COMP3605 Introduction to Data Analytics Cheat Sheet
based on lecture notes by Dr. Duc The Kieu

Christopher Sahadeo

2018-10-13

Contents

I Key Terms **3**

II Algorithms, Formulae and Examples **6**

1 Measures **6**

1.1 Distance Metrics 6

1.2 Means 7

1.3 Variance 7

2 Classification **8**

2.1 Decision Trees 8

2.1.1 Information Gain (ID3) 9

2.1.2 Gain Ratio (C4.5) 9

2.1.3 Gini Index (CART) 9

2.2 Naïve Bayesian Classification 10

2.3 Rule-Based Classification TBC 11

2.4 Evaluate Classifier Performance 12

3 Advanced Classification **13**

3.1 k-Nearest Neighbors Classification 13

3.2 Classification Using Frequent Patterns 13

3.3 Support Vector Machines (SVMs) 13

3.4 Classification by Backpropagation (ANNs) 13

3.5 Bayesian Belief Networks 13

3.6 Other Classification Methods 13

4 Clustering **14**

4.1 k-Means Algorithm (partitioning based) 14

4.2 DBSCAN Algorithm (density based) TBC 15

4.3 Evaluate Clustering Performance 16

5 Outlier Detection **17**

5.1 Classification-Based Approaches 17

5.2 Clustering-Based Approaches 17

5.3 Proximity-Based Approaches 17

5.4 Distance-Based Approaches 17

5.5 Density-Based Approaches 17

5.6 Statistical Approaches 17

Part I

Key Terms

Data classification	A two-step process, consisting of a learning (or training) step (where a classification model is constructed) and a classification step (where the model is used to predict class labels for given data)
Supervised Learning	The class label of each training tuple is provided
Unupervised Learning	The class label of each training tuple is not known
Accuracy	the percentage of test set tuples that are correctly classified by the (trained/learned) classifier
D	(parent set) is training set of class-labeled tuples
$X_{train} = (x_1, ..., x_n, y)$	A tuple in D; an attribute/feature
y	class label attribute
D_{Test}	Test data containing tuples $X_{test} = (x_1, ..., x_n)$ without the class attribute
m	The number of classes in D, each class is denoted as C_i (for $i = 1, ..., m$)
$C_{i,D}$	The set of tuples of class C_i in D
$\mid D \mid$	The number of tuples in D
$\mid C_{i,D} \mid$	The number of tuples in $C_{i,D}$
v	The number of distinct values of attribute A
a_j	A given value of attribute A (for $j = 1, ..., v$)
D_j	The set of tuples in D that have outcome a_j of A
p_i	The nonzero probability that an arbitrary tuple in D belongs to class C_i
$Info(D)$	Expected information needed to identify the class label of a tuple, **before** partitioning on A
$Info(D_j)$	Expected information needed to identify the class label of a tuple, **after** partitioning on A
$Info_A(D)$	Actual information **still** needed to identify the class label of a tuple, **after** partitioning on A
$Gain(A)$	The attribute with the highest value is chosen as the splitting attribute, biased toward tests with many outcomes
$GainRatio(A)$	The attribute with the highest value is chosen as the splitting attribute, overcomes the Information Gain bias, **but** it prefers unbalanced splits in which one partition is much smaller than the others
$Gini(D)$	The subset D_1 or D_2, upon binary split, that gives **minimum** Gini index for that attribute is selected as its splitting subset, overcomes the Gain Ratio bias, **but** is biased to multivalued attributes
H	A hypothesis that tuple X belongs to a class C
$P(H\|X)$	**posterior probability, posteriori probability** the probability that tuple X belongs to class C, given that we know the attribute description of X
$P(X\|H)$	The posterior probability that we can determine the description of X given that we know X belongs to class C

$P(H)$	**prior probability, priori probability** the probability that X belongs to C regardless of the description of X
$P(X)$	**A constant** the prior probability that we can determing the description of X regardless of what class X belongs to
$P(x_k \vert C_i)$	The number of tuples of class C_i in D having the value x_k for categorical attribute A_k, divided by $\vert C_{i,D} \vert$, the number of tuples of class C_i in D
R	A Rule, R : IF *condition* THEN *conclusion*
n_{covers}	The number of tuples covered by R, if the condition is satisfied
$n_{correct}$	The number of tuples correctly classified by R
$coverage(R)$	The percentage of tuples that are covered by the rule (i.e., their attribute values hold true for the rules antecedent)
$accuracy(R)$	The percentage of tuples (covered by the rule) that are correctly classified
TP	The number of positive tuples that were correctly labeled by the classifier
TN	The number of negative tuples that were correctly labeled by the classifier
FP	The number of negative tuples that were mislabeled as positive
FN	The number of positive tuples that were mislabeled as negative
Rule-Based Classification	Learned/trained model is represented as set of IF-THEN rules. Uses either **decision tree induction** or **sequential covering algorithm**.
Rule coverage (satisfied)	If condition (i.e., all the attribute tests) in rule antecedent holds true for given tuple. If a rule is satisfied, it is said to be **triggered**
Conflict Resolution	Two or more rules are triggered, we need a conflict resolution strategy to figure out which rule gets to fire and assign its class prediction to X
Size ordering	Assigns highest priority to triggering rule that has the toughest requirements
Classed-based Rule ordering	All the rules for the most prevalent (or most frequent) classes come first
Rule-based Rule ordering	Rules are organized into one long priority list, according to some measure of rule quality
Clustering	The process of grouping set of data objects into multiple clusters (or groups) so that objects within a cluster have high similarity, but are very dissimilar to objects in other clusters
cluster	A collection of data objects that are similar to one another within the cluster and dissimilar to objects in other clusters
p	An object in a cluster
c_i	The centroid (center/mean) of a cluster
m_i	The number of objects in cluster i
$dist(x, y)$	The Euclidean Distance between two points $x = (x_1, \ldots, x_d)$ and $y = (y_1, \ldots, y_d)$
E	Sum of squared error (SSE) between all objects in C_i and the centroid c_i
k-Means Algorithm	Distributes the objects in D into a set of k clusters $C_1, \ldots, C_k$ such that $C_i \subset D$ and $C_i \cap C_j = \emptyset$ for $1 \leq i, j \leq k$, designed to find spherical shaped clusters
DBSCAN	Density-Based Spatial Clustering of Applications with Noise, finds clusters of arbitrary shape, we can model clusters as **dense regions** in the data space, separated by **sparse regions**, discovers clusters of non-spherical shape, finds **core objects** (i.e., objects that have dense neighborhoods)
density	Number of objects close to o

ϵ-neighborhood	The space (region, area) within a radius ϵ centered at o, where $\epsilon > 0$ is a user-specified parameter
N	The set of objects in the ϵ-neighborhood of a point p under consideration
$\|N\|$	The number of elements in N
core object (point)	p is a core point if $\|N\| \geq MinPts$
Border point	q is not a core point (i.e., its ϵ-neighborhood contains less than $MinPts$ points, $\|N\| < MinPts$), but falls within the ϵ-neighborhood of a core point (i.e., its ϵ-neighborhood contains at least one core point)
Noise point	Any point that is neither a core point nor a border point

Part II

Algorithms, Formulae and Examples

1 Measures

1.1 Distance Metrics

Minkowski Denoted as $L_p(x,y), L^p(x,y), p \geq 2$

$$d_{Minkowski}(x,y) = \left(\sum_{j=1}^{d} \mid y_j - x_j \mid^p \right)^{\frac{1}{p}}$$

Manhattan (Minkowski with $p = 1$), denoted as as $L_1(x,y), L^1(x,y)$

$$d_{Manhattan}(x,y) = \sum_{j=1}^{d} \mid y_j - x_j \mid$$

Euclidean (Minkowski with $p = 2$), denoted as as $L_2(x,y), L^2(x,y)$

$$d_{Euclidean}(x,y) = \sqrt{\sum_{j=1}^{d} (y_j - x_j)^2}$$

Cosine

$$x \cdot y = \sum_{j=1}^{d} x_j y_j$$

$$\|x\| = \sqrt{\sum_{j=1}^{d} x_j^2}$$

$$d_{cosine}(x,y) = \frac{x \cdot y}{\|x\| \times \|y\|}$$

Mahalanobis

$$d_{Mahalanobis}(x,y) = \sqrt{(x-y)^T \sum^{-1} (x-y)}$$

where

- $\sum$ is a covariance matrix
- $\sum^{-1}$ is the inverse of $\sum$
- x^T is the transpose of x

1.2 Means

1.3 Variance

2 Classification

2.1 Decision Trees

$$O(n \times \mid D \mid \times \log(\mid D \mid))$$

Algorithm 1 Generate_decision_tree

Input: D, *attribute_list*, *Attribute_selection_method* $\in \{information\_gain, gain\_ratio, gini\_index\}$
Output: A decision tree

```
procedure GENERATE_DECISION_TREE(D, attribute_list, Attribute_selection_method)
    create a node N
    if X ∈ C,   ∀X ∈ D then
        return N as leaf node labeled with class C
    end if
    if attribute_list = ∅ then
        return N as leaf node labeled with the majority class in D
    end if
    apply ATTRIBUTE_SELECTION_METHOD(D, attribute_list) to  find  the "best" splitting_criterion
    if splitting_attribute is discrete  and  multiway splits allowed then
        attribute_list ← attribute_list − splitting_attribute
    end if
    for each outcome j of splitting_criterion do
        if D_j = ∅ then
            attach a leaf labeled with the majority class in D to node N
        else
            attach the node returned by GENERATE_DECISION_TREE(D_j, attribute_list) to node N
        end if
    end for
    return N
end procedure
```

2.1.1 Information Gain (ID3)

$$
\begin{aligned}
p_i &= \frac{\mid C_{i,D} \mid}{\mid D \mid} \\
Info(D) &= -\sum_{i=1}^{m} p_i \log_2(p_i) \\
p_{i,j} &= \frac{\mid C_{i,D_j} \mid}{\mid D_j \mid} \\
Info(D_j) &= -\sum_{i=1}^{m} p_{i,j} \log_2(p_{i,j}) \\
Info_A(D) &= \sum_{j=1}^{v} \frac{\mid D_j \mid}{\mid D \mid} \times Info(D_j) \\
Gain(A) &= Info(D) - Info_A(D) \\
0 \log_2 0 = 0
\end{aligned}
$$

Click here for Example

2.1.2 Gain Ratio (C4.5)

$$
\begin{aligned}
SplitInfo_A(D) &= -\sum_{j=1}^{v} \frac{\mid D_j \mid}{\mid D \mid} \times \log_2\left(\frac{\mid D_j \mid}{\mid D \mid}\right) \\
GainRatio(A) &= \frac{Gain(A)}{SplitInfo_A(D)}
\end{aligned}
$$

Click here for Example

2.1.3 Gini Index (CART)

$$
Gini(D) = 1 - \sum_{i=1}^{m} p_i^2
$$

Determines the splitting attribute and splitting subsets:

$$
Gini_A(D) = \frac{\mid D_1 \mid}{\mid D \mid} Gini(D_1) + \frac{\mid D_2 \mid}{\mid D \mid} Gini(D_2)
$$

Determines the reduction in impurtiy:

$$
\Delta Gini(A) = Gini(D) - Gini_A(D)
$$

Click here for Example

2.2 Naïve Bayesian Classification

$$P(x_k|C_i) = \frac{|\ C_{i,x_k}\ |}{|\ C_{i,D}\ |}$$

Laplacian Correction if $P(x_k|C_i) = 0$: If we have q counts to which we each add one, then we must remember to add q to the corresponding denominator used in the probability calculation

$$P(X|C_i) = \prod_{k=1}^{n} P(x_k|C_i)$$

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$$

Case 1 If $P(C_i)$ is known, then we calculate it by $P(C_i) = \frac{|\ C_{i,D}\ |}{|\ D\ |}$ and we maximize

$$P(X|C_i)P(C_i) > P(X|C_j)P(C_j) \text{ for } 1 \leq j \leq m, j \neq i$$

Case 2 If $P(C_i)$ is unknown, then we assume $P(C_i) = P(C_j)$ and we maximize

$$P(X|C_i) > P(X|C_j)$$

Click here for Example

2.3 Rule-Based Classification TBC

$$coverage(R) = \frac{n_{covers}}{\mid D \mid}$$
$$accuracy(R) = \frac{n_{correct}}{n_{covers}}$$

Algorithm 2 Sequential Covering Algorithm

Input: D, Att_vals
Output: A set of IF-THEN rules

```
procedure SEQUENTIAL COVERING(D, Att_vals)
    R_set = {}
    for each class c do
        repeat
            R = LEARN_ONE_RULE(D, Att_vals, c)
            remove tuples covered by R from D
            R_set = R_set + R
        until terminating condition
    end for
    return R_set
end procedure
```

$$FOIL\_Gain = pos' \times \left(\log_2 \frac{pos'}{pos' + neg'} - \log_2 \frac{pos}{pos + neg} \right)$$

2.4 Evaluate Classifier Performance

Predicted Class

		yes	no	Total
Actual class—	yes	TP	FN	P
	no	FP	TN	N
	Total	P'	N'	P + N

Confusion Matrix

$$accuracy = \frac{TP + TN}{P + N}$$

$$error\_rate = \frac{FP + FN}{P + N}$$

$$sensitivity, true\_positive\_rate, recall = \frac{TP}{P}$$

$$specificity, true\_negative\_rate = \frac{TN}{N}$$

$$precision = \frac{TP}{TP + FP}$$

$$F_1 = \frac{2 \times precision \times sensitivity}{precision + sensitivity}$$

$$F_\beta = \frac{(1 + \beta^2) \times precision \times sensitivity}{\beta^2 \times precision + sensitivity}, \beta \in \mathbb{R}, \beta \geq 0$$

Click here for Example

3 Advanced Classification

3.1 k-Nearest Neighbors Classification

3.2 Classification Using Frequent Patterns

3.3 Support Vector Machines (SVMs)

3.4 Classification by Backpropagation (ANNs)

3.5 Bayesian Belief Networks

3.6 Other Classification Methods

4 Clustering

4.1 k-Means Algorithm (partitioning based)

$$c_i = \frac{1}{m_i} \sum_{p \in C_i} p$$

$$dist(x, y) = \sqrt{\sum_{j=1}^{d} (y_j - x_j)^2}$$

$$E = \sum_{i=1}^{k} \sum_{p \in C_i} dist(p, c_i)^2$$

Algorithm 3 k-Means

Input: k, D

Output: A set of k clusters $C_1, \ldots, C_k$ such that $C_i \subset D$ and $C_i \cap C_j = \emptyset$ for $1 \leq i, j \leq k$

```
procedure k-MEANS(k, D)
    Randomly choose k objects from D as initial centroids
    repeat
        (re)assign each object to the cluster to which the object is the most similar
        recalculate the centroids
    until centroids do not change
end procedure
```

Click here for Example

4.2 DBSCAN Algorithm (density based) TBC

Density-Based Spatial Clustering of Applications with Noise

4.3 Evaluate Clustering Performance

5 Outlier Detection

5.1 Classification-Based Approaches

5.2 Clustering-Based Approaches

5.3 Proximity-Based Approaches

5.4 Distance-Based Approaches

5.5 Density-Based Approaches

5.6 Statistical Approaches