Name, Vorname: Wang, Jiahui	**Fach:** B.Sc.Mathematik	**Martriknr.**2992080
Name, Vorname: Zhao, Jiaqi	**Fach:** M.Sc.Autonome System	**Martriknr.** 3470190
Name, Vorname: Wang, Ziyin	**Fach:** M.Sc.Informatik	**Martriknr.** 3435397

Assignment 5

Task 1: Inductive Construction

1. Decision tree construction:

(1) Start from empty root node at depth 0:

root	-	+	total	entropy
	115	125	240	0.999

Tabelle 1: Entropy root node

Computing the information gain(IG) for each feature to choose the suitable root node:

F_1	-	+	total	entropy
0	15	65	80	0.696
1	30	50	80	0.954
2	70	10	80	0.544
IG(F_1) = 0.267				

Tabelle 2: IG for F_1

F_2	-	+	total	entropy
0	70	50	120	0.980
1	45	75	120	0.954
IG(F_2) = 0.032				

Tabelle 3: IG for F_2

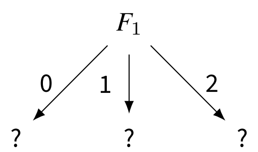
F_3	-	+	total	entropy
0	50	70	120	0.980
1	65	55	120	0.995
IG(F_3) = 0.011				

Tabelle 4: IG for F_3

F_4	-	+	total	entropy
0	50	70	120	0.980
1	65	55	120	0.954
IG(F_4) = 0.011				

Tabelle 5: IG for F_4

We can see that F_1 get the highest information gain, so we choose F1 as root node.
The graph of depth 0 is following:



(2) Computing the information gain(IG) for each feature to choose the suitable nodes at depth 1:

• F_1 = 0:

F_1 = 0	-	+	total	entropy
	15	65	80	0.696

Tabelle 6: Entropy F_1 = 0

F_2	-	+	total	entropy
0	15	25	40	0.954
1	0	40	40	0
IG(F_2) = 0.219				

Tabelle 7: IG for F_2

F_3	-	+	total	entropy
0	5	35	40	0.544
1	10	30	40	0.811
IG(F_3) = 0.0188				

Tabelle 8: IG for F_3

F_4	-	+	total	entropy
0	10	30	40	0.811
1	5	35	40	0.544
IG(F_4) = 0.0188				

Tabelle 9: IG for F_4

We can see that F_2 get the highest information gain, so we choose F_2 as the first split node.

• F_1 = 1:

F_1 = 1	-	+	total	entropy
	30	50	80	0.954

Tabelle 10: Entropy F_1 = 1

F_2	-	+	total	entropy
0	15	25	40	0.954
1	15	25	40	0.954
IG(F_2) = 0.000				

Tabelle 11: IG for F_2

F_3	-	+	total	entropy
0	15	25	40	0.954
1	15	25	40	0.954
IG(F_3) = 0.000				

Tabelle 12: IG for F_3

F_4	-	+	total	entropy
0	10	30	40	0.811
1	20	20	40	1.000
IG(F_4) = 0.0487				

Tabelle 13: IG for F_4

We can see that F_4 get the highest information gain, so we choose F_4 as the second split node.

- F_1 = 2:

F_1 = 2	-	+	total	entropy
	70	10	80	0.544

Tabelle 14: Entropy F_1 = 2

F_2	-	+	total	entropy
0	40	0	40	0.000
1	30	10	40	0.811
IG(F_2) = 0.138				

Tabelle 15: IG for F_2

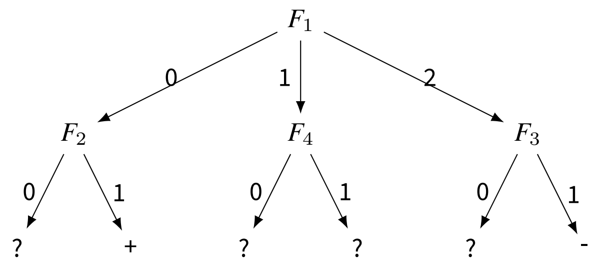
F_3	-	+	total	entropy
0	30	10	40	0.811
1	40	0	40	0.000
IG(F_3) = 0.138				

Tabelle 16: IG for F_3

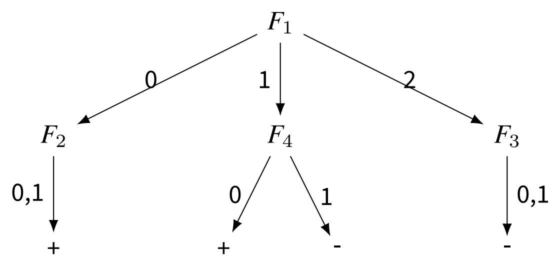
F_4	-	+	total	entropy
0	30	10	40	0.811
1	40	0	40	0.000
IG(F_4) = 0.138				

Tabelle 17: IG for F_4

We can see that F_2, F_3, F_4 get the same information gain, so we arbitrarily choose F_3 as the third split node. Now the graph of depth 1 looks below:



(3) One of the stop criteria of this task is the tree depth of 2, so we should finish the tree at depth 2. For the label of leaves, we use the label which is the majority of each node as the leaf. So for the F_2 node, we choose **+** label as leaf and for the F_3 node, we choose **-** label as leaf. For the F_4 node, the quantity difference between **+** and **-** is not obvious, and also the entropy of value **1** is 1.0, so we can first choose **+** as the label leaf of value **0**, then arbitrarily the label leaf of value **1**0 can be chosen as **-**, so the final graph of the decision tree is following:



2. We use the following tables to calculate the error rate:

F_2	0	1	total
-	15	0	15
+	25	40	65
total	40	40	80

Tabelle 18: node F_2

F_4	0	1	total
-	10	20	30
+	30	20	50
total	40	40	80

Tabelle 19: node F_2

F_3	0	1	total
-	30	40	70
+	10	0	10
total	40	40	80

Tabelle 20: node F_2

$$Error\_rate = \frac{15 + 10 + 20 + 10}{80 + 40 + 40 + 80} = \frac{55}{240} = 0.229$$

Task 2: Minimal Error Pruning

1. the error rate of the original tree:

$$e_0 = \frac{22+7+1}{2838+2628} = \frac{30}{5466} = \frac{5}{911} = 0.0055$$

Since a node is viable for pruning, if all its children are leaf nodes. So we should try to prune the 3 nodes:ödor = pungent","cap-color = pink","bruises = false".

2. the error rate of the tree after removing the node ödor = pungent":

$$e_1 = \frac{176+22+7+1}{2838+2628} = \frac{206}{5466} = \frac{103}{2733} = 0.0377$$

3. the error rate of the tree after removing the node "cap-color = pink":

$$e_2 = \frac{29+1}{2838+2628} = \frac{30}{5466} = \frac{5}{911} = 0.0055$$

4. the error rate of the tree after removing the node "bruises = false":

$$e_3 = \frac{22+7+21}{2838+2628} = \frac{50}{5466} = \frac{25}{2733} = 0.0091$$

So we should first removes the node "cap-color = pink", because it is the node with the lowest error rate. (Abbildung 1)

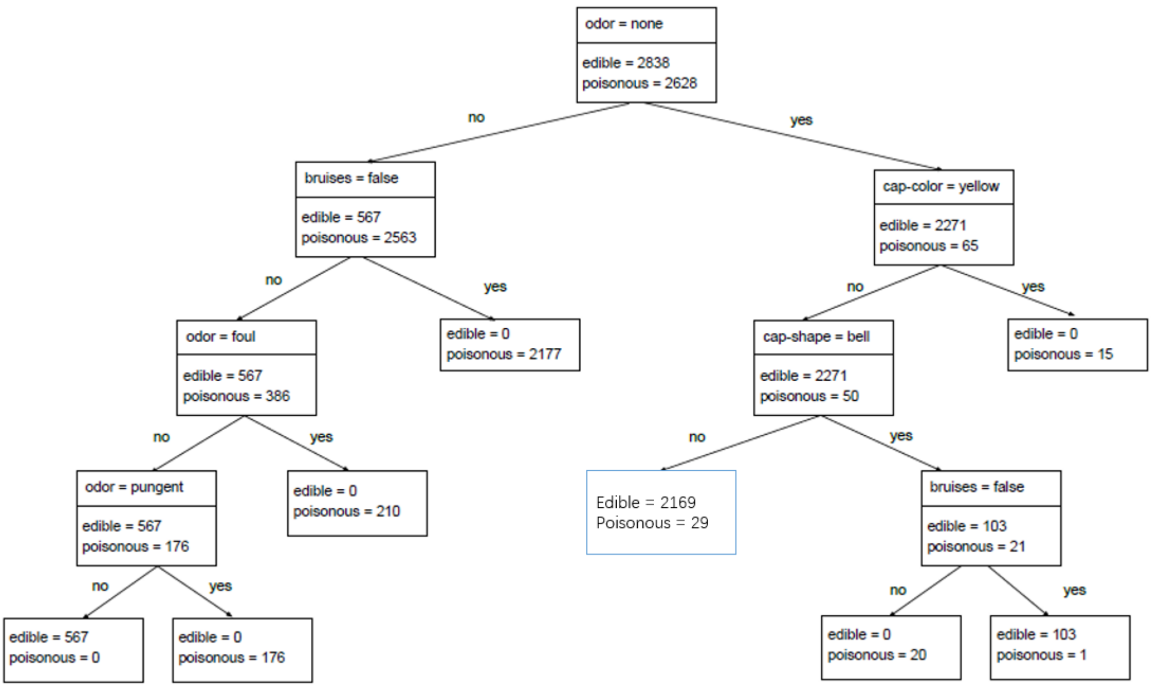


Abbildung 1: The decision tree after pruning the node "cap-color = pink

Then we should remove the node "bruises = false"because it has the lower error rate as ödor = pungent". So the decision tree after 2 prune operation is as followed:(Abbildung 2)

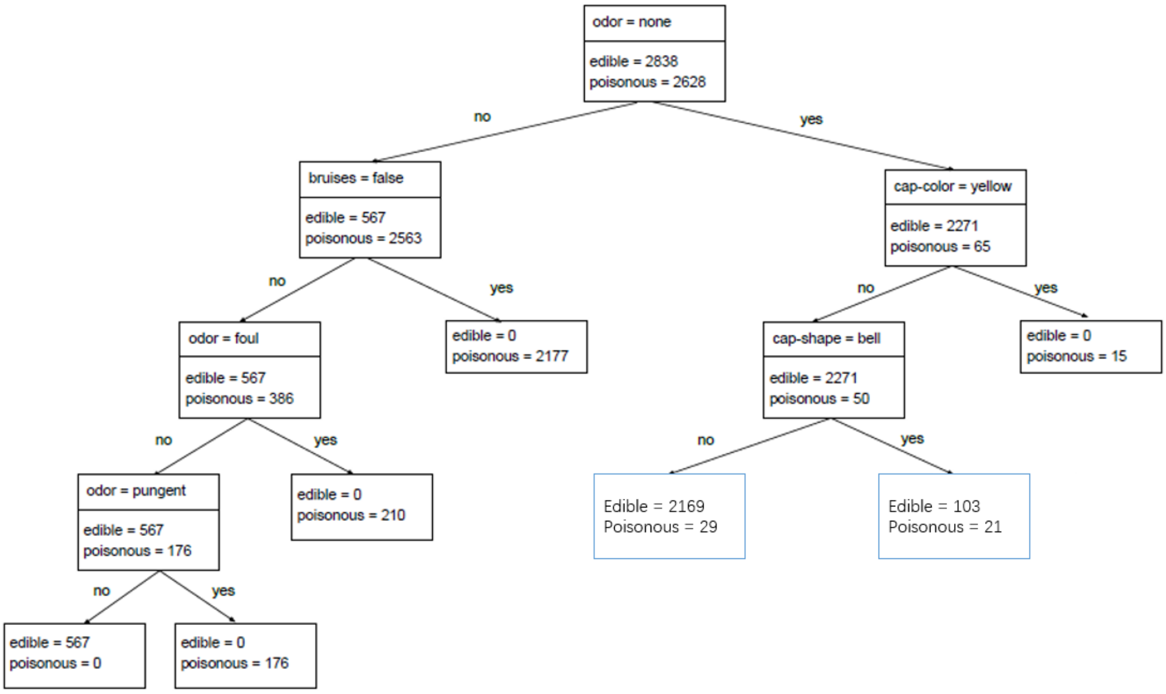


Abbildung 2: The decision tree after pruning the node "bruises = false"

Task 3. Regression with Decision Trees and kNN

3.1 How does the construction of regression trees di!er to classification trees? How is a prediction computed in a regression tree?

The structures of regression tree and decision tree are the same in the sense that they all use certain criteria to split the nodes, fit the input data into the tree and use certain property of corresponding leaf as output.
However, there are two main difference:

- Decision tree use entropy as criteria; while regression tree use the residual sum of squares (RSS) as criteria.
- Decision tree use the label value of elements in the leaf as output, which is normally a categorial value; while regression tree use the mean value of elements as output, which is normally a real number.

3.2 How can kNN be used for regression?

Instead use the majority of the labels of the k -nearest neighbors as output (prediction), the average value of the labels will be choose as prediction. In this way, the range of predictions are no longer the predefined category of labels but a real interval.