Nils Poethkow nipoet@taltech.ee

1 Distance Functions

This task was pretty straight forward, as discussed in the practice, all distance functions are combined in one function to avoid duplication and redundancy. All distance functions compute the distance between two points with any number of dimensions, which is determined in the function. Table 1 shows the calculated distances for two example points.

Table 1: Distances Between Two Points p1(1,2) and p2(4,6)

Distance Function	Distance
Euclidean	7
Manhattan	5
Chebyshev	4

2 Entropy and Fisher Score

The Entropy function accepts the label vector of a dataset as the only input parameter, since only the labels are needed for the calculation of the entropy.For the Fisher Score, the formula discussed in class was used, therefore the function accepts two parameters, one being the dataset along with the labels and the second being the feature for which the fisher score should be calculated for. To test both functions, a 2D dataset with two classes was created which are separable by one feature and non-separable by the other feature. The Fig. 1 shows the used

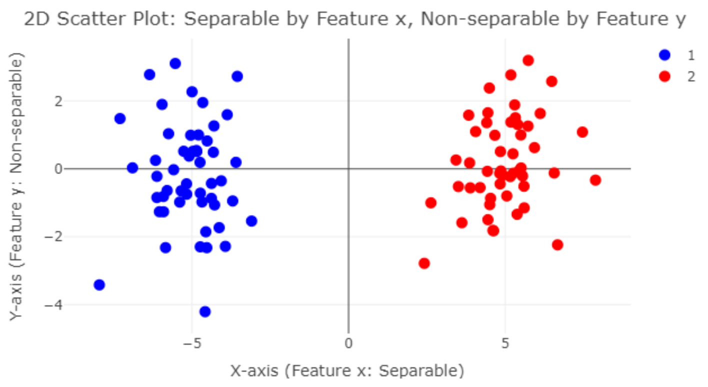

Figure 1: Plot of the Dataset used for testing

dataset. It's clearly visible, that classes 1 and 2 are separable by feature x and non-separable by feature y. Results of the test show, that the entropy of the entire dataset is clearly 1, since the dataset is split equally between class 1 and 2. For the Fisher Score, the results can be seen is Table 2. As anticipated,

Table 2: Fisher Score

.

Feature	Fisher Score
Feature x	21.47333
Feature y	0.003366095

the Fisher Score for feature "x" is significantly higher than the one for feature "y", indicating feature "x" should be used for classification.

3 Classification with Decision Tree

For the decision tree, essentially two functions are needed, one for building the tree and one for predicting the class labels of given data. To build a tree, the "train_tree" function is called with the dataset and the separate label vector along with the current depth of the tree (needed for recursion) and the max depth (set by the user, default is 4). The function then determines the feature with the highest information gain, which is calculated in the function "information_gain". This function uses the before implemented entropy function to determine both the feature and the best split for each feature, and returns the best feature along with the best possible split (threshold). It's inspired by the information gain function from Practice 3, yet different in the way that it performs the split based on the indices of the shuffled dataset. A possible improvement for this function could be an implementation of an early stopping functionality when determining the best split. Once the best split is determined, the "train_tree" function performs the split and calls the "train_tree" function recursively for the left branch with the left subset and the right branch with the right subset. All the data points in the right subset are greater and in the left subset less or equal to the determined split (threshold). When the "train_tree" function is called it first checks the stopping criteria, being the reach of max depth, the case of a subdataset with the length of one (meaning it can't be split anymore) or the case that of the data points in the subdataset are of the same class (there is no need for further splitting at this branch). When one of these cases occur, the function returns the label of the data points at this terminal node using a majority vote. This way the tree gets build in the depth first approach and the result is the linked list (using R "list" object) where every node consists of the used feature for splitting, the threshold for the split and a "link" to the left and right branch of the node. In case of a terminal node, all this data is missing and just a label is accessible in the list object. For first testing the algorithm, a 3D dataset was created and a tree of max depth equal to 4 was build, it's shown in Fig. 2. We haven't implemented goodness metrics yet, therefore no train-test-split was performed, so this is the next step to determine whether this tree is already overfitting on the training data. As seen

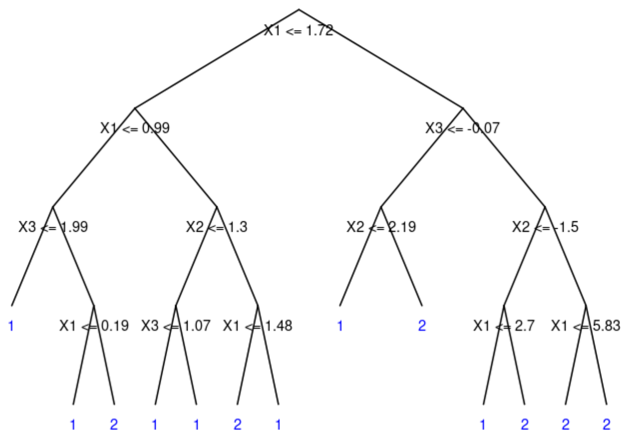

Figure 2: Example Decision Tree with max Depth of 4

in the Fig. 2 above, the tree performs multiple splits for the same feature when it results in the best gain.

3.1 Goodness Metrics

For the Goodness Metrics, the calculations for the True Positives, True Negatives, etc. are performed in a separate function, again to avoid redundancy. All Goodness Metrics are calculated in one function and can then be accessed separately.

3.2 Cross-validation

The different Goodness Metrics can now be used to evaluate the model with the further help of cross-validation. In the following, a cross-validation was performed on an example 2d Dataset (for simplification) and the outcome can be seen in Fig. 3. This plot implies, that after an optimal max depth of about 4, the accuracy of the tree on the test data steadily decreases with increasing depth. In this case, accuracy can be used because the dataset is evenly split between the two classes. The reason behind the accuracy increasing again could be the small size of the dataset. Nevertheless, is simple example shows the decision tree will overfit with a max depth set too high.

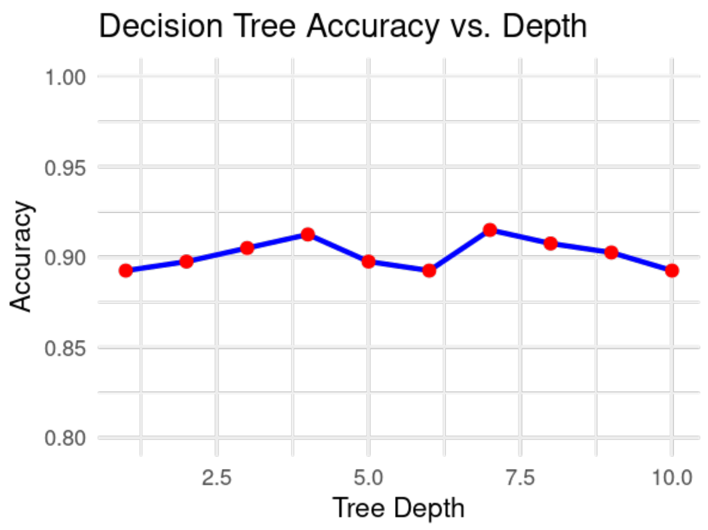

Figure 3: Accuracy of trees with different depths using cross validation

4 Linear Regression

For linear Regression, a forward selection Algorithm was implemented and tested with a ten dimensional dataset with one target feature. To visualize the outcome of the regression, the features selected by the algorithm are displayed as well as the intercept with are plotted in a bar plot, seen in Fig. 4. As the bar plot shows, the algorithm models the linear function

$$y = 2 + 0.3 \cdot x_1 - 0.5 \cdot x_2 + 0.1 \cdot x_3 \qquad (1)$$

perfectly. For testing the model, we could perform a train-test split on the dataset to investigate overfitting of the model.

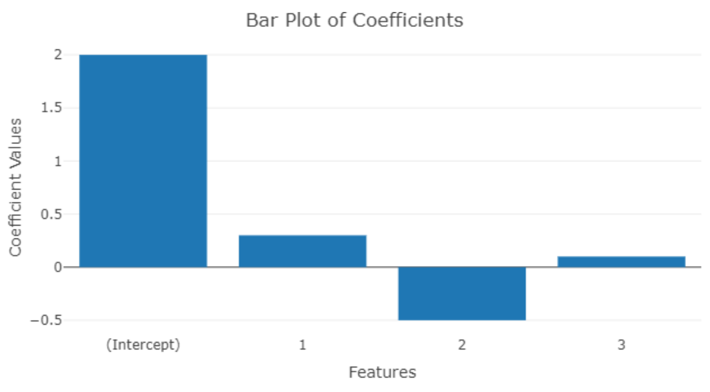

Figure 4: Feature selection with forward selection