
svr	0.44	0.58	**0.64**	**0.69**	**0.69**	**0.70**	**0.69**	**0.69**	**0.63**	**0.60**	0.54	**0.74**	**0.65**	1.00

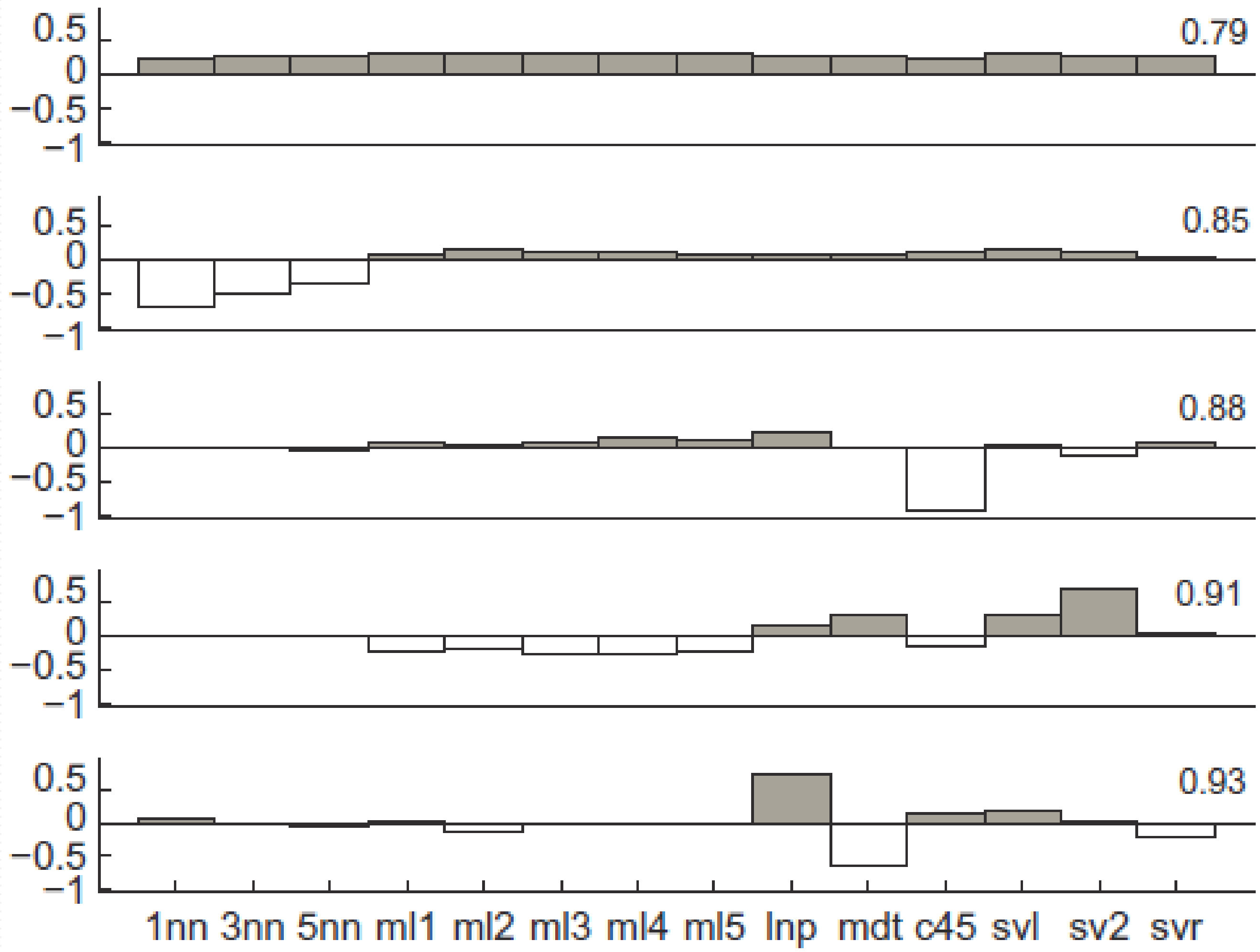
Correlation due to data

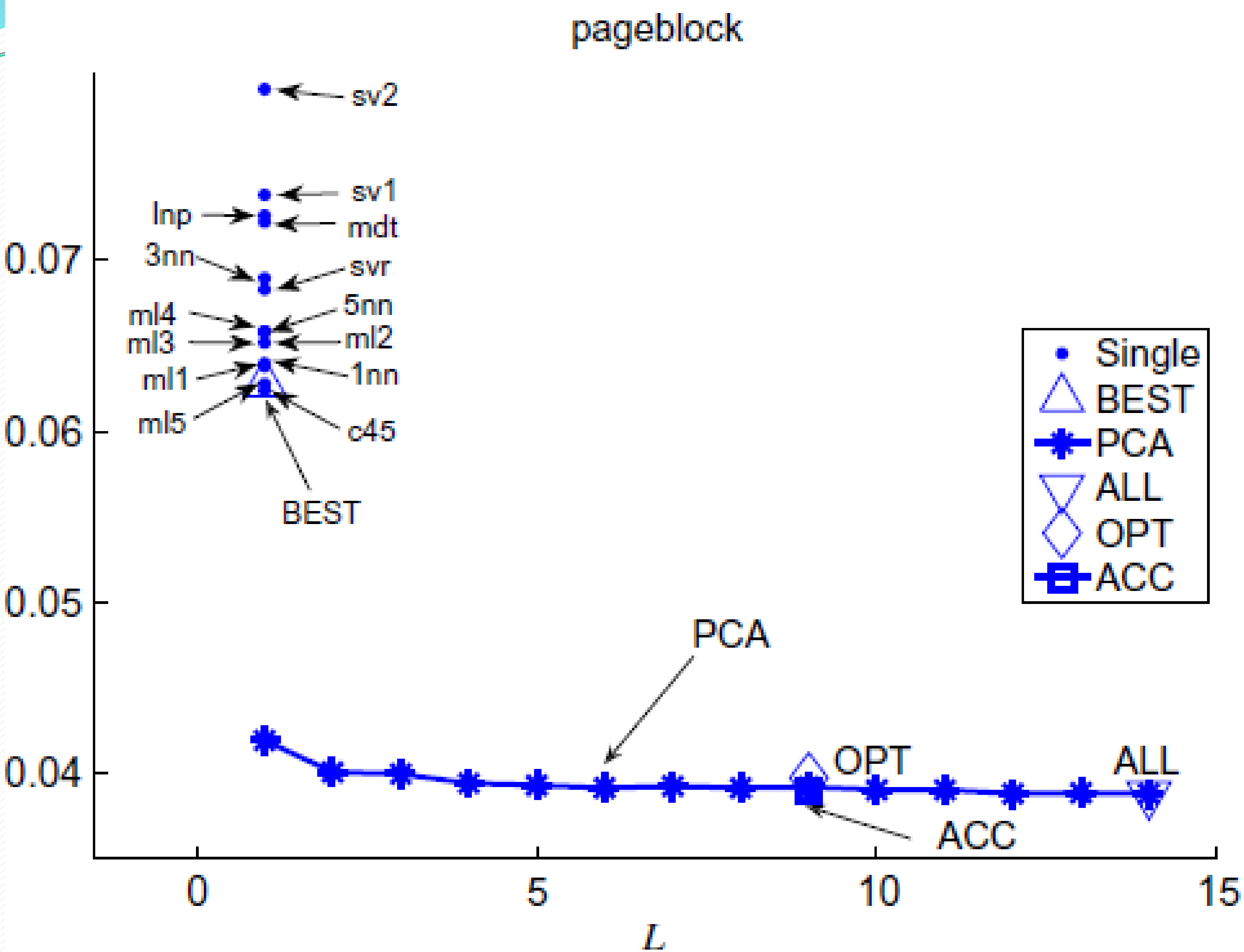


Correlation due to features

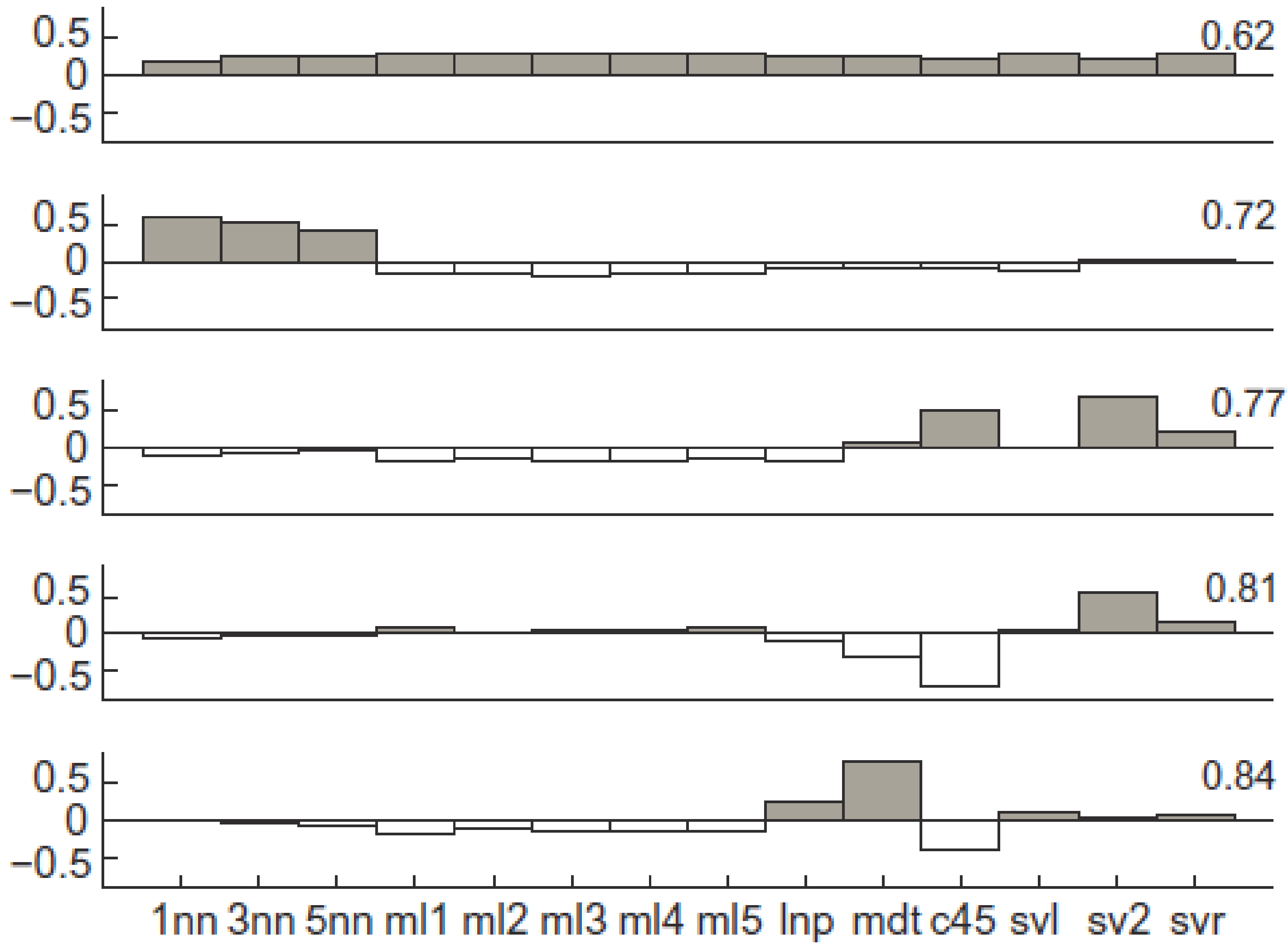


Pageblock data set

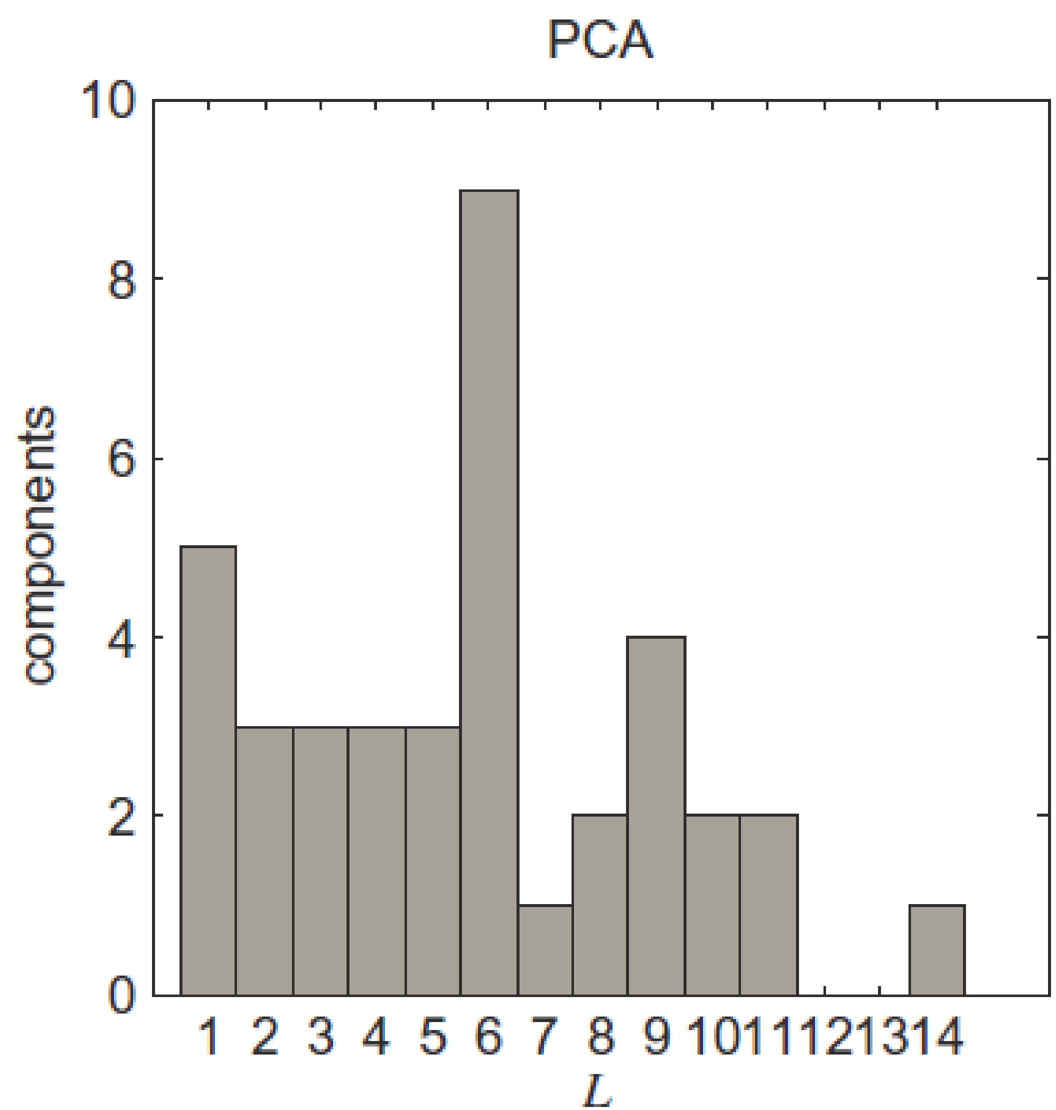



pageblock
sv2
sv1
lnp
mdt
3nn
svr
5nn
ml4
ml3
ml2
ml1
1nn
ml5
c45
BEST
0.07
0.06
0.05
0.04
Single
BEST
PCA
ALL
OPT
ACC
PCA
OPT
ALL
ACC
0
5
10
15
L

All datasets



How many eigenclassifiers?



Multiple Kernel Learning

- Fixed kernel combination

$$K(\mathbf{x},\mathbf{y}) = \begin{cases} cK(\mathbf{x},\mathbf{y}) \\ K_1(\mathbf{x},\mathbf{y}) + K_2(\mathbf{x},\mathbf{y}) \\ K_1(\mathbf{x},\mathbf{y})K_2(\mathbf{x},\mathbf{y}) \end{cases}$$

- Adaptive kernel combination

$$K(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{m} \eta_i K_i(\mathbf{x},\mathbf{y})$$

$$L_d = \sum_t \alpha^t - \frac{1}{2}\sum_t \sum_s \alpha^t \alpha^s r^t r^s \sum_i \eta_i K_i(\mathbf{x}^t, \mathbf{x}^s)$$

$$g(\mathbf{x}) = \sum_t \alpha^t r^t \sum_i \eta_i K_i(\mathbf{x}^t, \mathbf{x})$$

Localized Multiple Kernel Learning

Ref: M. Gönen, E. Alpaydin (2008) "Localized Multiple Kernel Learning," *ICML'08* , Helsinki, Finland, July, 352-359.

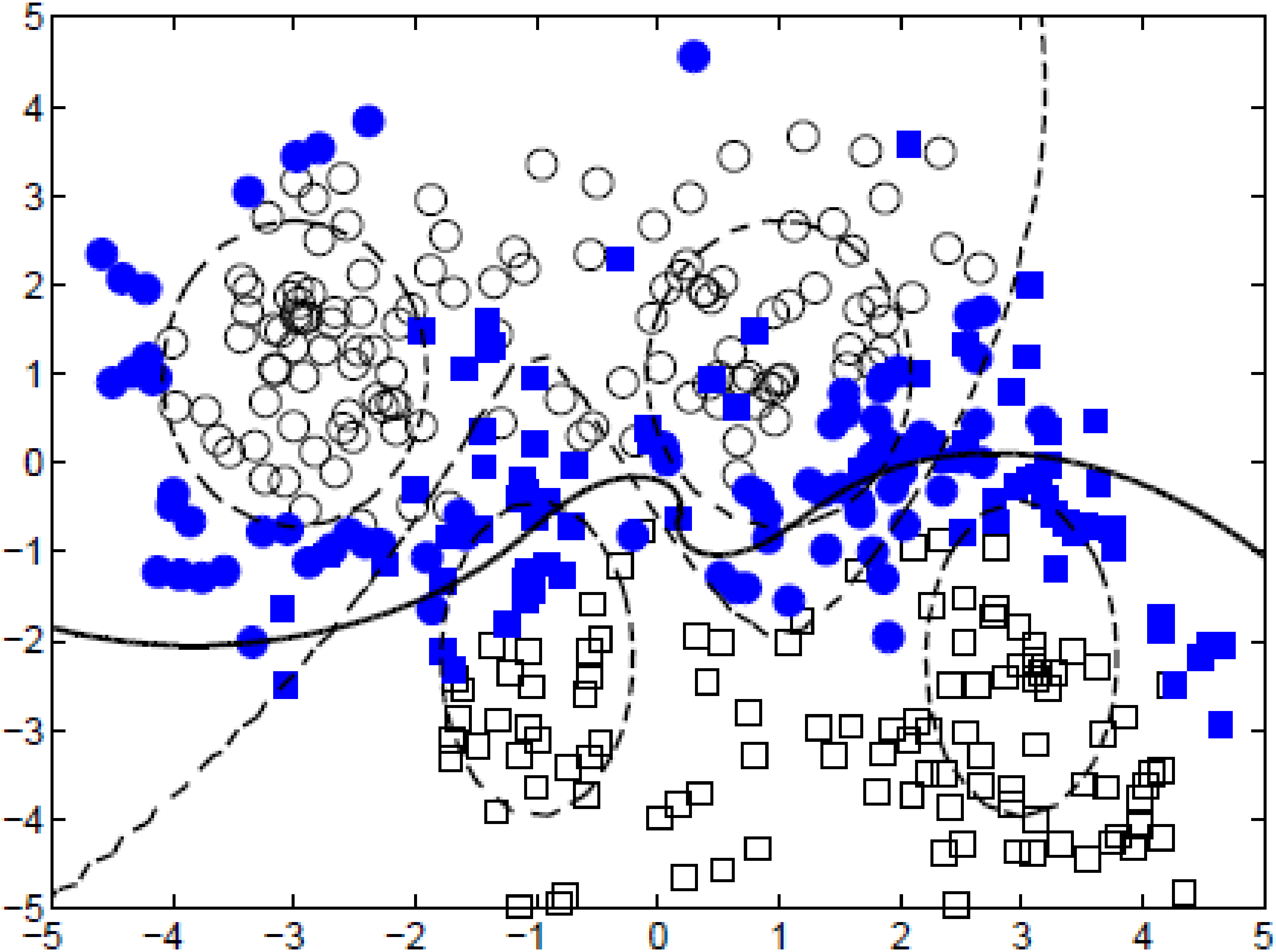
$$\min \frac{1}{2}\sum_{m=1}^{p} \|\boldsymbol{w}_m\|^2 + C\sum_{i=1}^{n}\xi_i$$

$$\text{w.r.t. } \boldsymbol{w}_m, b, \boldsymbol{\xi}, \eta_m(\boldsymbol{x})$$

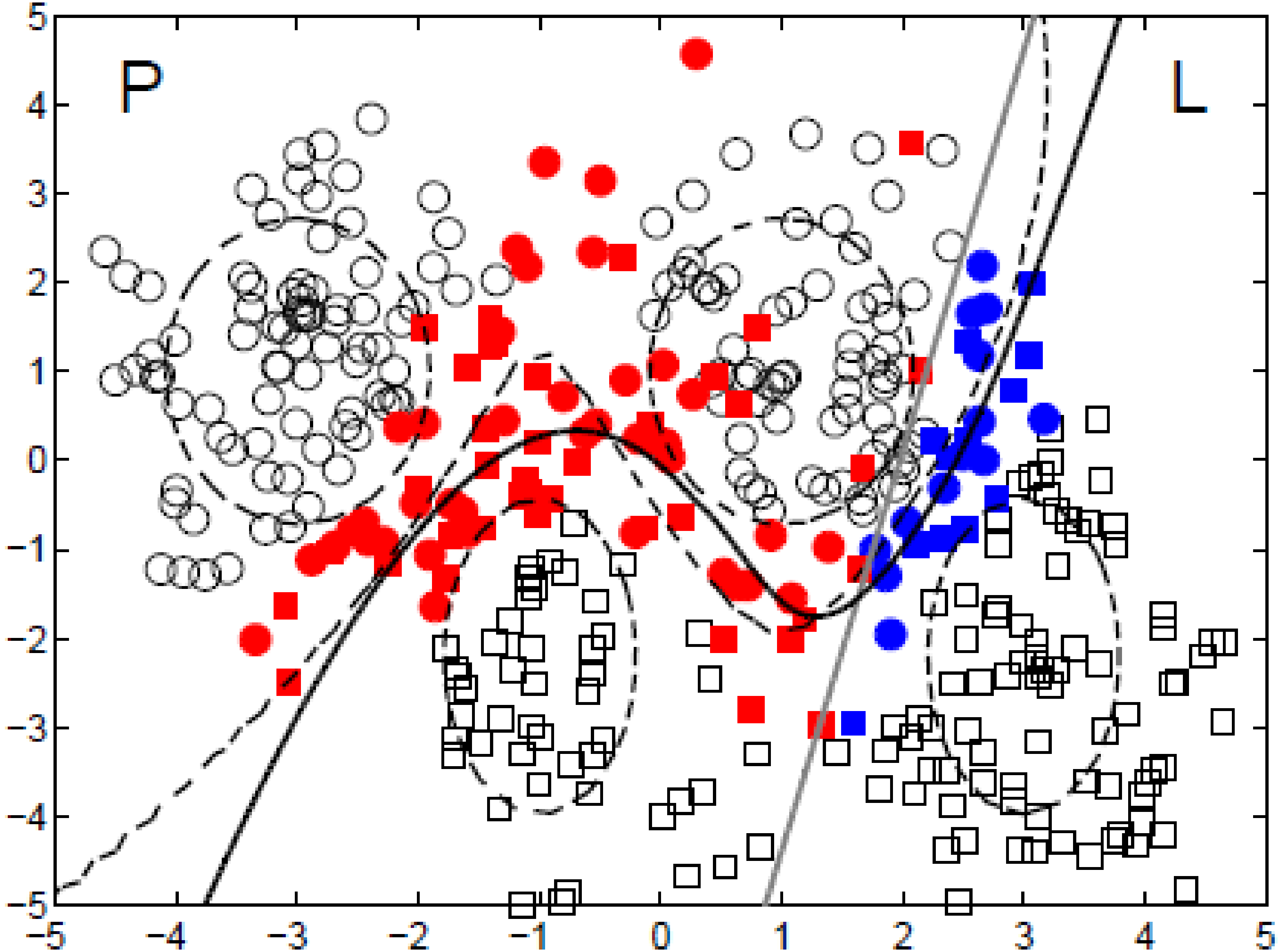
$$\text{s.t. } y_i\left(\sum_{m=1}^{p}\eta_m(\boldsymbol{x}_i)\langle\boldsymbol{w}_m, \Phi_m(\boldsymbol{x}_i)\rangle + b\right) \geq 1-\xi_i \quad \forall i$$

$$\xi_i \geq 0 \quad \forall i \qquad (5)$$

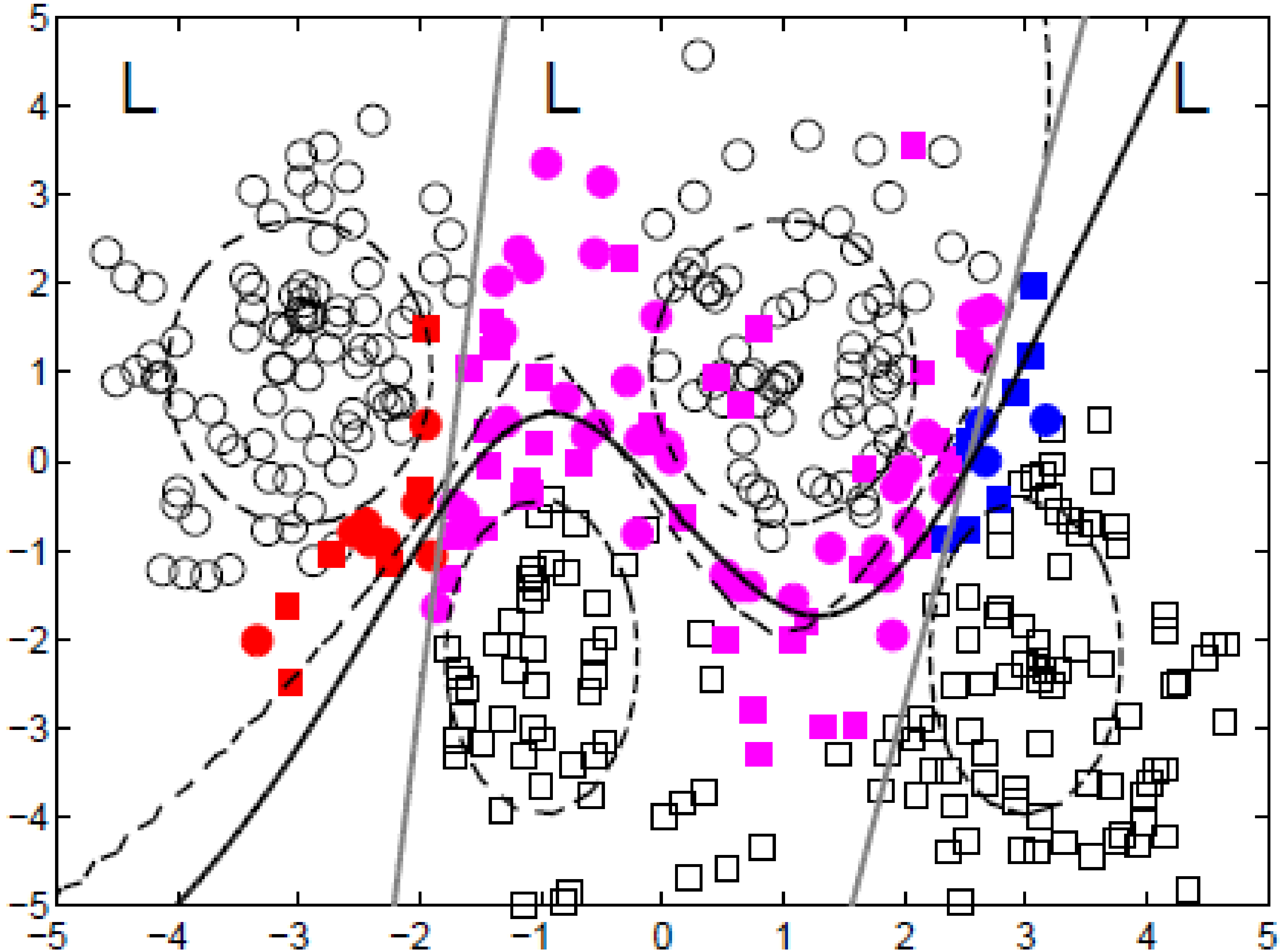
$$\eta_m(\boldsymbol{x}) = \frac{\exp(\langle\boldsymbol{v}_m, \boldsymbol{x}\rangle + v_{m0})}{\sum_{k=1}^{p}\exp(\langle\boldsymbol{v}_k, \boldsymbol{x}\rangle + v_{k0})}$$



(a) MKL with (K_L-K_P).



(b) LMKL with $(K_L\text{-}K_P)$.



(c) LMKL with (K_L-K_L-K_L).

	SVM				MKL		LMKL		SVM		LMKL	
	K_P		K_G		(K_P-K_G)		(K_P-K_G)		K_L		(K_L-K_L-K_L)	
Data Set	Acc.	SV	Acc.	SV	Acc.	SV	Acc.	SV	Acc.	SV	Acc.	SV
BANANA	56.51	75.99	83.57	92.67	81.99	93.39	83.84	83.97	59.18	93.99	**81.39**	**54.03**
GERMANNUMERIC	71.80	54.17	68.65	58.44	73.32	84.89	73.92	80.90	74.58	97.09	75.09	**57.21**
HEART	72.78	73.89	77.67	79.11	75.78	87.89	79.44	**81.44**	78.33	67.00	77.00	**58.44**
IONOSPHERE	91.54	38.55	94.36	61.71	93.68	64.10	93.33	53.33	86.15	36.58	87.86	49.06
LIVERDISORDER	60.35	69.83	64.26	74.43	63.39	93.57	64.87	92.52	64.78	85.65	64.78	78.35
PIMA	66.95	24.26	71.91	74.26	72.62	80.39	72.89	**73.63**	70.04	100.00	**73.98**	**53.09**
RINGNORM	70.66	53.91	98.82	40.68	98.86	57.68	98.69	56.69	76.91	78.68	78.92	**52.53**
SONAR	65.29	67.54	72.71	73.48	80.29	89.57	79.57	90.00	73.86	68.41	77.14	60.43
SPAMBASE	84.18	47.92	79.80	49.50	90.46	57.47	91.41	58.24	85.98	77.43	**91.18**	**34.93**
WDBC	88.73	27.11	94.44	54.74	95.50	58.11	95.98	**42.95**	95.08	**13.11**	94.34	21.89
5×2 cv Paired F Test (W-T-L)							0-10-0	3-7-0			3-7-0	6-4-0
Direct Comparison (W-T-L)							7-0-3	8-0-2			7-1-2	8-0-2
Wilcoxon's Signed Rank Test (W/T/L)							T	W			W	T

	SVM				MKL		LMKL		SVM		LMKL	
	K_P		K_G		(K_P-K_G)		(K_P-K_G)		K_L		(K_L-K_L-K_L)	
Data Set	Acc.	SV	Acc.	SV	Acc.	SV	Acc.	SV	Acc.	SV	Acc.	SV
ARABIDOPSIS	74.30	68.08	77.41	42.36	80.10	89.96	80.82	**65.41**	74.30	99.64	**81.29**	**68.66**
VERTEBRATES	75.50	68.54	75.72	41.64	78.67	90.46	77.67	68.14	75.50	99.02	**78.69**	**67.41**

Combining Learners

- Combining does not always improve accuracy; it *always* increases cost
- Need to find learners that are complementary/diverse so that accuracy improves
- Best to combine multiple sources of information (modalities) rather than algorithms, hyperparameters or data folds
- Combining features, algorithms and kernels