Assignment 3 – Solution

Machine Learning
MSc Business Analytics

Wolfram Wiesemann

1 Individual Assignment

For the tree with the root node only, the purity is

$$H(R) = -\frac{11}{25}\log_2\left(\frac{11}{25}\right) - \frac{14}{25}\log_2\left(\frac{14}{25}\right) = 0.989.$$

If we split on the 'day' predictor, we get

$$\begin{array}{c}\text{weekday} \to 13 \text{ no, } 7 \text{ yes;}\\ \text{weekend} \to 1 \text{ no, } 4 \text{ yes.}\end{array}$$

The information gain is

$$\begin{aligned}0.989 + &\frac{20}{25}\left(0.65 \cdot \log_2(0.65) + 0.35 \cdot \log_2(0.35)\right)\\ + &\frac{5}{25}\left(0.8 \cdot \log_2(0.8) + 0.2 \cdot \log_2(0.2)\right) = 0.097.\end{aligned}$$

If we split on the 'weather' predictor, we get

$$\begin{array}{c}\text{rainy} \to 1 \text{ no, } 6 \text{ yes;}\\ \text{sunny} \to 13 \text{ no, } 5 \text{ yes.}\end{array}$$

The information gain is

$$\begin{aligned}0.989 + &\frac{7}{25}\left(0.14 \cdot \log_2(0.14) + 0.86 \cdot \log_2(0.86)\right)\\ + &\frac{18}{25}\left(0.277 \cdot \log_2(0.277) + 0.723 \cdot \log_2(0.723)\right) = 0.21.\end{aligned}$$

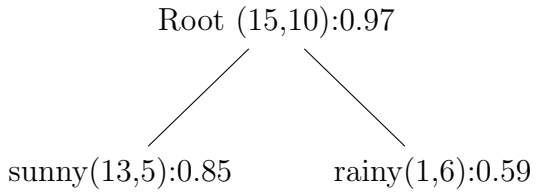
If we split on the 'time' predictor, we get

$$\begin{array}{c}\text{8am} \to 7 \text{ no 5, yes;}\\ \text{1pm} \to 7 \text{ no, } 6 \text{ yes.}\end{array}$$

The information gain is

$$0.989 + \frac{12}{25}\left(0.42 \cdot \log_2(0.42) + 0.58 \cdot \log_2(0.58)\right) + \frac{13}{25}\left(0.53 \cdot \log_2(0.53) + 0.47 \cdot \log_2(0.47)\right) = 0.056.$$

We therefore first split on the 'weather' predictor and obtain:



Here, the notation of the nodes is "predictor value (# of samples with no traffic, # of sample with traffic) : Impurity". Now let's split the left leaf node first.[1] The purity of this node is

$$-\frac{13}{18}\log_2\left(\frac{13}{18}\right) - \frac{5}{18}\log_2\left(\frac{5}{18}\right) = 0.85.$$

If we split on the 'day' predictor, we obtain

$$\begin{array}{c}\text{weekday} \to 13 \text{ no, } 2 \text{ yes;} \\ \text{weekend} \to 0 \text{ no, } 3 \text{ yes.}\end{array}$$

The information gain is

$$0.85 + \frac{15}{18}\left(0.14 \cdot \log_2(0.14) + 0.86 \cdot \log_2(0.86)\right) = 0.37.$$

Similarly, if we split on the 'time' predictor, we obtain

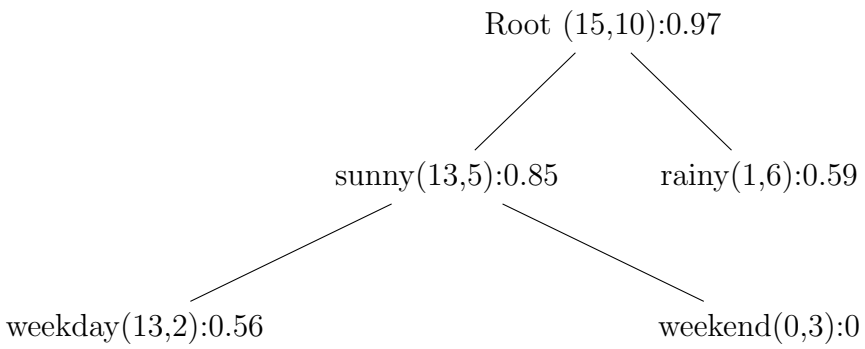
$$\begin{array}{c}\text{8am} \to 6 \text{ no, } 3 \text{ yes;} \\ \text{1pm} \to 7 \text{ no, } 2 \text{ yes.}\end{array}$$

The information gain is

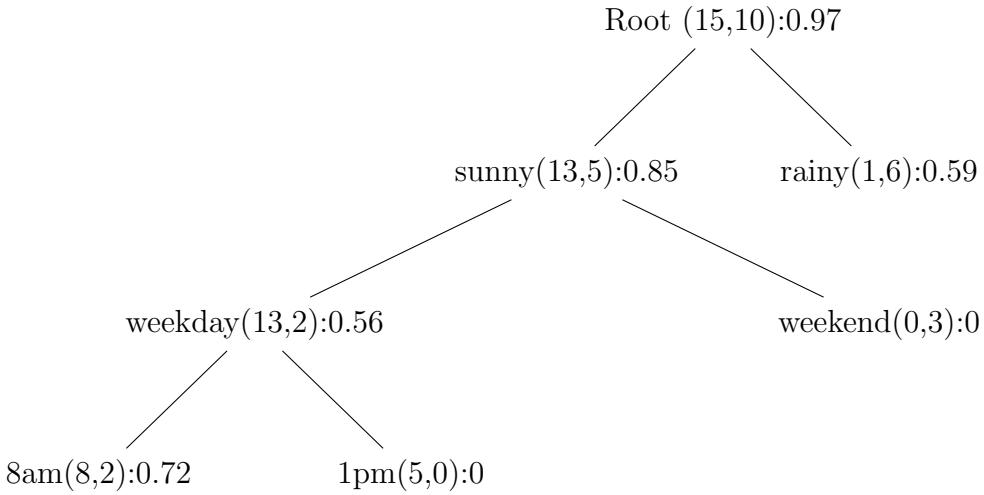
$$0.85 + \frac{9}{18}\left(0.66 \cdot \log_2(0.66) + 0.33 \cdot \log_2(0.33)\right) + \frac{9}{18}\left(0.28 \cdot \log_2(0.28) + 0.72 \cdot \log_2(0.72)\right) = 0.011.$$

Therefore we split on the 'day' predictor:

[1] Note that since we are growing a complete tree, we do not have to select the node with the best split – each 'splittable' node will be split anyway.



We can now split the leftmost leaf node on the remaining 'time' predictor:



Finally we need to decide how to split the 'rainy(1,6):0.59' node. The purity of the node is 0.59. If we split on the 'day' predictor, we obtain

$$\begin{array}{c}\text{weekday} \rightarrow 0 \text{ no, } 5 \text{ yes;} \\ \text{weekend} \rightarrow 1 \text{ no, } 1 \text{ yes}\end{array}$$

with an information gain of

$$0.59 + \frac{2}{7}\left(0.5 \cdot \log_2(0.5) + 0.5 \cdot \log_2(0.5)\right) = 0.274.$$

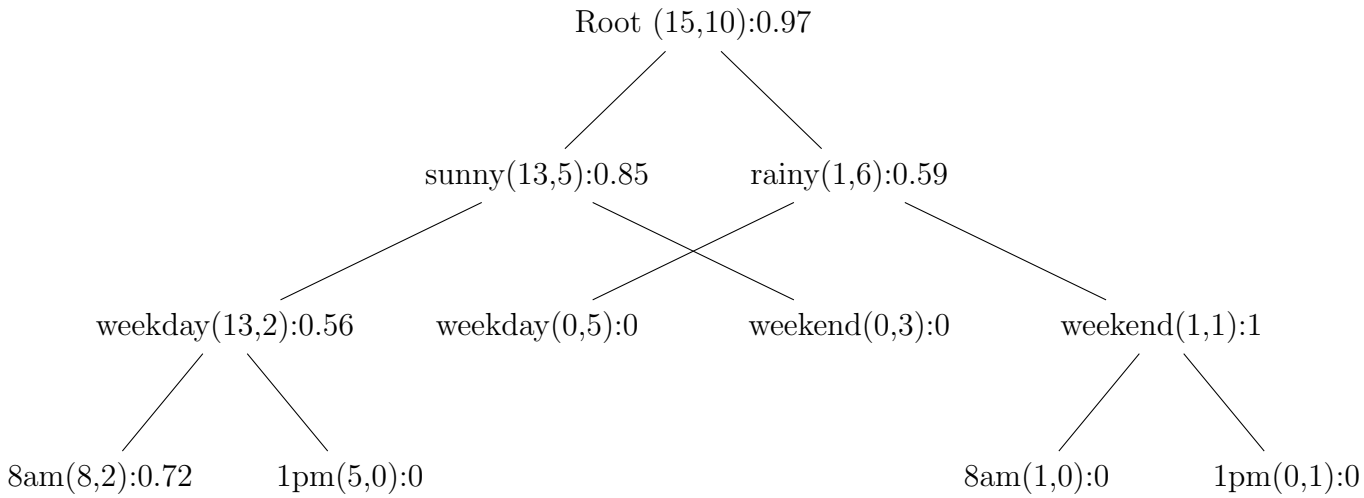
If we split on the 'time' predictor, we obtain

$$\begin{array}{c}\text{8am} \rightarrow 1 \text{ no, } 2 \text{ yes;} \\ \text{1pm} \rightarrow 0 \text{ no, } 4 \text{ yes}\end{array}$$

with an information gain of

$$0.59 + \frac{3}{7}\left(0.66 \cdot \log_2(0.66) + 0.33 \cdot \log_2(0.33)\right) = 0.2.$$

We thus first split on the 'day' predictor and then on the 'time' predictor:



The tree misclassifies 2 'traffic' samples out of the 25 training examples. The sensitivity is 2/11, and the specificity is 14/14.

The predictions on the test set are:

day	**weather**	**time**	**traffic**	**prediction**
weekend	rainy	8am	no	no
weekday	sunny	8am	yes	no
weekend	sunny	1pm	yes	yes
weekday	sunny	8am	no	no
weekend	sunny	1pm	yes	yes
weekday	rainy	8am	no	yes
weekday	sunny	8am	yes	no
weekday	sunny	1pm	no	no
weekday	sunny	1pm	no	no
weekday	sunny	1pm	no	no
weekend	rainy	1pm	yes	yes
weekday	sunny	8am	yes	no
weekday	sunny	1pm	no	no
weekday	rainy	1pm	yes	yes
weekday	sunny	1pm	no	no

The classification tree misclassifies 4 out of the 15 samples. The sensitivity is 4/7, and the specificity is 7/8.

2 Group Assignment

1. *Import the pre-processed data set in R. Shuffle the records and split them into a training set (20,000 records), a validation set (8,000 records) and a test set (all remaining records).*

This can be done as follows:

```
credit = read.csv ("Loans_processed.csv")
train_sample = sample (37226, 20000)
credit_train = credit[train_sample,]
credit_remain = credit[-train_sample,]
val_sample = sample (17226, 8000)
credit_val = credit_remain[val_sample,]
credit_test = credit_remain[-val_sample,]
```

2. *Using a classification tree (look at the* `C50` *library), try to predict with an accuracy greater than* $\frac{\#\text{ of repaid loans}}{\#\text{ of repaid loans} + \#\text{ of charged off loans}}$ *if a loan will be repaid. Do you manage to achieve this performance on the validation set? What about the training set?*

If the library is not already installed, we can install it with

```
install.packages ("C50")
```

and load it with

```
library (C50)
```

To train the decision tree and see how it performs in sample, we type:

```
credit_model = C5.0 (credit_train[-8], credit_train$loan_status)
credit_model
summary (credit_model)
```

The '-8' excludes from the input the 8th column, which is the outcome. Both on the training and the validation set, our decision tree coincides with the naïve majority predictor 'all loans will be paid off'. The misclassification rate is 14%, which is exactly the percentage of defaults in the training set.

3. *Experiment with different cost matrices to achieve a sensitivity (also known as* recall*) of approximately* 25%*,* 40% *and* 50% *in your validation set. Also report the percentage of the loans* $\frac{n_{11}}{n_{11}+n_{21}}$ *you would recommend to the bank for re-evaluation that were indeed charged off (also known as* precision*).*

Our initial solution has a sensitivity of 0%. To increase the sensitivity, we will increase the cost of false negatives and decrease the cost of false positives. For example, let's put a weight of 3 to a false negative, and let's see what we obtain in our train and validation sets:

```
c = matrix (1, 2, 2)
c[1,1] = 0
c[2,2] = 0
c[1,2] = 1
c[2,1] = 3
credit_model = C5.0 (credit_train[-8], credit_train$loan_status, costs = c)
summary (credit_model)
library (gmodels)
```

```
summary (credit_model)
credit_pred_train = predict (credit_model, credit_train[-8])
CrossTable (credit_train$loan_status, credit_pred_train)
credit_pred_val = predict(credit_model, credit_val)
CrossTable (credit_val$loan_status, credit_pred_val)
```

On the training set, we obtain

	Charged off (pred.)	**Paid (pred.)**
Charged off (act.)	1,199	1,601
Paid (act.)	2,045	15,155

On the validation set, we obtain

	Charged off (pred.)	**Paid (pred.)**
Charged off (act.)	333	819
Paid (act.)	946	5,902

Thus, in the validation set we have a sensitivity of $\frac{333}{333+819} = 0.289$ and a precision of 0.26. In the training set the sensitivity is 0.428, while the precision is 0.370. Using a cost multiplier of 5, we obtain

	Charged off (pred.)	**Paid (pred.)**
Charged off (act.)	2,072	728
Paid (act.)	5,128	12,072

on the training set and

	Charged off (pred.)	**Paid (pred.)**
Charged off (act.)	647	505
Paid (act.)	2,232	4,616

on the validation set. Thus, in the validation set we have a sensitivity of 0.562 and a precision of 0.225, while we have a sensitivity of 0.74 and a precision of 0.288 in the training set. We try a few more values and we report the sensitivity and the precision in the validation set:

cost coefficient	**sensitivity**	**precision**
1.4	0	–
1.8	0.06	0.337
2.2	0.087	0.318
2.6	0.221	0.29
3	0.289	0.26
4	0.401	0.242
5	0.562	0.225

4. *Pick a cost parameter matrix that you assess as the most appropriate for identifying loan applications that deserve further examination.*

This is a subjective choice – we decide to pick a cost coefficient of 3.

5. *Evaluate the performance of your cost parameter matrix on the test set.*

We evaluate our classifier on the test set with

```
credit_pred = predict (credit_model, credit_testing)
CrossTable (credit_testing$loan_status, credit_pred)
```

We obtain the following confusion matrix:

	Charged off (pred.)	**Paid (pred.)**
Charged off (act.)	397	1,083
Paid (act.)	1,179	8,012

The sensitivity is 0.268, and the precision is 0.252.