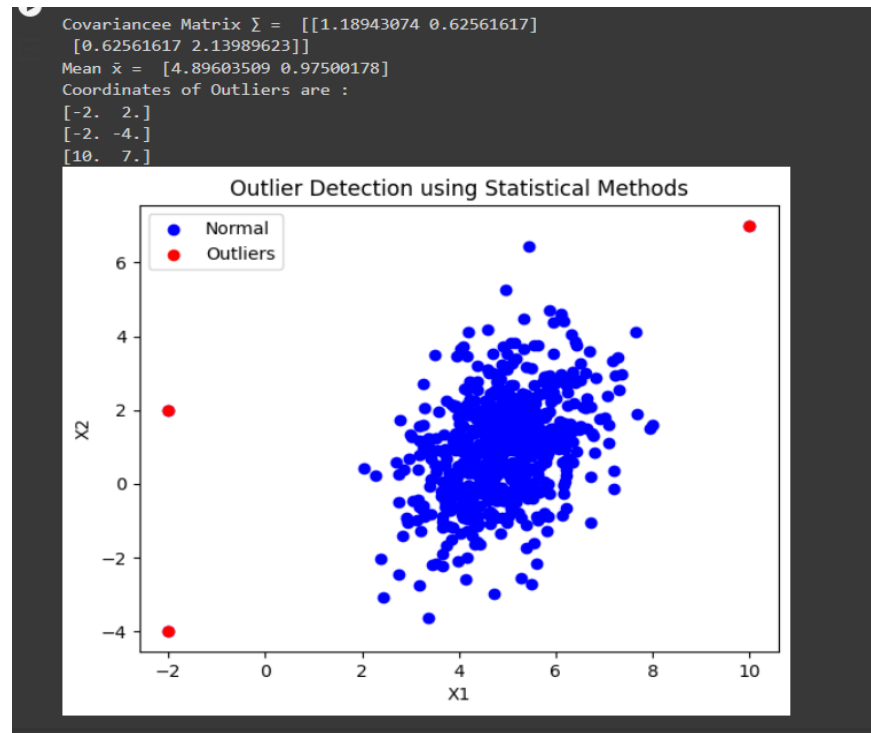
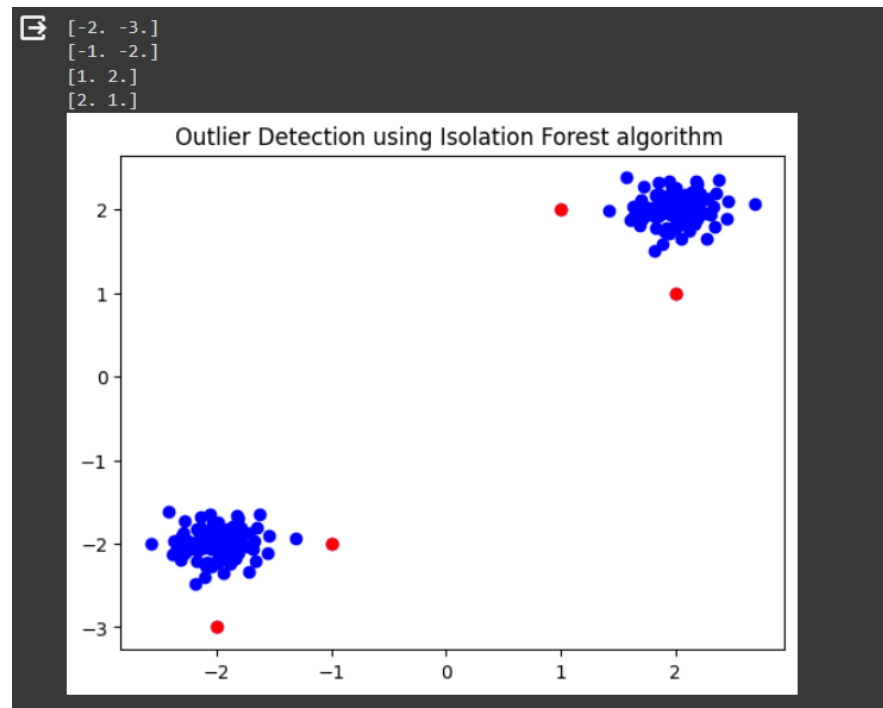


1 Outlier Detection (40pts)

1.1)



1.2)



2 Recommender Systems

- 1) In an e-commerce platform with users and items, suppose the characteristics of items are available, and the historical records (purchasement of items) of users are available, briefly introduce how to build a content-based recommender system.

→ To create a Content-based Recommender System we will need two key Components.

- 1) Item Profile
- 2) user Profile

Item Profile describes the items with features.

User Profile describes how the user likes those features.

Here, we are given the characteristics of each item and historical data of users is also available. As we are given the characteristics of each item, we are here providing with item Profile.

Now to make User Profile we have historical data of user's purchases.

We can create a user profile by doing weighted average of purchased item profile.

For a given user, we have $r_{u,i}$ given which is purchase history or rating of item I by User.

$$\mathbf{x} = \frac{\sum_i r_{u,i} \mathbf{i}}{\sum_i r_{u,i}}.$$

User Profile = \mathbf{x}

Now for a new item profile i we can estimate,

$$f(\mathbf{x}, i) = \text{similarity}(\mathbf{x}, i).$$

We can use inner product or Cosine function to find similarity.

For Example ,

If a user X has purchased two Items (RED CIRCLE) & (RED TRIANGLE) previously. And we have many other similar items available,

Like GREEN CIRCLE, RED HEXAGON, BLUE RECTANGLE & GREEN SQUARE.

So, for each item we have their characteristics.

Suppose for the given items we have 7 features available $[X1, X2, X3, X4, X5, X6, X7]$ which are

Features	X1	X2	X3	X4	X5	X6	X7
Items	Red	Green	Blue	Circle	Square	Triangle	Hex
Red Circle	1			1			
Red Triangle	1					1	
Green Circle		1		1			
Red Hex	1						1
Blue Square			1		1		
Green Square		1			1		

Now we can create the user profile from the purchased two items,

So, User Profile $X = [2, 0, 0, 1, 0, 1, 0]$

Now if we want to recommend a new item a user than we can use inner product to define similarity.

For Example,

$f(X, \text{RED HEX}) = 2$ (taking inner product between them)

$F(X, \text{GREEN CIRCLE}) = 1$

$F(X, \text{BLUE SQUARE}) = 0$

$F(X, \text{GREEN SQUARE}) = 0$

Here, for the **RED HEX**, we have the highest score of **2**, so we can recommend that item to that user.

In this way we can recommend new items to a user based on historical purchases.

- 2) In a e-commerce platform with users and items, suppose we only have the historical ratings given by some users to some items, briefly introduce how to build a recommender system.

→

Here, we have only historical ratings given by some users on some items. We do not have characteristics of items. In this case, to estimate the rating of user u on item i . We can use Collaborative Filtering.

We can use either item-item Collaborative filtering or User-User Collaborative filtering.

In User-User Collaborative Filtering we estimate user u 's rating based on the Users in N , where N is a set of similar users to user u who have rated item i .

We can find the similarity using the following formula.

$$\text{similarity}(r_1, r_2) = \frac{\sum_{i \in S} (r_{1,i} - \bar{r}_1)(r_{2,i} - \bar{r}_2)}{\sqrt{\sum_{i \in S} (r_{1,i} - \bar{r}_1)^2} \sqrt{\sum_{i \in S} (r_{2,i} - \bar{r}_2)^2}}$$

Where, $S_{1,2}$ = Common set of Items

r_1 = Average rating of User 1

r_2 = Average rating of User 2

$r_{1,s}$ = Rating is user 1 on item s

$r_{2,s}$ = Rating is user 2 on item s

We can predict rating of User u on item i using Following Formula,

$$r_{u,i} = \frac{1}{|N|} \sum_{v \in N} r_{v,i}$$
$$r_{u,i} = \frac{\sum_{v \in N} \text{sim}_{u,v} \cdot r_{v,i}}{\sum_{v \in N} \text{sim}_{u,v}}$$

Apart from this, we can also use item-item Collaborative Filtering.

In Item-Item Collaborative filtering, for item i , we find other similar items. We estimate the rating for item i based on rating for similar items. We use similar metrics and rating functions as in the user-user view.

$$r_{u,i} = \frac{\sum_{j \in N} sim_{i,j} \cdot r_{u,j}}{\sum_{j \in N} sim_{i,j}},$$

Where, $Sim_{i,j}$ = Similarity between item i and j

N is the set of items rated by User u similar to i

Also In Practice, Item-Item Collaborative Filtering works better than user-user Collaborative filtering because in practice users have multiple tastes.

3 Recommendation Evaluation

Name : Abhishek. Milind. Patwardhan
ID : 811271359

①

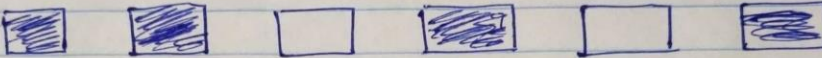
Data Science - 2

HomeWork - 4.

Q.3] Recommendation Evaluation

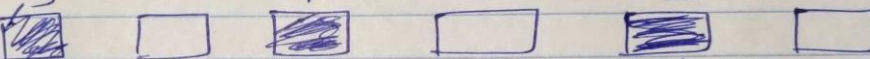
1]

→ Precision for $K = 4 \Rightarrow 3/4$
 $K = 5 \Rightarrow 3/5$
 $K = 6 \Rightarrow 4/6$

2] Q_1 

→ precision is: $1/1$ $2/2$ $2/3$ $3/4$ $3/5$ $4/6$

Average Precision $AP(Q_1) = (1 + 1 + 3/4 + 4/6) / 4 = 0.8541$

Q_2 
precision $1/1$ $1/2$ $2/3$ $2/4$ $3/5$ $3/6$

Average Precision $AP(Q_2) = (1 + 2/3 + 3/5) / 3 = 0.755$

MAP (Mean AP) = $(0.8541 + 0.755) / 2 = \underline{\underline{0.8045}}$

3]

$$\rightarrow DCC_5 = 3 + \frac{0}{\log 2} + \frac{1}{\log 3} + \frac{0}{\log 4} + \frac{2}{\log 5}$$

$$= 3 + 0.630 + 0.8613 + 0 + 0.4471 = 4.491$$

$$DCC_{max} = 3 + \frac{1}{\log 2} + \frac{2}{\log 3} + 0 + 0$$

$$= 3 + 1 + 1.261 = 5.261$$

$$\therefore NDCG_5 = \frac{DCC_5}{DCC_{max}} = \frac{4.491}{5.261} = \underline{\underline{0.853}}$$

4)



Normalized Discounted Cumulative Gain (NDCG) has some clear advantages over Precision@K and Mean Average Precision (MAP) in the world of information retrieval and recommender systems.

- When dealing with graded relevance: NDCG considers the varying degrees of relevance for items, while Precision@K and MAP only care about whether an item is relevant or not. This means NDCG gives higher scores to items that are not just relevant, but highly so. So, if we're dealing with items that have different levels of importance, NDCG offers a more detailed evaluation.
- When focusing on the ranking quality of top-ranked items: NDCG is tough on getting the ranking right, especially for those important items. If a highly relevant item is buried deep in the list, NDCG will show it no mercy. Precision@K and MAP, on the other hand, might not mind as much, if those items make it onto the list somewhere. So, if you want to make sure the cream rises to the top in your recommendations, NDCG is your friend.
- When dealing with imbalanced datasets: NDCG can handle datasets where there's an uneven number of relevant and non-relevant items. Precision@K and MAP might struggle with this, leading to skewed evaluations. NDCG steps in with its normalization factor to level the playing field, making sure all items get a fair shake regardless of the dataset's quirks.