CS 541-A Artificial Intelligence: Final Exam

Instructor: Jie Shen

12/15/2020, 6:30 pm - 9:00 pm EST

Instructions:

- Open book exam, feel free to use any lecture notes;
- Discussion is not permitted;
- Always give your answer and explain it (guaranteed 5 point for nonempty answer);
- 20 points per problem, totally 110 points (20 * 5 + 10).

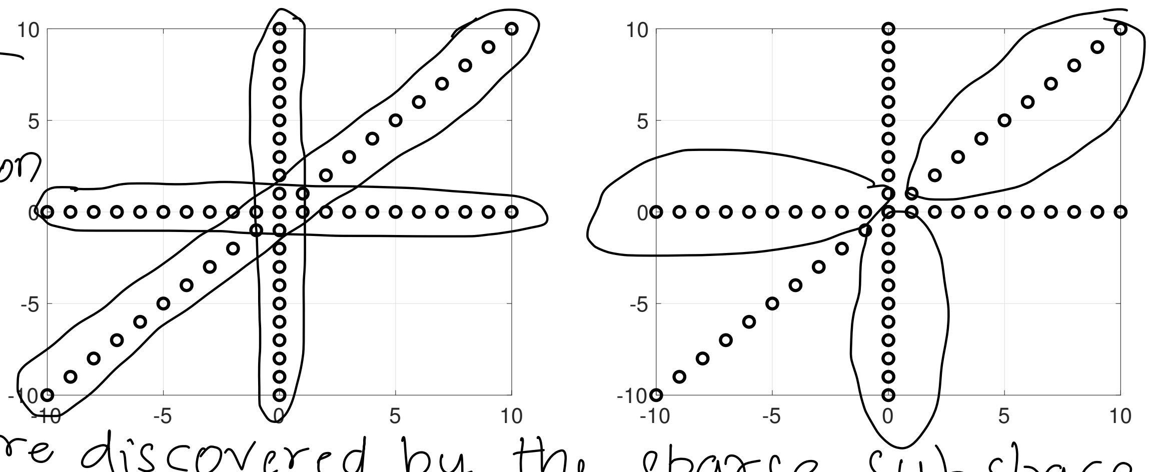
0. Write down your name. (10 pts)

Avirat Belekar

1. Consider the following data points (represented by circles) in 2-dimensional space.

- Illustrate the group structure discovered by sparse subspace clustering in the left panel;
- Illustrate the clustering result of single iteration of k-means with $k = 3$ and initial centers $(-10, 0)$, $(0, -10)$, $(10, 10)$ in the right panel.

This illustration shows the group structure discovered by the sparse subspace cl



✓**2.** Suppose we have gathered data from n patients as in Table 1, where some entries in the column "Blood Pressure" are missing (represented by the symbol "?"), and other columns are fully observed. Our goal is to estimate these missing values based on the current data matrix. Formulate it as a machine learning problem and state how we can make prediction.

Table 1: Patient data.

	Age	Weight	Height	Gender	Blood Pressure	$\cdots$	Sharp Pain
Patient 1	z_{11}	z_{12}	z_{13}	z_{14}	**?**	$\cdots$	z_{1m}
Patient 2	z_{21}	z_{22}	z_{23}	z_{24}	z_{25}	$\cdots$	z_{2m}
Patient 3	z_{31}	z_{32}	z_{33}	z_{34}	**?**	$\cdots$	z_{3m}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Patient n	z_{n1}	z_{n2}	z_{n3}	z_{n4}	z_{n5}	$\cdots$	z_{nm}

✓**3.** Now suppose we have another set of data from n patients as in Table 2, where for each column and each row there are some missing entries (represented by the symbol "?"). State when and how we can estimate all the missing values.

Table 2: Patient data.

	Age	Weight	Height	Gender	Blood Pressure	$\cdots$	Sharp Pain
Patient 1	**?**	z_{12}	z_{13}	z_{14}	**?**	$\cdots$	z_{1m}
Patient 2	**?**	z_{22}	**?**	z_{24}	z_{25}	$\cdots$	**?**
Patient 3	z_{31}	**?**	z_{33}	**?**	**?**	$\cdots$	z_{3m}
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Patient n	**?**	z_{n2}	z_{n3}	**?**	z_{n5}	$\cdots$	**?**

✓**4.** The error type of false positive is defined as follows: an algorithm outputs positive for a sample but in reality its label is negative. In some real-world applications, a learning algorithm should never incur the error of false positive. For example, the face recognition system of a laptop is designed such that it always denies unauthorized access to confidential data. Likewise, a self-driving car shall never recognize a red traffic light as green (but it is fine to classify green as red). Given a set of training samples $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^n \subset \mathbb{R}^d \times \{+1, -1\}$, state a proper formulation which is capable of preventing false positive.

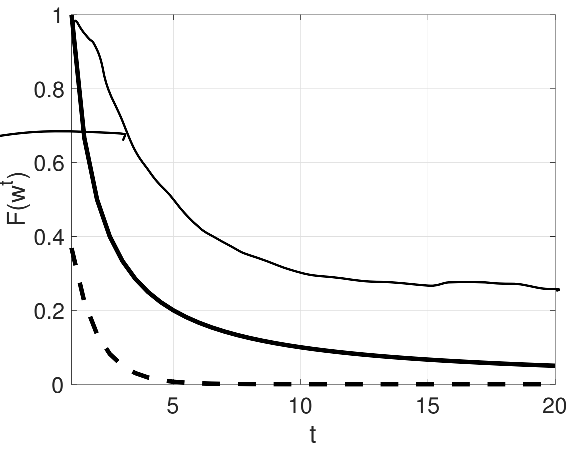
✓**5.** Consider two functions $F_1(\boldsymbol{w})$ and $F_2(\boldsymbol{w})$: both of them are strongly convex, but F_1 is smooth and F_2 is non-smooth. Suppose we apply GD to optimize these two functions. The following figure shows two convergence curves: a solid line and a dashed line. One is for $F_1(\boldsymbol{w}^t)$ and another for $F_2(\boldsymbol{w}^t)$.

- Explain which curve may correspond to F_1;
- Plot another possible convergence curve of applying GD to optimize F_2 with a different initial iterate or learning rate.

Answer:
Since F_1 is smooth and strongly convex the α value of F_1 would $\leq 0.5^+$
This means that the a_2 smooth and strongly convex function converges faster.

As we can see in the diagram below, the dashed line converges faster at time
$t =$ $t = 5$

Since f_2 is strongly non-convex and non-smooth
the α which is the convergence factor will $\frac{1}{t^2}$



This graph shows a much slower convergence of f_2 since it is non smooth and strongly convex.

2: Machine learning Algorithm

Let the data matrix be X.

Let A be a set of Columns that contains missing values column which in this Case is the blood pressure column

Now the machine learning problem can be defined as a k-nearest neighbor problem as follows:

$$X_n^* = \underset{\forall a, b \in X}{\arg\min} \ \underset{\text{Euclidian}}{d\left(X_A^{(a)}, X_{(A)}^{(b)}\right)}$$

where d denotes the distance function.

Now by Nature of design we know X_A^* we know the corresponding blood pressure column of X^*

We simply assign the mean value of

$$X_{\text{Bloodpressuremissing}} = \frac{1}{K} \sum_{j=1}^{K} X^{*}\text{blood pressure}$$

3. Let M be a binary data Matrix
Ω be the index set of
Observed data in M
X is real valued true preference matrix.

To estimate the missing values in this data matrix we will have to treat it as One bit recommender System where we can recommend the missing values based on certain preferences.

To estimate the missing values we will have to define an objective function which will learn the prediction matrix.

The objective function below is a non-convex program whose gradients will enable us to learn the prediction matrix.

$$\min_{U,V} F(U,V) = \frac{1}{2} \sum_{i,j \in \Omega_1} (M_{ij} - u_i v_j^T)^2 + \frac{\lambda}{2}\left(\|U\|_F^2 + \|V\|_F^2\right)$$

where M_{ij} is the (i,j)th entry of M

u_i and V_j are the i^{th} and j^{th} row of U and V.

To get the Objective function wrt to U and V we will have differentiate the above non convex program.

$$\frac{\partial F(U,V)}{\partial U} = \sum_{(i,j) \in \Omega_1} (u_i v_j^t - M_{ij})v_j + \lambda u_i$$

$$\frac{\partial F(U,V)}{\partial V} = \sum_{(i,j)\Omega_1} u_i (u_i V_j^T - M_{ij}) + \lambda V_j$$

Decribing the update rule that will minimize the loss and help in making better predictions.

$$U_T = U_{T-1} - \eta \frac{\partial F(U,V)}{\partial U_{T-1}}$$

$$V_T = V_{T-1} - \eta \frac{\partial F(U,V)}{\partial V_{T-1}}$$

4. Let the cost function associated with binary classification problem be denoted by

$L(y, f(x)$ where $f(x)$ is the prediction.

Since this is a binary setting, the loss function can be further split into loss function each dealing with two possible classes of +1 or −1

$L(y, f(x)) = L_1(y = -1, f(x) + L_2(y=1, f(x))$

But since we donot want any false positives, we should assign a higher penalty when the algorithm classifies negative samples incorrectly

So assign a penalty term $\lambda > 1$ to L_1

So our loss function is rewritten as :-

$$L(y, f(x) = \lambda L_1(y=-1, f(x)) + L_2(y=1, f(x)) \quad \ldots\ldots (1)$$

This subsequently changes the weight update rule as well. We Know the given set of parameter s of f can be updated :

$$\theta_{new} = \theta_{old} - \alpha \frac{\partial L (y, f(x; \theta))}{\partial \theta}$$

But from (1)

$$\theta_{new} = \theta_{old} - \alpha \Big[\lambda \frac{\partial L_1 (y = -1, f(x;\theta))}{\partial \theta}$$

$$+ \frac{\partial L_2 (y = 1, f(x;\theta))}{\partial \theta}$$

$$\theta_{new} = \theta_{old} - \alpha \lambda \frac{\partial L_1 (y = -1, f(x;\theta))}{\partial \theta}$$

$$- \alpha \frac{L_2 (y = +1, f(x;\theta))}{\partial \theta} \Big)$$

Notice that the learning rate
$\alpha\lambda > \alpha$
$\therefore \lambda > 1$

The negative class loss function is effectively increased which causes the loss function to converge faster and assign a higher penalty to loss function associate d with negative class.