

# CSE 446: Machine Learning Winter 2018

## Assignment 1

from  
Lukas Nies  
University of Washington

01/18/18

## Contents

<b>0</b>	<b>Policies</b>	<b>1</b>
0.1	List of Collaborators . . . . .	1
0.2	List of Acknowledgments . . . . .	1
0.3	RTFM . . . . .	1
<b>1</b>	<b>Problem: Criteria for Choosing a Feature to Split</b>	<b>1</b>
1.1	Not Splitting . . . . .	1
1.2	Splitting . . . . .	1
1.3	Mutual Information . . . . .	2
<b>2</b>	<b>Decision Stumps and Trees</b>	<b>3</b>
2.1	Built two decision stumps . . . . .	3
2.2	Build Decision Tree . . . . .	3
2.3	Multivariate Decision Tree . . . . .	3
2.4	Discussion . . . . .	4
<b>3</b>	<b>A "Baby" no Free Lunch Theorem</b>	<b>4</b>
3.1	The Number of Inputs . . . . .	5
3.2	The Number of Function . . . . .	5
3.3	Obtaining low expected loss . . . . .	5
3.4	Discussion . . . . .	5
<b>4</b>	<b>Implementation: <math>K</math>-Means</b>	<b>5</b>
4.1	Putting it all Together . . . . .	6

## 0 Policies

### 0.1 List of Collaborators

### 0.2 List of Acknowledgments

### 0.3 RTFM

I have read and understood these policies.

## 1 Problem: Criteria for Choosing a Feature to Split

We build a tree with the dataset  $D$  consisting  $n$  negative examples (label 0) and  $p$  positive examples (label 1).

### 1.1 Not Splitting

If we are at the bottom of our decision tree and don't have any features left to split, consider the subset  $D'$  of data  $D$  with  $n'$  negative and  $p'$  positive examples. The smallest number of mistakes we can make in this subset is given by

$$\text{err}(D') = \min(n', p') = \begin{cases} p' & \text{if } p' < n' \\ n' & \text{if } n' < p' \end{cases} \quad (1)$$

To achieve the minimum the node must be labeled accordingly, with 1 if  $n' < p'$  or with 0 if  $n' > p'$ .

### 1.2 Splitting

Now we have a new feature  $\Phi$  which splits the subsection  $D'$  according to the contingency table: By splitting we generate two new sub-nodes:  $(n_0, p_0)$  and  $(n_1, p_1)$ .

y	$\Phi$	
	0	1
0	$n_0$	$n_1$
1	$p_0$	$p_1$

Table 1: Contingency table for  $\Phi'$

The error for splitting is given by the sum of the errors of both nodes

$$\text{err}(D') = \min(n_0, p_0) + \min(n_1, p_1), \quad (2)$$

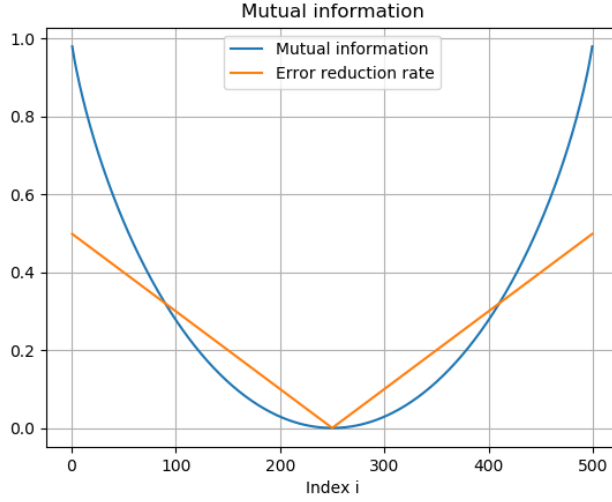


Figure 1: Comparison of mutual information between feature and label and error reduction rate for splitting the dataset.

where  $n' = n_0 + n_1$  and  $p' = n_1 + p_1$ . The error reduction rate (err\_red) is given by the reduction of error if comparing the error of "not splitting" with the error of "splitting", divided by the total number of examples in node  $D'$ :

$$\text{err\_red}(D') : \frac{\min(n', p') - (\min(n_0, p_0) + \min(n_1, p_1))}{|D'|}. \quad (3)$$

Consider the maximal possible error (in this case for binary data) when  $n' = p' = 0.5|D'|$

$$\text{err\_max}(D') = \min(n', p') = 0.5|D'|, \quad (4)$$

then the maximal possible error reduction is given by

$$\text{err\_red}(D') : \frac{\min(0.5|D'|, 0.5|D'|) - (\min(n_0, p_0) + \min(n_1, p_1))}{|D'|} = 0.5, \quad (5)$$

where either  $p_0 = 0$  or  $n_0 = 0$  and  $p_1 = 0$  or  $n_1 = 0$  (maximal information gain).

### 1.3 Mutual Information

The mutual information is a measure of information gain when splitting a dataset. In figure 1 the mutual information and the error reduction rate are plotted for problem 1.3. Both functions are symmetric around index 250 which represents equally distributed examples in form of  $n_0 = n_1 = p_0 = p_1 = 250$ .

As we showed in the previous part of this problem, the error reduction rate is maximal for the case where the error of a node is 0 after splitting. Corresponding to this the information gain in form of the mutual information is highest.

For increasing indices the information gain and the error reduction rate decrease until a minimum in  $i = 250$ , afterwards rising again.

## 2 Decision Stumps and Trees

### 2.1 Built two decision stumps

In figure 2 the comparison of error rates and mutual information for different threshold splits and features are plotted.

The top row shows the error as a function threshold. The feature "age" generates two minimums corresponding to two possible splits to minimize the error when splitting. For feature "salary" we get only one minimum but the error reduction is larger.

The mutual information is plotted in the bottom row. Since the mutual information gives the information gain for a split we have to search for maximums. Both features give one maximal values which we can choose as a threshold to split the node.

In figure 3 the decision stumps are plotted for the best thresholds found in figure 2. The error rate depends on choosing a feature to split and on the method to calculate the best threshold. An

### 2.2 Build Decision Tree

In figure 4 the greedily built decision trees are shown. We can see that the lower tree (built by using mutual information) only needs six nodes to reach an overall error of zero on the training set while the upper tree (built by counting errors) needs ten nodes.

### 2.3 Multivariate Decision Tree

To build the multivariate decision tree we can utilize some of the basic geometric foundations used in the Perceptron algorithm. We have to learn the function

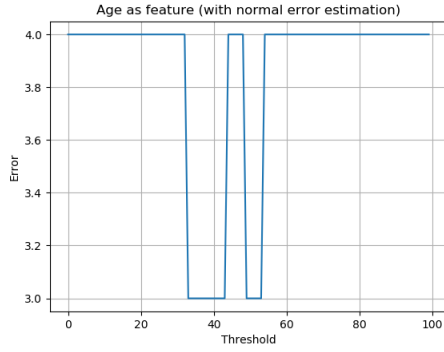
$$f(\vec{x}) = \left[ \sum_{d=1}^2 w_d x_d \right] - 1 = w_1 x_{\text{age}} + w_2 x_{\text{salary}} - 1. \quad (6)$$

Since we work in two dimensions it's most convenient to just "guess" the hyperplane of the data which is in this case of dimension one, a line. Let  $\vec{h}$  be the vector of this hyperplane. By drawing a straight line to separate negative and positive examples we can calculate the slope. This yields in

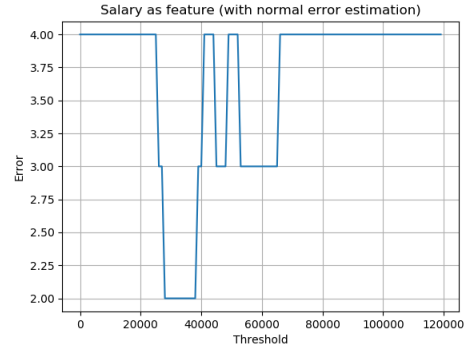
$$\vec{h} = \begin{pmatrix} 35 \\ 40000 \end{pmatrix}$$

where the first component corresponds to the age and the second component corresponds to the salary. According to the Perceptron the weight vector  $\vec{w}$  is perpendicular to the hyperplane. To get  $\vec{w}$  we can perform a rotation of  $\vec{h}$  by applying a 2D rotation matrix:

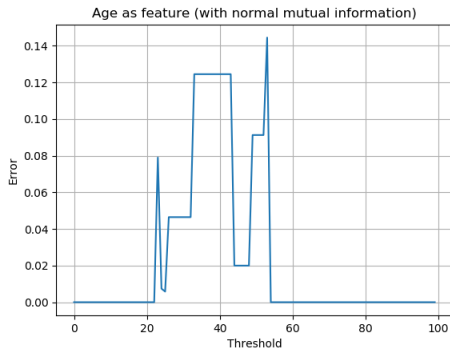
$$\vec{w} = \text{rot}(90)$$



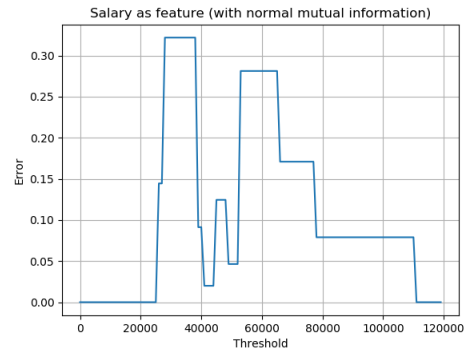
(a) Error for the feature "age"



(b) Error for the feature "salary"



(c) Mutual information for the feature "age"



(d) Mutual information for the feature "salary"

Figure 2: Comparison of error rates and mutual information for different threshold splits and features. For generating this data a script has been written to scan through different thresholds with a fine reasonable step width. The error respectively mutual information was calculated as shown in problem 1.

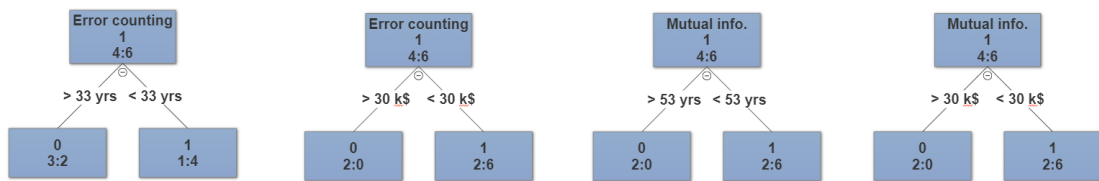


Figure 3: Decision stumps built after searching for the best threshold (Fig 2).

## 2.4 Discussion

## 3 A "Baby" no Free Lunch Theorem

Considering a function

$$f : \{0, 1\}^m \rightarrow \{0, 1\} \quad (7)$$

mapping a binary string of length  $m$  to a label, either 0 or 1.

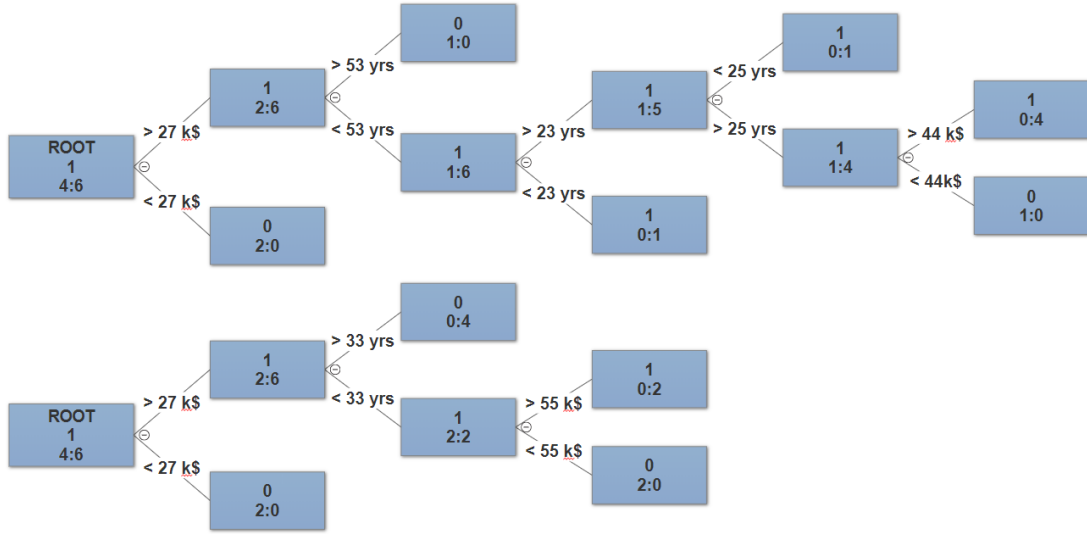


Figure 4: Greedily built decision trees. The splits for the upper tree were decided by searching for the best threshold to lower the total error. The splits for the lower tree were motivated by the best threshold corresponding to the largest information gain (mutual information). The label in the graphic for each node depicts the given label (upper row) and the abundance of negative examples (lower row, left value) and positive examples (lower row, right value). The errors for both trees are vanishing for deep nodes.

### 3.1 The Number of Inputs

The number of possible input strings is given by

$$N_{\text{input}} = 2^m \quad (8)$$

and for the length  $m = 5$  we have 32 different input strings.

### 3.2 The Number of Function

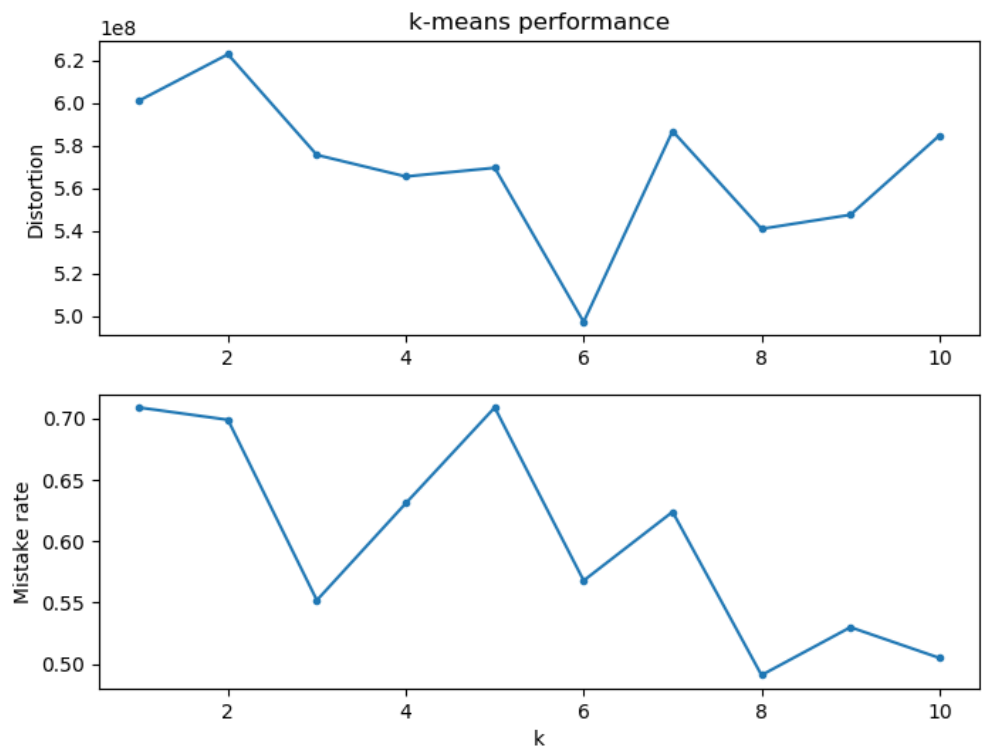
The number of possible functions to map the input string to a label is given by:

### 3.3 Obtaining low expected loss

### 3.4 Discussion

## 4 Implementation: *K*-Means

The script loops over  $K \in \{1, 2, \dots, 10\}$ , randomly initializes five times for each  $K$  and returns the result with the lowest within-group sum of squares.



4.1 Putting it all Together

References