PATRT I

1.**Solution:**

Given points are:

A1 (2, 10), A2(2, 5), A3(8, 4)

B1(5, 8), (B2(7, 5), B3(6, 4)

C191, 2), C2(4, 8)

Euclidean distance formulae for points

(x1, y1), (x2, y2) is:

D (x, y) =

Distance matrix table:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A1 | A2 | A3 | B1 | B2 | B3 | C1 | C2 |  |
| A1 | 0 | 5 | 8.49 | 3.6 | 7.07 | 7.21 | 8.08 | 2.23 |
| A2 | 5 | 0 | 6.03 | 4.24 | 5 | 4.12 | 3.16 | 4.42 |
| A3 | 8.49 | 6.08 | 0 | 5 | 1.41 | 2 | 8.06 | 6.4 |
| B1 | 3.6 | 4.24 | 5 | 0 | 3.6 | 4.12 | 7.21 | 1.41 |
| B2 | 7.07 | 5 | 1.41 | 3.60 | 0 | 1.41 | 6.70 | 1.41 |
| B3 | 7.21 | 4.12 | 2 | 4.12 | 1.41 | 0 | 5.38 | 5.38 |
| C1 | 8.06 | 3.16 | 3.06 | 7.21 | 6.70 | 5.38 | 0 | 7.61 |
| C2 | 2.23 | 4.42 | 6.4 | 1.41 | 5 | 5.38 | 7.61 | 0 |

For example, we consider distance between A1 & A3 is A1(2, 10) A3(8, 4)

By using the formulae

= = 8.49

Distance between B2 & C2 is B2(7, 5), C2(4, 8)

d (B2, C2) = = = = 5

Like above we will calculate the values for each point.

Now, initial centroid is given is,

A1(2, 100, B1(5, 8), C1(1, 2)

We construct the table by referring the previous table.

We will form 3 clusters d1, d2, & d3 with centroid A1(2, 10), B1(5, 8), C1(1,2)

**First round K-mean (1-st iteration):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | d(A1) | d(B1) | d(C1) |  |
| A1 | 0 | 3.6 | 8.06 | d1 |
| A2 | 5 | 4.24 | 3.16 | D3 |
| A3 | 8.49 | 5 | 8.06 | d2 |
| B1 | 3.6 | 0 | 7.21 | d2 |
| B2 | 7.07 | 3.60 | 6.70 | d2 |
| B3 | 7.21 | 4.12 | 5.38 | d2 |
| C1 | 8.06 | 7.12 | 0 | d3 |
| C2 | 2.23 | 1.41 | 7.61 | d2 |

1. **The three clusters with cluster points are:**

**Cluster d1** comprises only one point. So, it is centroid remain same i.e, **A1 (2, 10).**

**Cluster d2** comprises of **A3(8, 4), b1(5, 8), b2(7, 5), b3(6, 4), C2(4, 9)**

Centroid (d2) = = (6, 6)

**Cluster d3** comprises of **A2(2, 5), C1(1, 2)**

Centroid (d3) = = (1.5, 3.5)

**2nd round of K-mean (2nd iteration):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | d (2, 10) | d (6, 60 | d (1.5, 3.5) | Cluster Assigned |
| A1 | 0 | 5.6 | 6.51 | d1 |
| A2 | 5 | 4.12 | 1.58 | d3 |
| A3 | 8.48 | 2.82 | 6.51 | d2 |
| B1 | 3.6 | 2.23 | 5.7 | d2 |
| B2 | 7.07 | 1.41 | 5.7 | d2 |
| B3 | 7.21 | 2 | 4.52 | d2 |
| C1 | 8.06 | 6.40 | 1.58 | d3 |
| C2 | 2.23 | 3.60 | 6.04 | D1 |

In the previous table, each value is calculated by finding the distance below each point & each centroid of the cluster using Euclidean distance formulae.

New centroid points are:

Cluster d1 comprises A1(2, 10), C2(4, 8)

d1= = (3, 9.5)

Cluster d2 comprises of A3(8, 4), B1 (5, 8), B2(7, 5), B3(6, 4)

D2 = = (6.5, 5.25)

Cluster d3 comprises of A2(2, 5), C1(1, 2)

D3 = ( )= (1.5, 3.5)

**3-rd round of K- mean (3-rd iteration):**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | d1(3, 9.5) | d2(6.5, 5.25) | d3(1.5, 3.5) | Cluster Assigned |
| A1 | 1.11 | 6.54 | 6.51 | d1 |
| A2 | 4.60 | 4.50 | 1.58 | d3 |
| A3 | 7.43 | 1.95 | 6.51 | d2 |
| B1 | 2.5 | 3.13 | 5.7 | d1 |
| B2 | 6.02 | 0.55 | 5.7 | d2 |
| B3 | 7.03 | 0.90 | 4.52 | d2 |
| C1 | 7.76 | 6.38 | 1.58 | d3 |
| C2 | 1.11 | 6.04 | 6.04 | d1 |

1. **After the third iteration the final three clusters and their new centroid Points are:**

**Cluster d1**- **A1(2, 10), B1(5, 8), c2(4, 8)**

d1 = = (3.66, 9)

**Cluster d2** – **A3(8, 4), B2(7, 5), B3(6, 4)**

d2 = = (7, 4.33)

**Cluster d3 - A2(2, 5), C1(1, 2)**

d3 = = (1.5, 3.5)

**2. Answer:**

Support Vector Machine (SVM) is the classification technique which used to process on large training data. The Big and complex data can be left to the SVM since the result of SVM will be greatly influenced when there is too much noise in the datasets. SVM provides with an optimized algorithm to solve the problem of over fitting. SVM is an effective classification model is useful to handle those complex data. SVM can make use of certain kernels to reveal efficiently in quantum form the largest eigenvalues and corresponding eigenvectors of the training data overlap (kernel) and covariance matrices.

SVM have high training performance and low generalization error which pointed out the potential problems of SVMs when the training set is noisy and imbalanced. The SVM is not that much scalable on large data sets because it takes time for multiple scanning of data sets hence it is too expensive to perform. To overcome this problem, Clustering-Based SVM (CB-SVM) comes into picture for scalability and reliability of SVM classification. Clustering-Based SVM (CB-SVM) is the SVM technique that is designed for handling large data sets which applies on hierarchical micro-clustering algorithm that scans the entire data set only once to provide the high quality of samples. CB-SVM is scalable if and only if the efficiency of training maximizing the performance of SVMs.

The authors have illustrated the characteristics and description of hierarchical micro-clustering algorithm.

In sketching the algorithm, they said that “while selective sampling needs to scan the entire data set at each round to select the closest data point, CB-SVM runs based on the CF tree which can be constructed in a single scan of the entire data set and is carrying the statistical summaries that facilitates constructing an SVM boundary efficiently and effectively”. They projected steps of SVM algorithm which is as follows:

1. construct two CF trees from positive and negative data set independently.

2. train an SVM boundary function from the centroids of the root entries – entries in the root node – of the two CF trees. If the root node contains too few entries, train from the entries of the nodes in the second levels of the trees.

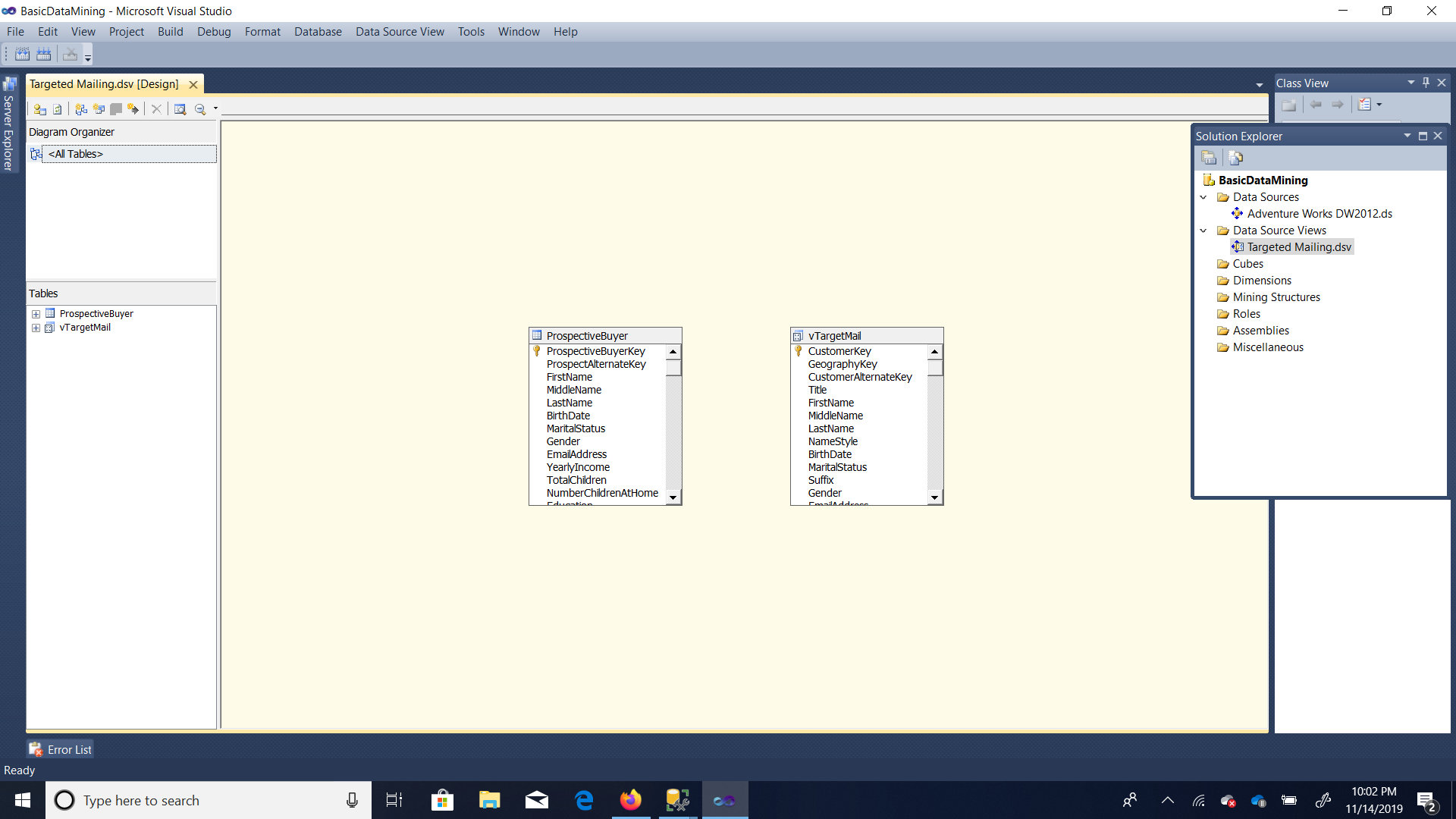
3. de-cluster the entries near the boundary into the next level, and the children entries de-clustered from the parent entries are accumulated into the training set with the non-de-clustered parent entries.

4. construct another SVM from the centroids of the entries in the training set and repeat from step 3 until nothing is accumulated.

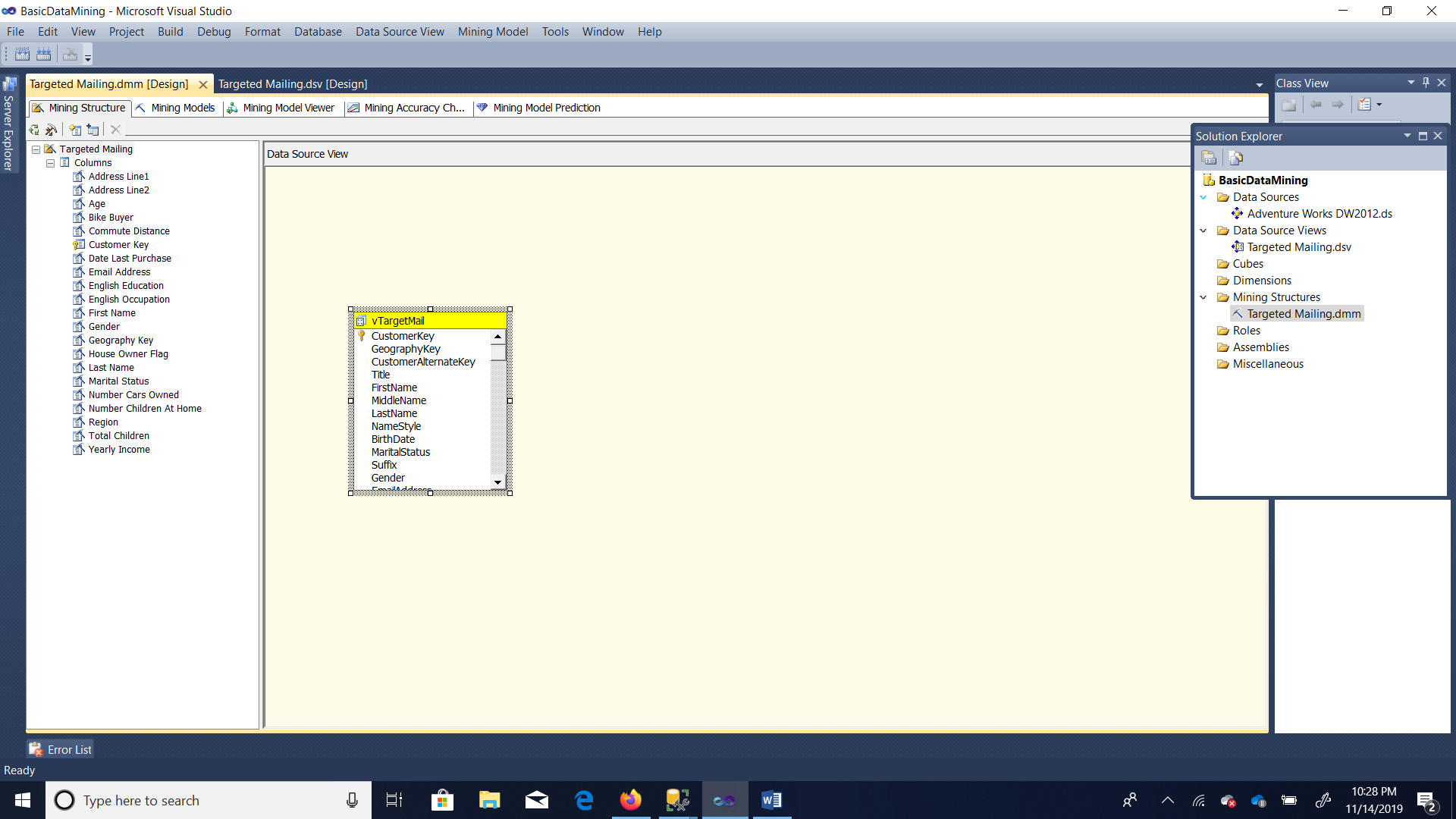
They also explained that the CF tree is a suitable base structure for CB-SVM to perform the selective de-clustering efficiently. Not only that they also provided theorem for it and theorem 2 states that CB-SVM trains asymptotically times faster than a standard SVM given a CF tree. The training time of CB-SVM is asymptotically equal to that of a standard SVM only if all the training data points become the SVs. The rate of the SVs is variant, depending on the type of problems, the type of kernels, the number of dimensions, the number of data points, and the SVM parameters. However, mostly, especially for very large data sets. So, the performance difference between CB-SVM and a standard SVM goes higher as the data set becomes larger. The implementation techniques of many algorithm, the most effective heuristics to speed up SVM training are to divide the original QP problem into small pieces, thereby reducing the size of each QP problem. Chunking, decomposition, and sequential minimal optimization are most well-known techniques. Our CB-SVM algorithm runs on top of these techniques to handle very large data sets by condensing further the training data into the statistical summaries of large data groups such that coarse summary is made for “unimportant” data and fine summary is made for “important” data.

**PART II**

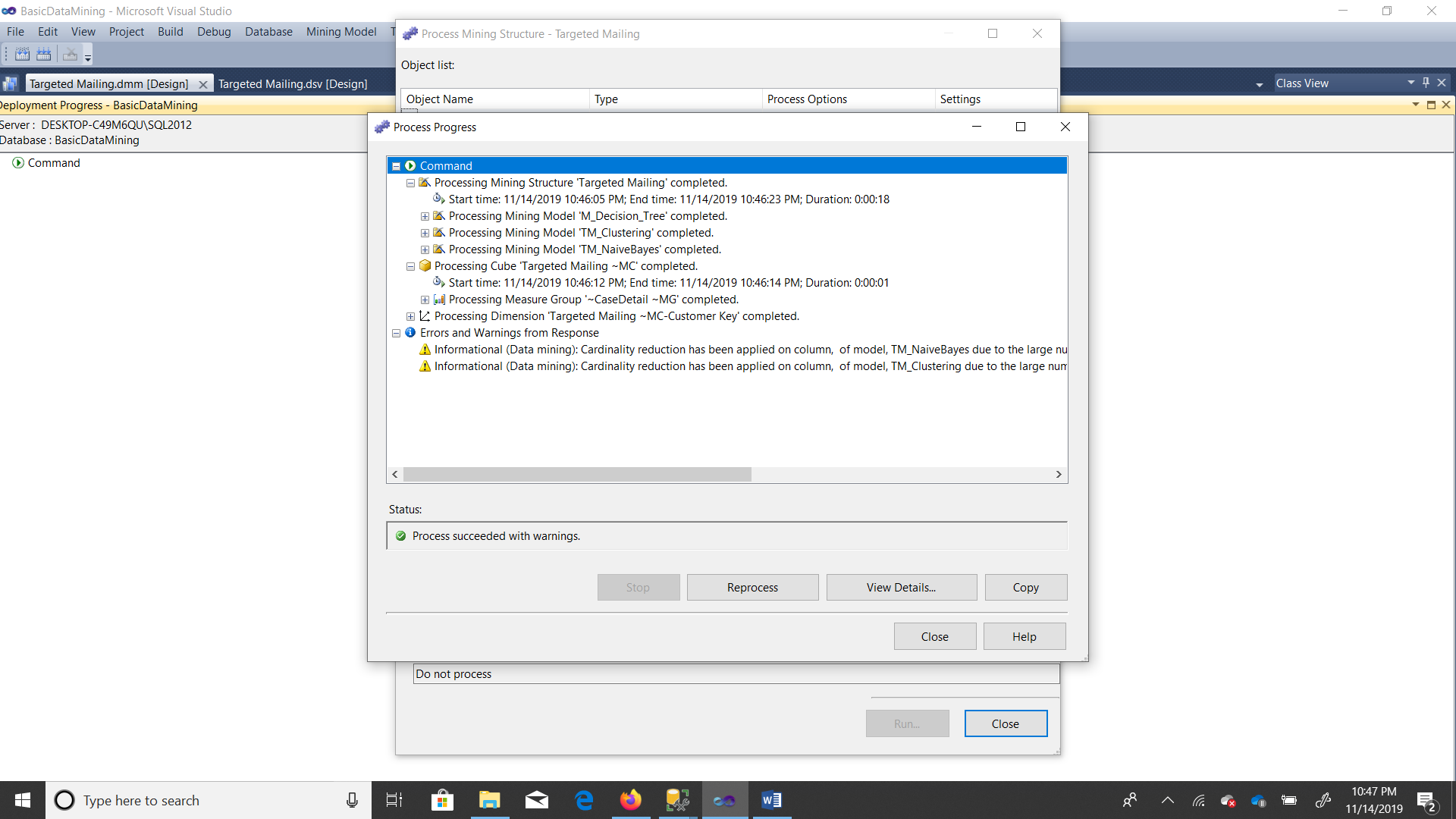
1. Already reviewed
2. All lessons from 1 through 6 has been completed, however, screenshot has not been taken. Some of the example screenshots has been provided below:

Lesson 1:

Lesson II:

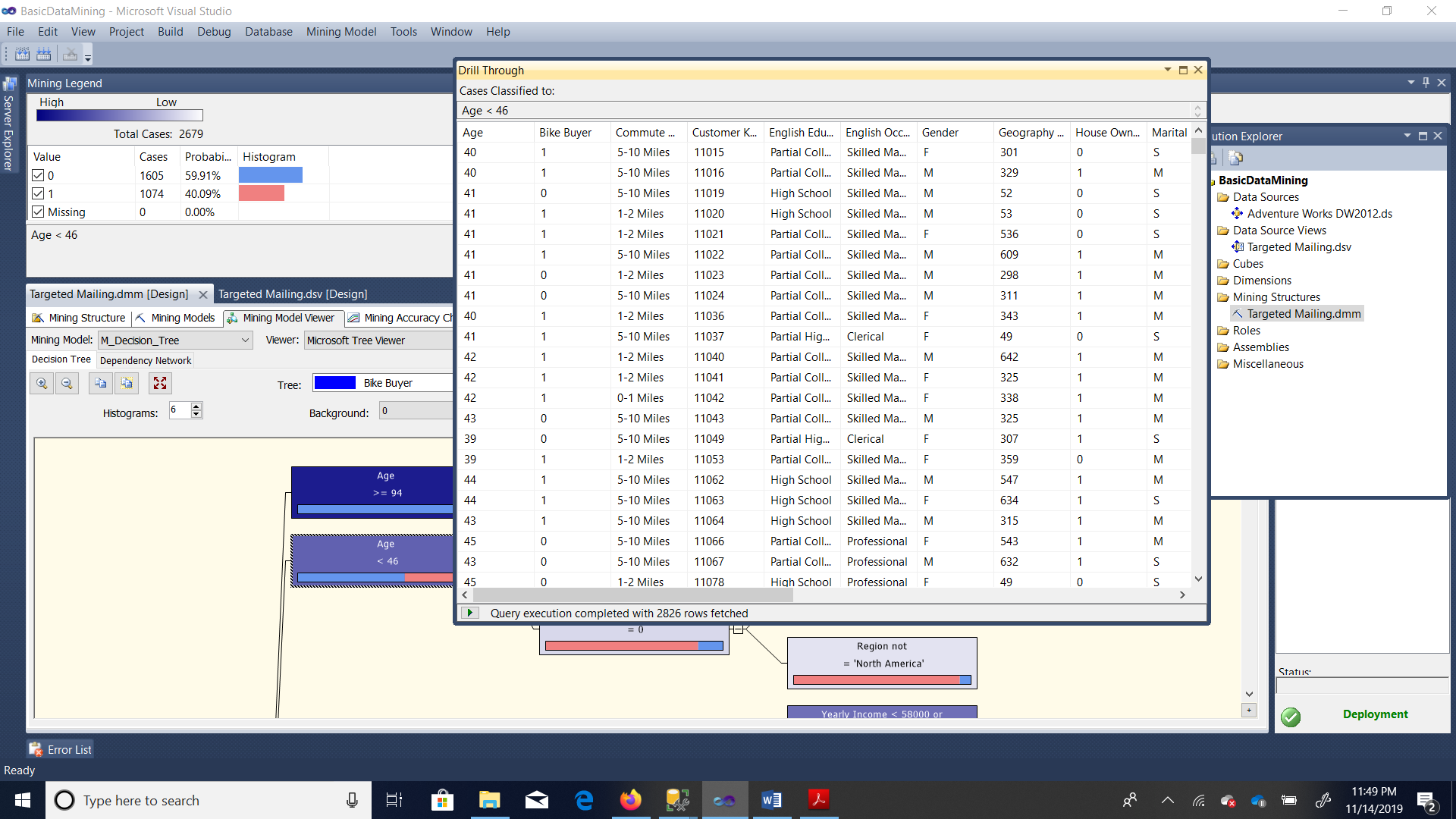


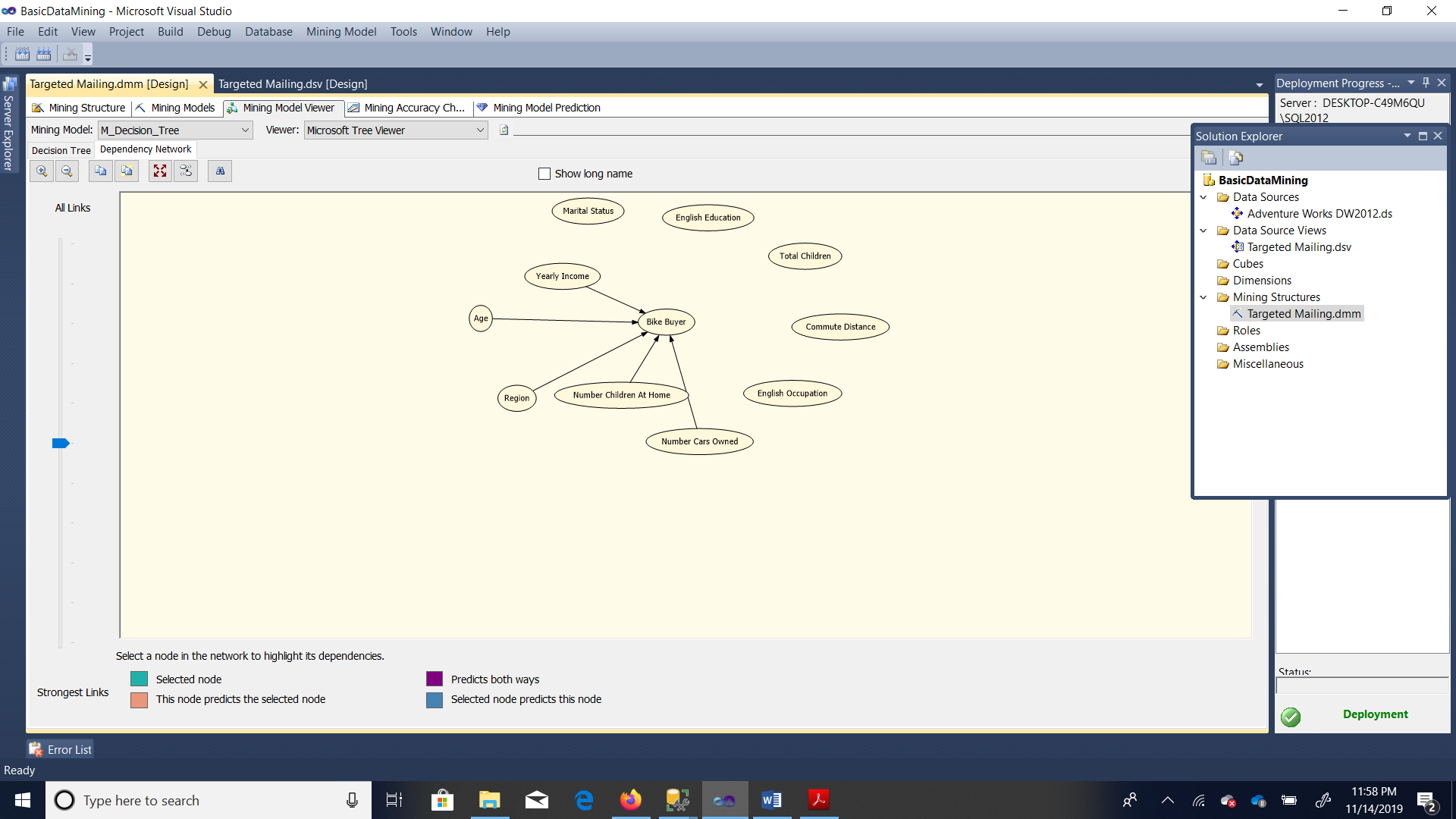
Lesson III

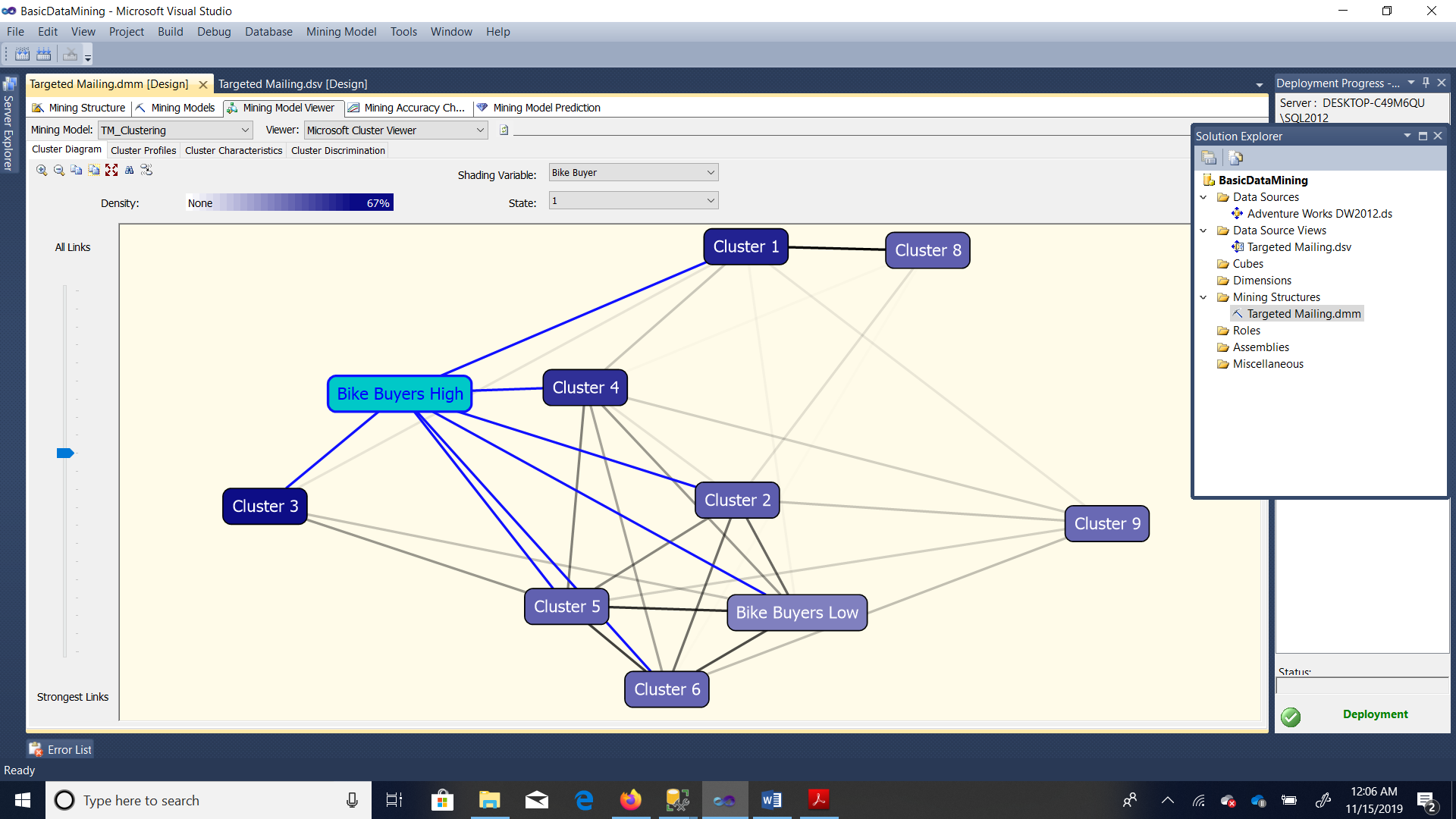


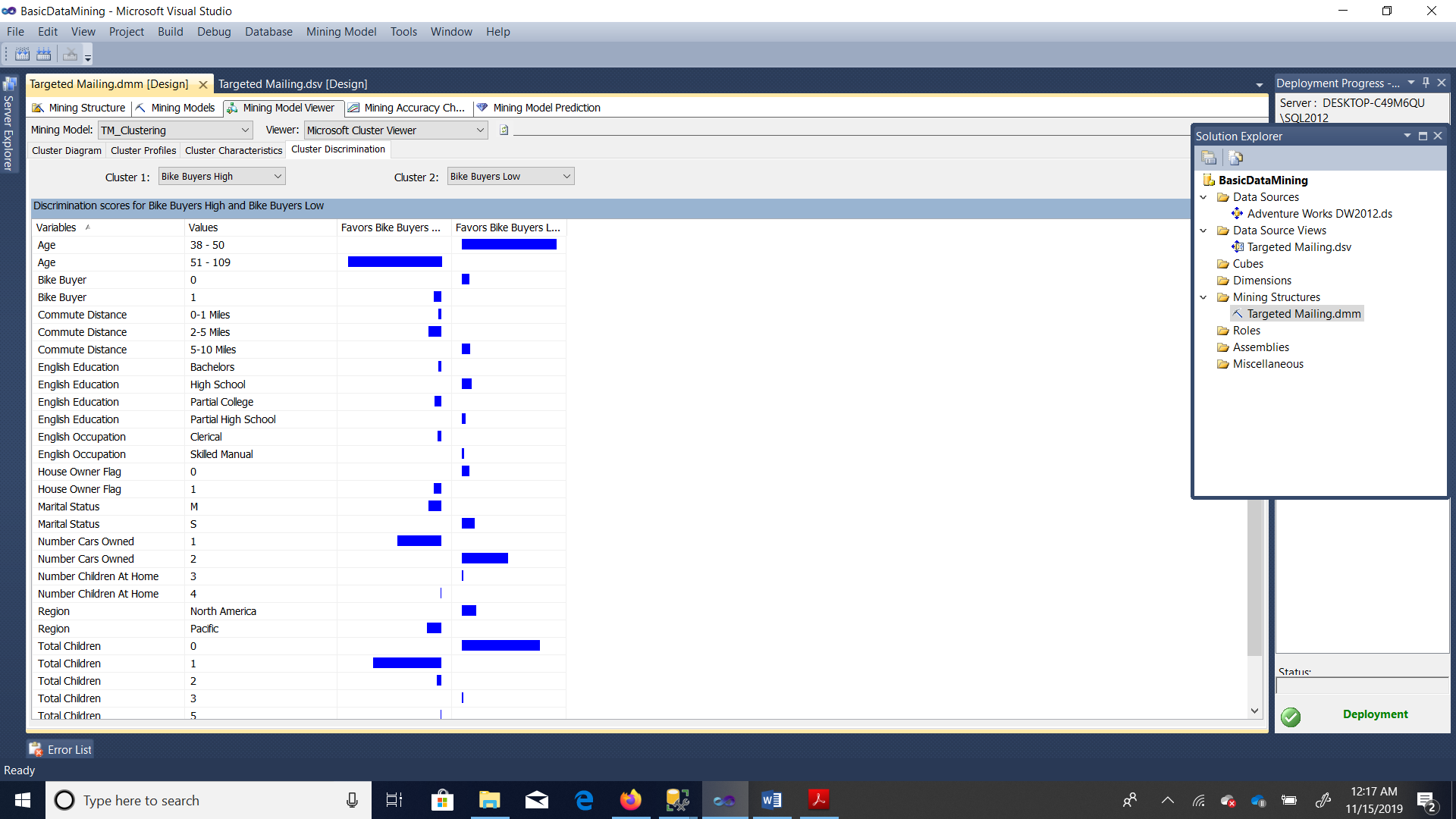
Lesson IV:

Decision Tree:

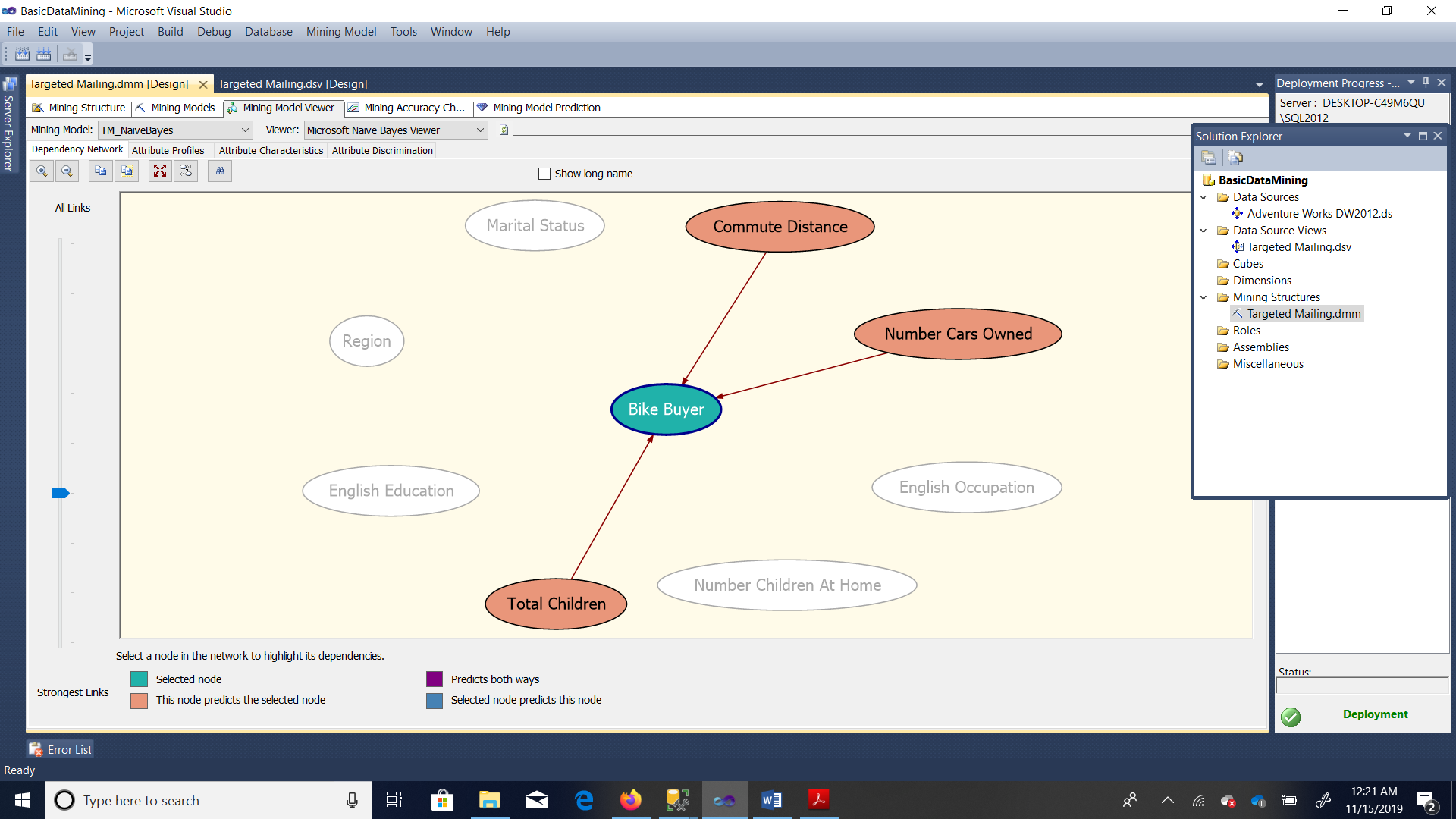




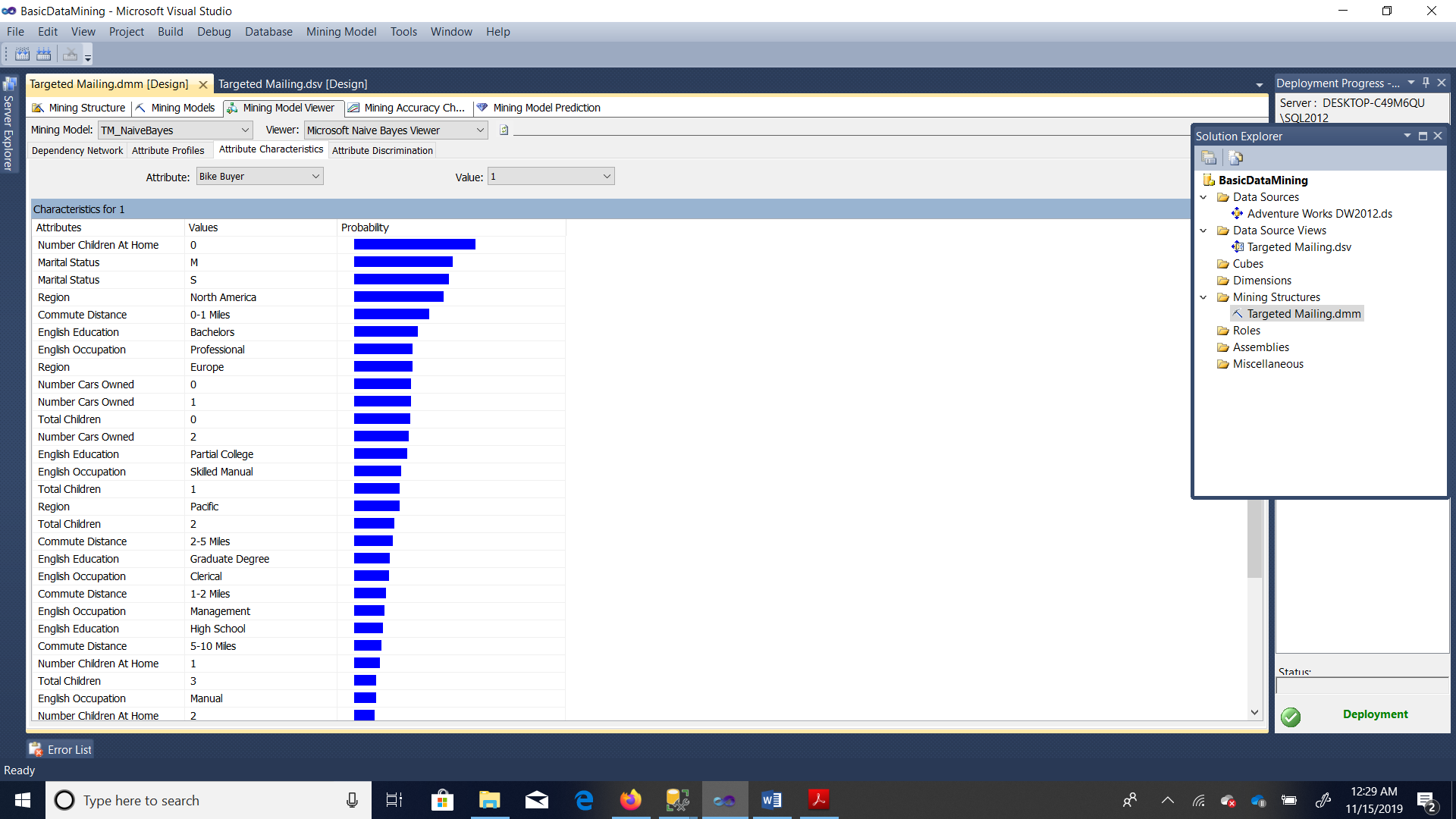




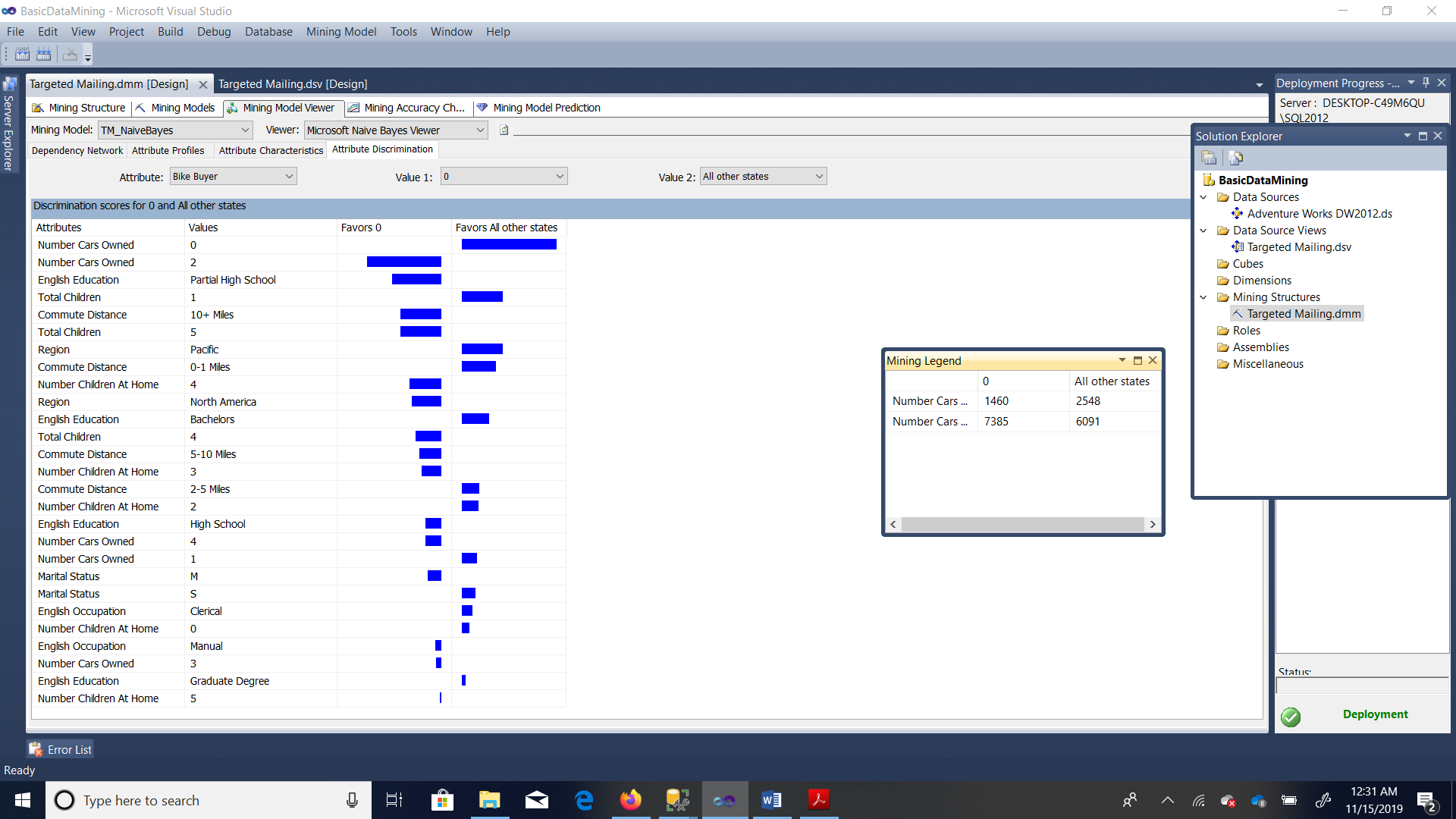
Naïve beyes

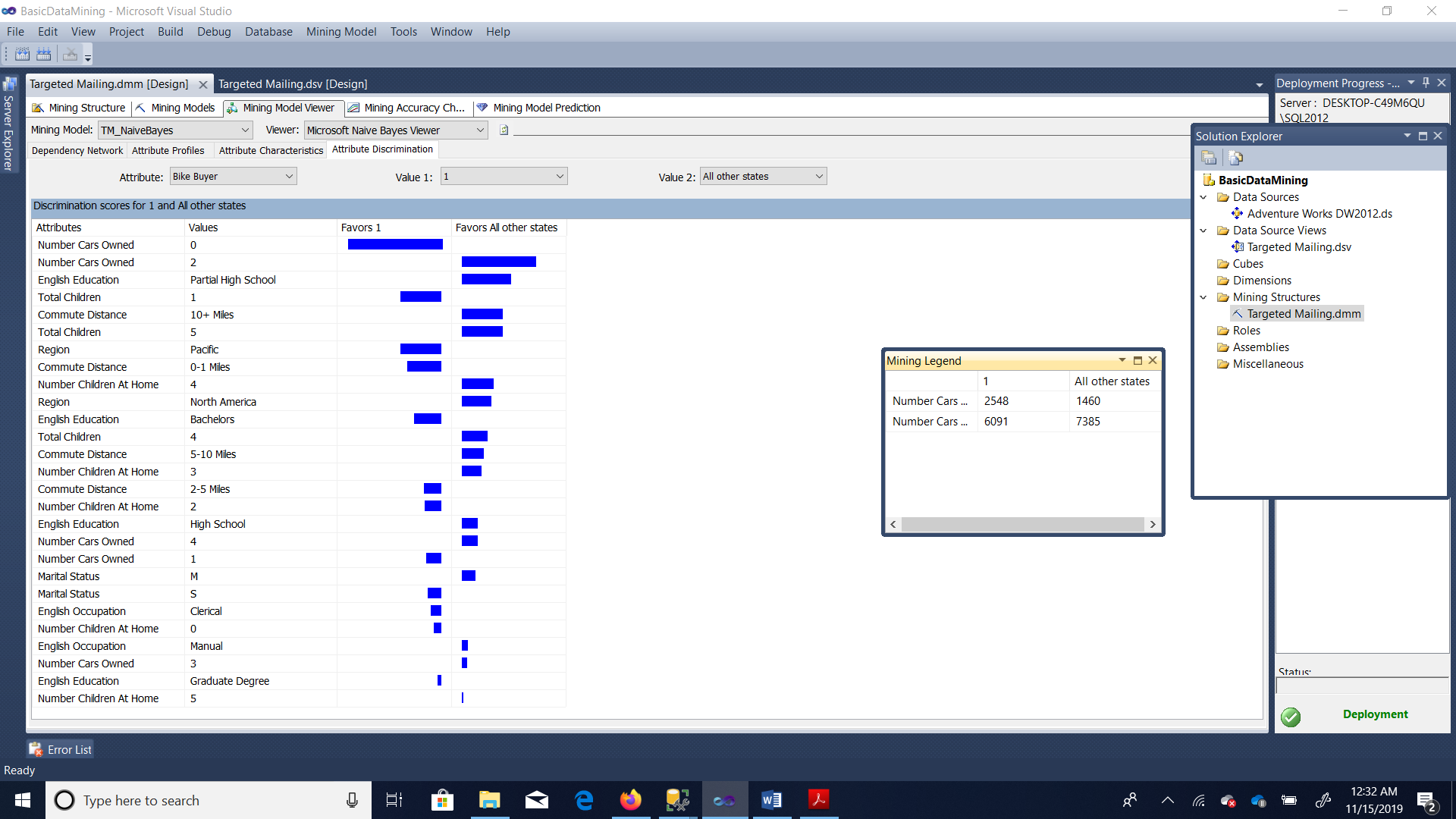


Naïve Beyes Attribute Characteristics

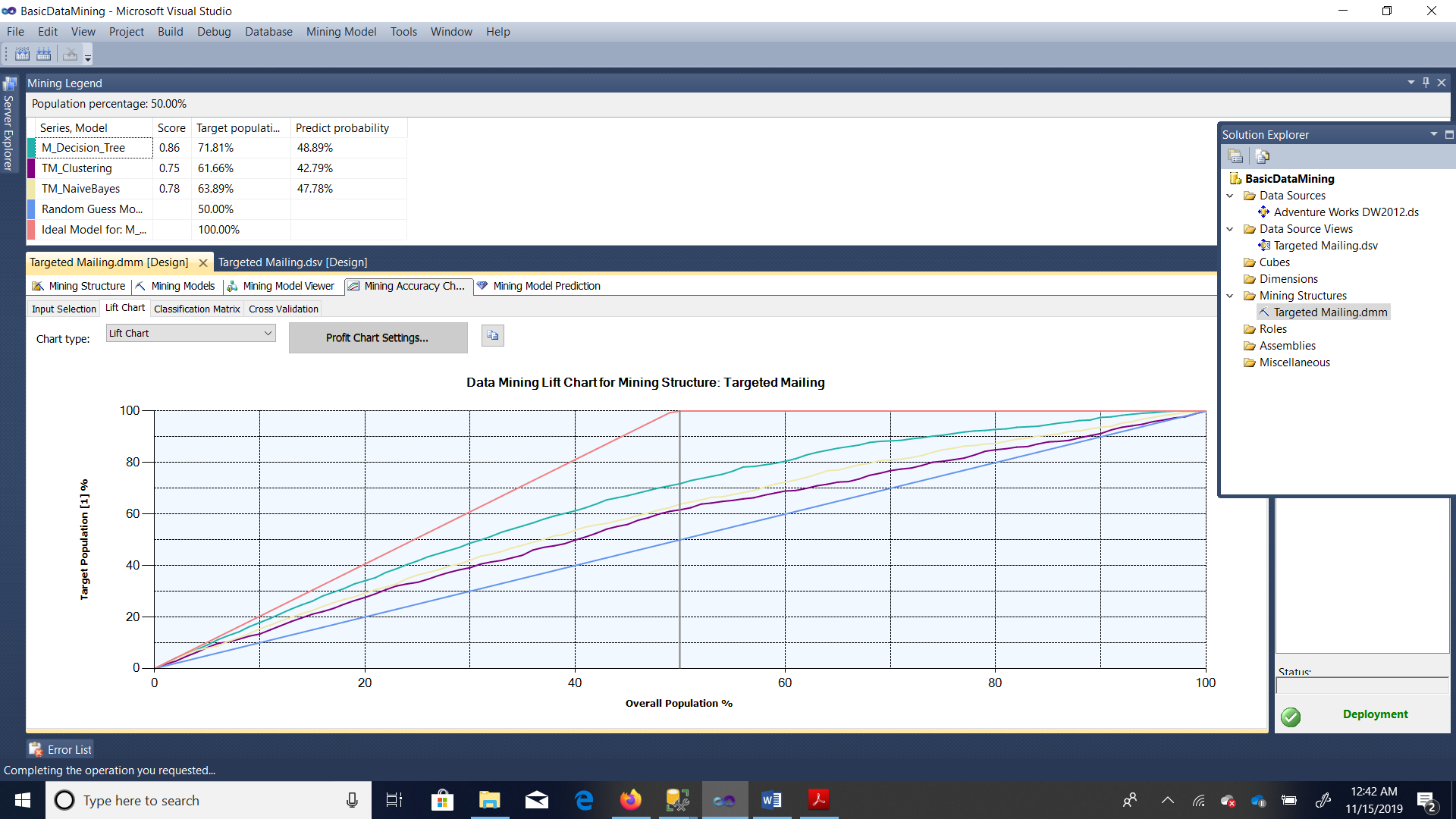


Naïve Beyes Attribute Discrimination

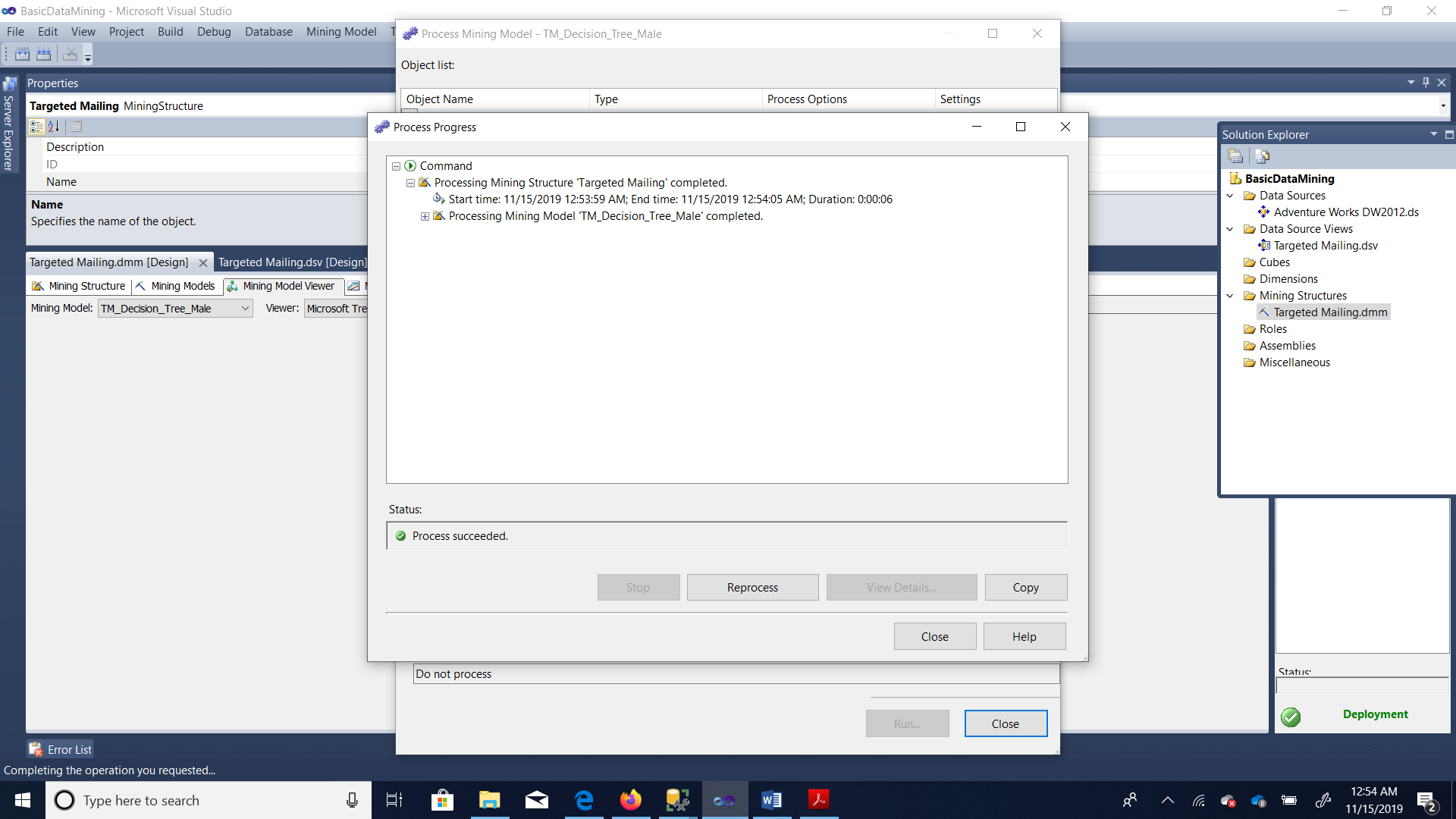




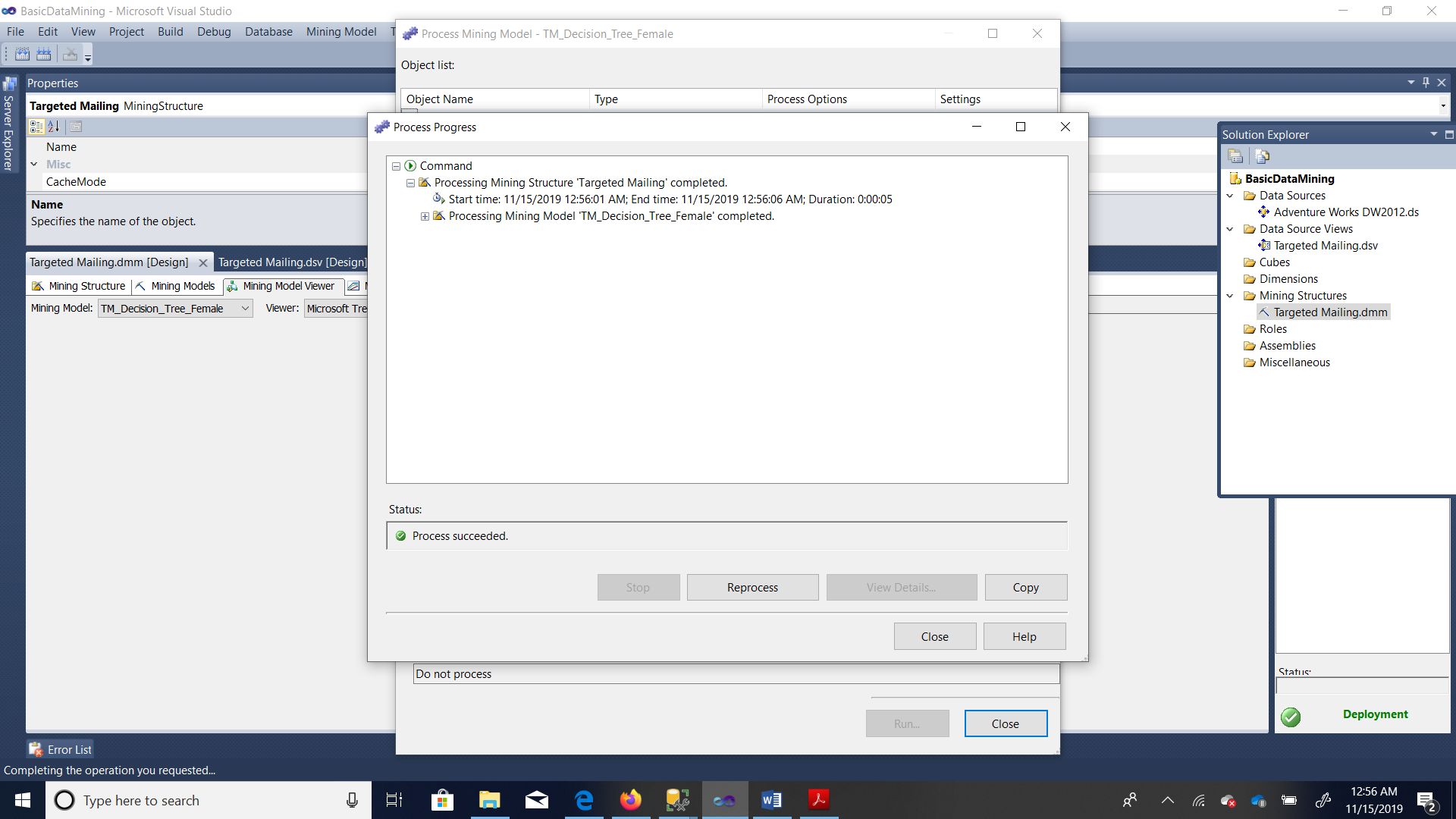
Lesson V:

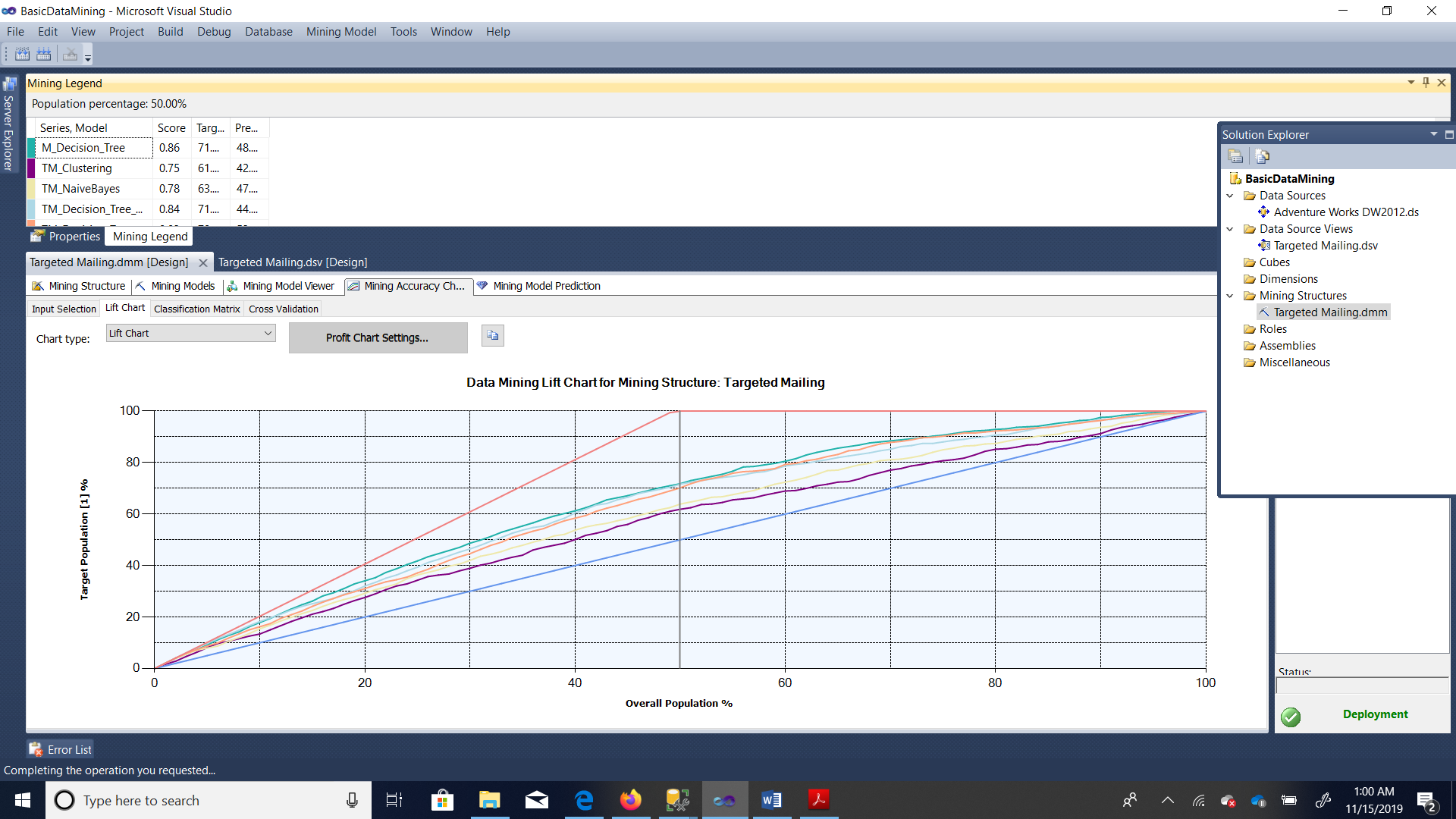


TM\_Decision\_Tree\_Male



TM\_Decision\_Tree\_female





Lesson Vi:

