6339- Advanced topics in database systems

Methods of Knowledge Discovery in Healthcare

Project 1

Team-

Ruchir Kadam- 1001227805

Maithili Deshpande- 1001230528

Task 1

1. **What is the most common disease for each age group?**

   Solution:

   Query:

```
select D.age,d.diagnosis_code_1,d.countn from
(SELECT B.`AGE`,B.`DIAGNOSIS_CODE_1` , MAX(B.COUNTN) COUNTN  FROM
(SELECT A.`AGE`,A.`DIAGNOSIS_CODE_1`, COUNT(DIAGNOSIS_CODE_1) COUNTN
FROM (SELECT `AGE`,`DIAGNOSIS_CODE_1`
FROM `PATIENT_INFO`UNION ALL SELECT `AGE`,`DIAGNOSIS_CODE_2` FROM `PATIENT_INFO`) AS A
GROUP BY A.`AGE`,A.`DIAGNOSIS_CODE_1`
ORDER BY AGE ASC, COUNTN DESC) AS B
GROUP BY B.`AGE`)C,
(SELECT A.`AGE`,A.`DIAGNOSIS_CODE_1`, COUNT(DIAGNOSIS_CODE_1) COUNTN
FROM (SELECT `AGE`,`DIAGNOSIS_CODE_1`
FROM `PATIENT_INFO`UNION ALL SELECT `AGE`,`DIAGNOSIS_CODE_2` FROM `PATIENT_INFO`) as A
GROUP BY A.`AGE`,A.`DIAGNOSIS_CODE_1`
ORDER BY AGE ASC, COUNTN DESC)D
where d.age=c.age
and d.countn=c.countn
```

Output:

age	diagnosis_code_1	countn
1	2767	1
1	51884	1
1	4168	1
1	7580	1
1	5856	1
1	2153	1
1	65571	1
1	64291	1
2	V6284	5
3	486	15
4	5856	25
5	486	14
6	486	22
7	486	22
8	5990	20
9	5849	11

Max query on the count of common diagnosis code (1 & 2) returns most common disease for every age group, where group by function is used to group results according to age.

2. **What is the prevalence of the top three diseases for each age group?**

Solution:

Query:

```
SELECT `q2`.`AGE`, `q2`.`DIAGNOSIS_CODE`, `q2`.`AFFECTED`,`population`.`total`,
`q2`.`AFFECTED`/`population`.`total` PREVALENCE FROM `q2`,`population`
WHERE `q2`.`AGE`= `population`.`AGE`
```

VIEWS CODES:

Q2: Returns count of patients affected from top 3 diagnosis within every age group

```
CREATE VIEW Q2 AS
SELECT `AGE`,`DIAGNOSIS_CODE`, COUNT(`AGE`) AFFECTED FROM `AGE_DIAG`
WHERE `DIAGNOSIS_CODE` IN
(SELECT * FROM TOP3)
GROUP BY `AGE`,`DIAGNOSIS_CODE`
```

ORDER BY `AGE`

AGE_DIAG: Returns combined result of age, diagnosis_code_1 and diagnosis_code_2

CREATE VIEW AGE_DIAG AS

SELECT `AGE`,`DIAGNOSIS_CODE_1` DIAGNOSIS_CODE FROM `PATIENT_INFO`

UNION ALL

SELECT `AGE`,`DIAGNOSIS_CODE_2` DIAGNOSIS_CODE FROM `PATIENT_INFO`

DIAG: Returns combined result of Diagnosis_code_1 and Diagnosis_code_2

CREATE VIEW DIAG AS

SELECT `DIAGNOSIS_CODE_1` DIAGNOSIS_CODE FROM `PATIENT_INFO`

UNION ALL

SELECT `DIAGNOSIS_CODE_2` DIAGNOSIS_CODE FROM `PATIENT_INFO`

TOP3: Returns top 3 frequent diagnosis codes.

CREATE VIEW TOP3 AS

SELECT DIAG.`DIAGNOSIS_CODE` FROM DIAG

GROUP BY DIAG.`DIAGNOSIS_CODE`

ORDER BY COUNT(DIAG.`DIAGNOSIS_CODE`) DESC

LIMIT 3

POPULATION: Returns count of patients in every age group.

CREATE VIEW POPULATION AS

SELECT `AGE`, COUNT(`AGE`) total FROM `PATIENT_INFO`

GROUP BY `AGE`

Output:

Query calculates the prevalence of top3 diagnosis, by finding the ratio of affected patients and total patients in each age group for respective diagnosis code.

AGE	DIAGNOSIS_CODE	AFFECTED	total	PREVALENCE
2	0389	2	68	0.0294
2	486	3	68	0.0441
2	5849	1	68	0.0147
3	0389	8	228	0.0351
3	486	15	228	0.0658
3	5849	13	228	0.0570
4	0389	12	220	0.0545
4	486	14	220	0.0636
4	5849	11	220	0.0500
5	0389	13	216	0.0602
5	486	14	216	0.0648
5	5849	9	216	0.0417
6	0389	9	212	0.0425
6	486	22	212	0.1038
6	5849	12	212	0.0566
7	0389	13	220	0.0591
7	486	22	220	0.1000
7	5849	14	220	0.0636
8	0389	12	188	0.0638
8	486	13	188	0.0691
8	5849	14	188	0.0745
9	0389	6	121	0.0496
9	486	8	121	0.0661
9	5849	11	121	0.0909

3. **What is the average death_on_discharge ratio for each admitting diagnosis code?**

   Solution:

   Query:

```
select a.c as 'Admitting Diagnosis code', b.cnt/(b.cnt+a.cnt) as 'Death on discharge ratio'
from (select `ADMITTING_DIAGNOSIS_CODE` as c, DISCHARGE_STATUS, count(`DISCHARGE_STATUS`) cnt
from `patient_info`
where `DISCHARGE_STATUS`='A'
group by c,`DISCHARGE_STATUS`) a,
(select ADMITTING_DIAGNOSIS_CODE as c, DISCHARGE_STATUS, count(`DISCHARGE_STATUS`) cnt
from `patient_info`
```

where `DISCHARGE_STATUS`='B'

group by c,`DISCHARGE_STATUS`) b

where a.c=b.c

Output:

Admitting Diagnosis code	Death on discharge ratio
0389	0.2000
2989	0.0833
40391	0.3333
4280	0.0370
431	0.2500
4329	0.5000
43491	0.1818
486	0.0217
5070	0.3333
51881	0.3500
51884	0.2000
5722	0.3333
5781	0.1429
586	0.5000
59960	0.5000
71945	0.1538
7802	0.0357
78060	0.0526
78079	0.0323
78097	0.0444
78605	0.0159
78650	0.0182
V5811	0.2500

The inner select queries count the discharge status (alive or dead) of patients for every admitting diagnosis code. The outer query calculates the average of death on discharge by dividing the number of patients who are dead with the total number of patients from respective admitting diagnosis codes.

4. What is the most common value combination for the source_of_admission and discharge_destination?

Solution:

Query:

SELECT `SOURCE_OF_ADMISSION`,`DISCHARGE_DESTINATION`, COUNT(*) NUM FROM

`PATIENT_INFO`

```
GROUP BY `SOURCE_OF_ADMISSION`,`DISCHARGE_DESTINATION`
ORDER BY NUM DESC
LIMIT 1
```

Output:

SOURCE_OF_ADMISSION	DISCHARGE_DESTINATION	NUM
1	1	619

The most common combination for source of admission and discharge destination is 1 for both (Physician referral and discharge to home) which is repeated for 619 times.

5. Compare the in-hospital mortality of men and women

Solution:

Query:

Mortality rate of men AND women:

```
CREATE VIEW DEAD AS
SELECT `SEX`, COUNT(`SEX`) DEAD FROM `patient_info`
WHERE `DISCHARGE_STATUS`= 'B'
GROUP BY `SEX`
CREATE VIEW SEX_INFO AS
SELECT `SEX`, COUNT(*) TOTAL FROM `patient_info`
GROUP BY `SEX`

SELECT A.`SEX`,A.`DEAD`,B.`TOTAL`, A.`DEAD`/B.`TOTAL` MORTALITY_RATE FROM `dead`AS A,`SEX_INFO` AS B
WHERE A.`SEX` = B.`SEX`
```

Output:

SEX	DEAD	TOTAL	MORTALITY_RATE
1	18	644	0.0280
2	29	833	0.0348

Mortality rate is calculated by dividing number of dead patients with total number of patients in the hospital. These patients are grouped by sex. The output shows that the mortality rate for women is higher than mortality rate for men.

6. What is the most common long stay primary diagnosis (DIAGNOSIS_CODE_1) and which is the most common short stay primary diagnosis (DIAGNOSIS_CODE_1)

Solution:

Query:

```
(select `STAY_INDICATOR`,`DIAGNOSIS_CODE_1`, count(`DIAGNOSIS_CODE_1`) as cnt
from `patient_info`
where `STAY_INDICATOR`='L'
group by STAY_INDICATOR,DIAGNOSIS_CODE_1
order by cnt desc limit 1)
union all
(select `STAY_INDICATOR`,`DIAGNOSIS_CODE_1`, count(`DIAGNOSIS_CODE_1`) as cnt
from `patient_info`
where `STAY_INDICATOR`='S'
group by STAY_INDICATOR,DIAGNOSIS_CODE_1
order by cnt desc limit 1)
```

Output:

STAY_INDICATOR	DIAGNOSIS_CODE_1	cnt
L	V5789	18
S	0389	65

Output is the union of the most common short stay diagnosis code and the most common long stay diagnosis code.

V5789 is the most common long stay diagnosis code whereas, 0389 is the most common short stay diagnosis code.

7. What is the average total cost (total charges) for each length of stay?

Solution:

Query:

```
select `LENGTH_OF_STAY`, avg(`TOTAL_CHARGES`)
```

from `patient_info`

group by `LENGTH_OF_STAY`

Output:

LENGTH_OF_STAY	avg(`TOTAL_CHARGES`)
1	18305.1159
2	20836.3945
3	26814.1084
4	27489.2429
5	35170.9545
6	36867.8077
7	49794.0303
8	47305.7222
9	49811.3333
10	63270.4000
11	73906.9355
12	67934.7647
13	48245.0625
14	82098.1538
15	72132.8824
16	72939.0000
17	59692.2000
18	76199.6250
19	92606.2500
20	111915.1250
21	273411.2500
22	163407.5000
23	342789.0000
24	179778.0000
25	118402.0000
27	184175.5000
28	151800.5000
29	40754.0000
31	96186.0000
33	561217.0000
34	127264.5000
35	254410.0000
37	228115.0000
38	208234.3333
39	316960.0000
41	262986.0000
42	168752.0000
51	389386.0000
55	135805.0000
58	217458.5000
110	934443.0000

Group by clause using length of stay, categorizes total charges for respective groups. We applied avg function to calculate the average of total charges for respective groups.

8. Compare the total charges for each admission diagnosis code, for the age group 5

Solution:

Query:

select admitting_diagnosis_code,sum(Total_charges) 'Total charges'

from patient_info

where age=5

group by admitting_diagnosis_code

Output:

admitting_diagnosis_code	Total charges
00845	18238
0389	192240
1589	16597
179	13284
2113	64983
2352	27830
2409	66525
24200	11699
25000	13034
25010	19864
25080	4368
2760	40939
2761	19644
27801	12259
2841	15837
28489	10533
2859	35735
28800	109804
29620	83074
29624	17667

Output shows a part of the records where, Total charges is the sum of total charges for respective admission diagnosis codes for the age group 5

9. Compare the average length of stay for type of admission=1 between men and women

Solution:

Query:

```
SELECT avg(`LENGTH_OF_STAY`) as 'Average Length of Stay', 'Men' as Gender  FROM `patient_info`

WHERE `TYPE_OF_ADMISSION`=1 and `SEX`=1

union

SELECT avg(`LENGTH_OF_STAY`) as 'Average Length of Stay', 'Women' as Gender FROM `patient_info`

WHERE `TYPE_OF_ADMISSION`=1 and `SEX`=2
```

Output:

Average Length of Stay	Gender
5.1753	Men
5.4960	Women

Output shows that the average length of stay of type of admission 1 is longer in case of women as compared to men.

10. Find the top 3 most expensive DRG codes (i) based on the DRG price and (ii) based on the total charges.

Solution:

Query: Below query shows top 3 most expensive DRG codes based on DRG price.

```
select `DRG_CODE`, max(`DRG_PRICE`) price
from `patient_info`
group by `DRG_CODE`
order by price desc limit 3
```

Output:

DRG_CODE	price
1	224066
4	126095
3	111683

Query: Below query shows top 3 most expensive DRG codes based on total charges.

```
select `DRG_CODE`, max(`total_charges`) charge
from `patient_info`
group by `DRG_CODE`
order by charge desc limit 3
```

Output:

DRG_CODE	charge
811	1161274
1	934443
207	665482

Task 2

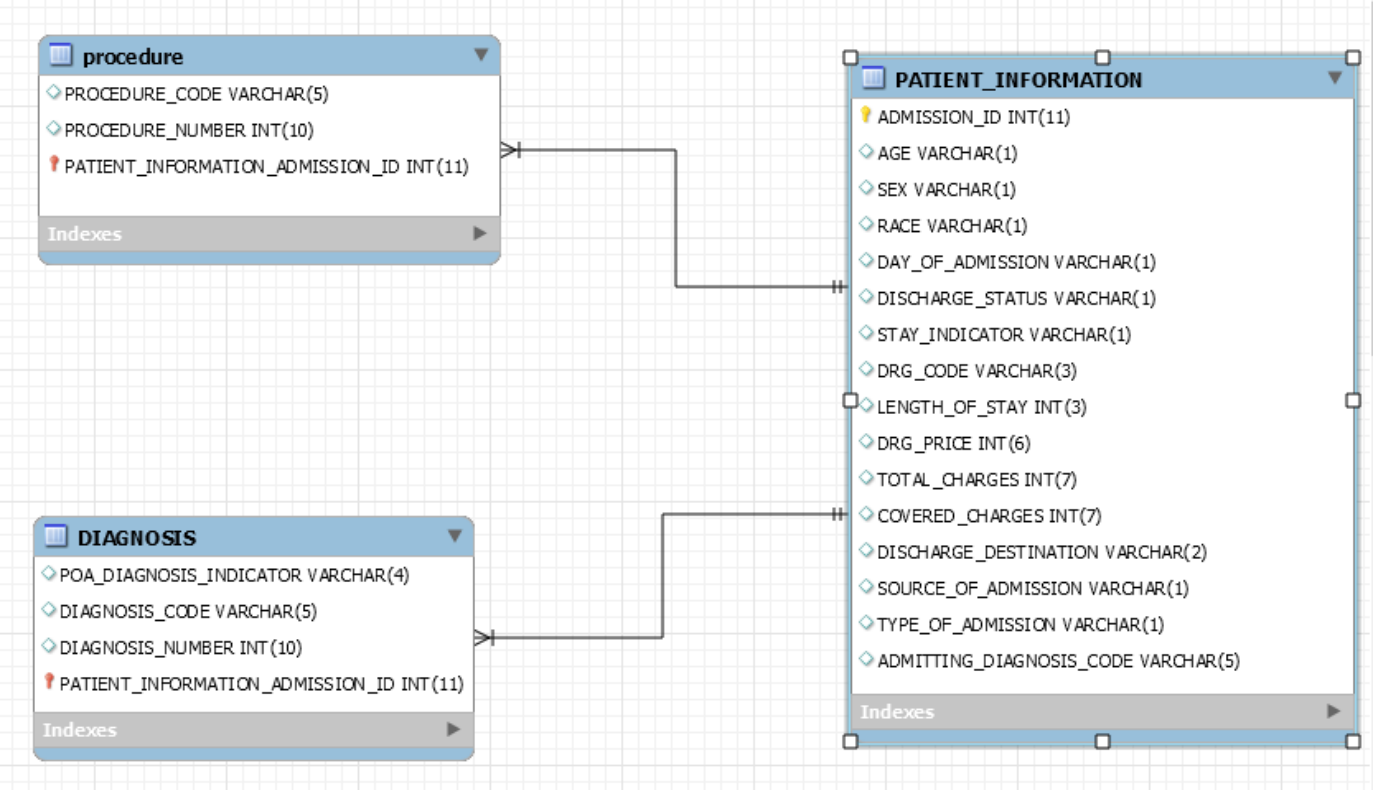
1. **Observe the data and design an appropriate relational schema. Then create the schema on the DBMS system of your choice (preferably MySQL server). The schema should have the following merits:**

    a. Normalization principles should be applied to avoid duplicates

    b. Appropriate data types should be defined

Solution:

Schema Diagram:



Schema on DBMS system:

We have used phpMyAdmin to create database schema-6339. We imported 6339.csv file into phpMyAdmin and applied normalization principles on the table 'patient_info'.

1NF Normalization-

We identified columns with noisy data/missing values/null values from patient_info.

POA_DIAGNOSIS_INDICATOR_1, POA_DIAGNOSIS_INDICATOR_2, DIAGNOSIS_CODE_1, DIAGNOSIS_CODE_2,

PROCEDURE_CODE_1, PROCEDURE_CODE_2.

Query:

```
UPDATE `patient_info` SET `PROCEDURE_CODE_1` = NULL WHERE `PROCEDURE_CODE_1`=''

UPDATE `patient_info` SET `PROCEDURE_CODE_2` = NULL WHERE `PROCEDURE_CODE_2`=''

UPDATE `patient_info` SET `POA_DIAGNOSIS_INDICATOR_1`= NULL WHERE
`POA_DIAGNOSIS_INDICATOR_1` = ''

UPDATE `patient_info` SET `POA_DIAGNOSIS_INDICATOR_2`= NULL WHERE
`POA_DIAGNOSIS_INDICATOR_2` = ''

UPDATE `patient_info` SET `DIAGNOSIS_CODE_1` = NULL WHERE `DIAGNOSIS_CODE_1` = ''

UPDATE `patient_info` SET `DIAGNOSIS_CODE_2` = NULL WHERE `DIAGNOSIS_CODE_2` = ''
```

2NF Normalization-

ADMISSION_ID has been added as a primary key into patient_info table.

Query:

```
ALTER TABLE `patient_information` ADD `admission_id` INT AUTO_INCREMENT PRIMARY KEY;
```

3NF Normalization-

Appropriate tables/data types are defined to remove transitive dependency.

I. Created table patient_information using patient_info table.
II. Created table Diagnosis(Admission_Id,POA_Diagnosis_Indicator,Diagnosis_Code,Diagnosis_Number) using phpMyAdmin.
III. Created table Procedure(Admission_id, Procedure_code, Procedure_Number) using phpMyAdmin.
IV. Dropped columns POA_Diagnosis_Indicator_1,Diagnosis_Code_1, POA_Diagnosis_Indicator_2,Diagnosis_Code_2, Procedure_code_1, Procedure_code_2 from patient information.

Query:

CREATE TABLE `patient_information` LIKE `patient_info`;

CREATE TABLE Procedure(Admission_id INT(11), Procedure_code VARCHAR(5),Procedure_Number INT(10))

CREATE TABLE Diagnosis(Admission_id INT(11), POA_Diagnosis_Indicator VARCHAR(4),Diagnosis_Code VARCHAR(5),Diagnosis_Number INT(10))

ALTER TABLE `procedure` ADD FOREIGN KEY (ADMISSION_ID) REFERENCES `patient_information`(ADMISSION_ID)

ALTER TABLE `diagnosis` ADD FOREIGN KEY (ADMISSION_ID) REFERENCES `patient_information`(ADMISSION_ID)

Patient_information

#	Name	Type
1	AGE	varchar(1)
2	SEX	varchar(1)
3	RACE	varchar(1)
4	DAY_OF_ADMISSION	varchar(1)
5	DISCHARGE_STATUS	varchar(1)
6	STAY_INDICATOR	varchar(1)
7	DRG_CODE	varchar(3)
8	LENGTH_OF_STAY	int(3)
9	DRG_PRICE	int(6)
10	TOTAL_CHARGES	int(7)
11	COVERED_CHARGES	int(7)
12	DISCHARGE_DESTINATION	varchar(2)
13	SOURCE_OF_ADMISSION	varchar(1)
14	TYPE_OF_ADMISSION	varchar(1)
15	ADMITTING_DIAGNOSIS_CODE	varchar(5)
16	admission_id	int(11)

Procedure

#	Name	Type
1	ADMISSION_ID	int(11)
2	PROCEDURE_CODE	varchar(5)
3	PROCEDURE_NUMBER	int(10)

Diagnosis

#	Name	Type
1	ADMISSION_ID	int(11)
2	POA_DIAGNOSIS_INDICATOR	varchar(4)
3	DIAGNOSIS_CODE	varchar(5)
4	DIAGNOSIS_NUMBER	int(10)

2. Import the data into your newly developed schema

Solution:

Data is imported in the Diagnosis and the Procedure table in such a way that the null values are eliminated from patient_information table.

Patient_Information

insert into `patient_information`

(`AGE`, `SEX`, `RACE`, `DAY_OF_ADMISSION`, `DISCHARGE_STATUS`, `STAY_INDICATOR`, `DRG_CODE`, `LENGTH_OF_STAY`, `DRG_PRICE`, `TOTAL_CHARGES`, `COVERED_CHARGES`, `POA_DIAGNOSIS_INDICATOR_1`, `POA_DIAGNOSIS_INDICATOR_2`, `DIAGNOSIS_CODE_1`, `DIAGNOSIS_CODE_2`, `PROCEDURE_CODE_1`, `PROCEDURE_CODE_2`, `DISCHARGE_DESTINATION`, `SOURCE_OF_ADMISSION`, `TYPE_OF_ADMISSION`, `ADMITTING_DIAGNOSIS_CODE`)

SELECT

`AGE`, `SEX`, `RACE`, `DAY_OF_ADMISSION`, `DISCHARGE_STATUS`, `STAY_INDICATOR`, `DRG_CODE`, `LENGTH_OF_STAY`, `DRG_PRICE`, `TOTAL_CHARGES`, `COVERED_CHARGES`, `POA_DIAGNOSIS_INDICATOR_1`, `POA_DIAGNOSIS_INDICATOR_2`, `DIAGNOSIS_CODE_1`, `DIAGNOSIS_CODE_2`, `PROCEDURE_CODE_1`, `PROCEDURE_CODE_2`, `DISCHARGE_DESTINATION`, `SOURCE_OF_ADMISSION`, `TYPE_OF_ADMISSION`, `ADMITTING_DIAGNOSIS_CODE`

FROM `patient_info`

Diagnosis:

insert into `diagnosis`
(`ADMISSION_ID`,`POA_DIAGNOSIS_INDICATOR`,`DIAGNOSIS_CODE`,`DIAGNOSIS_NUMBER`)

select `ADMISSION_ID`,`POA_DIAGNOSIS_INDICATOR_1`,`DIAGNOSIS_CODE_1`,1 from patient_information

where POA_DIAGNOSIS_INDICATOR_1 is not null and DIAGNOSIS_CODE_1 is not null

insert into `diagnosis`
(`ADMISSION_ID`,`POA_DIAGNOSIS_INDICATOR`,`DIAGNOSIS_CODE`,`DIAGNOSIS_NUMBER`)

```
select `ADMISSION_ID`,`POA_DIAGNOSIS_INDICATOR_2`,`DIAGNOSIS_CODE_2`,2 from
patient_information

where POA_DIAGNOSIS_INDICATOR_2 is not null and DIAGNOSIS_CODE_2 is not null
```

Procedure:

```
insert into `procedure` (`ADMISSION_ID`,`PROCEDURE_CODE`,`PROCEDURE_NUMBER`)

select `ADMISSION_ID`,`PROCEDURE_CODE_1`,1 from patient_information

where PROCEDURE_CODE_1 is not null


insert into `procedure` (`ADMISSION_ID`,`PROCEDURE_CODE`,`PROCEDURE_NUMBER`)

select `ADMISSION_ID`,`PROCEDURE_CODE_2`,2 from patient_information

where PROCEDURE_CODE_2 is not null
```

3. Create the following queries/series of queries. You are querying the NORMALIZED schema

a. An appropriate query which returns a result which is identical to the given csv file. This way you are demonstrating how one can extract data from an Electronic Medical Record database, to use for data analysis.

Solution:

Below query returns the result identical to the given csv file. Left join returns all records from patient_information table and returns null values for Diagnosis and procedure table when there is no match.

Query:

```
SELECT
pat.age,pat.sex,pat.`RACE`,pat.`DAY_OF_ADMISSION`,pat.`DISCHARGE_STATUS`,pat.`STAY_INDICATOR
`,pat.`DRG_CODE`,pat.`LENGTH_OF_STAY`,pat.`DRG_PRICE`,pat.`TOTAL_CHARGES`,pat.`COVERED_CHA
RGES`,diag.`POA_DIAGNOSIS_INDICATOR`,diag.`DIAGNOSIS_CODE`,pro.procedure_code,pat.discharge
_destination,pat.source_of_admission,pat.type_of_admission,pat.admitting_diagnosis_code

from patient_information pat

left join diagnosis diag

on diag.admission_id=pat.`ADMISSION_ID`

left join `procedure` pro

on pro.`ADMISSION_ID`=diag.`ADMISSION_ID`

group by pat.`admission_id`
```

b. Queries which return the Coverage Ratio (Coverage Ratio = COVERED_CHARGES/TOTAL_CHARGES) of patients who stayed in the hospital for a period of time longer than 5 days. How does this compare to the Coverage Ratio of patients with a Long Stay?

Solution:

Query:

```
select 5 as 'Length of stay' ,sum(covered_charges)/sum(total_charges) coverage_ratio
from `patient_information`
where `LENGTH_OF_STAY`>5
```

Output:

Length of stay	coverage_ratio
5	0.9611

Query:

```
select 'Long Stay' as 'Length of stay', sum(covered_charges)/sum(total_charges) coverage_ratio
from `patient_information`
where `STAY_INDICATOR`='L'
```

Output:

Length of stay	coverage_ratio
Long Stay	0.9908

We get the above results when we divide sum of covered charges with sum of total charges for patients with a long stay and patients who stay for more than 5 days. The coverage ratio (0.96) of patients who stayed for a time longer than 5 days is less compared to the coverage ratio (0.99) of patients with a long stay.

c. Is there any variation in the average Length of Stay of patients admitted to the hospital in different days of the week (DAY_OF_ADMISSION)? Showcase this with an appropriate SQL query and design an appropriate graph comparing the average Length of Stay between Friday admissions (DAY_OF_ADMISSION=6) and Monday admissions (DAY_OF_ADMISSION=2).

Solution:

Query:

```
SELECT `DAY_OF_ADMISSION`, AVG(`LENGTH_OF_STAY`) FROM `patient_info`
GROUP BY `DAY_OF_ADMISSION`
```

Output:

DAY_OF_ADMISSION	AVG(`LENGTH_OF_STAY`)
1	4.7013
2	4.7376
3	5.5572
4	5.0536
5	6.5855
6	5.5602
7	6.7500

Above results show that the average length of stay increases with the day of admission of patients. If the patients are admitted on Monday (4.74), their length of stay is expected to be lesser than the patients admitted on Friday (5.56). We can verify this result from the graph given below.

Query:

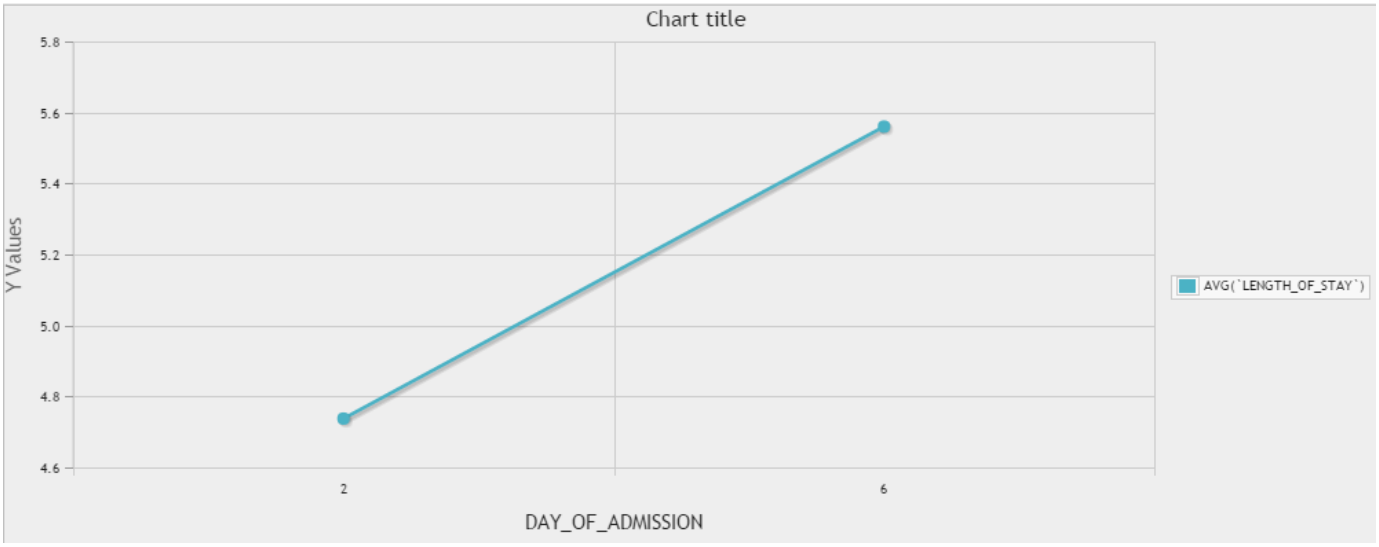
```
SELECT `DAY_OF_ADMISSION`, AVG(`LENGTH_OF_STAY`) FROM `patient_info`

GROUP BY `DAY_OF_ADMISSION`

HAVING `DAY_OF_ADMISSION` = '2' or `DAY_OF_ADMISSION` = '6'
```

Output:

DAY_OF_ADMISSION	AVG(`LENGTH_OF_STAY`)
2	4.7376
6	5.5602



d. Count all the distinct DRG Price values for each DRG code. For example for DRG code 1234 there might be the following DRG prices in dollars: $23,100 (12 occurrences), $19,000 (15 occurrences), and $15,320 (20 occurrences).

Solution:

Below query shows the count of all distinct DRG prices which appear for every DRG code.

Query:

```
select distinct drg_code `DRG CODE`, drg_price `DRG PRICE`, count(DRG_PRICE) `OCCURANCES`
from patient_information
group by drg_code,drg_price
```

Output:

DRG CODE	DRG PRICE	OCCURANCES
1	172453	1
1	224066	1
100	4929	1
100	7062	1
100	9583	1
100	10800	1
100	13678	1
101	17	1
101	172	1
101	4424	1
101	4466	1
101	4575	1
101	4625	1
101	4639	1
101	5010	1
11	45823	1
114	5977	1
115	6565	1
137	0	1
146	14891	1
149	3401	1
149	3841	1

Task 3

Using the Weka implementation of k-means, please find out:

(a) Is clustering a supervised or an unsupervised data mining technique? Please explain your answer focusing on what differentiates unsupervised learning from the supervised techniques.

Solution:

Supervised learning is the method where the relation between input-output attributes (from training data) is used to develop a model. This model can be further used to test the provided data (testing data). In supervised learning, we classify objects into predefined groups.

For example, if users are given a basket of fruits (apple, strawberry, kiwi, banana). In supervised learning, users already know everything about fruits. So, they classify based upon the previously defined knowledge.

Unsupervised learning method differentiates data without any previous knowledge of hidden characteristics. Clustering is an unsupervised data mining technique because, it is a procedure of grouping objects together in such a way that the objects in one group are more similar to each other than the objects of another group. While distributing the objects, there are no predefined groups of there is no previous knowledge about the hidden data.

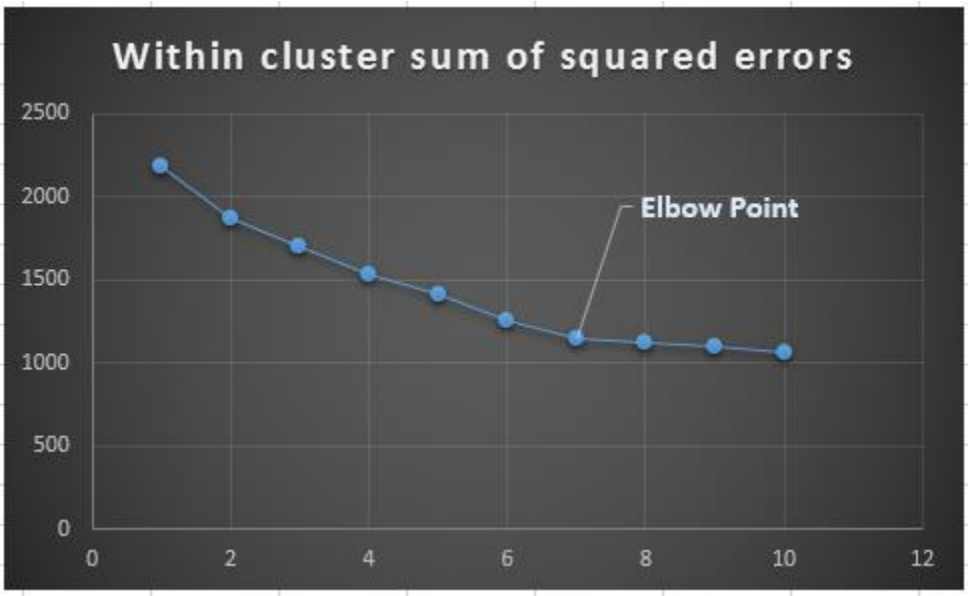
For example, if users are given a basket of fruits (apple, strawberry, kiwi, banana). In unsupervised learning, users will not have any previous knowledge. So, they will cluster fruits based on different characteristics first (based on size or color etc.). This clustering will group same types of fruits together. Clustering follows this approach and hence, it is an unsupervised data mining technique.

(b) The appropriate number of clusters which are required to properly cluster the following admission attributes: source of admission, type of admission, age group. Use the elbow method to define the number, by evaluating the 'within cluster sum of squared errors' you get as a result in your Weka output. Draw an appropriate graph to explain your answer.

Solution:

Number of clusters to cluster the admission attributes such as source of admission, type of admission, age group are calculated using Weka as below:

No of Cluster	Within cluster sum of squared errors
1	2177
2	1873
3	1697
4	1532
5	1411
6	1257
7	1143
8	1124
9	1097
10	1061



In the above graph X-axis denotes the number of cluster whereas, Y-axis denotes 'Within cluster sum of squared errors'.

Elbow point- Cluster 7

(c) Based on the number of clusters you specified in the above step, please calculate those clusters and explain how one may interpret the result.

Solution:

No of Cluster	Within cluster sum of squared errors
1	2177
2	1873
3	1697
4	1532
5	1411
6	1257
7	1143
8	1124
9	1097
10	1061

We have calculated 10 clusters to group admission attributes such as source of admission, type of admission, age group. We have plotted these values along with 'Within cluster sum of squared errors'. The main goal is to identify optimal number of clusters to represent these values. We have to calculate that how much variance/difference we get if we go from one cluster to another. From the graph, we can identify that the change of difference is minimized once we cross cluster 7. From this point, value of 'Within cluster sum of squared errors' is not changing much. That is why we defined 7 as an elbow point.

Task 4

(a) Use any appropriate method to modify the class attribute values to be only of two values, either zero (DRG price less than $80,000) or one (DRG price more than $80,000) so that the problem will be binary classification. Integrate the new attribute (DRG_PRICE_BINARY) into your dataset.

Solution:

Query:

Table Task_4: Create table task_4 and add a column for DRG_PRICE_BINARY

```
create table task_4 like patient_info


insert into task_4 select * from `table 1`
ALTER TABLE task_4
ADD DRG_PRICE_BINARY int(1)
```

Set values of drg_price_binary:

```
update task_4
set drg_price_binary=0
where drg_price<80000

update task_4
set drg_price_binary=1
where drg_price>80000
```

(b) Observe the available features and explain how the data are acquired in a temporal manner during the healthcare procedure in the real hospital. Specifically, define what do clinicians/administrators already know:

1. At the time when the patient enters the hospital

Solution:

At the time when the patient is admitted to the hospital, initial data is collected by the administrator staff from the information provided by the patient. It covers following attributes-

AGE, SEX, RACE, DAY_OF_ADMISSION, SOURCE_OF_ADMISSION, TYPE_OF_ADMISSION, ADMITTING_DIAGNOSIS_CODE

2. At the time when the patient is discharged from the hospital

Solution:

At the time when the patient is discharged from the hospital, data is collected from the diagnosis information provided by the clinicians/doctors. It covers following attributes-

LENGTH_OF_STAY, DISCHARGE_STATUS, STAY_INDICATOR, POA_DIAGNOSIS_INDICATOR_1, POA_DIAGNOSIS_INDICATOR_2, DIAGNOSIS_CODE_1, DIAGNOSIS_CODE_2, PROCEDURE_CODE_1, PROCEDURE_CODE_2, DISCHARGE_DESTINATION

Therefore, at the time of discharge administrators/clinicians know all the above data along with the information provided at the time of admission.

Some of the attributes can be calculated **only after** the patient is discharged which consists of the following attributes:

DRG_CODE, DRG_PRICE, TOTAL_CHARGES, COVERED_CHARGES

Scenario 1: we only know the admission information about the patient. In other words, we only know the Features with ID 1,2,3,4,19,20,21. Only keep those features in Weka before you proceed.

Scenario 2: we know what was known in scenario 1 plus all the information the clinicians and administrators acquired during the hospital care. In other words, we know the Features with ID 1,2,3,4,5,6,8,12,13,14,15,16,17,18,19,20,21. Only keep those features in Weka before you proceed.

(c) Answer, for the two above scenarios, the following questions:

1. You are going to notice that for both scenarios we manually excluded the features 7, 10 and 11. Please explain why we excluded each one of these features.

Solution:

Excluded features consists of DRG_CODE, DRG_PRICE, TOTAL_CHARGES, COVERED_CHARGES. We are using data mining to predict if the DRG code price is going to be higher or lower. DRG price is calculated based on DRG code and DRG code is determined once the patient has been discharged from the hospital. Therefore, we can verify that the TOTAL_CHARGES and COVERED_CHARGES are based on DRG_PRICE which in turn depends upon the DRG code. In the above 2 scenarios, we only know the information collected at the time of admission and during the hospital stay. We are unaware of the features which are identified after the discharge that is why we excluded features 7,9,10 and 11 from both the scenarios.

2. Undergo the feature selection process CfsSubsetEval to select the appropriate features for each of the two scenarios. This way, you will be able to know which features will be included in your classification later on.

Scenario 1:

Using Weka, we went through the feature selection process CfsSubsetEval to select the appropriate features for the first scenario (1,2,3,4,19,20,21)

=== Run information ===

Instances: 1477

Attributes: 8

AGE

SEX

RACE

DAY_OF_ADMISSION

SOURCE_OF_ADMISSION

TYPE_OF_ADMISSION

ADMITTING_DIAGNOSIS_CODE

DRG_PRICE_BINARY

Evaluation mode: evaluate on all training data

=== Attribute Selection on all input data ===

Search Method:

Best first.

Start set: no attributes

Search direction: forward

Stale search after 5 node expansions

Total number of subsets evaluated: 30

Merit of best subset found: 0.008

Attribute Subset Evaluator (supervised, Class (nominal): 8 DRG_PRICE_BINARY):

CFS Subset Evaluator

Including locally predictive attributes

Selected attributes: 1,2,3,4,5,6,7 : 7

AGE

SEX

RACE

DAY_OF_ADMISSION

SOURCE_OF_ADMISSION

TYPE_OF_ADMISSION

ADMITTING_DIAGNOSIS_CODE

Scenario 2:

Using Weka, we went through the feature selection process CfsSubsetEval to select the appropriate features for the first scenario (1,2,3,4,5,6,8,12,13,14,15,16,17,18,19,20,21)

=== Run information ===

Instances: 1477

Attributes: 18

AGE

SEX

RACE

DAY_OF_ADMISSION

DISCHARGE_STATUS

STAY_INDICATOR

LENGTH_OF_STAY

POA_DIAGNOSIS_INDICATOR_1

POA_DIAGNOSIS_INDICATOR_2

DIAGNOSIS_CODE_1

DIAGNOSIS_CODE_2

PROCEDURE_CODE_1

PROCEDURE_CODE_2

DISCHARGE_DESTINATION

SOURCE_OF_ADMISSION

TYPE_OF_ADMISSION

ADMITTING_DIAGNOSIS_CODE

DRG_PRICE_BINARY

Evaluation mode: evaluate on all training data

=== Attribute Selection on all input data ===

Search Method:

Best first.

Start set: no attributes

Search direction: forward

Stale search after 5 node expansions

Total number of subsets evaluated: 91

Merit of best subset found: 0.101

Attribute Subset Evaluator (supervised, Class (nominal): 18 DRG_PRICE_BINARY):

CFS Subset Evaluator

Including locally predictive attributes

Selected attributes: 1,5,7 : 3

AGE

DISCHARGE_STATUS

LENGTH_OF_STAY

3. Use the classifiers (i) Naïve Bayes and (ii) Logistic Regression to classify the DRG_PRICE_BINARY, for each scenario, by using the features you found to be useful during feature selection.

Scenario 1:

i> Naive Bayes:

Using Weka, we went through the feature selection process CfsSubsetEval to select the appropriate features for the first scenario (1,2,3,4,19,20,21). After that, we used Naive Bayes classifier to classify DRG_PRICE_BINARY:

=== Confusion Matrix ===

a	b	<-- classified as
1470	0	a = 0
7	0	b = 1

The confusion matrix shows that the Naïve Bayes accurately predict values of DRG_PRICE_BINARY less than 80000. But, some values are predicted as greater than 80000 incorrectly.

ii> Logistic Regression:

Using Weka, we went through the feature selection process CfsSubsetEval to select the appropriate features for each of the first scenario (1,2,3,4,19,20,21). After that, we used Logistic Regression classifier to classify DRG_PRICE_BINARY:

=== Confusion Matrix ===

a	b	<-- classified as
1369	101	a = 0
4	3	b = 1

The confusion matrix shows that the Logistic Regression predicts most of the values of DRG_PRICE_BINARY less than 80000 correctly. But, it wrongly predicts major number of DRG_PRICE_BINARY as greater than 80000 even if they are less than 80000.

Scenario 2:

i> Naive Bayes:

Using Weka, we went through the feature selection process CfsSubsetEval to select the appropriate features for the first scenario (1,2,3,4,5,6,8,12,13,14,15,16,17,18,19,20,21). After that, we used Naive Bayes classifier to classify DRG_PRICE_BINARY:

=== Confusion Matrix ===

a	b	<-- classified as
1446	24	a = 0
6	1	b = 1

Accuracy of Naïve Bayes in second scenario is less than that of the first scenario even if it classifies most of the DRG_PRICE_BINARY less than 80000 correctly. Because, it wrongly classifies some DRG_PRICE_BINARY which are less as well as greater than 80000.

Ii> Logistic Regression

Using Weka, we went through the feature selection process CfsSubsetEval to select the appropriate features for the first scenario (1,2,3,4,5,6,8,12,13,14,15,16,17,18,19,20,21). After that, we used Logistic Regression classifier to classify DRG_PRICE_BINARY:

=== Confusion Matrix ===

a	b	<-- classified as
1449	21	a = 0
7	0	b = 1

In the confusion matrix of Logistic regression, DRG_PRICE_BINARY <80000 are classified correctly but, class b wrongly predicts some DRG_PRICE_BINARY >80000.

4. Answer the following questions regarding the accuracy of the classification for Naïve Bayes in each scenario:

Scenario 1:

Overall Accuracy=99.5261

Precision for DRG_PRICE_BINARY (0) = 0.995

Precision for DRG_PRICE_BINARY (1) = 0.000

Recall for DRG_PRICE_BINARY (0) = 1.000

Recall for DRG_PRICE_BINARY (1) =0.000

f-measure for DRG_PRICE_BINARY (0)=0.998

f-measure for DRG_PRICE_BINARY (1)=0.000

Naïve Bayes classification for scenario 1 is highly accurate. Precision for classifying DRG_PRICE_BINARY (0) tells that 99.5% of the predicted values are actually less than 80000 whereas, recall for classifying DRG_PRICE_BINARY (0) tells that 100% of actual values less than 80000 are classified correctly.

Precision for classifying DRG_PRICE_BINARY (1) tells that no values are predicted as greater than 80000 whereas, recall for classifying DRG_PRICE_BINARY (1) tells that not a single DRG_PRICE_BINARY is greater than 80000 and it is classified correctly.

f-measure determines that a little tradeoff is required to manage precision and recall for classifying DRG_PRICE_BINARY(0).

Scenario 2:

Overall Accuracy=97.96

Precision for DRG_PRICE_BINARY (0) =0.996

Precision for DRG_PRICE_BINARY (1) =0.04

Recall for DRG_PRICE_BINARY (0) =0.984

Recall for DRG_PRICE_BINARY (1) =0.143

f-measure for DRG_PRICE_BINARY (0) =0.990

f-measure for DRG_PRICE_BINARY (1) =0.063

Naïve Bayes classification for scenario 2 is less accurate. Precision for classifying DRG_PRICE_BINARY (0) tells that 99.6% of the predicted values are actually less than 80000 whereas, recall for classifying DRG_PRICE_BINARY (0) tells that only 98.4% of actual values less than 80000 are classified correctly.

Precision for classifying DRG_PRICE_BINARY (1) tells that only 4% of the predicted values are actually greater than 80000 whereas, recall for classifying DRG_PRICE_BINARY (1) tells that 14.3% of values greater than 80000 are actually classified correctly.

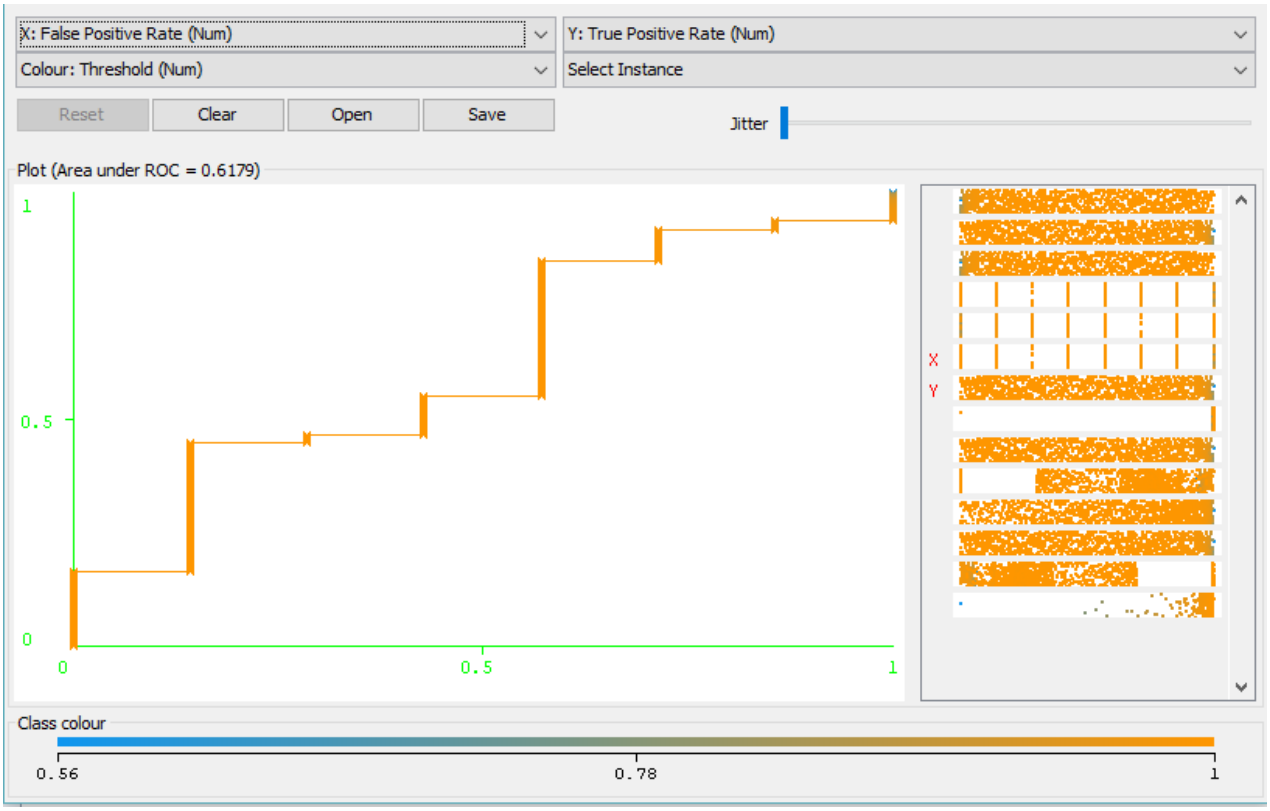
f-measure determines that the performance of the classifier needs to be improved. Precision and recall values for DRG_PRICE_BINARY<80000 are closer to each other but, tradeoff is required to improve performance to classify DRG_PRICE_BINARY>80000 correctly.

5. Design the ROC curve and calculate the ROC area of Naïve Bayes for both scenarios

Solution:

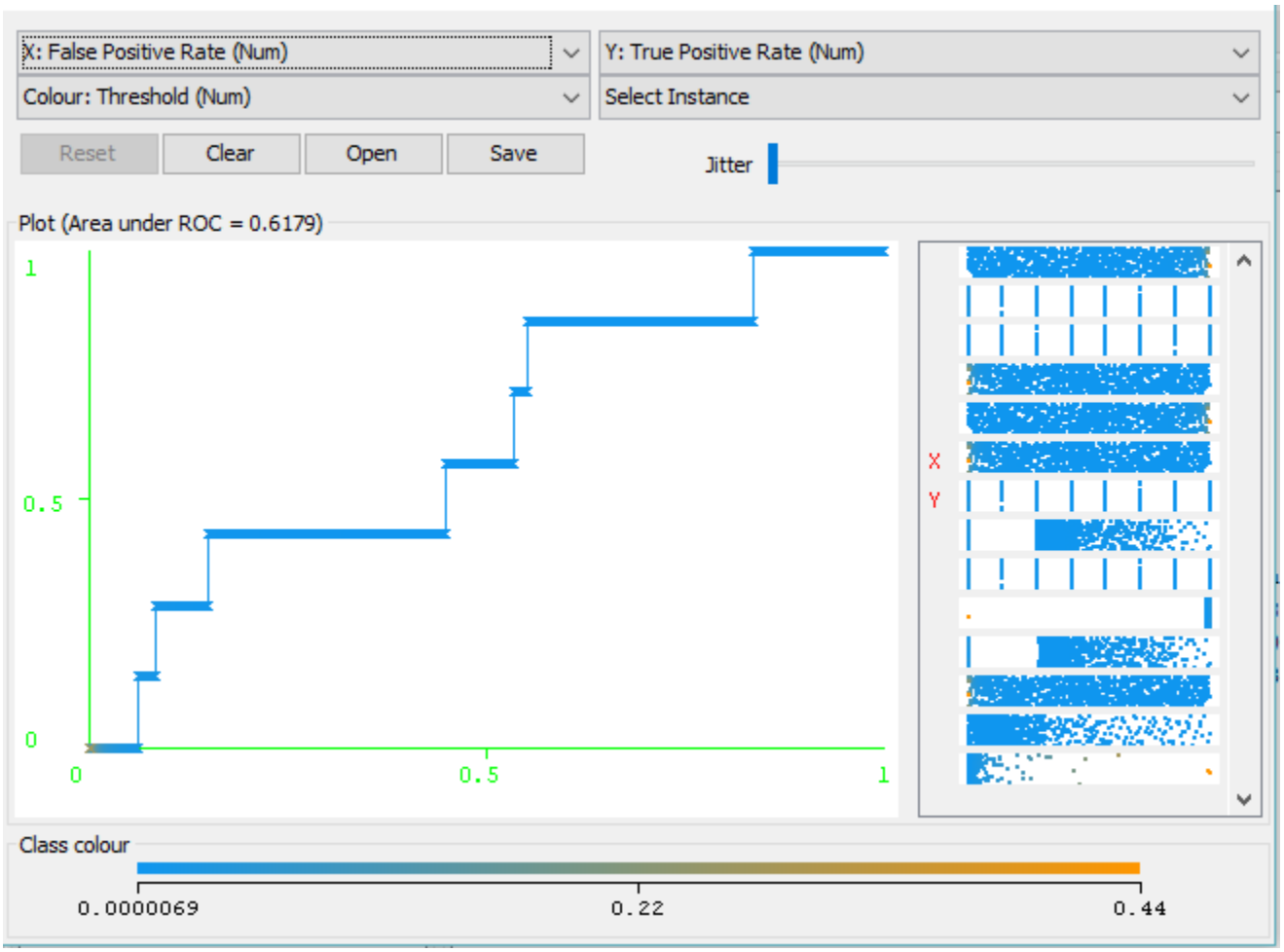
ROC curve tells if your classifier is appropriate or not. Area of ROC curve is calculated using Weka after DRG_PRICE_BINARY is classified using Naïve Bayes. Area is calculated with Visualize threshold curve for DRG_PRICE_BINARY>80000 as well as DRG_PRICE_BINARY <80000

Scenario 1:



ROC area for DRG_PRICE_BINARY (0) =0.6179

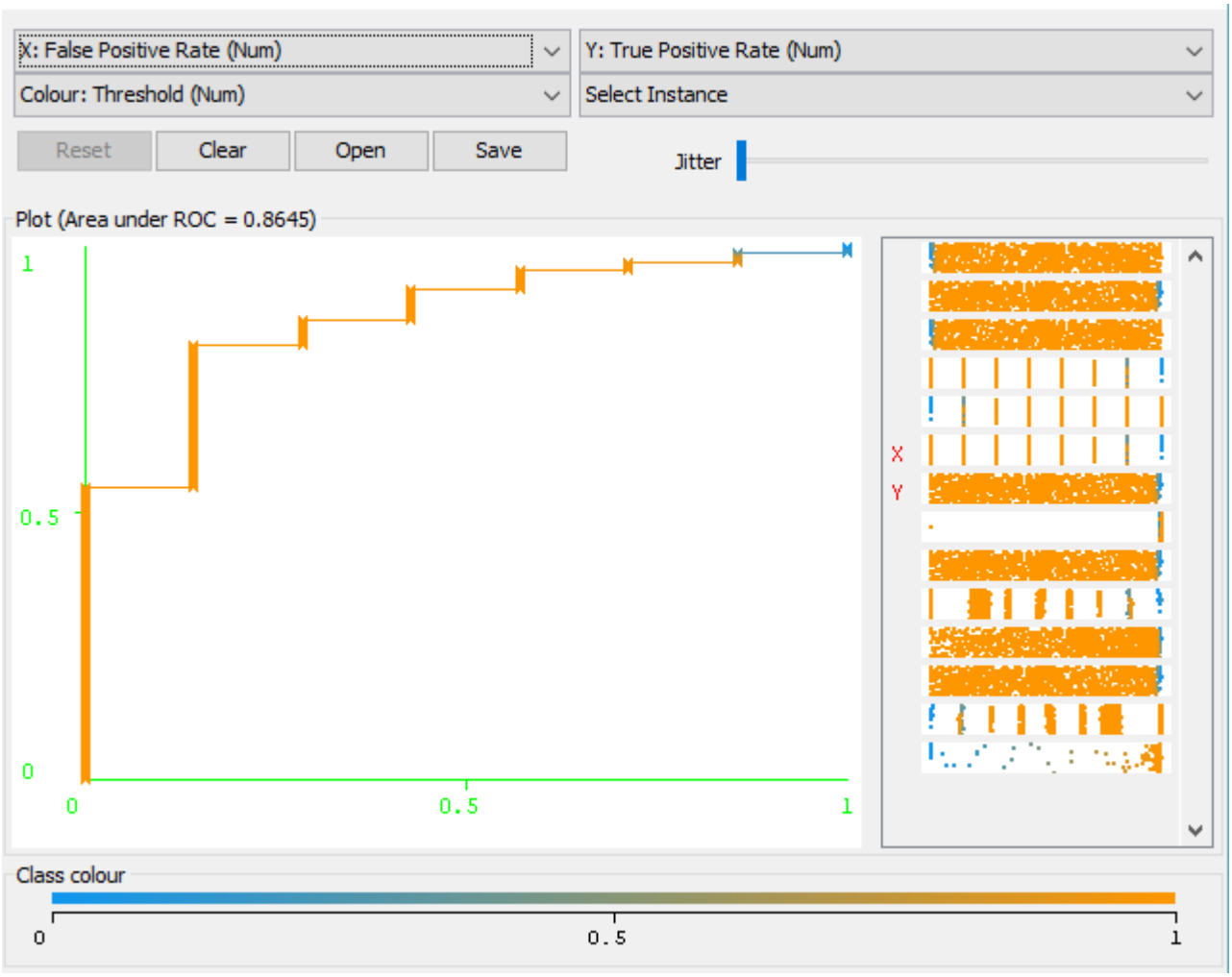
ROC area for scenario 1 tells that the classifier is not appropriate enough.



ROC area for DRG_PRICE_BINARY (1) =0.6179

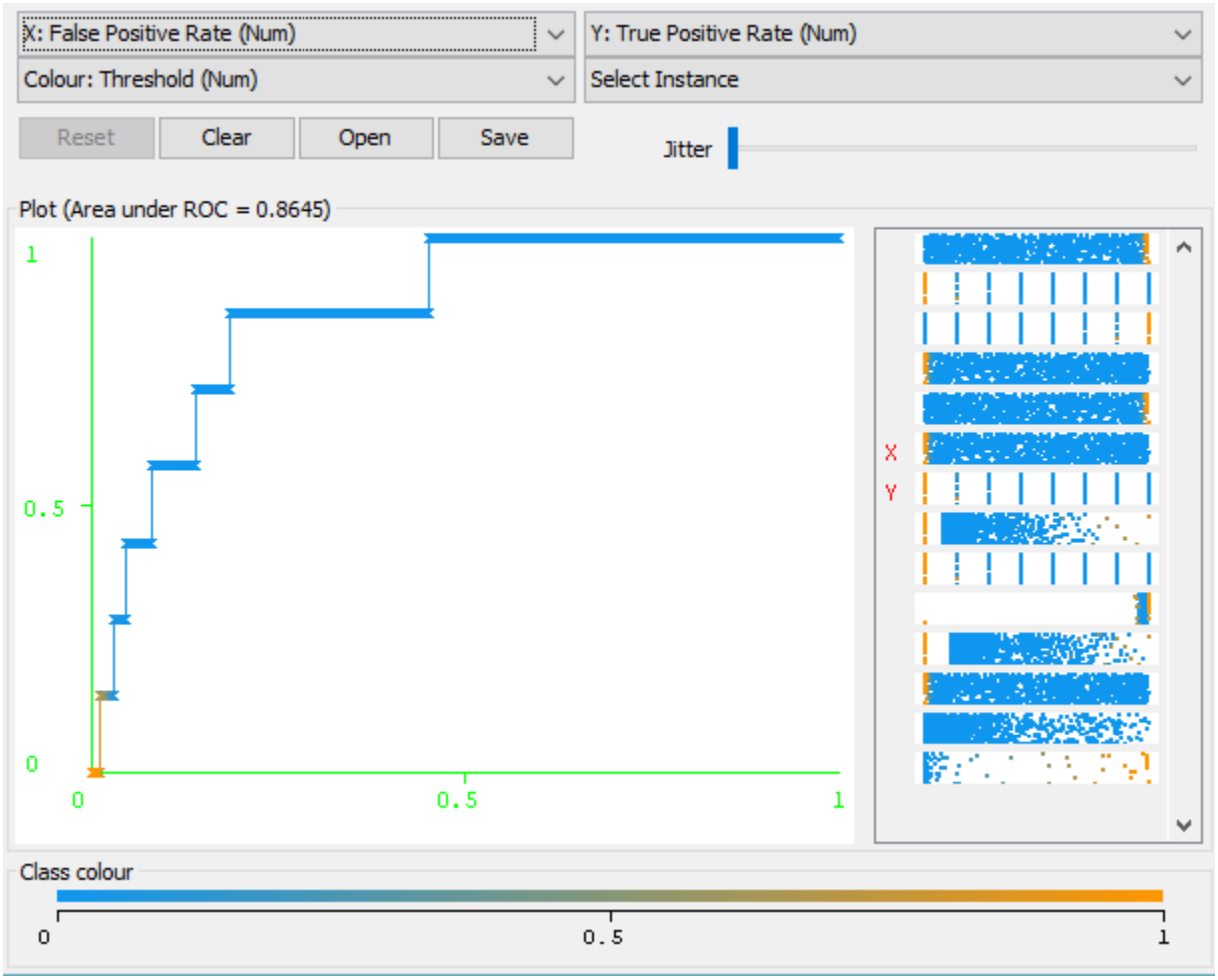
ROC area for scenario 1 tells that the classifier is not appropriate enough.

Scenario 2:



ROC area for DRG_PRICE_BINARY (0) =0.8645

ROC area for scenario 1 tells that it is reasonably appropriate classifier as the curve is more inclined towards Y-axis which represents true positive rate.



ROC area for DRG_PRICE_BINARY (1) =0.8645

ROC area for scenario 1 tells that it is reasonably appropriate classifier as the curve is more inclined towards Y-axis which represents true positive rate.