Capture the data, then put them into excel.

1. Wrong value in class:

1. testedpositive should be tested\_positive.
2. testneg according to analysis it should be tested\_negative.

2. Wrong value in pedi:

1. 248 use “the most possible value to replace” should be 0.248
2. 875 use “the most possible value to replace” should be 0.875

3. Transfer the class into dummies (1=positive, 0=negative)

4. Handling missing value (Using Mode to fill missing values)

Missing value in column plas: filling by 99(column mode)

Missing value in column skin: filling by 0(column mode)

Missing value in column pedi: filling by 0.254(column mode)

Missing value in column age: filling by 22(column mode)

5. Binning the variable with bigger min and max difference. (plas, pres, skin, insu)

For all of them choose 8 bins and equal interval

6. Partition the data into Training(60%) and validation set(40%)

Q2.

There is a list of possible method to reduce the data, according to the characteristic our dataset I group them by applicable and unapplicable.

Applicable:

PCA (principal components analysis)

Do correlation analysis to remove variables that are strongly correlated. (There are no correlated variables in our dataset)

Unapplicable:

Reducing the number of categories in categorical variable (there is no categories in our dataset)

According to my own analysis, I think we do not have too many data records, (training 461, validation 307) and we do not need to apply PCA to our dataset. In addition, I apply a correlation matrix for all the variables.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | *column 1* | *column 2* | *column 3* | *column 4* | *column 5* | *column 6* | *column 7* | *column 8* |
| column 1 | 1 |  |  |  |  |  |  |  |
| column 2 | 0.12955 | 1 |  |  |  |  |  |  |
| column 3 | 0.141282 | 0.152024 | 1 |  |  |  |  |  |
| column 4 | -0.08167 | 0.057576 | 0.207371 | 1 |  |  |  |  |
| column 5 | -0.07353 | 0.331284 | 0.088933 | 0.436783 | 1 |  |  |  |
| column 6 | 0.017683 | 0.22124 | 0.281805 | 0.392573 | 0.197859 | 1 |  |  |
| column 7 | -0.0312 | 0.140048 | 0.042662 | 0.180716 | 0.184432 | 0.142191 | 1 |  |
| column 8 | 0.544226 | 0.263755 | 0.239471 | -0.11411 | -0.042 | 0.036232 | 0.033331 | 1 |

As we can see, column1 (pregnant times), and column 8 (age) are somehow correlated to each other, we can delete one of these two variable, but we only have 8 variables. Therefore, after all the analysis, I think we should just keep all the things of the dataset, and do no reduction on the dataset.

Q3.

Classification

Q4.

Q5.

For k-Nearest Neighbor algorithm: (Classification)

1. We first select all the 8 variables into the prediction group.
2. Put the class into output variable (because the class variable is the result).
3. Use the default cutoff, which is 0.5.
4. Set the number of k as 6, and score best k between 1 and 6. This step is in order to find the best K that balance between overfitting and ignoring the predictor information.

According to the result we can see the best K is 5, and the confusion table for training and validation data are shown as below.

|  |  |  |
| --- | --- | --- |
| **Confusion Matrix (training)** | | |
|  | **Predicted Class** | |
| **Actual Class** | **1** | **0** |
| **1** | 103 | 62 |
| **0** | 16 | 280 |

|  |  |  |
| --- | --- | --- |
| **Confusion Matrix (validation)** | | |
|  | **Predicted Class** | |
| **Actual Class** | **1** | **0** |
| **1** | 52 | 51 |
| **0** | 34 | 170 |

Error rate =16.92% Error rate =27.69%

Q6.

For Naïve Bayes algorithm: (Classification)

1. In order to run Naïve Bayes algorithm we have to bin the variables that have more than 100 distinct values, that means we have to bin column “mass”, and column “pedi” (still using 8 bins and equal interval). After binning them, we repartition the dataset.
2. Then, we select all the 8 variables into the prediction group.
3. Put the class into output variable (because the class variable is the result).
4. Use the default cutoff, which is 0.5

The confusion table for training and validation data are shown as below.

|  |  |  |
| --- | --- | --- |
| **Confusion Matrix (training)** | | |
|  | **Predicted Class** | |
| **Actual Class** | **1** | **0** |
| **1** | 113 | 52 |
| **0** | 36 | 260 |

|  |  |  |
| --- | --- | --- |
| **Confusion Matrix (validation)** | | |
|  | **Predicted Class** | |
| **Actual Class** | **1** | **0** |
| **1** | 51 | 52 |
| **0** | 56 | 148 |

Q7.

For Logistic Regression algorithm: (classification)

1. We first select all the 8 variables into the prediction group.
2. Put the class into output variable (because the class variable is the result).
3. Use the default cutoff, which is 0.5.
4. At the last step we choose covariance matrix of coefficients.

The confusion table for training and validation data are shown as below.

|  |  |  |
| --- | --- | --- |
| **Confusion Matrix (training)** | | |
|  | **Predicted Class** | |
| **Actual Class** | **1** | **0** |
| **1** | 98 | 67 |
| **0** | 35 | 261 |

|  |  |  |
| --- | --- | --- |
| **Confusion Matrix (validation)** | | |
|  | **Predicted Class** | |
| **Actual Class** | **1** | **0** |
| **1** | 66 | 37 |
| **0** | 37 | 167 |

Also according to the p-value below we can get that the mass (body mass index), and plas seems to have bigger contribution to the multiple linear regression model, which means they will influence the result more.

|  |  |
| --- | --- |
| **Input Variables** | **P-Value** |
| **Intercept** | 3.98E-19 |
| **preg** | 0.004826 |
| **mass** | 1.95E-07 |
| **pedi** | 0.009188 |
| **age** | 0.036413 |
| **Binned\_plas** | 5.34E-12 |
| **Binned\_pres** | 0.040209 |
| **Binned\_skin** | 0.615827 |
| **Binned\_insu** | 0.604477 |

Q8.

For CART (classification) Classify – Classification tree – Single tree

1. We first select all the 8 variables into the prediction group.
2. Put the class into output variable (because the class variable is the result).
3. Use the default cutoff, which is 0.5.
4. Set the minimum number of record in the terminal node as default.
5. Choose maximum number of levels to display equals to 7.

According to the full tree we can know that the first three important predictor are: plas, mass, age, and preg (sequence is according to the importance). The importance sequence for the predictor we get from CART algorithm is similar with the result we got from Logistic Regression algorithm.

The confusion table for training and validation data are shown as below.

|  |  |  |
| --- | --- | --- |
| **Confusion Matrix (training)** | | |
|  | **Predicted Class** | |
| **Actual Class** | **1** | **0** |
| **1** | 141 | 24 |
| **0** | 54 | 242 |

|  |  |  |
| --- | --- | --- |
| **Confusion Matrix (validation)** | | |
|  | **Predicted Class** | |
| **Actual Class** | **1** | **0** |
| **1** | 80 | 23 |
| **0** | 68 | 136 |

Q9.

Tune the algorithm K-NN

My choice to tune the K-NN algorithm is to change value of K to balance between overfitting and ignoring the predictor information. In Q3, I use 6 neighbors and the best k was 5, this time I am going to use double time of the neighbors. The process of XLMiner is shown as below:

1. We first select all the 8 variables into the prediction group.
2. Put the class into output variable (because the class variable is the result).
3. Use the default cutoff, which is 0.5.
4. Set the number of k as 12, and score best k between 1 and 12. This step is in order to find the best K that balance between overfitting and ignoring the predictor information.

|  |  |  |
| --- | --- | --- |
| **Confusion Matrix (validation)** | | |
|  | **Predicted Class** | |
| **Actual Class** | **1** | **0** |
| **1** | 62 | 41 |
| **0** | 37 | 167 |

As the result shown that, the best k this time is 10, the confusion table for training and validation data are shown as below.

|  |  |  |
| --- | --- | --- |
| **Confusion Matrix (training)** | | |
|  | **Predicted Class** | |
| **Actual Class** | **1** | **0** |
| **1** | 109 | 56 |
| **0** | 36 | 260 |

Error rate ==19.96% Error rate ==25.41%

To compare with Q3 (best k=5), this time we got the error rate for validation set equals to 25.41%, which is smaller than Q3 (27.69%), although for the training set the error rate rise a little bit, but the prediction capacity for the model has been increased.

Q10.

Tune the algorithm Naïve Bayes

The basic idea of tuning the NB algorithm is changing the bins number or the binning method.

First, I tried to bin all the variables except age and pregnancy time into 12 bins and the method is equal interval. The result shows that the error rate for training and validation set are all bigger than before.

Second, I tried to bin into 5 bins and the method is equal interval. The result shows that the error rate for validation set is still bigger than before.

|  |  |  |
| --- | --- | --- |
| **Confusion Matrix (validation)** | | |
|  | **Predicted Class** | |
| **Actual Class** | **1** | **0** |
| **1** | 55 | 48 |
| **0** | 55 | 149 |

Third, this time I tried to bin all 8 variables into 8 bins and using equal interval. In this time the error rate for training set is bigger than before, but the error rate for validation set is smaller than before, although the error rate for training rise a little bit but the prediction capacity for the model has been increased. The confusion table for training and validation data are shown as below.

|  |  |  |
| --- | --- | --- |
| **Confusion Matrix (training)** | | |
|  | **Predicted Class** | |
| **Actual Class** | **1** | **0** |
| **1** | 107 | 58 |
| **0** | 39 | 257 |

Error rate: 21.04% Error rate: 33.55%

Q11.

Tune the algorithm Logistic Regression

The basic idea of tuning the Logistic Regression algorithm can divided into two parts first is build the correlation matrix for all the variables, and then try to delete one of them and make a comparison with the error rate to see which one is better.

According to the correlation matrix which I built in question 2, we can get that, the pregnant times, and ages are somehow correlated to each other.

1. Delete the pregnant times column. The confusion table for training and validation data are shown as below. (The basic steps are same with before)

The result shows that in this way, the error rate will increase.

|  |  |  |
| --- | --- | --- |
| **Confusion Matrix (validation)** | | |
|  | **Predicted Class** | |
| **Actual Class** | **1** | **0** |
| **1** | 66 | 37 |
| **0** | 32 | 172 |

1. Delete the age column. The confusion table for training and validation data are shown as below. (The basic steps are same with before)

|  |  |  |
| --- | --- | --- |
| **Confusion Matrix (training)** | | |
|  | **Predicted Class** | |
| **Actual Class** | **1** | **0** |
| **1** | 101 | 64 |
| **0** | 28 | 268 |

Error rate=19.96% Error rate=22.47

To compare with the algorithm without tune in Q7, the error rate in bot training and validation are reduced.

Q12.

Tune the algorithm CART

The main idea of tuning the CART algorithm is using the best pruned tree, in this way it will not only improve the model’s efficiency, but also can avoid of overfitting.

1. We first select all the 8 variables into the prediction group.
2. Put the class into output variable (because the class variable is the result).
3. Use the default cutoff, which is 0.5.
4. Set the minimum number of record in the terminal node to 1, and choose prune tree (in case of overfitting).
5. Choose maximum number of levels to display equals to 7, and choose the best pruned tree (make the model more efficient and in this way could save a lot of time when the data’s scale is very big).

According to the result, the best pruned is 4 (has 4 decision nodes), and min error tree is 5 (has 5 decision nodes). In addition, according to the best pruned tree we can know that the first four important predictor are: plas, mass, age, and pedi (sequence is according to the importance). The importance sequence for the predictor we get from CART algorithm is same from which we got from Logistic Regression algorithm.

The confusion table for training and validation data are shown as below.

|  |  |  |
| --- | --- | --- |
| **Confusion Matrix (training)** | | |
|  | **Predicted Class** | |
| **Actual Class** | **1** | **0** |
| **1** | 165 | 0 |
| **0** | 0 | 296 |

Error rate =0

|  |  |  |
| --- | --- | --- |
| **Confusion Matrix (validation)** | | |
|  | **Predicted Class** | |
| **Actual Class** | **1** | **0** |
| **1** | 57 | 46 |
| **0** | 29 | 175 |

Error rate=24.43

To compare with the method before, the error rate for both validation and training set are reduced. However, this time the error rate of training set is 0 therefore, it seems a little bit overfitting.

Q13.

In order to find the best algorithm and its result for this dataset, in my opinion, we should first think about the characteristics of our dataset. Our dataset has 769 records in total, which means the scale is not too big, and it has 8 variables, also not very much, and all of the variables are numerical variables. Second, the purpose of the data mining process is to classify a patient into 2 groups (sick or not sick), and finding which predictor is important to classify the patient. In order to evaluate the performance of each algorithm to our dataset, I create a table as below.

|  |  |  |  |
| --- | --- | --- | --- |
|  | small scale dataset | numerical variables | relation between variables |
| KNN | works better when we got large training dataset | works good | works normal |
| Naïve Bayes | Need large scale of dataset to improve accuracy | better works for categorical data | works normal |
| CART | Need large scale of dataset to build more accurate model | works fine | likely miss relationships between predictors |
| logistic Regression | works fine | works good | emphasis the relationship between predictors(variables) and response(classify groups) |

According to the table and the error rate, I would recommend the logistic Regression’s (after tuning) result. It’s not only resulting lesser mistakes, but also works fine for our dataset.