Assignment 4

Note: Show all your work.
Problem 1 Consider the following confusion matrix.

	predicted class			Sum
actual class		C1	C2	
	C1	770	230	1000
	C2	150	250	400
Sum		920	480	1400

Note: C1 is positive and C2 is negative.
Compute *sensitivity*, *specificity*, *precision*, *accuracy*, *F-meassure*, and F_2.
Answer:
Sensitivity (TP rate or recall) = TP/P (# true positives / # positives) = 770 / 1000 = 0.77
Specificity *(*TN *rate)* = TN / N (# true negatives / # negatives) = 250 / 400 = 0.625
Precision = TP / (TP+FP) (# true positives / # tuples classified as positives) = 770 / 920 = 0.837
Accuracy (recognition rate) = (TP+TN) / (P+N) (# correctly classified tuples / # all tuples)
= (770 + 250) / (1000 + 400) = 1020 / 1400 = 0.728
F-measure (*F*1 or *F*-score) = (2×precision×recall) / (precision + recall) (harmonic mean of precision and recall) = (2 × 0.837 × 0.77) / (0.837 + 0.77) = 0.80

F2 = (recall has 2 times as much weight as precision)

$[(1+2^2) \times \text{precision} \times \text{recall}] / (2^2 \times \text{precision} + \text{recall}) = (5 \times 0.837 \times 0.77) / (4 \times 0.837 + 0.77)$
= 3.22 / 4.11 = 0.78

Problem 2 Suppose you built two classifier models *M*1 and *M*2 from the same training dataset and tested them on the same test dataset using 10-fold crossvalidation. The error rates obtained over 10 iterations (in each iteration the same training and test partitions were used for both *M*1 and *M*2) are given in the table below. Determine whether there is a significant difference between the two models using the statistical method discussed in Section 7 of the online lecture Module 4 (also in Section 8.5.5, pp 372-373 of the textbook). Use a significance level of 1%. If there is a significant difference, which one is better?

Iteration	M1	M2
1	0.12	0.15
2	0.21	0.18
3	0.05	0.1
4	0.12	0.18
5	0.1	0.08
6	0.16	0.13
7	0.08	0.09
8	0.21	0.2
9	0.11	0.18
10	0.14	0.21

Answer:

Iteration	M1	M2	M1-M2	ave(M1)-ave(M2)	((M1-M2)-(ave(M1)-ave(M2)))^2
1	0.12	0.15	-0.03	-0.02	0.0001
2	0.21	0.18	0.03	-0.02	0.0025
3	0.05	0.1	-0.05	-0.02	0.0009
4	0.12	0.18	-0.06	-0.02	0.0016
5	0.1	0.08	0.02	-0.02	0.0016
6	0.16	0.13	0.03	-0.02	0.0025
7	0.08	0.09	-0.01	-0.02	0.0001
8	0.21	0.2	0.01	-0.02	0.0009
9	0.11	0.18	-0.07	-0.02	0.0025
10	0.14	0.21	-0.07	-0.02	0.0025
Average	0.13	0.15			0.00152

Var (M1−M2) = 0.00152

$$t = \frac{ave(M_1)-ave(M_2)}{\sqrt{\frac{Var(M_1-M_2)}{k}}} = \frac{0.13-0.15}{\sqrt{\frac{0.00152}{10}}} = -1.62$$

The significance level *of* 1% (or *sig* = 0.01) means we want to assert that the difference between the two error rates is significantly different for 99% of the population. We use $z = sig/2 = 0.005$. From the *t*-distribution table, $t0.005, 9 = 3.250$. Since $|t| = |-1.62| < t0.005, 9 = 3.250$, we cannot reject the null hypothesis and conclude that there is no significant difference in the error rates of *M*1 and *M*2. If we assume there is a significant difference, then M1 is better than M2 because M1 has less average error rate (0.13) than M2 (0.15).

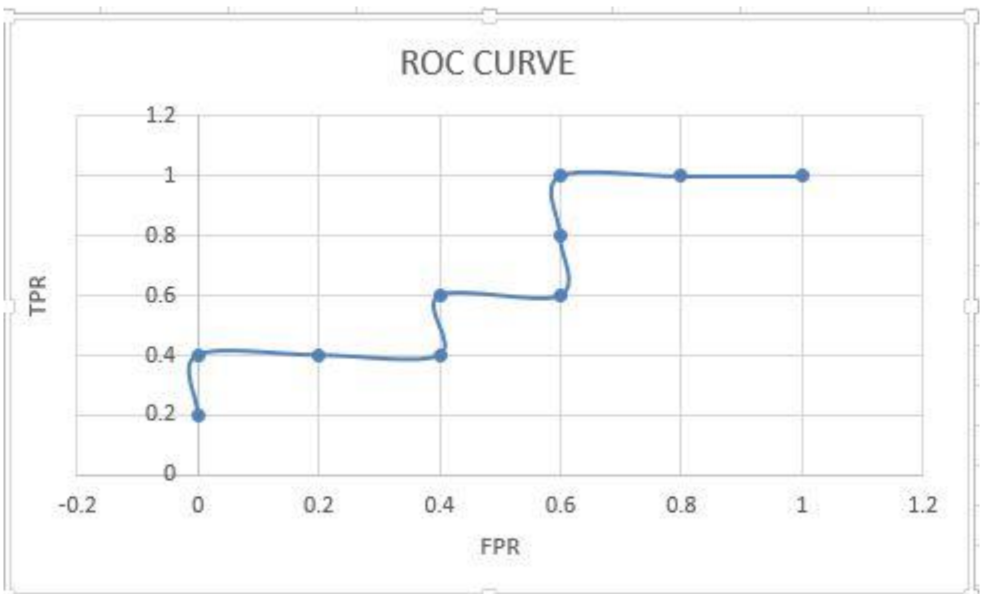
Problem 3. The following table shows a test result of a classifier on a dataset.

Tuple_id	Actual Class	Probability
1	N	0.79
2	P	0.95
3	P	0.82
4	N	0.86
5	P	0.73
6	N	0.69
7	N	0.87
8	N	0.71
9	P	0.75
10	P	0.90

Problem 3-1. For each row, compute *TP*, *FP*, *TN*, *FN*, *TPR*, and *FPR*.

Tuple_d	Actual Class	Probability	TP	FP	TN	FN	TPR	FPR
1	P	0.95	1	0	5	4	0.2	0
2	P	0.9	2	0	5	3	0.4	0
3	N	0.87	2	1	4	3	0.4	0.2
4	N	0.86	2	2	3	3	0.4	0.4
5	P	0.82	3	2	3	2	0.6	0.4
6	N	0.79	3	3	2	2	0.6	0.6
7	P	0.75	4	3	2	1	0.8	0.6
8	P	0.73	5	3	2	0	1	0.6
9	N	0.71	5	4	1	0	1	0.8
10	N	0.69	5	5	0	0	1	1

Problem 3-2. Plot the ROC curve for the dataset.



Problem 4. For this problem, you will run Naïve Bayes and RandomForest classification algorithms on *heart-disease-cs699.arff* dataset and compare the performance of the models built by the two algorithms. Make sure that you select "Cross-validation" for "Test options."

Problem 4-1. First, run Naïve Bayes and RandomForest on *heart-disease-cs699.arff* from Weka Explorer. For each classifier model, capture the screenshot of a part of Classifier Output window that shows "Correctly Classified Instances" and "Confusion Matrix" and include them in your submission. Compare and discuss the performance of the two models using the performance measures in Weka.

```
Time taken to build model: 0.04 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         253               83.4983 %
Incorrectly Classified Instances        50               16.5017 %
Kappa statistic                          0.6655
Mean absolute error                      0.1889
Root mean squared error                  0.365
Relative absolute error                 38.0385 %
Root relative squared error             73.2364 %
Total Number of Instances              303

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.884    0.223    0.824      0.884    0.853      0.668    0.894     0.904     0
                 0.777    0.116    0.850      0.777    0.812      0.668    0.894     0.880     1
Weighted Avg.    0.835    0.174    0.836      0.835    0.834      0.668    0.894     0.893

=== Confusion Matrix ===

   a   b   <-- classified as
 145  19 |   a = 0
  31 108 |   b = 1
```

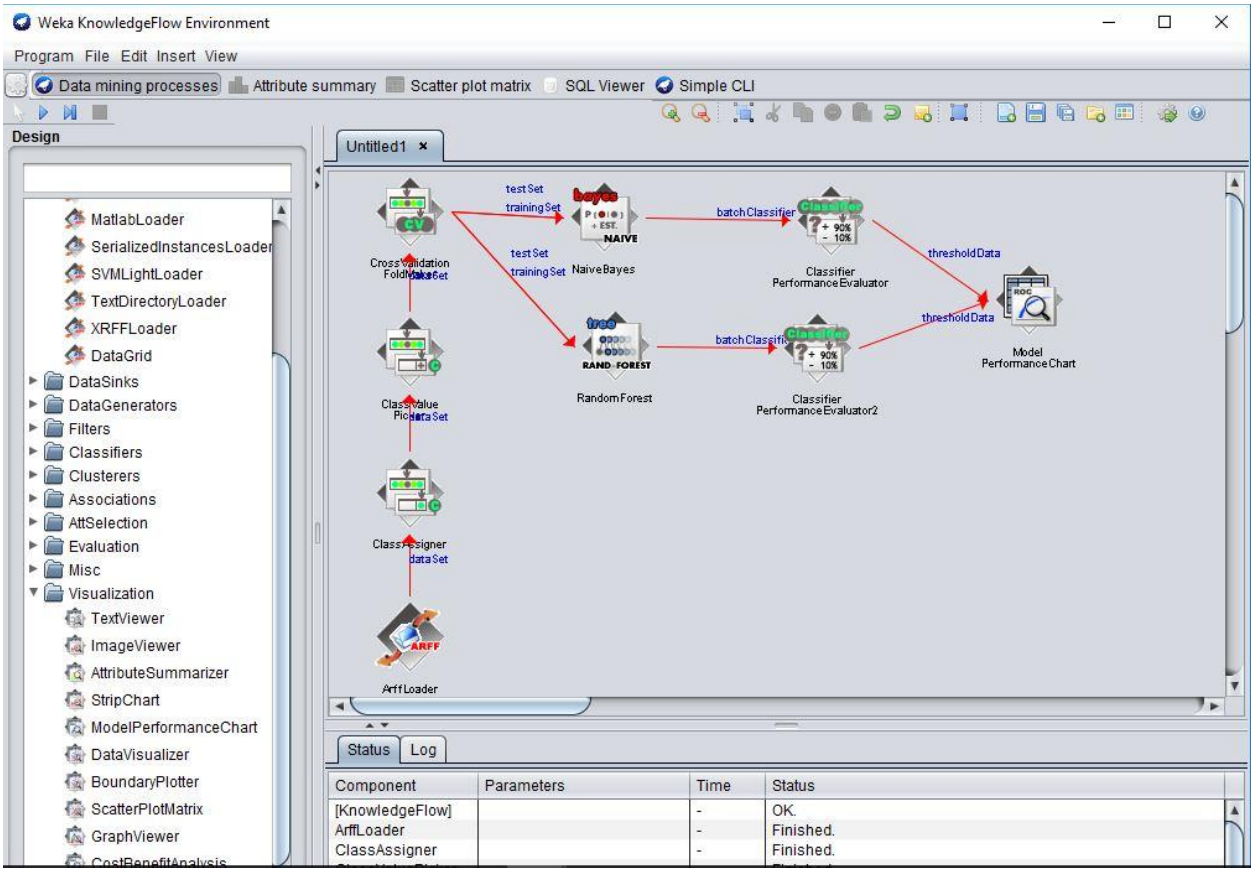
```
weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities

Time taken to build model: 0.44 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         245               80.8581 %
Incorrectly Classified Instances        58               19.1419 %
Kappa statistic                          0.6137
Mean absolute error                      0.267
Root mean squared error                  0.3589
Relative absolute error                 53.7585 %
Root relative squared error             72.0189 %
Total Number of Instances              303

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.835    0.223    0.815      0.835    0.825      0.614    0.898     0.910     0
                 0.777    0.165    0.800      0.777    0.788      0.614    0.898     0.894     1
Weighted Avg.    0.809    0.196    0.808      0.809    0.808      0.614    0.898     0.902

=== Confusion Matrix ===

   a   b   <-- classified as
 137  27 |   a = 0
  31 108 |   b = 1
```

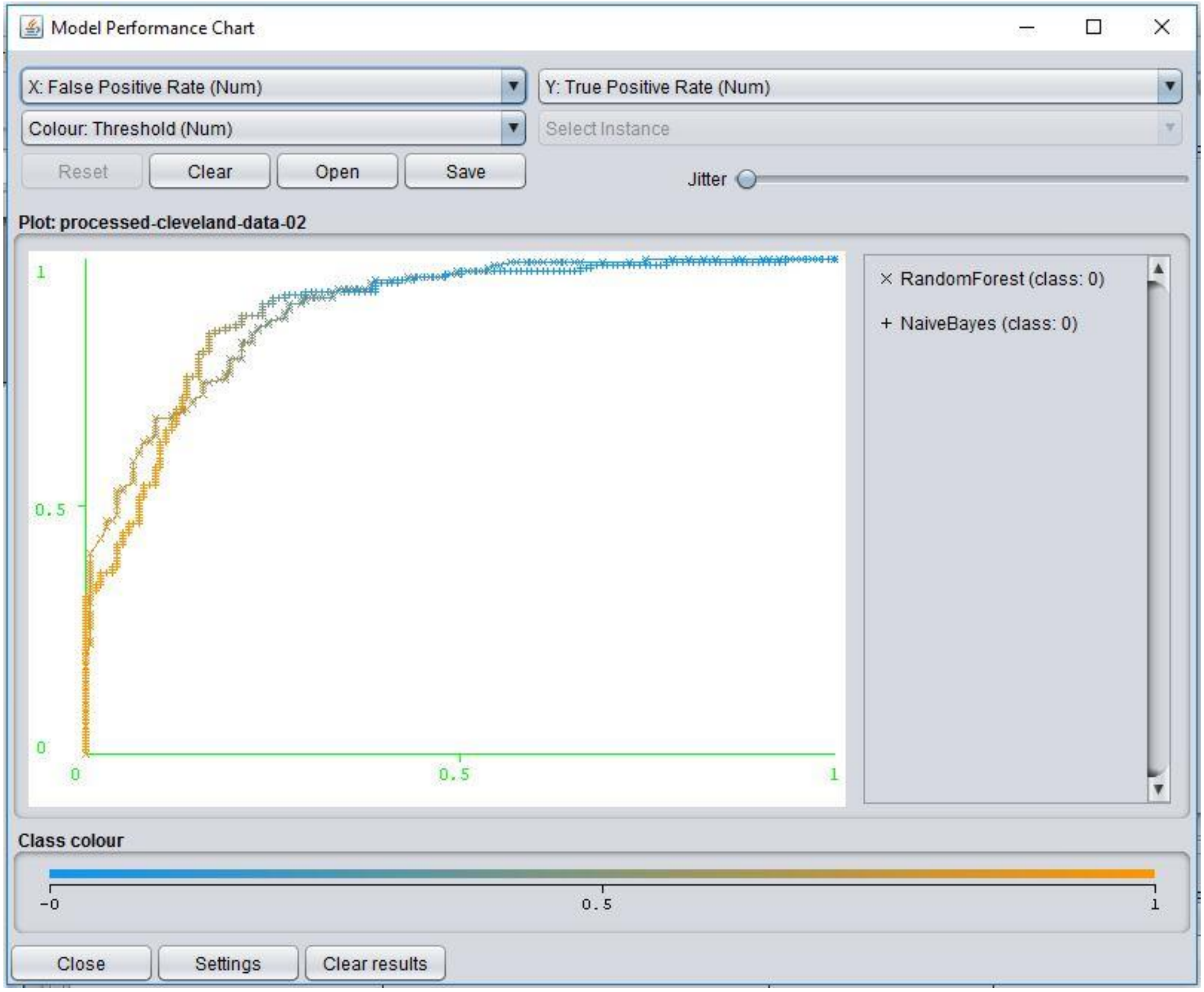
In overall seeing, Naïve Bayes is better choosing than Random Forest in this particular case, because it has better accuracy performance (%83.4983) than Random Forest which has %80.8581 accuracy performance.The table below details the indicators is compared.

	CLASS	Naïve Bayes	Random Forest
TP	0	0.884	0.835
	1	0.777	0.777
Ave		0.835	0.809
FP	0	0.223	0.223
	1	0.116	0.165
Ave		0.174	0.196
Precision	0	0.824	0.815
	1	0.85	0.8
Ave		0.836	0.808
F-Measure	0	0.853	0.825
	1	0.812	0.788
Ave		0.834	0.808
MCC	0	0.668	0.614
	1	0.668	0.614
Ave		0.668	0.614
ROC Area	0	0.894	0.898
	1	0.894	0.898
Ave		0.894	0.898
PRC Area	0	0.904	0.91
	1	0.88	0.894
Ave		0.893	0.902

According to the table above we can see in the most of the performance measures, Naïve Bayes is better than Random Forest in this sample. For example Naïve Bayes has more sensitivity, F-Measure and precision measures than Random Forest. Precision can be thought of as a measure of exactness and F-Measure is the harmonic mean of precision and recall. The Matthews Correlation Coefficient or MCC is the geometric mean of the regression coefficients of the problem and its dual, which Naïve Bayes has better performance in this measure.
Area under ROC curve is often used as a measure of quality of the classification models. In this case Random Forest has better performance. And finally area under PRC curve or Precision Recall Curves indicates how the classifier is behaving on one class. In this measure, Random Forest has better performance.

Problem 4-2. This is a practice of comparing performance of classifier models using ROC curves. You can plot ROC curves using Weka Knowledge Flow. On the Blackboard course web site, I posted a Weka Manual under Discussion board – Common Area. How to use Knowledge Flow is described in Section 7. Following the instruction in the manual (especially Section 7.4.2), build and test Naïve Bayes and RandomForest classifiers on *heart-disease-cs699.arff* dataset, and capture the screenshot which shows two ROC curves. Include this screenshot in your submission. Compare and discuss the performance of the two models using the ROC curves.





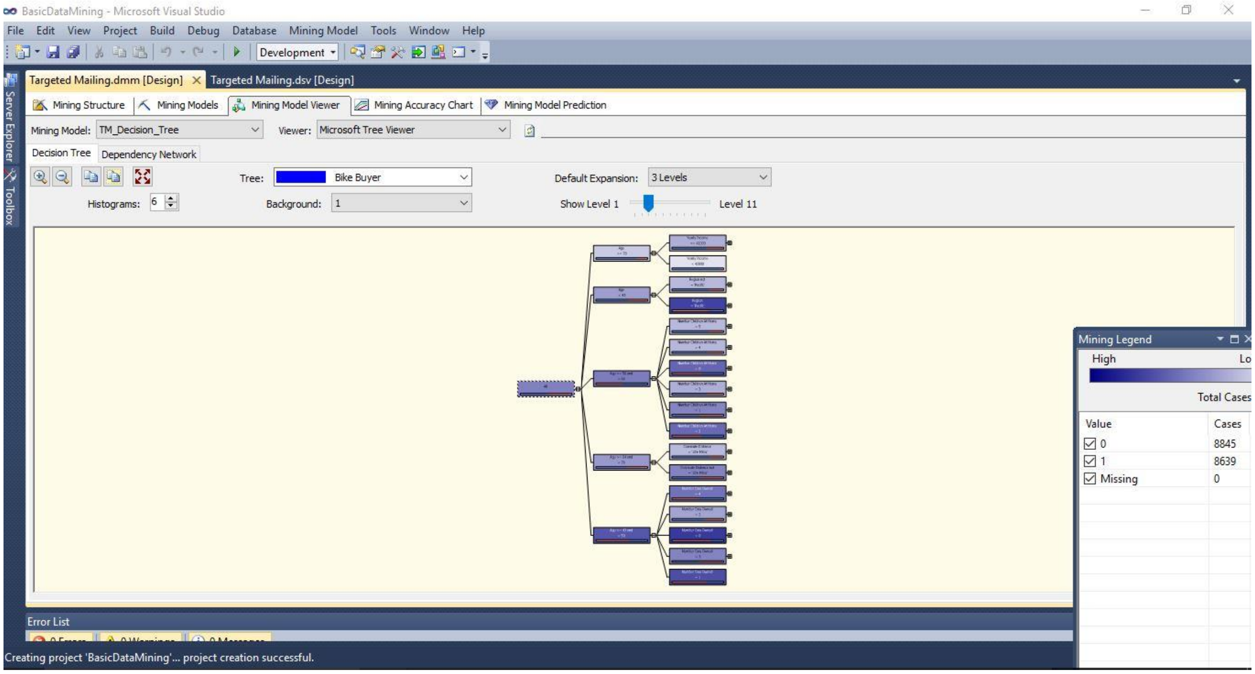
As I mentioned before, Area under ROC curve is often used as a measure of quality of the classification models. Ideally, the curve will climb quickly toward the top-left meaning the model correctly predicted the cases, thus in this case Naïve Bayes is better, because its curve is upper than Random Forest.

Problem 5. This problem has two sections. Problem 5-1 is for SQL Server and Problem 5-2 is for Oracle. Choose one of the two. This problem will take some time – one to three hours depending on your speed.

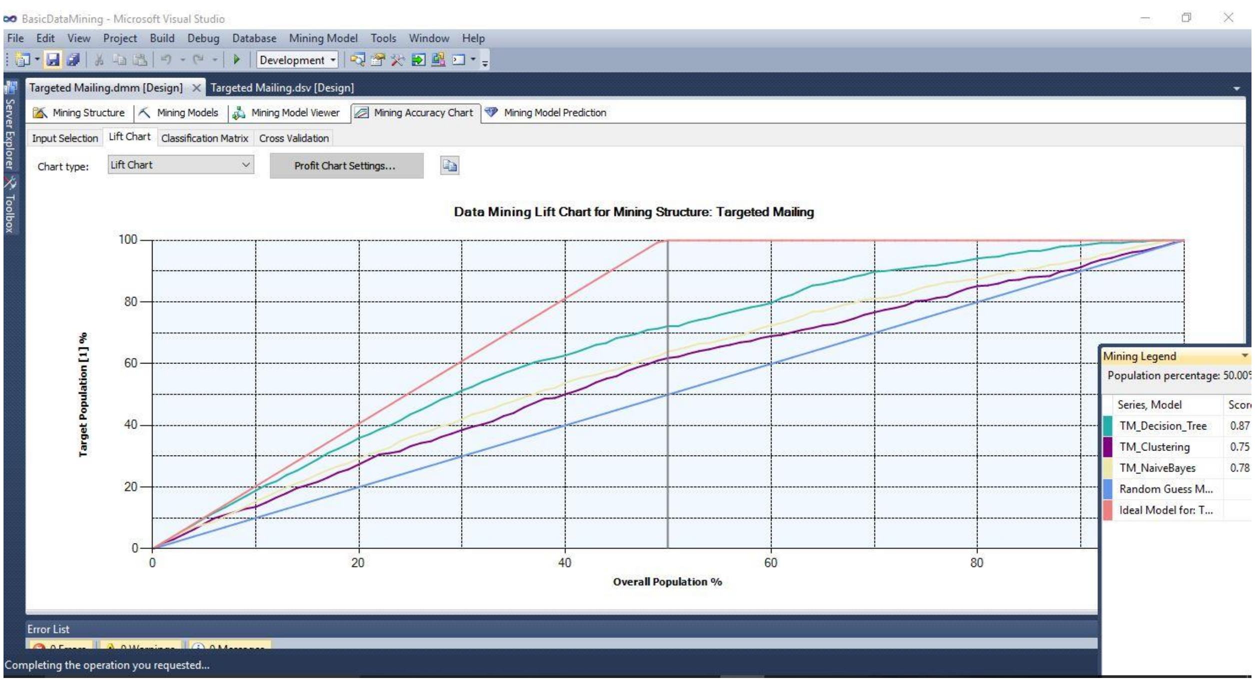
Problem 5-1 (SQL Server). This problem is a practice of building data mining models with SQL Server 2012.

(1) Go to the SQL Server 2012 Data Mining Tutorial web site. The link to the tutorial is: http://msdn.microsoft.com/en-us/library/bb677206(v=sql.110).aspx
(2) Click Basic Data Mining tutorial.
(3) Follow all steps in the Basic Data Mining tutorial.
(4) Capture the following screenshots and paste them onto your submission:
(a) A screen that has the decision tree you built. You will see your decision tree in

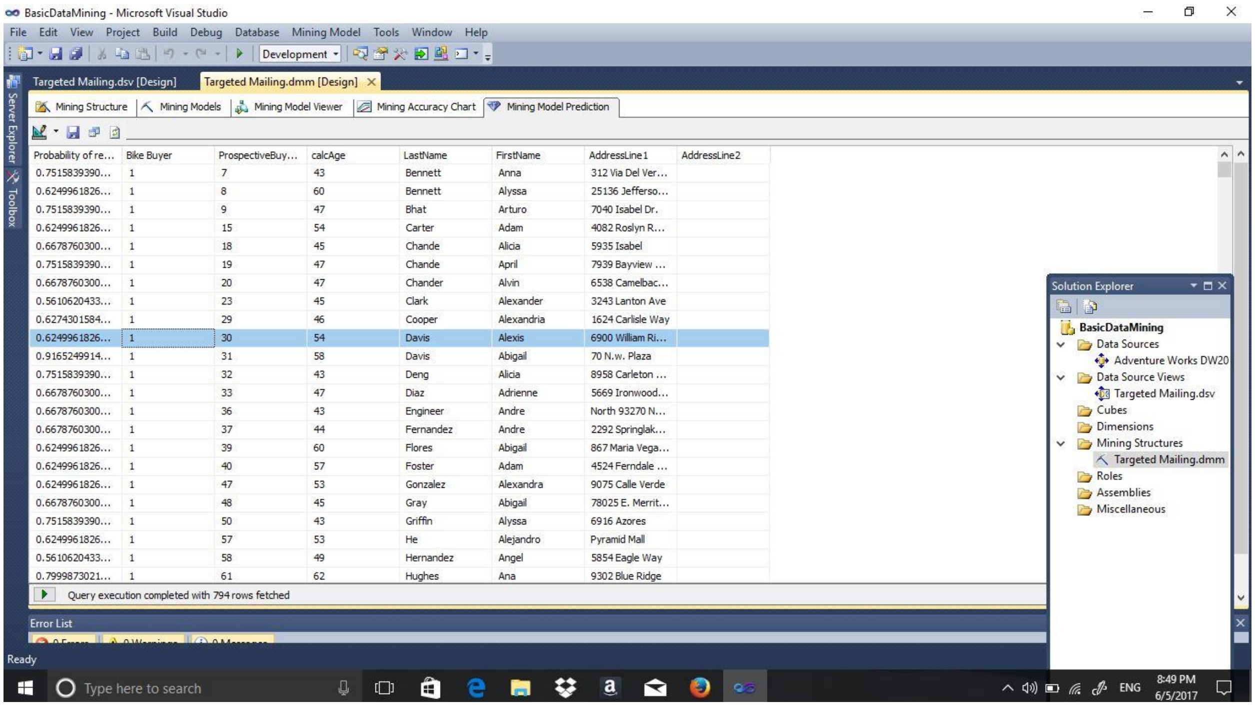
the first task (Exploring the Decision Tree Model) of Lesson 4 (Exploring the Targeted Mailing Models).



(b) A screen that shows lift charts for all three mining models (decision tree, naïve Bayes, and clustering). You will see this screen at the end of the first task (Testing Accuracy with Lift Charts) of Lesson 5 (Testing Models).



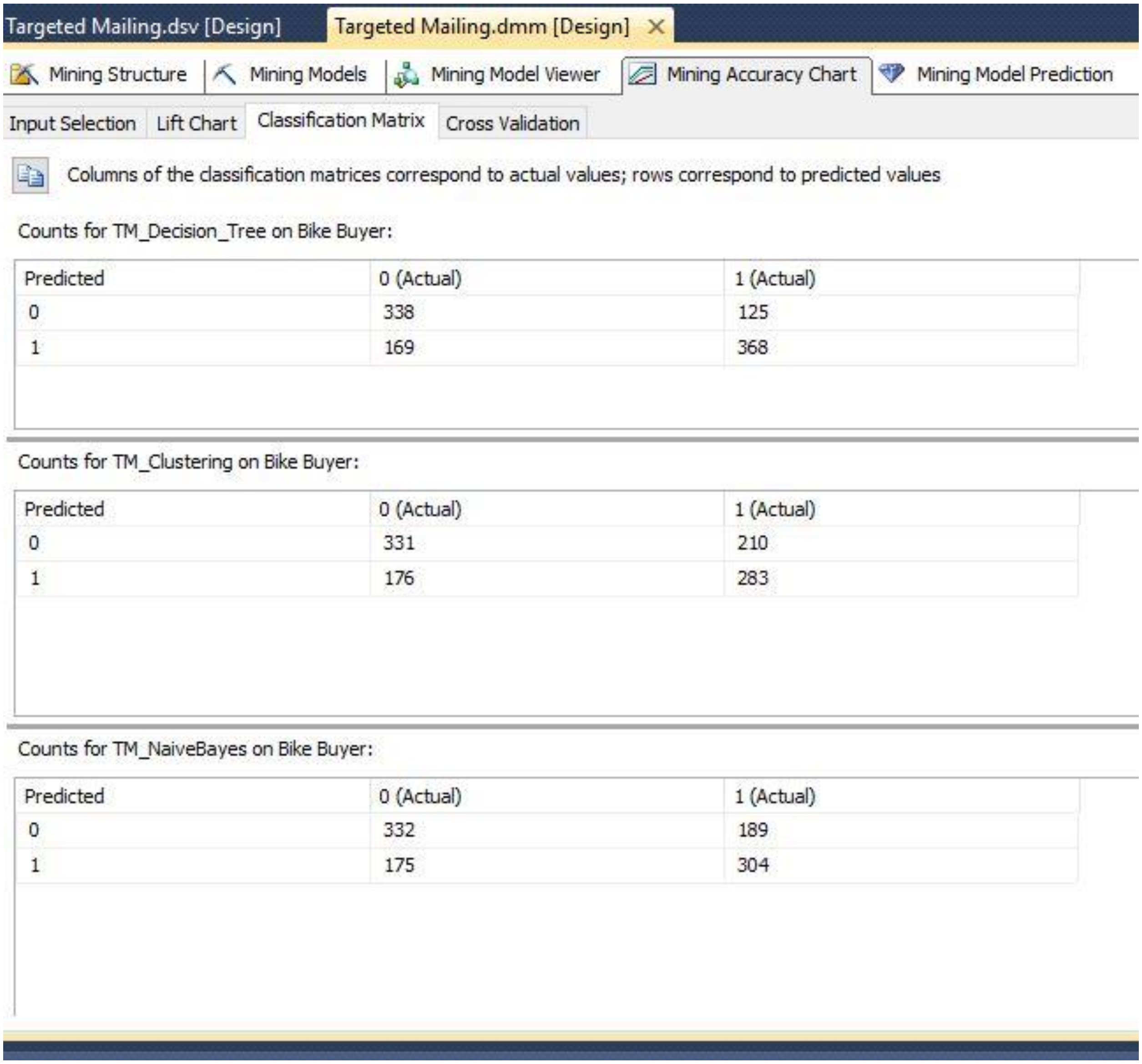
(c) At the end of the first task (Creating Predictions) of Lesson 6 (Creating and Working with Predictions), you will have the result of a query. Capture the screen that shows the query result.



(5) You will see many graphs in the lift chart (in Lesson 5) – one for a random guess model, one or more for ideal models, and one each for the models you built. Explain the differences among a random guess model graph, an ideal model graph, and the graphs for the models you built.

Random guess shows a diagonal straight line where for every true positive of such a model, and is the baseline against which to evaluate lift. Area under random guess is 0.5. Ideal line or best line peaks at around 50 percent, meaning that if I had a perfect model, I could reach 100 percent of my targeted customers by sending a mailing to only 50% of the total population. TM_Decision_Tree model when I target 50 percent of the population is 87 percent, meaning I could reach 87 percent of my targeted customers by sending the mailing to 50 percent of the total customer population.

(6) If you click *Classification Matrix* under *Mining Accuracy Chart* (in Lesson 5), you will see confusions matrices for the models you built. Show all your confusion matrices in your submission, compute the accuracies of all models from the confusion matrices, and compare them.



Targeted Mailing.dsv [Design] | Targeted Mailing.dmm [Design]

Mining Structure | Mining Models | Mining Model Viewer | Mining Accuracy Chart | Mining Model Prediction

Input Selection | Lift Chart | Classification Matrix | Cross Validation

Columns of the classification matrices correspond to actual values; rows correspond to predicted values

Counts for TM_Decision_Tree on Bike Buyer:

Predicted	0 (Actual)	1 (Actual)
0	338	125
1	169	368

Counts for TM_Clustering on Bike Buyer:

Predicted	0 (Actual)	1 (Actual)
0	331	210
1	176	283

Counts for TM_NaiveBayes on Bike Buyer:

Predicted	0 (Actual)	1 (Actual)
0	332	189
1	175	304

TM_Desicion_Tree:

Accuracy = (TP+TN)/ALL = (368+338)/1000 = 0.71
Sensitivity = TP/P = 368/(169+368) = 0.68
Specificity = TN/N = 338/(125+338) = 0.73
Precision = TP/(TP+FP) = 368/(368+125) = 0.74
F = 2*precision*recall / (precision + recall) = 2*0.74*0.68/(0.74+0.68) = 0.70

TM_Clustring:

Accuracy = (TP+TN)/ALL = (283+331)/1000 = 0.61
Sensitivity = TP/P = 283/(283+176) = 0.62
Specificity = TN/N = 331/(331+210) = 0.61
Precision = TP/(TP+FP) = 283/(283+210) = 0.57
F = 2*precision*recall / (precision + recall) = 2*0.57*0.62/(0.57+0.62) = 0.59

TM_NaiveBayes:

Accuracy = (TP+TN)/ALL = (304+332)/1000 = 0.64
Sensitivity = TP/P = 304/(304+175) = 0.63
Specificity = TN/N = 332/(332+189) = 0.63
Precision = TP/(TP+FP) = 304/(304+189) = 0.61
F = 2*precision*recall / (precision + recall) = 2*0.61*0.63/(0.61+0.63) = 0.62

With an overview can be concluded Decision Tree has a better performance than other methods.

Counts for TM_Decision_Tree_Male on Bike Buyer:

Predicted	0 (Actual)	1 (Actual)
0	335	160
1	172	333

Counts for TM_Decision_Tree_Female on Bike Buyer:

Predicted	0 (Actual)	1 (Actual)
0	333	124
1	174	369