CSCI-4/5588, ML-II
Fall, 2020
Complete Study Guide for Test #3

(Please don't distribute this study guide.
The guide is for your study purpose only)

---------- **Chapter 5: Unsupervised Learning** ----------

1. What is unsupervised learning? (Chapter #5, ~ page 1-2)

2. Write down the *k*-means clustering algorithm. (Chapter #5, ~page 4)
Ans:

Algorithm: *K-means Clustering*

Inputs: Parameter K, dataset D $\{(x_1), \ldots, (x_N)\}$.

1. Initialize cluster centroids $\{m_1, m_2, \ldots, m_K\}$ randomly.

2. For $\forall i$ compute the cluster assignment,

$$C(i) = \arg\min_{1 \le k \le K} \|x_i - m_k\|^2$$

3. For $\forall k$ compute,

$$m_k = \frac{\sum_{i=1}^{N} I\{C(i) = k\}\, x_i}{\sum_{i=1}^{N} I\{C(i) = k\}}$$

where, I is the indicator function. *Note, for any m_k if $\sum_{i=1}^{N} I\{C(i) = k\} = 0$ then we reset m_k randomly.

4. Goto step 2 to repeat, unless cluster assignments ($C(i)$) do not change.

---

3. How can we find the optimal clusters (K^*) using the K-means clustering algorithm?
Ans: (Hints: See "Practical Issues", ~ pages 6 to 7 in the lecture notes)

---------- **Chapter 6: Random Forest** ----------

4. What are the basic ideas of a bootstrap method? (Chapter#6, page 1-2)
Ans:
A bootstrap is a general tool for assessing statistical accuracy.

Suppose we have a model fit to a set of training data. We denote the training set by $Z = (z_1, z_2, \ldots, z_N)$ where $z_i = (x_i, y_i)$. The basic idea is to

(1) randomly draw datasets ***with replacement*** from the training data,
(2) this is done B times ($B = 100$ say), producing B bootstrap datasets.
(3) Then we refit the model to each of the bootstrap datasets and examine the fits over the B replications' behavior.

5. What is bagging? (Chapter #6, page 2-3)

6. Write down the steps of AdaBoost.M1 Algorithm (chapter#6, page 4)
Ans:

Algorithm: AdaBoost.M1.

---

1. Initialize the observation weights $w_i = 1/N$, $i = 1, 2, \ldots, N$.
2. For $m = 1$ to M
   (a) Fit a classifier $G_m(x)$ to the training data using weights w_i.
   (b) Compute
$$err_m = \frac{\sum_{i=1}^{N} w_i I(y_i \neq G_m(x_i))}{\sum_{i=1}^{N} w_i}$$
   (c) Compute $\alpha_m = \log((1 - err_m)/err_m)$.
   (d) Set $w_i \leftarrow w_i \cdot \exp[\alpha_m \cdot I(y_i \neq G_m(x_i))]$, $i = 1,2,\ldots,N$.
3. Output $G(x) = \text{sign}\left[\sum_{m=1}^{M} \alpha_m G_m(x)\right]$.

---

7. Write down the steps of Random Forest for regression and classification (chapter#6, page 10)

Ans:

Algorithm: *Random Forest for Regression or Classification.*

---

1. For b = 1 to B:

   (a) Draw a bootstrap sample 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 m variables at random from the p variables.

      ii) Pick the best variable/split-point among them.

      iii) Split the node into two daughter nodes.

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

To make a new prediction at a new point *x*:

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

Classification: Let $\hat{C}_b(x)$ be the class prediction of the b^{th} random-forest tree.

Then $\hat{C}_{rf}^B(x) =$ majority vote $\{\hat{C}_b(x)\}_1^B$.

---

8. Write down the motivation behind the development of the Random Forest algorithm.

Ans: The essential idea in bagging is to average many noisy but approximately unbiased models and hence reduce the variance. Trees are ideal candidates for bagging since they can capture complex interaction structures in the data, and if grown sufficiently deep, have a relatively low bias. Since trees are notoriously noisy, they benefit greatly from the averaging. Moreover, since each tree generated in bagging is identically distributed (*i.d.*), the expectation of an average of B such trees is the same as the expectation of any one of them. This means the bias of bagged trees is the same as that of the individual trees, and the only hope of improvement is through variance reduction.

---------- **Chapter 7: Recommender Systems** ----------

9. What is the long tail phenomenon? Explain.

Ans**:** See slide 6 and slide 7

10. What are the major classes of recommender systems, and how do they work?

Ans: see slide 10.

11. What is the utility matrix, and how would you formally define it?

Ans: see slide 12 and 11.

12. What are the key problems that a recommender systems address?
Ans: Slide 15.

13. How will you gather a rating for the utility matrix? Explain.
Ans: Slide 16.

14. How will you build the profile for (a) item containing the movie and (b) item containing text – explain.

Ans: Slide 21-26.

15. Suppose the only features of movies are the set of actors and the average rating. Consider two movies with five actors each. Two of the actors are in both movies. Also, one movie has an average rating of 3 and the other an average of 4. The two vectors are given below (row-wise):

0	1	1	0	1	1	0	1	3α
1	1	0	1	0	1	1	0	4α

Here, the last component shown represents the average rating, which has been shown to have an unknown scaling factor α. Explain how you will compute the similarity of the two movies and discuss the impact of the possible values of α.

Ans: Slide 27-28.

16. Describe the pros and cons of the content-based recommender system.
Ans: Slide 33-34.

17. Given the following utility matrix, among the 03 similarity measurement approaches: Jaccard similarity, cosine similarity, normalized or centered cosine similarity – which method will be most appropriate to apply and why? Explain.

	HP1	HP2	HP3	TW	SW1	SW2	SW3
A	4			5	1		
B	5	5	4				
C				2	4	5	
D		3					3

Ans: Slide 37-43.

18. Provide and explain two possible options to compute rating prediction in collaborating filtering.

Ans: Slide 45.

19. How will you use the item-item collaborative filtering to compute the rating of user *x* on item *i*? Explain.

Ans: Slide 46.

20. Describe the pros and cons of collaborative filtering.

Ans: Slide 52.

21. Describe the common practice in collaborative filtering where global baseline estimation is linearly combined to compute the ratings of user *x* on item *i*.

Ans: Slide 56, 57.

---- x ----