CAB430 ASSESSMENT 2

JiYan Zhu, Shu Du

QUT SEM1 29/05/2021

JiYan Zhu: n10415483, Shu Du: n10505024.

Contents

Task 1

We have 3 structures, first one is for prediction 1, second one for prediction 2, and last one is for prediction 3. Each structure contains different columns.

Prediction 1 structure

For prediction 1, the question is asking to predict customer's demographic attributes based on customers' demographic information and previously rented cars. This means the structure should contains order id, the customer's age, gender, occupation, and car's model.

Below is the screen shot of the first structure.

```
CREATE MINING STRUCTURE [Car_Rentalsv1]
(
    [Order_ID] LONG KEY,
    [Customer_Age] LONG DISCRETIZED(AUTOMATIC, 10),
    [Customer_Gender] TEXT DISCRETE,
    [Customer_Occupation] TEXT DISCRETE,
    [Car_Model] TEXT DISCRETE
)
WITH HOLDOUT (30 PERCENT or 1000 CASES)
```

Since the prediction is on customer's information base on the his orders, we should use order id to find all the other information and use it to prediction one for them, for example, using gender, age, cat's model to predict the customers' occupation. Below is the script for the screen shot.

```
CREATE MINING STRUCTURE [Car_Rentalsv1]
(
	[Order_ID] LONG KEY,
	[Customer_Age] LONG DISCRETIZED(AUTOMATIC, 10),
	[Customer_Gender] TEXT DISCRETE,
	[Customer_Occupation] TEXT DISCRETE,
	[Car_Model] TEXT DISCRETE
)
WITH HOLDOUT (30 PERCENT or 1000 CASES)
```

Prediction 2 structure

For prediction 2, the question is asking to use a batch query to predict the top-3 cars for each customer based on the customer's demographic information and previously rented cars. This means the structure should contains customer id, the customer's age, gender, occupation, a table for the top 3 car's model.

Below is the screen shot for the second structure.

```
CREATE MINING STRUCTURE [Car_Rentalsv2]
(
    [Customer_ID] LONG KEY,
    [Customer_Age] LONG DISCRETIZED(AUTOMATIC, 10),
    [Customer_Gender] TEXT DISCRETE,
    [Customer_Occupation] TEXT DISCRETE,
    [Products] TABLE (
        [Model] TEXT KEY
    )
)
```

Since the prediction is on the top-3 cars that each customer would most likely to rent based on the previous rental. We will use customer ID to prediction each customer. The Products table is created to fit the top 3 car's model that the customer would most likely to rent. Below is the script for the screen shot.

```
CREATE MINING STRUCTURE [Car_Rentalsv2]
(
	[Customer_ID] LONG KEY,
	[Customer_Age] LONG DISCRETIZED(AUTOMATIC, 10),
	[Customer_Gender] TEXT DISCRETE,
	[Customer_Occupation] TEXT DISCRETE,
	[Products] TABLE (
		[Model] TEXT KEY
	)
)
```

Prediction 3 structure

The last prediction that the question is asking to do is very similar to prediction 2, the only difference is require stores, demographic information and previously rented cars. This means the structure should contains all the columns that is added from prediction 2's structure, with a new table containing stores, because a customer can rent from different stores.

Below is the screen shot for prediction 3 structure.

```
CREATE MINING STRUCTURE [Car_Rentalsv3]
(
    [Customer_ID] LONG KEY,
    [Customer_Age] LONG DISCRETIZED(AUTOMATIC, 10),
    [Customer_Gender] TEXT DISCRETE,
    [Customer_Occupation] TEXT DISCRETE,
    [Stores] TABLE(
        [StoreName] TEXT KEY
    ),
    [Products] TABLE (
        [Model] TEXT KEY
    )
)
```

There is not much different between the second prediction's structure and the last one, the only difference between them is the last structure added a new table call stores, which contain the store name. Below is the script for the last structure.

```
CREATE MINING STRUCTURE [Car_Rentalsv3]
(
	[Customer_ID] LONG KEY,
	[Customer_Age] LONG DISCRETIZED(AUTOMATIC, 10),
	[Customer_Gender] TEXT DISCRETE,
	[Customer_Occupation] TEXT DISCRETE,
	[Stores] TABLE(
		[StoreName] TEXT KEY
	),
	[Products] TABLE (
		[Model] TEXT KEY
	)
)
```

Task 2

After creating all 3 structures, the next task is to create model that can predict the column that we were been asked to do.

Prediction 1 model

The first prediction is asking to prediction the customer's demographic attributes based on customer's demographic information and previously rented cars. We decided to predict the customer's occupation, this means that we should use PREDICT_ONLY for Customer Occupation, because we will not be using Occupation. Furthermore, we will be using all other columns to predict the customer's occupation, this includes customer's age, gender, and the car that the customer rented before.

Below is the screenshot of the model.

```
ALTER MINING STRUCTURE [Car_Rentalsv1]
ADD MINING MODEL [pred1]
(

    [Order_ID],
    [Customer_Age],
    [Customer_Gender],
    [Customer_Occupation] PREDICT_ONLY,
    [Car_Model]
)USING Microsoft_Association_Rules
WITH DRILLTHROUGH
GO
INSERT INTO [pred1]
```

Below is the script of the screen shot.

```
ALTER MINING STRUCTURE [Car_Rentalsv1]
ADD MINING MODEL [pred1]
(

        [Order_ID],
        [Customer_Age],
        [Customer_Gender],
        [Customer_Occupation] PREDICT_ONLY,
        [Car_Model]
)USING Microsoft_Association_Rules
WITH DRILLTHROUGH
GO
INSERT INTO [pred1]
```

Prediction 2 model

Prediction 2 is asking to prediction the top-3 cars that the customer will most likely to rent based on the customer's demographic information and previously rented cars. This we should use PREDICT to predict the top three cars. The reason why we will not be using PREDICT_ONLY is because we want to use the car column since we will prediction using the customer's rented car. Moreover, we should also add the customer's information, age, gender, and occupation.

Below is the screen shot of the model.

```
ALTER MINING STRUCTURE [Car_Rentalsv2]
ADD MINING MODEL [pred2]
(
 [Customer_ID],
[Customer_Age],
[Customer_Gender],
[Customer_Occupation],
   [Products] PREDICT(
        [Model]
    )
)
Using Microsoft_Association_Rules
WITH DRILLTHROUGH
GO
INSERT INTO [Car_Rentalsv2]
```

Below is the script of the screenshot

```
ALTER MINING STRUCTURE [Car_Rentalsv2]
ADD MINING MODEL [pred2]
(
 [Customer_ID],
[Customer_Age],
[Customer_Gender],
[Customer_Occupation],
   [Products] PREDICT(
        [Model]
    )
)
Using Microsoft_Association_Rules
WITH DRILLTHROUGH
GO
INSERT INTO [Car_Rentalsv2]
```

Prediction 3 model

Prediction 3 is predicting the same column as prediction 2, with additional column to be consider, which is store. The model is very similar to prediction's 2 model, using the same command PREDICT with the same reason.

Below is the screen shot of the model.

```
ALTER MINING STRUCTURE [Car_Rentalsv3]
ADD MINING MODEL [pred3]
(
 [Customer_ID],
[Customer_Age],
[Customer_Gender],
[Customer_Occupation],
[Stores](
[StoreName]
),
   [Products] PREDICT(
        [Model]
    )
)
Using Microsoft_Association_Rules
WITH DRILLTHROUGH
GO
INSERT INTO [Car_Rentalsv3]
```

Below is the script of the screen shot.

```
ALTER MINING STRUCTURE [Car_Rentalsv3]
ADD MINING MODEL [pred3]
(
 [Customer_ID],
[Customer_Age],
[Customer_Gender],
[Customer_Occupation],
[Stores](
[StoreName]
),
   [Products] PREDICT(
        [Model]
    )
)
Using Microsoft_Association_Rules
WITH DRILLTHROUGH
GO
INSERT INTO [Car_Rentalsv3]
```

JiYan Zhu: n10415483, Shu Du: n10505024.

Task 3

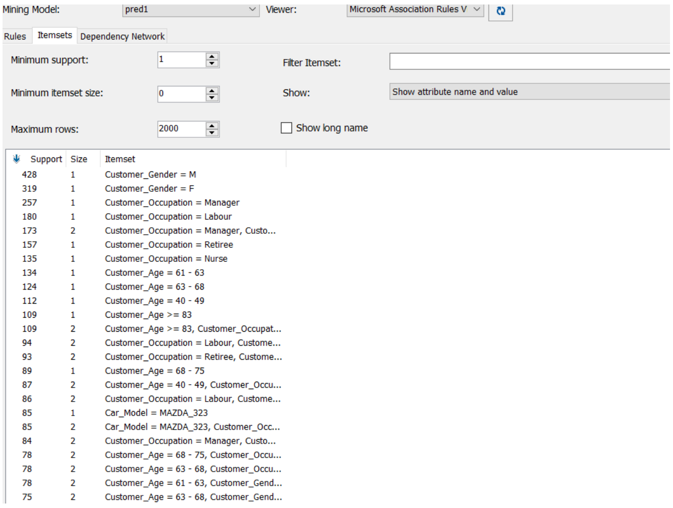
Once the structures and the structures' models have been created, the next task is to inserting source from the data base.

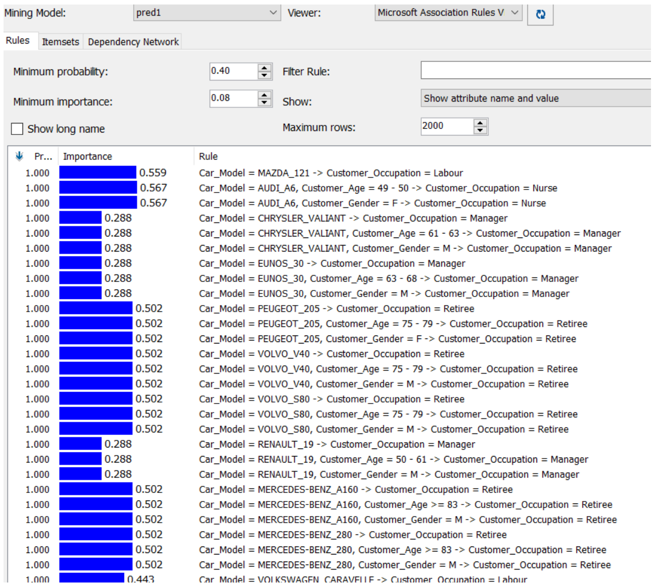
Prediction 1

The screenshot for inserting source from the data base to the first structure.

```
INSERT INTO MINING STRUCTURE [Car_Rentalsv1]
(
    [Order_ID],
    [Customer_Age],
    [Customer_Gender],
    [Customer_Occupation],
    [Car_Model]
)
OPENQUERY(CarRentals,
'SELECT [Order_ID], cust.Customer_Age, cust.Customer_Gender, cust.Customer_Occupation, car.Car_Model FROM dbo.Fact_Rentals AS x
INNER JOIN dbo.Customer AS cust ON cust.Customer_ID = x.Order_Customer
INNER JOIN dbo.Car as car ON car.Car_ID = x.Order_Car')
```

By inner join 3 tables, fact table, customer table and car table, it allows us to find the link them together and use the model to predict the customer's occupation based on those columns.





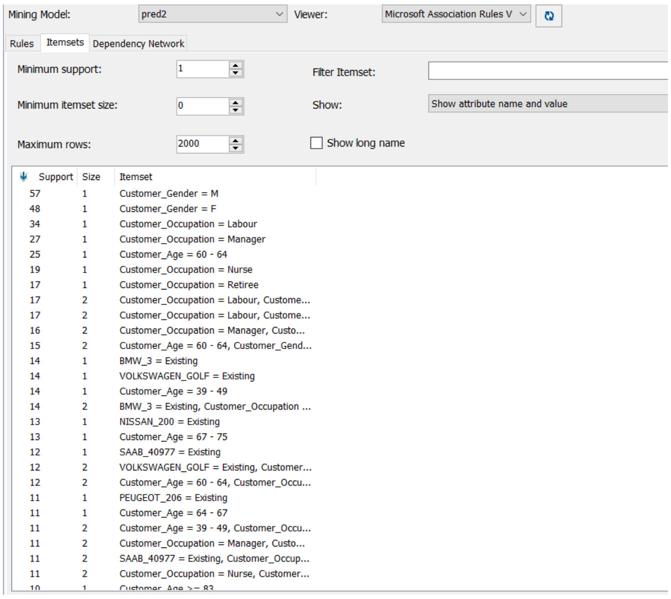
JiYan Zhu: n10415483, Shu Du: n10505024.

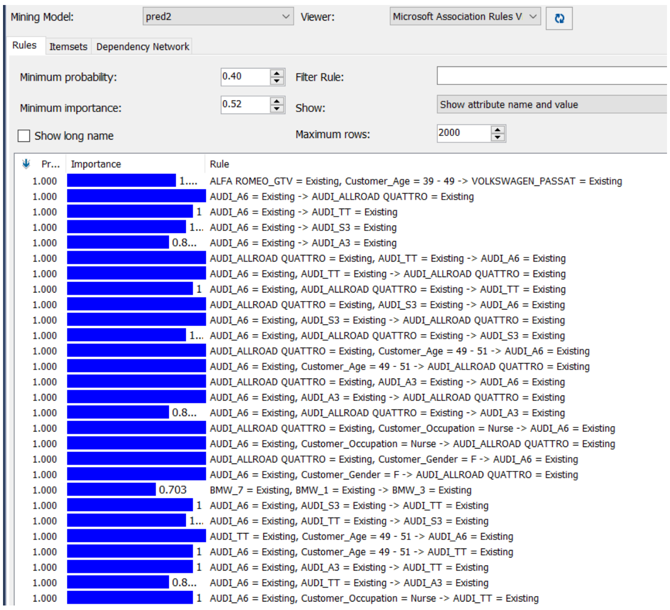
Prediction 2

The screenshot for inserting source from the data base to the second structure.

```
INSERT INTO MINING STRUCTURE [Car_Rentalsv2]
(
   [Customer_ID], [Customer_Age], [Customer_Gender], [Customer_Occupation],
   [Products] (SKIP, [Model])
)
SHAPE {
  OPENQUERY([CarRentals],'SELECT DISTINCT [Customer_ID],cust.Customer_Age, cust.Customer_Gender, cust.Customer_Occupation
FROM dbo.Fact_Rentals AS x
INNER JOIN dbo.Customer AS cust ON cust.Customer_ID = x.Order_Customer
INNER JOIN dbo.Car AS car ON car.Car_ID = x.Order_Car')
}
APPEND
(
    {OPENQUERY([CarRentals],'SELECT Customer_ID, car.Car_Model AS [Model]  FROM
dbo.Fact_Rentals as j
INNER JOIN dbo.Customer AS cust ON cust.Customer_ID = j.Order_Customer
INNER JOIN dbo.Car AS car ON car.Car_ID = j.Order_Car ORDER BY Customer_ID')
  }
RELATE Customer_ID to Customer_ID
) AS [Products]
```

First, inner join three tables, which are fact table, customer table, car table, and append another table joined by fact table, customer table and car table to order by customer ID as products table.





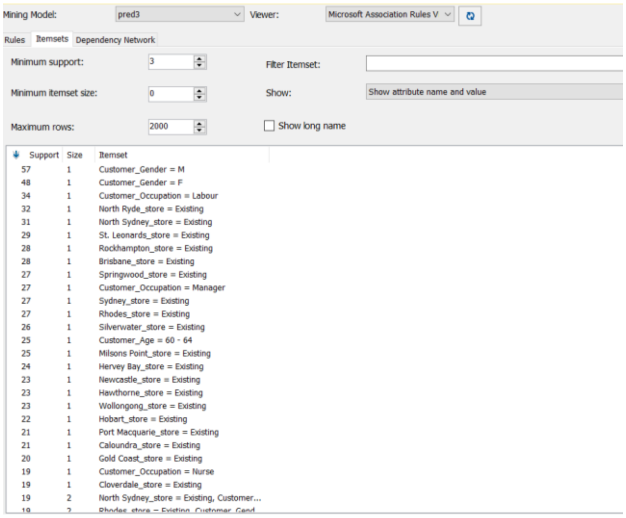
Prediction 3

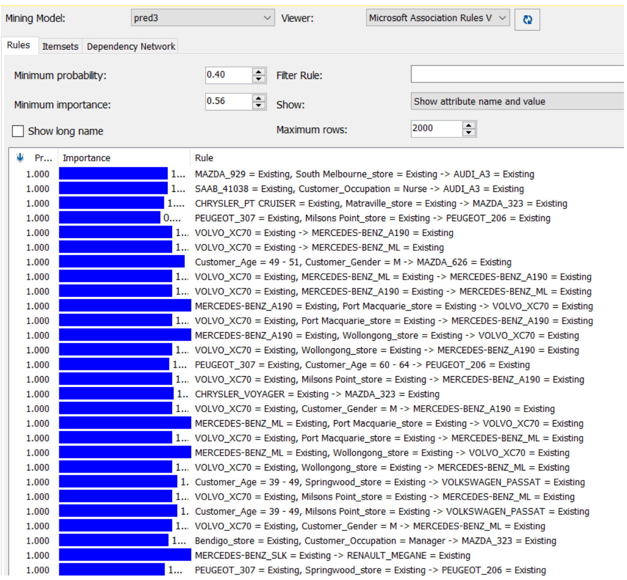
The screenshot for inserting source from the data base to the last structure.

```
INSERT INTO MINING STRUCTURE [Car_Rentalsv3]
(
   [Customer_ID], [Customer_Age], [Customer_Gender], [Customer_Occupation], [Stores] (SKIP,[StoreName]),
   [Products] (SKIP, [Model])
)
SHAPE {
  OPENQUERY([CarRentals],'SELECT DISTINCT [Customer_ID],cust.Customer_Age, cust.Customer_Gender, cust.Customer_Occupation
FROM dbo.Fact_Rentals AS x
INNER JOIN dbo.Customer AS cust ON cust.Customer_ID = x.Order_Customer
INNER JOIN dbo.Car AS car ON car.Car_ID = x.Order_Car
')
}
APPEND(
    {OPENQUERY([CarRentals],'SELECT Customer_ID, st.Store_Name AS StoreName FROM
dbo.Fact_Rentals as z
INNER JOIN dbo.Customer AS cust ON cust.Customer_ID = z.Order_Customer
INNER JOIN dbo.Store AS st ON st.Store_ID = z.Order_Store ORDER BY Customer_ID')
    }
RELATE Customer_ID to Customer_ID
)AS [Stores],

    ({OPENQUERY([CarRentals],'SELECT Customer_ID, car.Car_Model AS [Model]  FROM
dbo.Fact_Rentals as j
INNER JOIN dbo.Customer AS cust ON cust.Customer_ID = j.Order_Customer
INNER JOIN dbo.Car AS car ON car.Car_ID = j.Order_Car ORDER BY Customer_ID')}
RELATE Customer_ID to Customer_ID
) AS [Products]
```

First, inner join three tables, which are fact table, customer table and car table, then append the second table by inner join fact table, customer table, and store table order by customer id as store table. Lastly, append the last table by inner join fact table, customer table and car table order by customer id as products table.





Task 4

Task 4 require us to show the screen shots of the results for each prediction.

Prediction 1

Prediction

```
SELECT t.Customer_Age, t.Customer_Gender,  t.Car_Model,
    PREDICT([Customer_Occupation]) AS [Prediction on Car_Rentalsv1]
    FROM
    [pred1]
NATURAL PREDICTION JOIN
(SELECT 60 AS Customer_Age, 'M' AS Customer_Gender, 'BMW_1' AS Car_Model) as t
```

83 %

Messages | Results

Customer_Age	Customer_Gender	Car_Model	Prediction on Car_Rentalsv1
60	M	BMW_1	Labour

In this screen shot I given the information if a customer is 60 yeas old, male, and rented BMW_1, what is his occupation, and the predicted answer is Labour

Batch query to against the cases

```
SELECT t.Customer_Age, t.Customer_Gender, t.Car_Model,
  PREDICT([Customer_Occupation]) AS [Prediction on Car_Rentalsv1]
From
  [pred1]
NATURAL PREDICTION JOIN
(SELECT * FROM [pred1].CASES WHERE IsTestCase()
 ) AS t
```

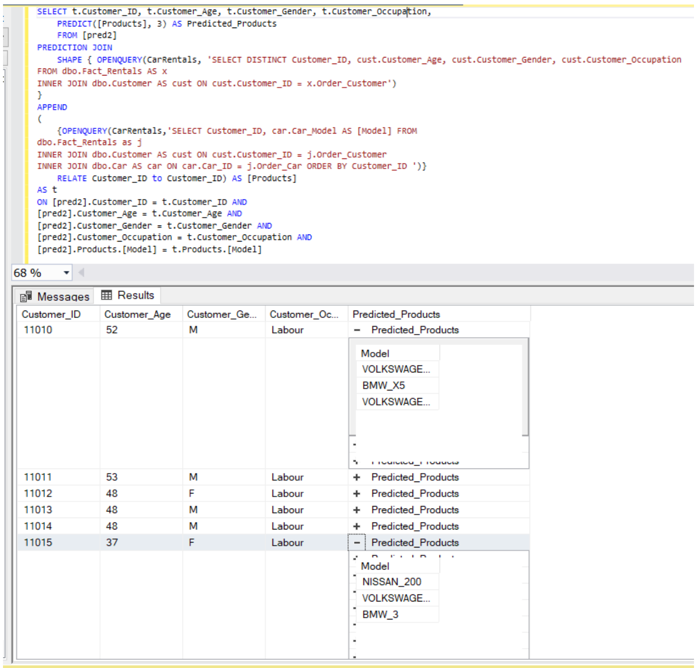
83 %

Messages | Results

Customer_Age	Customer_Ge...	Car_Model	Prediction on ...
38	M	BMW_3	Labour
38	M	BMW_5	Labour
44	F	BMW_3	Labour
66	F	BMW_3	Labour
66	F	BMW_3	Labour
38	F	BMW_3	Labour
72	M	BMW_3	Labour
38	F	BMW_3	Labour
44	M	BMW_5	Labour
66	F	BMW_3	Labour
66	F	BMW_3	Labour
66	F	BMW_X5	Labour
66	F	MAZDA_121	Labour
72	M	MAZDA_121	Labour
56	M	VOLKSWAGE...	Labour
56	M	VOLKSWAGE...	Labour
56	F	VOLKSWAGE...	Labour
56	M	VOLKSWAGE...	Labour
56	M	VOLKSWAGE...	Labour
56	M	VOLKSWAGE...	Labour
56	M	VOLKSWAGE	Labour

In this screen show I given a batch query to test against all the testing data sets.

Prediction 2

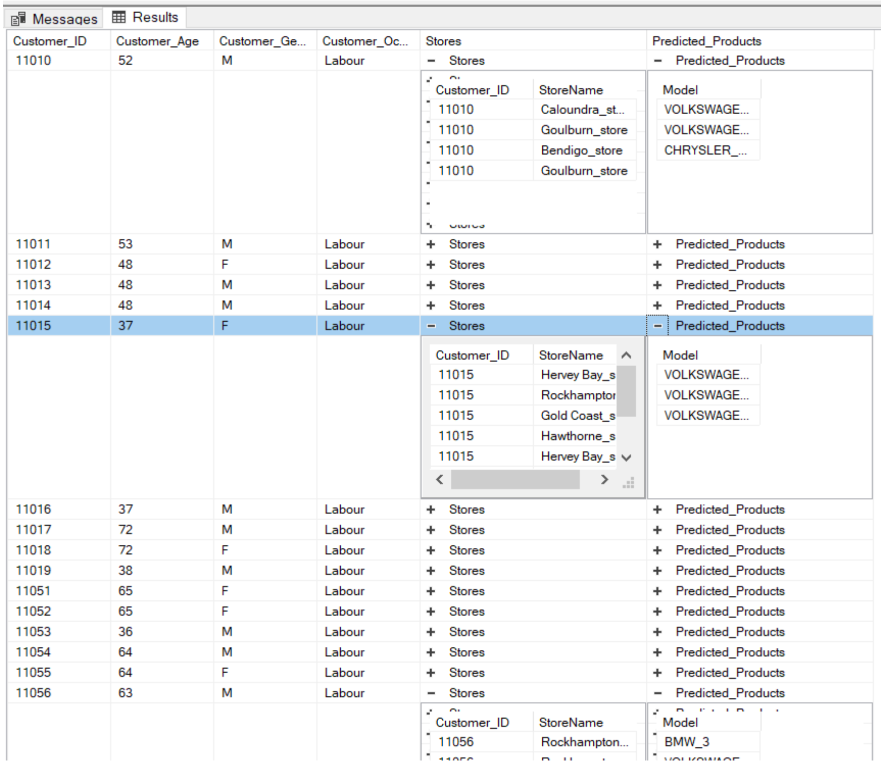


In this screen shot I showed the prediction on what's would the costumer rent based on his demographic information and rented cars.

JiYan Zhu: n10415483, Shu Du: n10505024.

Prediction 3

```
SELECT t.Customer_ID, t.Customer_Age, t.Customer_Gender, t.Customer_Occupation, t.Stores,
PREDICT([Products], 3) AS Predicted_Products
    FROM [pred3]
PREDICTION JOIN
SHAPE {
  OPENQUERY([CarRentals],'SELECT DISTINCT [Customer_ID],cust.Customer_Age, cust.Customer_Gender, cust.Customer_Occupation
FROM dbo.Fact_Rentals AS x
INNER JOIN dbo.Customer AS cust ON cust.Customer_ID = x.Order_Customer
INNER JOIN dbo.Car AS car ON car.Car_ID = x.Order_Car
')}
APPEND(
    {OPENQUERY([CarRentals],'SELECT Customer_ID, st.Store_Name AS StoreName FROM
dbo.Fact_Rentals as z
INNER JOIN dbo.Customer AS cust ON cust.Customer_ID = z.Order_Customer
INNER JOIN dbo.Store AS st ON st.Store_ID = z.Order_Store ORDER BY Customer_ID')
    }
RELATE Customer_ID to Customer_ID
)AS [Stores],
({OPENQUERY([CarRentals],'SELECT Customer_ID, car.Car_Model AS [Model]  FROM
dbo.Fact_Rentals as j
INNER JOIN dbo.Customer AS cust ON cust.Customer_ID = j.Order_Customer
INNER JOIN dbo.Car AS car ON car.Car_ID = j.Order_Car ORDER BY Customer_ID')}
RELATE Customer_ID to Customer_ID
) AS [Products]
AS t
ON [pred3].Customer_ID = t.Customer_ID AND
[pred3].Customer_Age = t.Customer_Age AND
[pred3].Customer_Gender = t.Customer_Gender AND
[pred3].Customer_Occupation = t.Customer_Occupation AND
[pred3].Stores.[StoreName] = t.Stores.[StoreName] AND
[pred3].Products.[Model] = t.Products.[Model]
```



Customer_ID	Customer_Age	Customer_Ge...	Customer_Oc...	Stores	Predicted_Products
11010	52	M	Labour	− Stores	− Predicted_Products
11011	53	M	Labour	+ Stores	+ Predicted_Products
11012	48	F	Labour	+ Stores	+ Predicted_Products
11013	48	M	Labour	+ Stores	+ Predicted_Products
11014	48	M	Labour	+ Stores	+ Predicted_Products
11015	37	F	Labour	− Stores	− Predicted_Products
11016	37	M	Labour	+ Stores	+ Predicted_Products
11017	72	M	Labour	+ Stores	+ Predicted_Products
11018	72	F	Labour	+ Stores	+ Predicted_Products
11019	38	M	Labour	+ Stores	+ Predicted_Products
11051	65	F	Labour	+ Stores	+ Predicted_Products
11052	65	F	Labour	+ Stores	+ Predicted_Products
11053	36	M	Labour	+ Stores	+ Predicted_Products
11054	64	M	Labour	+ Stores	+ Predicted_Products
11055	64	F	Labour	+ Stores	+ Predicted_Products
11056	63	M	Labour	− Stores	− Predicted_Products

Stores (11010):

Customer_ID	StoreName
11010	Caloundra_st...
11010	Goulburn_store
11010	Bendigo_store
11010	Goulburn_store

Predicted_Products (11010):

Model
VOLKSWAGE...
VOLKSWAGE...
CHRYSLER_...

Stores (11015):

Customer_ID	StoreName
11015	Hervey Bay_s
11015	Rockhamptor
11015	Gold Coast_s
11015	Hawthorne_s
11015	Hervey Bay_s

Predicted_Products (11015):

Model
VOLKSWAGE...
VOLKSWAGE...
VOLKSWAGE...

Stores (11056):

Customer_ID	StoreName
11056	Rockhampton...

Predicted_Products (11056):

Model
BMW_3

In these screen shots I showed what car would the customer rent based on his demographic information, stores and rented cars. I showed customer ID on store table to proof that one some customer will rent in many different stores.

Task 5

The last task is to evaluate the performance of the first mining model, and tried to make it more accurate by changing the parameters. We decided to use both SystemGetAccuracyResults and SystemGetCrossValidationResults to proof that the model does not need any adjustment.

First I'll show you the pass and fail for the model without changes on the parameters.

```
CALL SystemGetAccuracyResults([Car_Rentalsv1],[pred1], 3, 'Customer_Occupation', NULL)
GO
CALL SystemGetCrossValidationResults([Car_Rentalsv1],[pred1], 2, 0, 'Customer_Occupation', NULL)
```

99 %

Messages | Results

ModelName	AttributeName	AttributeState	PartitionIndex	PartitionSize	Test	Measure	Value
pred1	Customer_Oc...		0	1067	Classification	Pass	1053
pred1	Customer_Oc...		0	1067	Classification	Fail	14
pred1	Customer_Oc...		0	1067	Likelihood	Log Score	-0.013137205...
pred1	Customer_Oc...		0	1067	Likelihood	Lift	1.419887977...
pred1	Customer_Oc...		0	1067	Likelihood	Root Mean S...	0.021590080...

ModelName	AttributeName	AttributeState	PartitionIndex	PartitionSize	Test	Measure	Value
pred1	Customer_Oc...		1	373	Classification	Pass	358
pred1	Customer_Oc...		1	373	Classification	Fail	15
pred1	Customer_Oc...		1	373	Likelihood	Log Score	-0.031727701...
pred1	Customer_Oc...		1	373	Likelihood	Lift	1.405350581...
pred1	Customer_Oc...		1	373	Likelihood	Root Mean S...	0.049365588...
pred1	Customer_Oc...		2	374	Classification	Pass	351
pred1	Customer_Oc...		2	374	Classification	Fail	23
pred1	Customer_Oc...		2	374	Likelihood	Log Score	-0.069898620...
pred1	Customer_Oc...		2	374	Likelihood	Lift	1.366593797...
pred1	Customer_Oc...		2	374	Likelihood	Root Mean S...	0.057745784...

The first table of the screen shot shows the overall pass and fall. According to the table, it shows that 1053 passed and 14 failed, which make it over accuracy is around all 98.7%. The table under the first table is spitting the data in two partition, and each partition size is around 373.5. the accuracy on those is very similar to the table above. The next step will be changing the parameters of this model, and try to make the accuracy higher by copy this mode, and set the minimum support = 50, and core method = 1.

```
SELECT INTO [pred1_1]
USING Microsoft_Decision_Trees(MINIMUM_SUPPORT = 50, SCORE_METHOD = 1)
FROM [pred1]
GO
INSERT INTO [pred1_1]
```

```
CALL SystemGetAccuracyResults([Car_Rentalsv1],[pred1_1], 3, 'Customer_Occupation', NULL)
GO
CALL SystemGetCrossValidationResults([Car_Rentalsv1],[pred1_1], 2, 0, 'Customer_Occupation', NULL)
```

99 %

Messages | Results

ModelName	AttributeName	AttributeState	PartitionIndex	PartitionSize	Test	Measure	Value
pred1_1	Customer_Oc...		0	1067	Classification	Pass	803
pred1_1	Customer_Oc...		0	1067	Classification	Fail	264
pred1_1	Customer_Oc...		0	1067	Likelihood	Log Score	-0.631713962...
pred1_1	Customer_Oc...		0	1067	Likelihood	Lift	0.801311220...
pred1_1	Customer_Oc...		0	1067	Likelihood	Root Mean S...	0.319620868...

ModelName	AttributeName	AttributeState	PartitionIndex	PartitionSize	Test	Measure	Value
pred1_1	Customer_Oc...		1	373	Classification	Pass	209
pred1_1	Customer_Oc...		1	373	Classification	Fail	164
pred1_1	Customer_Oc...		1	373	Likelihood	Log Score	-0.952846098...
pred1_1	Customer_Oc...		1	373	Likelihood	Lift	0.484232184...
pred1_1	Customer_Oc...		1	373	Likelihood	Root Mean S...	0.506077308...
pred1_1	Customer_Oc...		2	374	Classification	Pass	221
pred1_1	Customer_Oc...		2	374	Classification	Fail	153
pred1_1	Customer_Oc...		2	374	Likelihood	Log Score	-0.915649843...
pred1_1	Customer_Oc...		2	374	Likelihood	Lift	0.520842574...
pred1_1	Customer_Oc...		2	374	Likelihood	Root Mean S...	0.486320612...

After changing the parameters, the accuracy has dropped significantly, this can be proofed by the above screen shot, the accuracy went from 98.7% to 75%.

The below screen shots are the next method of changing the parameters, this time I changed the minimum support =1, and score method =1. The result of this change is clearly better than the second one; however, it is still not as well as the original model. This new model made from copied the first model accuracy is around 98.3%.

```
SELECT INTO [pred1_2]
USING Microsoft_Decision_Trees(MINIMUM_SUPPORT = 1, SCORE_METHOD = 1)
FROM [pred1]
GO
INSERT INTO [pred1_2]
```

```
CALL SystemGetAccuracyResults([Car_Rentalsv1],[pred1_2], 3, 'Customer_Occupation', NULL)
GO
CALL SystemGetCrossValidationResults([Car_Rentalsv1],[pred1_2], 2, 0, 'Customer_Occupation', NULL)
```

109 %

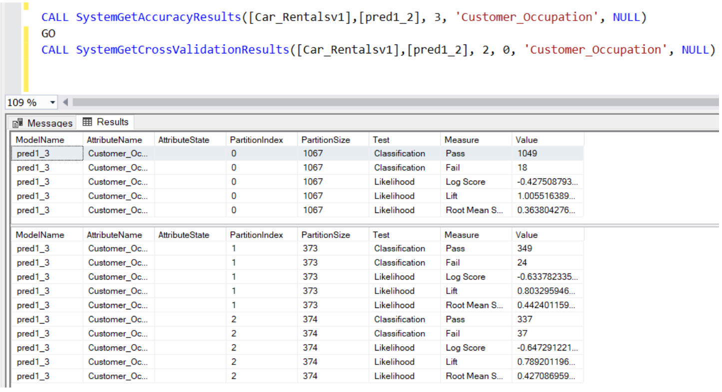
Messages | Results

ModelName	AttributeName	AttributeState	PartitionIndex	PartitionSize	Test	Measure	Value
pred1_2	Customer_Oc...		0	1067	Classification	Pass	1049
pred1_2	Customer_Oc...		0	1067	Classification	Fail	18
pred1_2	Customer_Oc...		0	1067	Likelihood	Log Score	-0.397314470...
pred1_2	Customer_Oc...		0	1067	Likelihood	Lift	1.035710712...
pred1_2	Customer_Oc...		0	1067	Likelihood	Root Mean S...	0.345368779...

ModelName	AttributeName	AttributeState	PartitionIndex	PartitionSize	Test	Measure	Value
pred1_2	Customer_Oc...		1	373	Classification	Pass	348
pred1_2	Customer_Oc...		1	373	Classification	Fail	25
pred1_2	Customer_Oc...		1	373	Likelihood	Log Score	-0.614303535...
pred1_2	Customer_Oc...		1	373	Likelihood	Lift	0.822774747...
pred1_2	Customer_Oc...		1	373	Likelihood	Root Mean S...	0.430465028...
pred1_2	Customer_Oc...		2	374	Classification	Pass	335
pred1_2	Customer_Oc...		2	374	Classification	Fail	39
pred1_2	Customer_Oc...		2	374	Likelihood	Log Score	-0.648052444...
pred1_2	Customer_Oc...		2	374	Likelihood	Lift	0.788439972...
pred1_2	Customer_Oc...		2	374	Likelihood	Root Mean S...	0.423579532...

The last change will be adding spilt method =2. This change did not make any big impact, the result is very similar to the changes made in the second method. However, the first partition accuracy has increased. The second method's second table's first partition result is 348 out of 373, where this time is 349 out of 373. But the last method still did not perform well than the original unchanged model, which make the original model the best at predicting customer's occupation.

```
SELECT INTO [pred1_3]
USING Microsoft_Decision_Trees(MINIMUM_SUPPORT = 1, SCORE_METHOD = 1, SPLIT_METHOD=2)
FROM [pred1]
GO
INSERT INTO [pred1_3]
```



```
CALL SystemGetAccuracyResults([Car_Rentalsv1],[pred1_2], 3, 'Customer_Occupation', NULL)
GO
CALL SystemGetCrossValidationResults([Car_Rentalsv1],[pred1_2], 2, 0, 'Customer_Occupation', NULL)
```

109 %

Messages Results

ModelName	AttributeName	AttributeState	PartitionIndex	PartitionSize	Test	Measure	Value
pred1_3	Customer_Oc...		0	1067	Classification	Pass	1049
pred1_3	Customer_Oc...		0	1067	Classification	Fail	18
pred1_3	Customer_Oc...		0	1067	Likelihood	Log Score	-0.427508793...
pred1_3	Customer_Oc...		0	1067	Likelihood	Lift	1.005516389...
pred1_3	Customer_Oc...		0	1067	Likelihood	Root Mean S...	0.363804276...

ModelName	AttributeName	AttributeState	PartitionIndex	PartitionSize	Test	Measure	Value
pred1_3	Customer_Oc...		1	373	Classification	Pass	349
pred1_3	Customer_Oc...		1	373	Classification	Fail	24
pred1_3	Customer_Oc...		1	373	Likelihood	Log Score	-0.633782335...
pred1_3	Customer_Oc...		1	373	Likelihood	Lift	0.803295946...
pred1_3	Customer_Oc...		1	373	Likelihood	Root Mean S...	0.442401159...
pred1_3	Customer_Oc...		2	374	Classification	Pass	337
pred1_3	Customer_Oc...		2	374	Classification	Fail	37
pred1_3	Customer_Oc...		2	374	Likelihood	Log Score	-0.647291221...
pred1_3	Customer_Oc...		2	374	Likelihood	Lift	0.789201196...
pred1_3	Customer_Oc...		2	374	Likelihood	Root Mean S...	0.427086959...

Overall, there are many methods to change the change the model, there are only a few of them. However, due to the number of ways to change, it makes this task very time consuming, and some of the ways are out of the scope of this unit. For now, we only used 3 methods, and out of those 3 methods, the original model has the bet prediction accuracy on predicting the customer's occupation, but this does not mean it is the best model, there are many ways to change it.

JiYan Zhu: n10415483, Shu Du: n10505024.

Student name	**ID Number**
JiYan Zhu	10415483
Shu Du	10505024
Tasks	**Statement of completeness**
Task 1	Finished
Task 2	Finished
Task 3	Finished
Task 4	Finished
Task 5	Finished