# Assignment 1

Vincent de Vos (0741795)
v.j.h.d.vos@student.tue.nl

Thom Hurks (0828691)
t.p.hurks@student.tue.nl

*March 2, 2015*

**Explain and motivate how you will use the available data to create the model.**

The available data contains the necessary labels, namely the class of an observation: either a transaction is fraudulent (label is 1) or it isn't fraudulent (label is 0). Therefore we can create our model using supervised learning.

First we looked into our data and have established that the data is balanced. We can see this inspecting the frequency of the classes in our data, which are distributed as follows.

| **Class** | **Frequency** | **Percentage** |
|---|---|---|
| Fraudulent: No | 4668 | 65.4% |
| Fraudulent: Yes | 2469 | 34.6% |
| **Total** | 7137 | 100% |

**Table 1: Data distribution.**
This is good distribution for which there is no need to balance this data.

We will then split up our original data set as follows:

- Cross validation data (80%) - this data is used to actually train and validate our model
- Final test data (20%) - used to validate the final output of our model

Since a bigger data set for training will yield a better model we picked our cross validation set to be quite big and our final test set quite small. We chose this division ratio because the emphasis for this assignment is primarily focussed on the performance of our model, which will now be improved by picking a bigger dataset that will be used to train our model. The holdout percentage of 15%-to-20% is also recommended in the course slides as an empirically good value. This initial partitioning is also stratified, to ensure both partitions have representative data.

Then we will use our cross validation dataset to train a model using the stratified ten-fold cross validation method. This method allows us to create a model with an accurate estimate because the stratification reduces the estimate's variance and because with ten-fold cross-validation all predictors are used for both training and validation, and all predictors are used for validation only once.

Extensive experiments have shown that stratified ten-fold cross-validation is the best choice to get an accurate estimate, as explained in the course slides. Yet another reason why ten-fold stratified cross validation is a good idea is that our dataset is not that big. With only 7137 observations from which 2469 are fraudulent. Using ten-fold cross validation we prevent ourselves from having to choose a fixed training subset which can be a not representative of the entire dataset or unseen values.

**Explain and motivate how you will evaluate the performance of the model.**

Using stratified ten-fold cross validation our cross validation dataset is split into 10 equally sized subsets. Of these 10 subsets a single subset is used as validation dataset for testing the model while the remaining 9 subsets are used as training data. The cross-validation process is then repeated 9 more times in the same way, with each of the 9 subsamples used exactly once as the validation dataset. The loss values of the 10 resulting models will then be averaged to give a single loss (classification error) estimation.

As explained in the question about the minimal number of leaf nodes, this ten-fold stratified cross validation procedure is actually run multiple times with different parameters for the model building procedure, in order to select the models parameters which yield the lowest average ten-fold loss.

Using this method instead of a repeated random sub-sampling validation method gives us the advantage that all observations are used for both training and validation while each observation is used for validation exactly once. This gives us a good grasp on the way our model is performing on our data set.

After iterating the ten-fold stratified cross validation procedure many times (we chose 200 iterations) to select the tree generation settings which yielded the lowest ten-fold loss, we generate the final tree by taking the entire training set which is 80% of the total dataset and use it with the best discovered settings to generate a final model. The advantage of this approach is that using more data for the final tree will give us a better model than if we had used less data. We still have the final test set of 20% available to measure the performance of this final tree.

We measure the performance by taking the 20% final test set and use our model to predict the label for the predictors in that set. We can compare the predictions with the actual labels and construct a confusion matrix with true positives, false negatives, etc.

We can then use the confusion matrix and a simple 0-1 loss function to calculate various interesting metrics, such as overall success rate, error, precision, recall and F1. For all those metrics it holds that a value close to 1 is very good and a value close to 0 is very bad, except for the error where it's the other way around.

**Decision tree model can accept some parameters. One of them is the minimal number of data points in the leaf node. Explain which values of this parameter you will consider and how those values will influence the model.**

In order to find the optimal number of leaf nodes we run the stratified ten-fold cross validation for every element in a linear set of elements ranging from 1 to 500, with a total of 200 samples. This is basically a brute-force method to determine the optimal number of leaf nodes. Experimentation found that beyond 500 min leaf size, the error only went up very fast and more than 200 samples yielded a very long running time, which is why these values are considered optimal in our situation.

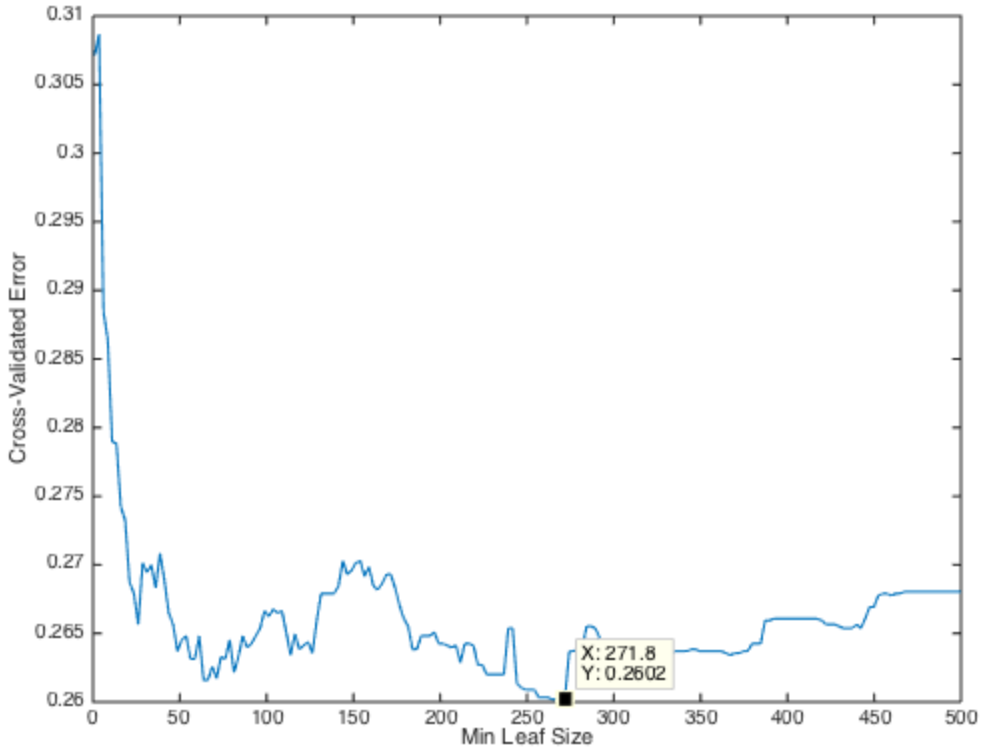


**Figure 1.** Number of leaf nodes with corresponding cross validation estimation error

You can see in the figure above that with a leaf size of 272 (rounded up from 271.8) that we have an estimated error of 0.2602 which is the smallest estimation error among all the other leaf sizes. Now that we know this optimal leaf size we will use this leaf size when we train our model, which will enhance the performance of our model.

**Describe and interpret the best decision tree model you obtained. What is the quality of this model? And its accuracy.**

The quality of our final model is expressed by the following metrics

| | |
|---|---|
| Overall success rate | 0.7239 |
| Error rate | 0.2761 |
| Precision rate | 0.6266 |
| Recall | 0.4970 |
| F1 | 0.5543 |

**Table 2: Quality metrics of the tree.**

From the overall success rate and error we can see that most of our predictions were actually correct, so we can consider the tree as a whole to be satisfactory.
The precision value means that of the fraud predictions, most were actual frauds, which is good. However, ideally the value would be much higher as we do not like false positives as customers could be wrongfully accused of fraud.
The recall means that of all actual frauds, we found more than half, which is pretty good, although we would of course like to discover all frauds.
The F1 measure is the harmonic mean of precision and recall, but we do not know exactly what a good threshold value where F1 can be considered good would be.

Beyond that, we can visually observe that the tree is relatively simple, which means it will likely generalize and give good predictions on other unseen data in the future.

Our final trained model yields a decision tree as shown in figure 2 below. Note that the annotations used in the tree correspond as follows with our observations

| **Tree annotation** | **Observation feature** |
|---|---|
| x1 | TL: Time elapsed since the last transaction |
| x2 | TT: Time elapsed from the first transaction |
| x3 | VAL: Transaction value |
| 0 and 1 | Non-fraud and fraud, respectively. |

**Table 3: Tree legend.**

You can see immediately that any transactions that have a elapsed time since the last transaction (TL) of more than 54.2143 are immediately labeled as non-fraudulent. Only when the TL is below 54.2143 the tree can label an observation as fraudulent.

Furthermore we see that Time elapsed from first Transaction (TT) is not of much influence on labeling an observation, it only occurs one time within the deepest level of our tree. However you can see that the decisions are mainly guided by TL and the transaction value. In fact any transaction value above 0.6795 will be labeled as a fraudulent once the TL falls below 54.2143. This means our tree will label all “high value” transactions that occur “too soon” as fraudulent and low-valued transactions < 0.34 as non-fraudulent. If the transaction value is then still of “medium value” then further distinctions are made; higher TL then means fraud, otherwise a higher value or TT can still indicate fraud. It is obvious that the most fine-grained distinctions are made in the bottom of the tree.

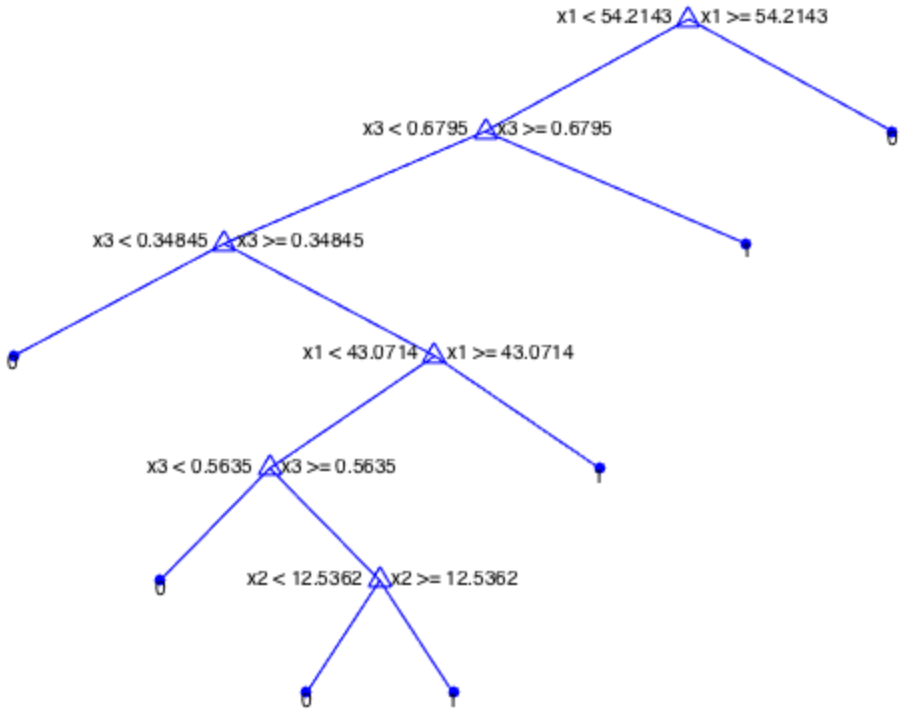


**Figure 2.** Decision tree of our final trained model

In the table below we’ve summed up the parameters that are influential for making a decision. The predictor importance is a measure that can be extracted from the Tree model in MatLab and estimates the predictor importance for the tree by summing changes in the mean squared error due to splits on every predictor and dividing the sum by the number of branch nodes. We can nicely see that the computed predictor importance matches our observations of the tree.

| Measure | TL | TT | VAL |
|---|---|---|---|
| # Decisions in Tree | 2 | 1 | 3 |
| Predictor Importance | 0.0132 | 0.0646e-04 | 0.0044 |

**Table 4: Tree decision metrics.**