

# Session 8

## Decision trees, rules, entropy, kappa and ROC

This session goes inside the classifier output: how the tree was built from entropy, what kappa adds to accuracy, how to read rules off the tree, and how ROC curves compare classifiers.

## Objectives

✕ Do not copy. Read for understanding and the viva

- Complete questions 17 to 18 of the manual: decision trees, rules, entropy, kappa and roc
- Prepare the deliverable before the lab and finish it during the session
- Be ready to explain every step in the viva

## Questions Covered

✕ Do not copy. Read for understanding and the viva

| Question | Requirement           | Status   |
|----------|-----------------------|----------|
| Q17      | Perform the following | Complete |
| Q18      | Perform the following | Complete |

## Preparation

✕ Do not copy. Read for understanding and the viva

- ID3 is under `trees.Id3` and needs nominal attributes only; J48 handles numeric ones. Compute the root entropy and the gain of the chosen root attribute by hand for the weather data.
- Extract rules: each root-to-leaf path is one if-then rule; write all of them and check they cover every instance.

- ROC: right-click the result, Visualize threshold curve, pick the class; record the AUC for each classifier and dataset in one comparison table.

## Question 17

### Problem Statement

■ Write in lab record

Perform the following:

- Demonstrate performing classification on various data sets.
- Load each dataset into WEKA and run ID3, J48 classification algorithm.
- Study the classifier output. Compute entropy values, Kappa statistic.
- Extract if-then rules from the decision tree generated by the classifier.
- Observe the confusion matrix.

### Solution

■ Write in lab record

### Steps

1. **Preprocess, Open file...**, `data/weather.nominal.arff` (14 instances, 5 nominal attributes, class `play` : 9 yes, 5 no).
2. `Id3` is not in a fresh WEKA 3.8 install. Open **Tools, Package manager** from the GUI Chooser, find `simpleEducationalLearningSchemes`, click **Install**, restart WEKA. It then appears under `trees.Id3`. `Id3` accepts only nominal attributes and no missing values, so for `student.arff` and `employee.arff` first apply `filters.supervised.attribute.Discretize` on the Preprocess tab.
3. **Classify, Choose**, `trees.Id3`, Test options **Use training set, Start**. Record the tree. Then switch to **Cross-validation** 10 folds and **Start** again.
4. **Choose**, `trees.J48`, **Start** with 10-fold cross-validation. Record the tree, the "Number of Leaves" and "Size of the tree" lines, the kappa and the confusion matrix. Right-click the result, **Visualize tree**.
5. Repeat step 4 for `student.arff` and `employee.arff` (from Session 7) and note each confusion matrix.

6. Compute the entropy of the root and the information gain of every attribute by hand from the formula sheet, then check with `entropy_gain.py`, which also prints the gain at every internal node and the rules of the finished tree.
7. Write one if-then rule per root-to-leaf path and check that the instance counts on the leaves add up to 14.

## Program

entropy\_gain.py

```

1  #!/usr/bin/env python3
2  """Entropy, information gain, gain ratio and the ID3 tree for weather.nomina
3  Every number WEKA's Id3 / J48 uses on this data set is printed, then the tre
4  if-then rules, then the kappa statistic from a confusion matrix by hand.
5  Run: python3 entropy_gain.py
6  """
7  import math
8  from collections import Counter
9
10 ATTRS = ['outlook', 'temperature', 'humidity', 'windy', 'play']
11 DATA = [r.split(',') for r in """sunny,hot,high,FALSE,no
12 sunny,hot,high,TRUE,no
13 overcast,hot,high,FALSE,yes
14 rainy,mild,high,FALSE,yes
15 rainy,cool,normal,FALSE,yes
16 rainy,cool,normal,TRUE,no
17 overcast,cool,normal,TRUE,yes
18 sunny,mild,high,FALSE,no
19 sunny,cool,normal,FALSE,yes
20 rainy,mild,normal,FALSE,yes
21 sunny,mild,normal,TRUE,yes
22 overcast,mild,high,TRUE,yes
23 overcast,hot,normal,FALSE,yes
24 rainy,mild,high,TRUE,no""".splitlines()]]
25
26
27 def entropy(rows):
28     n = len(rows)
29     return -sum(c / n * math.log2(c / n) for c in Counter(r[-1] for r in row
30
31
32 def gain(rows, j):
33     parts = {}
34     for r in rows:
35         parts.setdefault(r[j], []).append(r)
36     rem = sum(len(p) / len(rows) * entropy(p) for p in parts.values())
37     split = -sum(len(p) / len(rows) * math.log2(len(p) / len(rows)) for p in
38     return entropy(rows) - rem, rem, split, parts
39
40
41 def id3(rows, avail, depth=0, path=()):
42     counts = Counter(r[-1] for r in rows)
43     pad = '    ' * depth

```

```

44     if len(counts) == 1 or not avail:
45         label = counts.most_common(1)[0][0]
46         print(f"{pad}leaf: {label} ({len(rows)})")
47         return [(path, label, len(rows))]
48     print(f"{pad}node {list(path) or 'root': {len(rows)} rows, {dict(counts
49     best = None
50     for j in avail:
51         g, rem, split, parts = gain(rows, j)
52         ratio = g / split if split else 0
53         print(f"{pad} gain({ATTRS[j]}) = {entropy(rows):.4f} - {rem:.4f} =
54         if best is None or g > best[0]:
55             best = (g, j, parts)
56     g, j, parts = best
57     print(f"{pad} split on {ATTRS[j]}")
58     rules = []
59     for v, p in parts.items():
60         print(f"{pad}{ATTRS[j]} = {v}")
61         rules += id3(p, [a for a in avail if a != j], depth + 1, path + ((AT
62     return rules
63
64
65 def kappa(matrix):
66     n = sum(map(sum, matrix))
67     po = sum(matrix[i][i] for i in range(len(matrix))) / n
68     pe = sum(sum(matrix[i]) * sum(row[i] for row in matrix) for i in range(l
69     return po, pe, (po - pe) / (1 - pe)
70
71
72 if __name__ == '__main__':
73     print(f"root: 14 rows, {dict(Counter(r[-1] for r in DATA))}, H(S) = {ent
74     rules = id3(DATA, [0, 1, 2, 3])
75     print("\nif-then rules from the tree:")
76     for i, (path, label, n) in enumerate(rules, 1):
77         cond = ' and '.join(f"{a} = {v}" for a, v in path)
78         print(f"R{i}: if {cond} then play = {label} ({n} instances)")
79     assert sum(n for _, _, n in rules) == 14 # rules cover every row exactl
80     print("\nkappa by hand for J48's 10-fold confusion matrix on weather.nom
81     m = [[5, 4], [3, 2]]
82     po, pe, k = kappa(m)
83     print(f"matrix = {m} po = {po:.4f} pe = {pe:.4f} kappa = {k:.4f}")
84     assert abs(k + 0.0426) < 5e-4

```

## Output

python3 entropy\_gain.py (run for real):

```

1 root: 14 rows, {'no': 5, 'yes': 9}, H(S) = 0.9403
2
3 node root: 14 rows, {'no': 5, 'yes': 9}, H = 0.9403
4   gain(outlook) = 0.9403 - 0.6935 = 0.2467   splitinfo = 1.5774   gain ratio
5   gain(temperature) = 0.9403 - 0.9111 = 0.0292   splitinfo = 1.5567   gain r
6   gain(humidity) = 0.9403 - 0.7885 = 0.1518   splitinfo = 1.0000   gain rati
7   gain(windy) = 0.9403 - 0.8922 = 0.0481   splitinfo = 0.9852   gain ratio =
8   split on outlook
9 outlook = sunny
10   node [('outlook', 'sunny')]: 5 rows, {'no': 3, 'yes': 2}, H = 0.9710
11   gain(temperature) = 0.9710 - 0.4000 = 0.5710   splitinfo = 1.5219   ga
12   gain(humidity) = 0.9710 - 0.0000 = 0.9710   splitinfo = 0.9710   gain
13   gain(windy) = 0.9710 - 0.9510 = 0.0200   splitinfo = 0.9710   gain rat
14   split on humidity
15   humidity = high
16     leaf: no (3)
17   humidity = normal
18     leaf: yes (2)
19 outlook = overcast
20   leaf: yes (4)
21 outlook = rainy
22   node [('outlook', 'rainy')]: 5 rows, {'yes': 3, 'no': 2}, H = 0.9710
23   gain(temperature) = 0.9710 - 0.9510 = 0.0200   splitinfo = 0.9710   ga
24   gain(humidity) = 0.9710 - 0.9510 = 0.0200   splitinfo = 0.9710   gain
25   gain(windy) = 0.9710 - 0.0000 = 0.9710   splitinfo = 0.9710   gain rat
26   split on windy
27   windy = FALSE
28     leaf: yes (3)
29   windy = TRUE
30     leaf: no (2)
31
32 if-then rules from the tree:
33 R1: if outlook = sunny and humidity = high then play = no   (3 instances)
34 R2: if outlook = sunny and humidity = normal then play = yes   (2 instances)
35 R3: if outlook = overcast then play = yes   (4 instances)
36 R4: if outlook = rainy and windy = FALSE then play = yes   (3 instances)
37 R5: if outlook = rainy and windy = TRUE then play = no   (2 instances)
38
39 kappa by hand for J48's 10-fold confusion matrix on weather.nominal:
40 matrix = [[5, 4], [3, 2]]   po = 0.5000   pe = 0.5204   kappa = -0.0426

```

The same gains appear on WEKA's Preprocess tab under **Select attributes**,

InfoGainAttributeEval with Ranker : 0.2467 outlook, 0.1518 humidity, 0.0481 windy, 0.0292 temperature (the manual's Buys-computers example shows the same layout).

What WEKA prints for the two tree learners on weather.nominal.arff :

```

1  ≡≡≡ Classifier model (full training set) ≡≡≡
2
3  Id3
4
5  outlook = sunny
6  |  humidity = high: no
7  |  humidity = normal: yes
8  outlook = overcast: yes
9  outlook = rainy
10 |  windy = TRUE: no
11 |  windy = FALSE: yes
12
13 ≡≡≡ Evaluation on training set ≡≡≡
14 Correctly Classified Instances      14           100      %
15 Kappa statistic                     1

```

```

1 J48 pruned tree
2 -----
3
4 outlook = sunny
5 |   humidity = high: no (3.0)
6 |   humidity = normal: yes (2.0)
7 outlook = overcast: yes (4.0)
8 outlook = rainy
9 |   windy = TRUE: no (2.0)
10 |   windy = FALSE: yes (3.0)
11
12 Number of Leaves   :    5
13
14 Size of the tree   :    8
15
16 == Stratified cross-validation ==
17 == Summary ==
18
19 Correctly Classified Instances      7           50      %
20 Incorrectly Classified Instances    7           50      %
21 Kappa statistic                    -0.0426
22 Total Number of Instances          14
23
24 == Confusion Matrix ==
25
26 a b   ← classified as
27 5 4 | a = yes
28 3 2 | b = no

```

Confusion matrices for the tree learner on the Session 7 data sets ( `classify_cv.py` , 10-fold cross-validation): student, pass 19 0 and fail 1 10 (kappa 0.9268); employee, yes 15 1 and no 3 11 (kappa 0.7297).

## Explanation

- **Root entropy.** Nine yes and five no give  $H(S) = -\frac{9}{14}\log_2 \frac{9}{14} - \frac{5}{14}\log_2 \frac{5}{14} = 0.9403$  bits.
- **Gain of outlook.** Sunny has 2 yes and 3 no (entropy 0.9710), overcast 4 yes (0), rainy 3 yes and 2 no (0.9710). The weighted remainder is  $\frac{5}{14}(0.9710) + \frac{4}{14}(0) + \frac{5}{14}(0.9710) = 0.6935$ , so the gain is 0.9403 minus 0.6935, which is 0.2467, the largest of the four. ID3 therefore puts `outlook` at the root. Under `sunny` , `humidity` splits the five rows perfectly (gain 0.9710); under `rainy` , `windy` does. Both trees are identical here because the gain ratio picks the

same attributes; the ratio matters only when an attribute with many values (an identifier) would win on plain gain.

- **Rules.** Each path from the root to a leaf is one conjunction; the five rules cover  $3 + 2 + 4 + 3 + 2 = 14$  instances and no instance matches two rules, because the tests on a path are on distinct attributes and sibling branches are exclusive.
- **Kappa by hand** for the J48 cross-validation matrix: observed agreement  $p_o = (5 + 2)/14 = 0.5$ ; the row totals are 9 and 5, the column totals 8 and 6, so  $p_e = (9 \cdot 8 + 5 \cdot 6)/14^2 = 102/196 = 0.5204$  and  $\kappa = (0.5000 - 0.5204)/(1 - 0.5204) = -0.0426$ . A negative kappa says the cross-validated tree does worse than chance: with 14 rows, each fold trains on 12 or 13 and the tree changes shape every time. The training-set result of 100 percent is not evidence of anything; the data was used to build the tree.
- **Reading the confusion matrix.** Row = actual class, column = predicted. For student the tree misses exactly one row (the noisy row 28, attendance 70 and internal 14 but `fail`), and no `pass` student is predicted `fail`.

## Question 18

### Problem Statement

■ Write in lab record

Perform the following:

- Load each dataset into WEKA and perform Naive Bayes classification and k-Nearest Neighbour classification.
- Interpret the results obtained.
- Plot ROC Curves.
- Compare classification results of ID3, J48, Naive Bayes and k-NN classifiers for each dataset, and deduce which classifier is performing best and poor for each dataset and justify.

### Solution

■ Write in lab record

### Steps

1. Load each data set ( `weather.nominal.arff` , `student.arff` , `employee.arff` , `labor.arff` ) and run `bayes.NaiveBayes` and `lazy.IBk` (  $k = 1$ , then  $k = 3$  ) with 10-fold cross-validation, as in Session 7. Keep the `Id3` and `J48` entries in the Result list.
2. For each result, read the **Detailed Accuracy By Class** table: the `ROC Area` column is the AUC for that class; the `Weighted Avg.` row is the number to compare.
3. To plot the curve, right-click the result entry, **Visualize threshold curve**, pick the class ( `pass` , `yes` , `good` ). The window plots False Positive Rate on X against True Positive Rate on Y; the title bar shows the area. Set Colour to `Threshold` to see which probability cut-off each point corresponds to. Repeat for the other classifiers and compare the curves by eye: the curve nearer the top-left corner is better.
4. Fill one comparison table per data set with accuracy, kappa and ROC area for ID3, J48, NaiveBayes and k-NN, and name the best and the poorest with a reason.

## Output

`classify_cv.py` computes the ROC area exactly as WEKA does, by ranking the cross-validated probability of the first class (Mann-Whitney statistic). Student, employee and weather rows are from the script ( `python3 classify_cv.py student.arff employee.arff weather.nominal.arff` , run for real); on the 14 weather rows WEKA's own J48 result (50 percent, kappa -0.0426) is also listed, and it differs from the script's tree only because the two split the 14 rows into folds differently. Labor rows are the accuracies WEKA 3.8 prints with the default seed; copy its kappa and ROC area from your run.

| Data set        | Classifier         | Accuracy | Kappa  | ROC area | Verdict                                        |
|-----------------|--------------------|----------|--------|----------|------------------------------------------------|
| weather.nominal | Id3 (training set) | 100 %    | 1.000  | 1.000    | not comparable (no cross-validation)           |
| weather.nominal | Tree, script folds | 71.4 %   | 0.378  | 0.756    | best on these folds                            |
| weather.nominal | J48 in WEKA        | 50.0 %   | -0.043 | record   | worst on WEKA's folds: tree changes every fold |
| weather.nominal | NaiveBayes         | 64.3 %   | 0.186  | 0.667    | steady: averages over all attributes           |
| weather.nominal | IBk k=1            | 42.9 %   | -0.057 | 0.467    | poorest: below chance                          |
| student         | Id3/J48            | 96.7 %   | 0.927  | 0.945    | best: data follows a threshold rule            |
| student         | NaiveBayes         | 83.3 %   | 0.648  | 0.919    | poorest accuracy, second-best ranking          |
| student         | IBk k=1            | 90.0 %   | 0.781  | 0.883    | poorest ROC: only 0 or 1 scores                |
| employee        | Id3/J48            | 86.7 %   | 0.730  | 0.868    | joint best accuracy                            |
| employee        | NaiveBayes         | 86.7 %   | 0.735  | 0.969    | best: highest kappa and ROC                    |
| employee        | IBk k=1            | 76.7 %   | 0.537  | 0.772    | poorest: noisy neighbours                      |
| labor           | J48                | 73.7 %   | 0.442  | record   | poorest: 57 rows, unstable pruning             |
| labor           | NaiveBayes         | 89.5 %   | record | record   | best: tolerates missing values                 |

| Data set | Classifier | Accuracy   | Kappa  | ROC area | Verdict |
|----------|------------|------------|--------|----------|---------|
| labor    | IBk k=1    | about 82 % | record | record   | middle  |

Naive Bayes model block WEKA prints for `weather.nominal.arff`, which shows what the ranking is built from:

```

1 Naive Bayes Classifier
2
3           Class
4 Attribute    yes    no
5              (0.63) (0.38)
6 =====
7 outlook
8   sunny      3.0    4.0
9   overcast   5.0    1.0
10  rainy      4.0    3.0
11  [total]    12.0    8.0
12
13 temperature
14   hot        3.0    3.0
15   mild       5.0    3.0
16   cool       4.0    2.0
17  [total]    12.0    8.0
18
19 humidity
20   high       4.0    5.0
21  normal     7.0    2.0
22  [total]    11.0    7.0
23
24 windy
25   TRUE       4.0    4.0
26  FALSE      7.0    3.0
27  [total]    11.0    7.0

```

Each count is one more than the raw frequency (Laplace correction), so no probability is ever zero. For a sunny, cool, high, windy day the yes score is  $0.63 \times \frac{3}{12} \times \frac{4}{12} \times \frac{4}{11} \times \frac{4}{11} = 0.0069$  and the no score is  $0.38 \times \frac{4}{8} \times \frac{2}{8} \times \frac{5}{7} \times \frac{4}{7} = 0.0194$ , so the prediction is no with probability  $0.0194 / (0.0069 + 0.0194) = 0.74$ . That 0.74 is the score the ROC curve ranks.

## Explanation

- **Interpreting the results.** Accuracy counts hard decisions at the 0.5 cut-off; kappa corrects it for chance; the ROC area asks a different question: if one positive and one negative row are picked at random, how often does the classifier score the positive one higher? A classifier can have modest accuracy and a high ROC area (Naive Bayes on student: 83 percent but 0.92) when its probabilities are well ordered but its cut-off is off.
- **Why the best classifier differs per data set.** The tree wins on `student` because the class was generated by thresholds on `attendance` and `internal`, exactly the shape a tree draws. Naive Bayes wins on `employee` and `labor`: the class there is a smooth trend over correlated numeric attributes, and on `labor` the product form simply omits an attribute that is missing instead of imputing it. k-NN with  $k = 1$  is poorest on `employee` because the two deliberately mislabelled rows sit inside the promoted region and become the nearest neighbour of several test rows;  $k = 3$  with distance weighting lifts it to about 83 percent. On `weather.nominal` the verdict depends on the folds: 14 rows are too few for a stable estimate, which is why WEKA's J48 lands at 50 percent while the script's tree on different folds reaches 71 percent; k-NN with  $k = 1$  is below chance either way, because most rows have no near-identical neighbour. J48 is poorest on `labor` for the same reason: a tree pruned from about 50 rows differs from fold to fold.
- **ROC of k-NN with  $k = 1$ .** Every score is exactly 0 or 1, so the curve has a single corner and the area is roughly the average of the true positive and true negative rates; raising  $k$  gives graded scores and a smoother curve.
- **Where ID3 fits.** ID3 is unpruned and nominal-only, so on the discretized student and employee data it behaves like the unpruned tree of `classify_cv.py` (the numbers in the table); on data with unseen attribute values it fails to classify the instance, which WEKA counts as an error.

## Formula Sheet

✗ Do not copy. Read for understanding and the viva

## Association rules

For a rule  $X \Rightarrow Y$  over  $N$  transactions:

$$\text{support}(X \Rightarrow Y) = \frac{|X \cup Y|}{N}, \quad \text{confidence}(X \Rightarrow Y) = \frac{|X \cup Y|}{|X|}, \quad \text{lift}(X \Rightarrow Y) = \frac{\text{confidence}(X \Rightarrow Y)}{\text{support}(Y)}$$

WEKA's Apriori starts at the upper bound of minimum support and lowers it by the delta each pass until the requested number of rules is found or the lower bound is reached.

## Entropy, information gain and Gini

For a set  $S$  with class proportions  $p_1, \dots, p_c$ :

$$H(S) = - \sum_{i=1}^c p_i \log_2 p_i, \quad \text{Gain}(S, A) = H(S) - \sum_{v \in \text{values}(A)} \frac{|S_v|}{|S|} H(S_v), \quad \text{Gini}(S) = 1 - \sum_{i=1}^c p_i^2$$

ID3 splits on the attribute with the highest gain; J48 (C4.5) uses the gain ratio

$\text{Gain}(S, A) / \text{SplitInfo}(S, A)$  where  $\text{SplitInfo}(S, A) = - \sum_v \frac{|S_v|}{|S|} \log_2 \frac{|S_v|}{|S|}$ .

## Classifier evaluation

From the confusion matrix with true positives  $TP$ , false positives  $FP$ , false negatives  $FN$ , true negatives  $TN$ :

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}, \quad \text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}, \quad F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Kappa compares observed agreement  $p_o$  (accuracy) with the agreement expected by chance  $p_e$ :

$$\kappa = \frac{p_o - p_e}{1 - p_e}, \quad p_e = \sum_i \frac{(\text{row}_i \text{ total}) (\text{column}_i \text{ total})}{N^2}$$

The ROC curve plots true positive rate  $TP / (TP + FN)$  against false positive rate  $FP / (FP + TN)$ ; the area under it (AUC) is 0.5 for guessing and 1.0 for a perfect classifier.

## Naive Bayes and k-nearest neighbour

$$P(C | x_1, \dots, x_n) \propto P(C) \prod_{i=1}^n P(x_i | C)$$

k-NN assigns the majority class among the  $k$  nearest training records under Euclidean distance

$$d(\mathbf{a}, \mathbf{b}) = \sqrt{\sum_{i=1}^n (a_i - b_i)^2}$$

after normalising each attribute to  $[0, 1]$  with  $x' = (x - x_{\min}) / (x_{\max} - x_{\min})$ .

## Linear regression

$$\hat{y} = \beta_0 + \beta_1 x, \quad \beta_1 = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sum (x_i - \bar{x})^2}, \quad \beta_0 = \bar{y} - \beta_1 \bar{x}$$

WEKA reports the correlation coefficient, mean absolute error and root mean squared error  $\sqrt{\frac{1}{N} \sum (y_i - \hat{y}_i)^2}$ .

## Clustering

k-means minimises the within-cluster sum of squared errors over clusters  $C_1, \dots, C_k$  with centroids  $\mu_j$ :

$$SSE = \sum_{j=1}^k \sum_{\mathbf{x} \in C_j} \|\mathbf{x} - \mu_j\|^2, \quad \mu_j = \frac{1}{|C_j|} \sum_{\mathbf{x} \in C_j} \mathbf{x}$$

Hierarchical (agglomerative) clustering merges the two closest clusters each step; linkage defines closeness: single  $\min d(a, b)$ , complete  $\max d(a, b)$ , average  $\frac{1}{|A|+|B|} \sum d(a, b)$ .

DBSCAN calls a point a core point when at least minPts points lie within radius  $\epsilon$ ; clusters grow from core points, and points reachable from none are noise.

## Viva Questions

✗ Do not copy. Read for understanding and the viva

**Q:** What is entropy measuring here? **A:** The impurity of the class distribution in bits; 0 for a pure set, 1 for a 50/50 two-class set.

**Q:** Why does ID3 choose outlook at the root? **A:** Its information gain, 0.2467, is the largest; humidity gives 0.1518, windy 0.0481, temperature 0.0292.

**Q:** Why does C4.5 use gain ratio instead of gain? **A:** Gain favours attributes with many values (an ID attribute gives perfect gain); dividing by the split information penalises that.

**Q:** How many rules does a tree produce? **A:** One per leaf; each rule is the conjunction of the tests on the path from the root.

**Q:** What does a negative kappa mean? **A:** Agreement below what chance would give; the J48 cross-validation on 14 weather rows shows -0.0426.

**Q:** What are the axes of a ROC curve? **A:** False positive rate on X, true positive rate on Y; each point is one probability threshold.

**Q:** What is the AUC of a classifier that guesses? **A:** 0.5, the diagonal.

**Q:** Why is `Id3` missing from the Classify tab? **A:** In WEKA 3.8 it lives in the `simpleEducationalLearningSchemes` package and must be installed through the package manager.

## Common Mistakes

✗ Do not copy. Read for understanding and the viva

- Computing gain with natural logarithms; the formula sheet uses base 2 and WEKA's numbers only match with base 2.
- Forgetting to weight the branch entropies by branch size when computing the remainder.
- Writing rules for internal nodes instead of leaves, or omitting the `overcast` rule because it has no second test.
- Quoting training-set accuracy for ID3 alongside cross-validated accuracy for the others and calling ID3 the best.
- Reading the ROC area for the wrong class; pick the class named in the question, or use the weighted average.
- Comparing classifiers on accuracy alone when the classes are unbalanced; kappa and ROC area exist for that reason.

## Session Summary

■ Write in lab record

- Question 17: ID3 and J48 on weather.nominal, student and employee; root entropy 0.9403 and every gain computed by `entropy_gain.py`, five if-then rules extracted, kappa -0.0426 derived by hand from the J48 confusion matrix.
- Question 18: NaiveBayes and IBk on the same data sets, ROC curves plotted with Visualize threshold curve, comparison table of accuracy, kappa and ROC area across ID3, J48, NaiveBayes and k-NN with best and poorest justified per data set.

✎ Edit this page

< Previous  
**Session 7**

Next >  
**Session 9**