Data Set: Mushroom Data Set
Task   : To predict a mushroom to be poisonous/edible

Group 16

Anjana Babu

Drishya Praveen

Indu Sree

Asla Aboo

WHAT ALL DID WE DO WITH OUR DATA SET?

1. Data Cleaning
    - Handled Missing Values
    - Removed an unwanted attribute

2. Data Mining
    - Analysing the data using WEKA
    - Is our problem Classification or Clustering?
    - Visualization using WEKA
    - Adding Predicted attribute to data set
    - Building a Classifier
        - J48 Tree Classifier - extended C4.8
            - Pruning
                1. minNumObject
                2. confidenceFactor
                3. subtreeRaising
            - Feature Subset Reduction using Decision Tree
            - Testing
                1. Testing against training data
                2. Testing against test data
                3. Percentage Splitting/Hold out Method
                4. Cross validation/ Repeated Hold out Method
        - ID3 Tree Classifier
        - Naïve Bayes Classifier
            - Handling Zero Frequency Issue
        - Rule Based Classifier – oneR
            - Changing the split attribute
    - Test for Over-Fitting in Rule based Classifier!
        - Changing Bucket Size

DATA CLEANING

Tool	**: Data Wrangler (available online only)**
Data Set Size	**: 8124**
Attributes	**: 22 + 1**
Missing values	**: In stalk-root 11th attribute was marked with '?'**

How did we handle missing values?

- Copied the attribute value of the tuple just above it.
- Since the data set was not too large we didn't adopt deleting the tuples.
- Data Wrangler supports only 1000 rows & 40 columns at a time. We could enter only 25 tuples, so we divided the dataset among the group members (2000 tuples each) and did the cleaning.

DATA MINING

Tool : WEKA

What is WEKA?

- Open Source Software Tool for Data Mining
- Developed at University of Waikato
- **WEKA** stands for Waikato Environment for Knowledge Analysis
- WEKA interfaces
    - Explorer
    - Experimenter - Performance Comparison (for large datasets)
    - Knowledge Flow - Graphical Interface
    - Simple CLI - Command-line Interface

Analysing the Data using WEKA.

- Opened the Data Set in WEKA (Explorer->Open file).
    - It shows various information like number of instances, attributes etc.
    - If we select one attribute, we can know its name, type, number of instances of each label, whether it contain missing values, how many distinct values are present and so on.
- A histogram for the selected attribute is also displayed.
    - Blue colour signifies poisonous and red colour signifies edible for each attribute we can see how many instances are poisonous and how many are edible.
    - Visualize all gives the histogram of all attributes.
    - By changing the Class, you can change the target class to some attribute and see the histogram for it. (By default last attribute is the label in WEKA)
- Edit gives the option to make changes to the data set and save in .arff format.
    - We can also get a sorted view of the data set here.

Inference:

- Among the training dataset 3914/8124 tuples are poisonous and 4208/8124 are edible.
- In this stage we found out that the attribute “viel-type” took only value, hence we decided to remove that attribute.

Is our problem Classification or Clustering?

Our problem is a Classification based problem. The supporting reasons:

- **Classification**
  - Supervised Learning – We have the class labels assigned for our training set (independent) using which we create a MODEL that classifies test data.
  - Class Attribute - Discrete(Nominal) => Classification Problem
  - Class Attribute – Continuous(Numeric) => Regression Problem

Visualization using WEKA

- A plot matrix and a graph between various attributes are shown.
- We can also select a portion of from the graph which includes only some portion of the data set and analyse separately (Submit).
- Jitter adds randomness to points. We can view the instances corresponding to a cross by clicking on it.

Adding Predicted Attribute to Data Set

Preprocess -> Filter -> supervised -> attribute -> attributeclassification->(config) output prediction

We added the predicted attribute also to our data set so that with each tuple we can see the actual and predicted class labels.

Building a Classifier

1. **J48 Tree Classifier – extended C4.8**
   Splitting is based on Gain, such that entropy after splitting is less

   Test Against Training Data

   Scheme:weka.classifiers.trees.J48 -C 0.25 -M 2

Root	*: Odour*
# Leaves	*: 24*
# Nodes	*: 24+5*
Accuracy	*: 100% (as is tested against training set=>misleading results)*

   === Detailed Accuracy By Class ===

TP Rate	FP Rate	Precision	Recall	F-Measure	ROC Area	Class
1	0	1	1	1	1	p
1	0	1	1	1	1	e

   === Confusion Matrix ===

   ```
    a      b        <-- classified as
   3916    0          |   a = p
   0       4208     |   b = e
   ```

   Inference:

   - Actual and Predicted results where same.
     - Diagonal values - classified correctly
     - Off Diagonal values - classified incorrectly.

   - **Feature Subset Reduction By Decision Tree**
     - Attributes odor, spor-print-color, population, gill-size, gill spacing forms the splitting attribute. Hence we reduced our attribute size to 5 in this stage.
     - Judiciously removing attributes can increase performance (accuracy) and comprehensibility.
       Pics: J48-Tree-Normal
       J48-Tree-Tree-InstancesPerLeave

   - **Pruning our decision tree!**
     Pruning may show less accuracy on training set but works well with test set
     - ❖ minNumObject = 2(default)
       - ➢ *Don't continue splitting if nodes get very small*
     - The minimum amount of data separation per branching.
     - Separating one instance from 100 instances doesn't give you much information, so you set a minimum amount of separation. However, if you have a node with 4 branches, and 2 of them end up with 0 instances, the other two with 50 each, the branching still produced information.
     - Increasing the #of instances per leaf to 20, training set accuracy decreased to 99.8031%. 16 tuples where wrongly classified (to see them Visualise Classifier Error)

Scheme:weka.classifiers.trees.J48 -C 0.25 -M 20

Confusion Matrix

```
a    b   <-- classified as
3900   16 |    a = p
0      4208 |    b = e
```

We got a 100% accuracy with % splitting => pruning worked well with test data

- #of instances per leaf is the minimum number of instances to be present in a leaf so as to consider the splitting attribute.
- On doing so, Gill-spacing was removed as it didn't have min 2 branches with more than 20 instances per leaf.

❖ confidencFactor = 0.25 (default)

➢ *Build a full tree first, and then work back from leaves applying a statistical test at each stage*

- No notable change observed changing this value

❖ subtreeRaising = true (default)

➢ *Prune an interior node raising the sub tree beneath up one level*

- No notable change observed changing this value

<u>Adding Test Set</u>

Classify -> Set -> open file -> more options -> output predictions

Test set and training set both in same format.

Training & Test data should be different for reliable results else misleading results.

Test Set For J48 Model Set

p, w, b, u, n, ?

y, w, n, u, v, p

a, c, n, k, s, p

```
=== Predictions on test split ===
inst#,    actual, predicted, error, probability distribution
    1         ?       1:p      +   *1      0
    2       1:p       1:p          *1      0
    3       1:p       2:e       +   0     *1
```

+ indicates error i.e actual and predicated doesn't match

Percentage Split/Hold out Method

Percentage Split 66% means that 66% of the data set is taken as training set and a classifier is made, on which the remaining 34% of the data set is tested.

- @66%
  - Test Set is 2762 tuples
  - Accuracy = 100%
- @ 20%
  - Test Set is 6492 tuples
  - Accuracy = 99.8923%
- @10%
  - Test Set is 7304 tuples
  - Accuracy = 99.8906%

Inference:

- Estimated Accuracy = 99.9276% (good!)
- For increased performance and reliability of the estimate the training data set should be of large size. So that you can create a better classifier.
- As % split increases from 10% to 20% and so on, general trend is increased accuracy.

Repeated Training & Testing

We changed the Random seed value for a particular % split. We got different accuracies each time we run.

- 25% (1) - 99.885%
- 25% (2) - 100%
- 25% (10)- 99.885%

Cross Validation/Repeated Holdout

- We try to get the estimate of the expected accuracy, to get a better classifier.
- Reduces variation of estimate
- Stratified Cross Validation(each portion contains equal proportion of class labels) reduces it even further(default in WEKA)
  - ❖ Cross Validation Folds = 10
    - Dataset is divided into 10 parts each parts taking turns to test and rest to train.
    - Average the results
    - In WEKA, 10 times 90% of data goes into training & 11th time 100% goes into training.

Inference:

- Our data set gave 100% accuracy for 10, 5 folds and with different random seeds.
  - Reason for this may be because our test data set & training data set was not independent. Hence we used the results of percentage split.
- If you have large data set use % split else cross validation with 10 fold. 20 fold takes more time

2. **Naive Bayes classifier**
   - All attributes contribute independently in decision making unlike OneR
     - *All attributes are equally important*
     - *Knowing the value of one attribute we can't say about the value of other attribute (Not often true!)*
   - We got a naive Bayes model showing how many times each value in the attributes appears under poisonous and edible.
     - Sample:

```
Naive Bayes Classifier
                          Class
Attribute                p      e
                     (0.48) (0.52)
=========================================
cap-shape
  x                   1709.0 1949.0
  b                     49.0  405.0
  s                      1.0   33.0
  f                   1557.0 1597.0
  k                    601.0  229.0
  c                      5.0    1.0
  [total]             3922.0 4214.0
```

   - WEKA resolves zero frequency problem by adding one to the count of each value in the attribute set.
     - We intentionally added a ZERO FREQUENCY attribute label and observed this. See the "intentional" attribute value. No tuple has such a value.

```
Naive Bayes Classifier
                          Class
Attribute                p      e
                     (0.48) (0.52)
=========================================
cap-shape
  x                   1709.0 1949.0
  b                     49.0  405.0
  s                      1.0   33.0
  f                   1557.0 1597.0
  k                    601.0  229.0
  c                      5.0    1.0
  intentional            1.0    1.0
  [total]             3923.0 4215.0
```

   - Accuracy = 95.678% => 7771 correctly classified instances and 353 incorrectly classified instances.

=== Confusion Matrix ===

```
   a    b   <-- classified as
 3587  329 |   a = p
   24 4184 |   b = e
```

3. **ID3 classifier**
   Same results of J48 classifier

4. **Rule based classifier**
   - **One R: One attribute does all the work**
     - Here we find out the **error rate** of each attribute.
     - Then choose the attribute with min error rate.
     - Accuracy = 98.5229%

```
=== Classifier model (full training set) ===

odor:
        p       -> p
        a       -> e
        l       -> e
        n       -> e
        f       -> p
        c       -> p
        y       -> p
        s       -> p
        m       -> p
(8004/8124 instances correct)
=== Confusion Matrix ===
   a    b   <-- classified as
 3796  120 |    a = p
   0   4208|    b = e
```

Inference:

- Here, **odor** is the attribute having least error rate and is the best splitting attribute. So was the case with J48 also.

Test for Over-fitting in Rule base classifier!

To test whether the oneR classifier over fits data i.e. works well with training data but not with independent test data.
Reason why you should not test against training data set.
Does it have too many branching? NO

- For OneR classifier
  - Odor as the splitting attribute accuracy remained to be 98.5229% for testing against training data set and cross validation
  - We removed Odor attribute. Now spore-print-color became the splitting attribute, accuracy remained to be 86.8045% for both testing against training data and cross validation.
  - Even if we reduce the over fitting BUCKETSIZE attribute from 6 to 1 no change in result was observed.
    - Min bucket size is used for discretising numeric attributes. Our data set has only nominal attributes.

Inference:

- OneR classifier for the data set doesn't shows over fitting.