Statistical Methods in Artificial Intelligence
CSE471 - Monsoon 2015 : Lecture 24

Avinash Sharma

CVIT, IIIT Hyderabad

Lecture Plan

- Combining Classifiers
  - No Free Lunch Theorem
  - Overfitting Avoidance and Occam's Razor
  - Bias and Variance
  - Resampling for Classifier Design
  - Classifier combination Strategy
- End Semester Exam Syllabus

Introduction

Is some classifier better than all others?

How to compare classifiers?

Is comparison possible at all?

Is at least some classifier always better than random?

Do techniques exist which boost all classifiers?

No Free Lunch Theorem

$$\mathcal{E}[E|\mathcal{D}] = \sum_{h,F} \sum_{\mathbf{x} \notin \mathcal{D}} P(\mathbf{x})[1 - \delta(F(\mathbf{x}), h(\mathbf{x}))] P(h|\mathcal{D}) P(F|\mathcal{D}),$$

$$\mathcal{E}_k(E|F,n) = \sum_{\mathbf{x} \notin \mathcal{D}} P(\mathbf{x})[1 - \delta(F(\mathbf{x}), h(\mathbf{x}))] P_k(h(\mathbf{x})|\mathcal{D}).$$

Theorem 9.1 (No Free Lunch) *For any two learning algorithms $P_1(h|\mathcal{D})$ and $P_2(h|\mathcal{D})$, the following are true, independent of the sampling distribution $P(\mathbf{x})$ and the number n of training points:*

1. *Uniformly averaged over all target functions F, $\mathcal{E}_1(E|F,n) - \mathcal{E}_2(E|F,n) = 0$;*
2. *For any fixed training set $\mathcal{D}$, uniformly averaged over F, $\mathcal{E}_1(E|F,\mathcal{D}) - \mathcal{E}_2(E|F,\mathcal{D}) = 0$;*
3. *Uniformly averaged over all priors $P(F)$, $\mathcal{E}_1(E|n) - \mathcal{E}_2(E|n) = 0$;*
4. *For any fixed training set $\mathcal{D}$, uniformly averaged over $P(F)$, $\mathcal{E}_1(E|\mathcal{D}) - \mathcal{E}_2(E|\mathcal{D}) = 0$.**

No Free Lunch Theorem

Uniformly averaged over all target functions F,

$$\sum_F \sum_{\mathcal{D}} P(\mathcal{D}|F)\,[\mathcal{E}_1(E|F,n) - \mathcal{E}_2(E|F,n)] = 0,$$

Average over all possible target functions, the error will be the same for all classifiers.

Possible target functions: 2^5

For any fixed training set D, uniformly averaged over F,

$$\sum_F [\mathcal{E}_1(E|F,\mathcal{D}) - \mathcal{E}_2(E|F,\mathcal{D})] = 0.$$

Even if we know the training set D, the off-training errors will be the same.

	x	F	h_1	h_2
Training set D	000	1	1	1
	001	-1	-1	-1
	010	1	1	1
Off-Training set	011	-1	1	-1
	100	1	1	-1
	101	-1	1	-1
	110	1	1	-1
	111	1	1	-1

No Free Lunch Theorem

Conclusion: If no prior information about the target function F(x) is provided:

- No classifier is better than some other in the general case
- No classifier is better than random in the general case

Overfitting Avoidance and Occam's Razor

- If No Free Lunch Theorem conclude that there are no better classifiers then why classifiers with simpler hypothesis tend to perform empirically well ?

- The major reason is attributed to the natural preference of computationally simpler models for all practical reasons.

- A bias in design methodology to find only a "good enough" solution instead of a optimal solution for problem at hand.

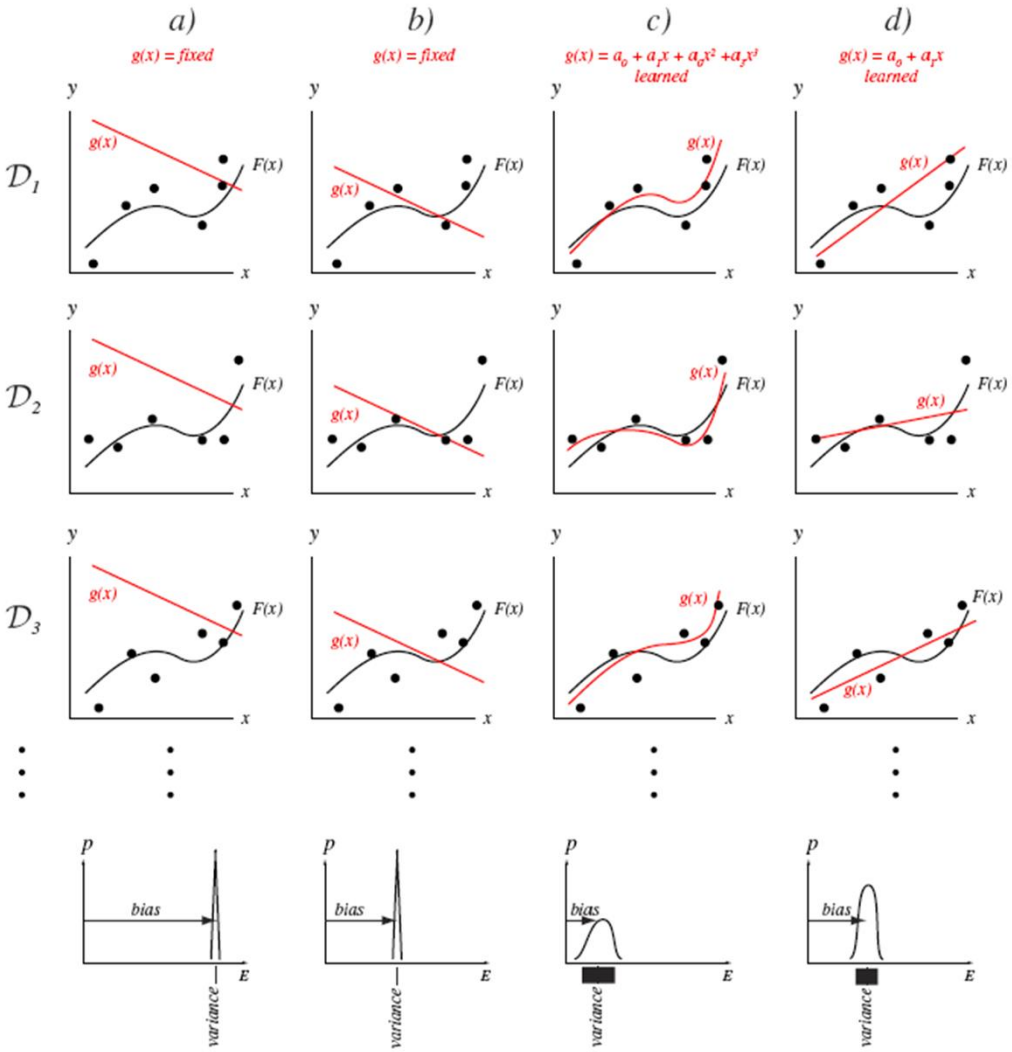
Bias and Variance

$$\varepsilon_D\left[\left(g(x,D)-F(x)\right)^2\right]$$

$$=\underbrace{\left(\varepsilon_D\left[g(x,D)-F(x)\right]\right)^2}_{bias^2}+\underbrace{\left(\varepsilon_D\left[\left(g(x,D)-\varepsilon_D\left[g(x,D)\right]\right)^2\right]\right)}_{\text{var } iance}$$

- Bias: given the training set D, we can accurately estimate F from D.
- Variance: given different training sets D, there will be no (little) differences between the estimations of F.
- Low bias means usually high variance
- High bias means usually low variance
- Best: low bias, low variance
  - **Only possible with as much as possible information about F(x).**

Bias and Variance



Resampling for Classifier Design

- Resampling of training data can be done in a way to learn a set of classifiers that together outperform single classifier learned with full training data.

- Idea is to learn an ensemble (a set) of classifiers (experts) and to allow them to vote.
  - Advantage: improvement in predictive accuracy.
  - Disadvantage: it is difficult to understand an ensemble of classifiers.

Why Ensembles work?

- Ensembles overcome three problems:
  - **The Statistical Problem** arises when the hypothesis space is too large for the amount of available data. Hence, there are many hypotheses with the same accuracy on the data and the learning algorithm chooses only one of them! There is a risk that the accuracy of the chosen hypothesis is low on unseen data!
  - **The Computational Problem** arises when the learning algorithm cannot guarantees finding the best hypothesis.
  - **The Representational Problem** arises when the hypothesis space does not contain any good approximation of the target class(es).

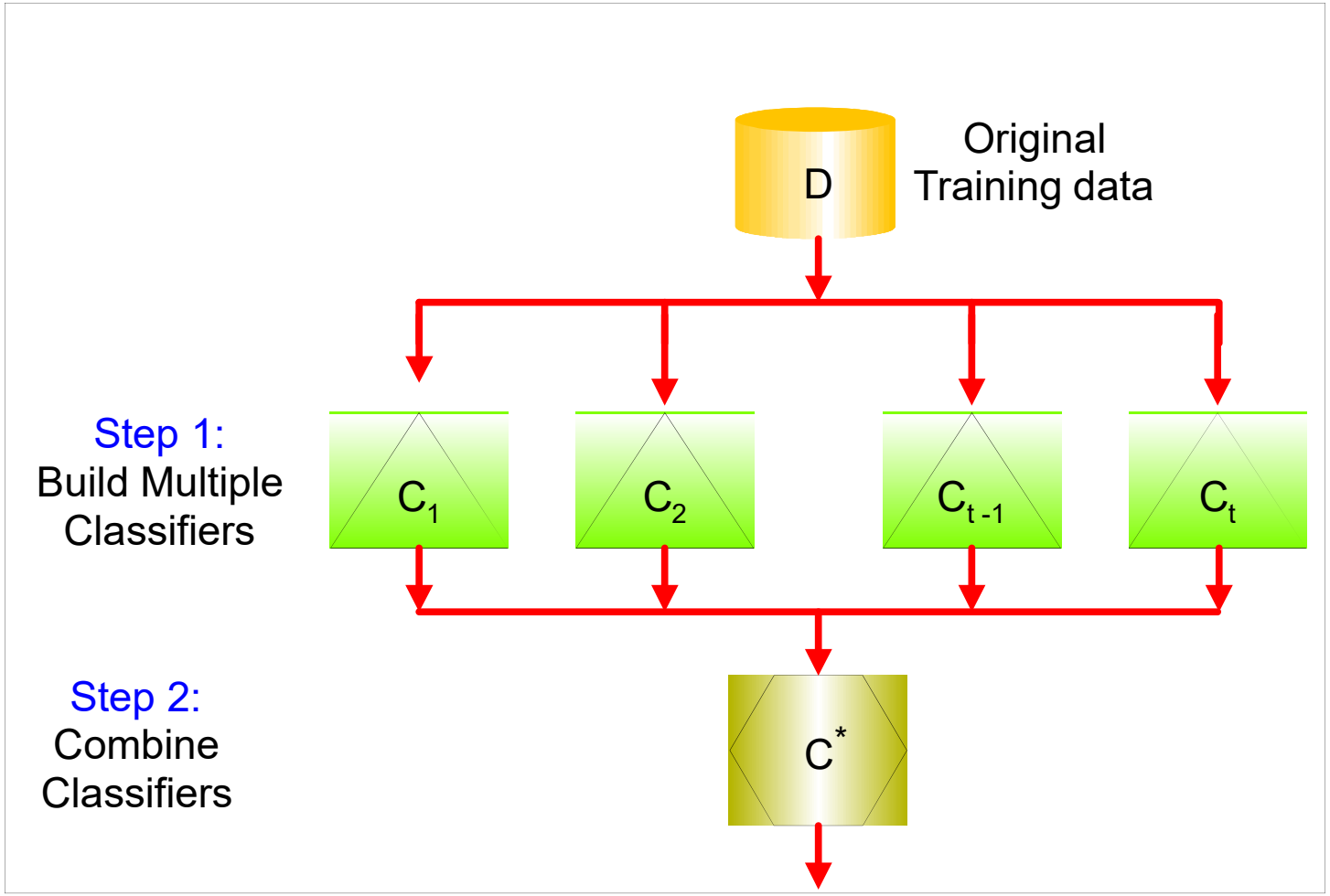
- The statistical problem and computational problem result in the variance component of the error of the classifiers!
- The representational problem results in the bias component of the error of the classifiers!

Methods for Constructing Ensembles

- **Independent Construction:** One way to force a learning algorithm to construct multiple hypotheses is to run the algorithm several times and provide it with somewhat different data in each run.
  - **Majority Voting**
  - **Bagging**
  - **Randomness Injection**
  - Feature-Selection Ensembles

- **Coordinated Construction:** The key idea is to learn complementary classifiers so that instance classification is realized by taking a weighted sum of the classifiers.
  - **Boosting**
  - Stacking

Majority Voting



Majority Voting

- Suppose there are 25 base classifiers
  - Each classifier has error rate, $\varepsilon = 0.35$
  - Assume errors made by classifiers are uncorrelated
  - Probability that the ensemble classifier makes a wrong prediction:

$$P(X \geq 13) = \sum_{i=13}^{25} \binom{25}{i} \varepsilon^i (1-\varepsilon)^{25-i} = 0.06$$

Bagging

- Draw n’ < n training points and train a different classifier
- Combine classifiers’ votes into end result.
- Classifiers are of same type: all neural networks, decision trees etc.
- Instability: small changes in the training sets leads to significantly different classifiers and/or results.
- Bagging reduces variance by voting/ averaging, thus reducing the overall expected error.
- Usually, the more classifiers the better performance.

Randomization Injection

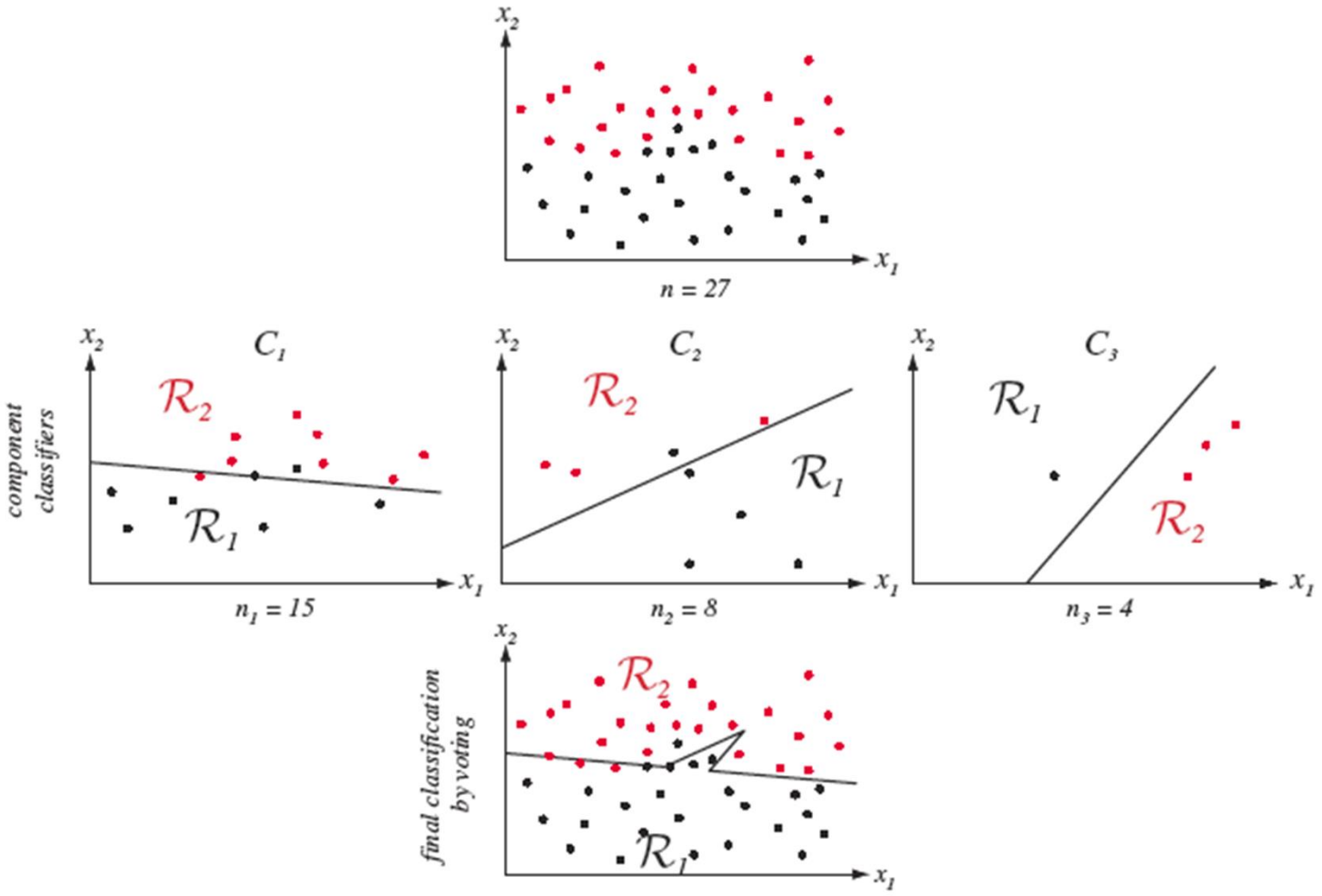
- Inject some randomization into a standard learning algorithm (usually easy):
  - Neural network: random initial weights
  - Decision tree: when splitting, choose one of the top *N* attributes at random (uniformly)

- Dietterich (2000) showed that 200 randomized trees are statistically significantly better than C4.5 for over 33 datasets!

Boosting

- Also uses voting/averaging but models are weighted according to their performance
- Iterative procedure: new models are influenced by performance of previously built ones
  - New model is encouraged to become expert for instances classified incorrectly by earlier models
  - Intuitive justification: models should be experts that complement each other
- Weak learners: Each individual classifier has accuracy only slightly better than random.
- There are several variants of this algorithm

Boosting


x_2
x_1
$n = 27$
component classifiers
C_1
$\mathcal{R}_2$
$\mathcal{R}_1$
$n_1 = 15$
C_2
$n_2 = 8$
C_3
$n_3 = 4$
final classification by voting

Adaboost

- Learning the Ensemble

1 **begin** **initialize** $\mathcal{D} = \{\mathbf{x}^1, y_1, \mathbf{x}^2, y_2, \ldots, \mathbf{x}^n, y_n\}, k_{max}, W_1(i) = 1/n, i = 1, \ldots, n$
2 $k \leftarrow 0$
3 **do** $k \leftarrow k + 1$
4 Train weak learner C_k using $\mathcal{D}$ sampled according to distribution $W_k(i)$
5 $E_k \leftarrow$ Training error of C_k measured on $\mathcal{D}$ using $W_k(i)$
6 $\alpha_k \leftarrow \frac{1}{2}\ln[(1 - E_k)/E_k]$
7 $W_{k+1}(i) \leftarrow \frac{W_k(i)}{Z_k} \times \begin{cases} e^{-\alpha_k} & \text{if } h_k(\mathbf{x}^i) = y_i \text{ (correctly classified)} \\ e^{\alpha_k} & \text{if } h_k(\mathbf{x}^i) \neq y_i \text{ (incorrectly classified)} \end{cases}$
8 **until** $k = k_{max}$
9 **return** C_k and α_k for $k = 1$ to k_{max} (ensemble of classifiers with weights)
10 **end**

- Classification Function

$$g(\mathbf{x}) = \left[\sum_{k=1}^{k_{max}} \alpha_k h_k(\mathbf{x})\right]$$

Remarks on Boosting

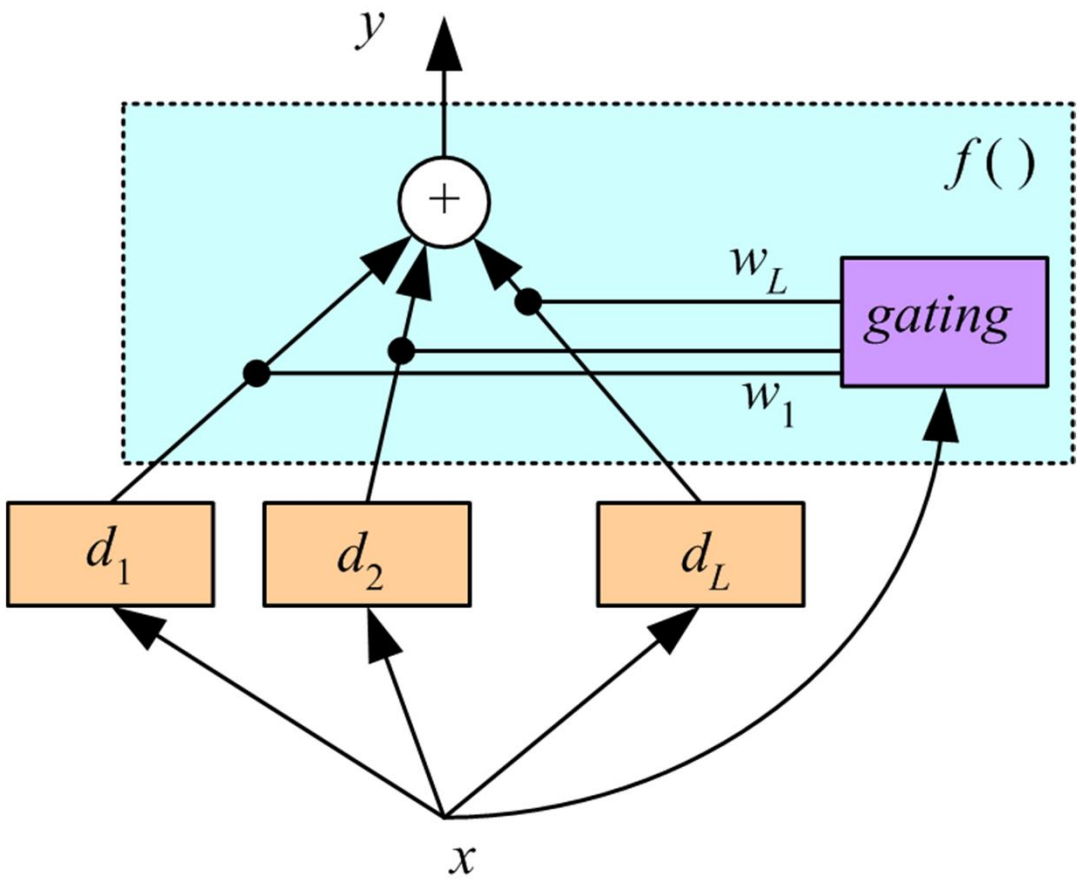
- Boosting can be applied without weights using re-sampling with probability determined by weights;
- Boosting decreases exponentially the training error in the number of iterations;
- Boosting works well if base classifiers are not too complex and their error doesn't become too large too quickly!
- Boosting reduces the bias component of the error of simple classifiers!

Classifier combination Strategies

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



End Semester Exam Syllabus

- Approximately equal contribution from 3 partitions (before mid 1, before mid 2 and after mid 2).
- What all is covered in the classes & tutorials.
- Chapter 1, Chapter 5 (LDF)
  - 5.1-5.7, 5.8.1, 5.9,
- Chapter 6 (MNN)
  - 6.1-6.5,6.8,6.10*,
- Chapter 4 (KNN) ,Chapter 2 (Normal Density, DF, Mahalanobis Distance)
  - 2.1—2.3, 2.5, 2.6, 2.8.3
- Chapter 3 (BPE, MLE, PCA, LDA)
  - 3.1, 3.2, 3.3, 3.4, 3.5, 3.5.1, 3.7, 3.8
- Chapter 5 (SVM, KSVM, Kernel s analysis)
  - 5.11, 5.12, (Refer to Chapter 1-3 of Kernel Methods for PA by Taylor & Chrtianini)
- Chapter 8 (Decision Trees)
  - 8.1--8.4
- Chapter 9 (NFL Theorem, Ensembles)
  - 9.1, 9.2, 9.2.1, 9.2.5, 9.3.1, 9.5.1, 9.5.2
- Chapter 10 (Clustering)
  - 10.4.3, 10.4.4, 10.6, 10.7, 10.9, 10.12
- Chapter 5 (SVM, Kernel SVM, Kernel definition/trick/properties)
  - 5.11, 5.12, (Refer to Chapter 1-3 of Kernel Methods for PA by Taylor & Chrtianini)
- Spectral Clustering (Referred tutorials from von-Luxburg and others)
- Graphical Models (Referred tutorials)
  - 2.11
- Please note that this syllabus is only indicative and hence the exam paper will not be strictly based on content in the listed sections of the text book. So please do refer to related public material from books/online resources.