Business Anaytics

M. Tech QROR – 2nd yr (2024)

Classification

(Predictive Analytics)

Dr. Prasun Das

SQC & OR Unit

Indian Statistical Institute

e-mail: prasun@isical.ac.in

Rule Induction

Deducing **IF-THEN decision rules** from a dataset / decision trees explain an inherent relationship between the attributes and labels.

An example: "***IF*** it is 8 a.m. on a weekday, ***THEN*** highway traffic will be heavy" and "***IF*** it is 8 p.m. on a Sunday, ***THEN*** the traffic will be light."

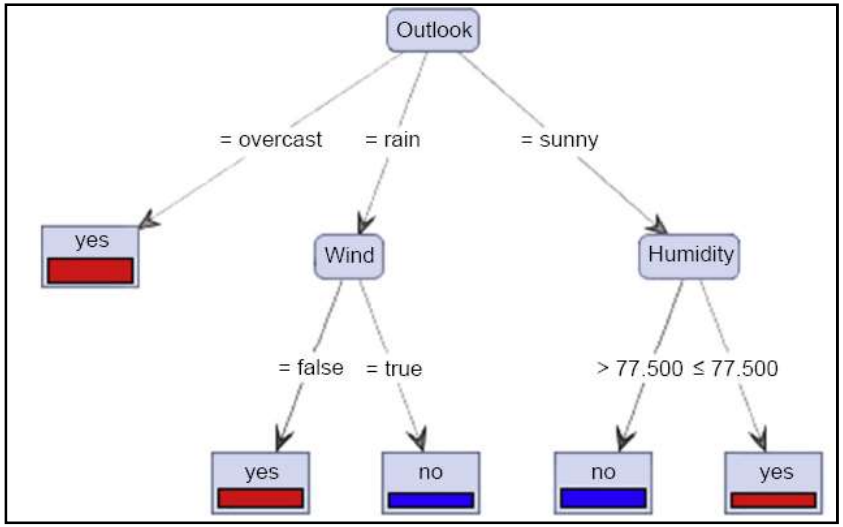
Apart from its use by classification of unknown data, rule induction is also used to describe the patterns in the data.

The description is in the form of simple *if-then* rules that can be easily understood by general users.

Rule Induction: Extraction of Rules

The easiest way to extract rules from a data set is from a decision tree that is developed on the same data set. If we trace back from the leaf to the root node, we can combine all the split conditions of decision tree to form a distinct rule.

Ex. Golf Data Set



Rule 1: if (Outlook = overcast) then Play = yes
Rule 2: if (Outlook = rain) and (Wind = false) then Play = yes
Rule 3: if (Outlook = rain) and (Wind = true) then Play = no
Rule 4: if (Outlook = sunny) and (Humidity > 77.5) then Play = no
Rule 5: if (Outlook = sunny) and (Humidity ≤ 77.5) then Play = yes

Rule Induction: Definition of Rules

Rule Set: R* = {$r_1 \cap r_2 \cap r_3 \cap \ldots \cap r_k$}** where, *each individual rule* r_i *is called a* ***disjunct *or* ***classification rule****.*

An individual ***disjunct*** can be represented as r_i = *if* (antecedent) *then* (consequent).

For ex., rule r_2: ***if*** (Outlook = rain) and (Wind = false) ***then*** Play = yes.

In this example, (Outlook = rain) and (Wind = false) are the ***antecedent*** of the rule. The condition of the rule can have many attributes and values, each separated by a logical AND operator.

Each attribute and its value is called the ***conjunct*** *of the rule.*

An example of a ***conjunct*** *is (Outlook* = rain).

The ***antecedent*** is a group of conjuncts with the AND operator. Each conjunct is a node in the equivalent decision tree.

Rule Induction: Properties of Rules

Mutually Exclusive Rule Set: This means that no example record will trigger more than one rule and hence the outcome of the prediction is definite.
Non-Mutually Exclusive Rule Set: If a record activates more than one rule in a rule set and all the class predictions are the same, then there is no problem. If the class predictions differ, ambiguity exists on which class is the prediction of the induction rule model.

Techniques used to resolve conflicting class prediction by more than one rule:

a) develop an ordered list of rules where, if a record activates many rules, the first rule in the order will take precedence.
b) where each active rule can "vote" for a prediction class, the predicted class with highest vote is the prediction of the rule set.

Exhaustive Rule Set: This means the rule set is activated for all the combinations of the attribute values in the record set, not just limited to training records.
Non-Exhaustive Rule Set: If so, then all bucket rule *"else Class = Default Class Value"* can be introduced to make the rule set *exhaustive.*

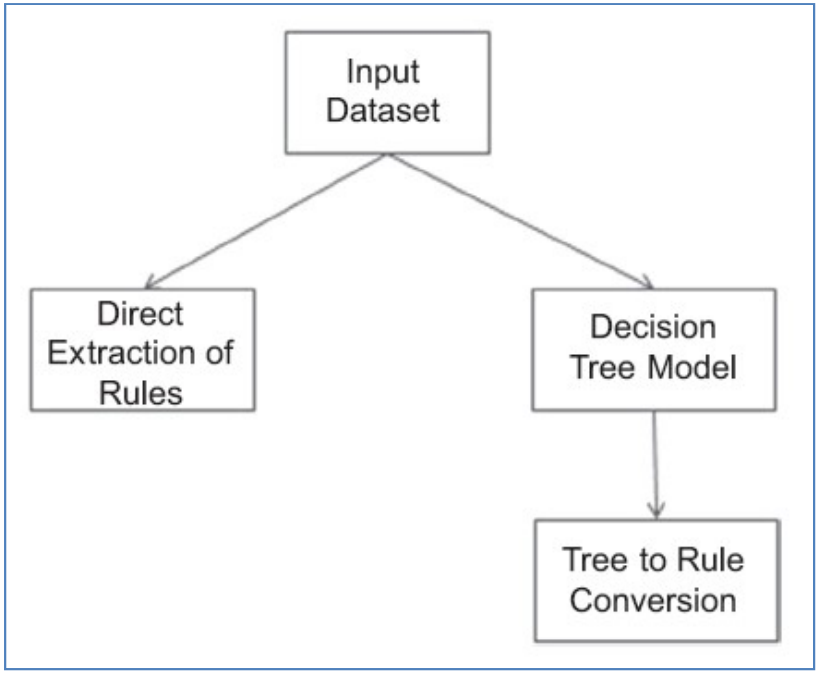
Rule Induction: Developing Rule Set

Direct method: This is built on leveraging the relationship between the attribute and class label in the data set.
Passive method: Deriving a rule set from a previously built *classifier*. Decision tree model is a passive or *indirect* approach.

Focus on Direct Method:

Sequential Covering Approach
(Tan *et al*. 2005)

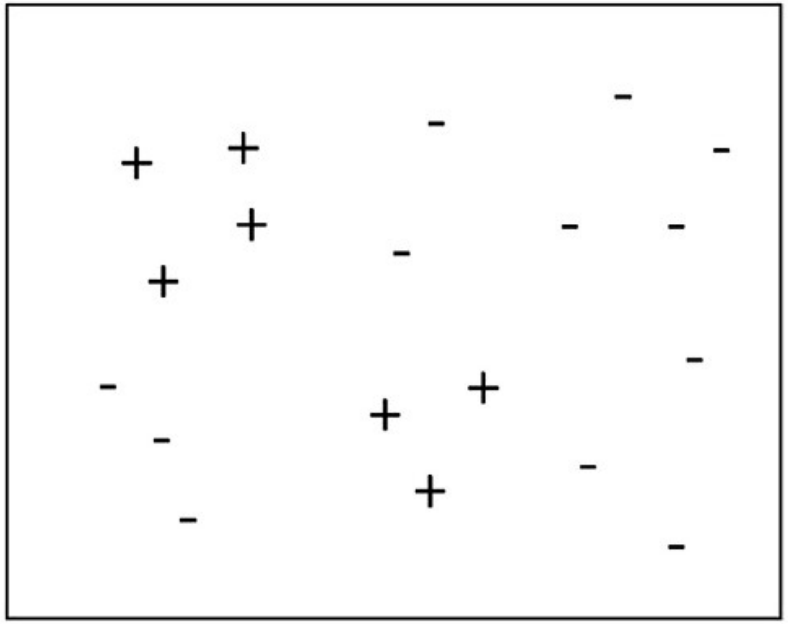
This is an iterative procedure of extracting rules from the data set. The sequential covering approach attempts to find all the rules in the data set class by class.



Rule Induction: Sequential Covering

Step 1: Class Selection

The algorithm starts with selection of class labels one by one. The rule set is class-ordered where all the rules for a class are developed before moving on to next class. The first class is usually the least-frequent class label. From the figure, the least frequent class is "+" and the algorithm focuses on generating all the rules for "+" class.

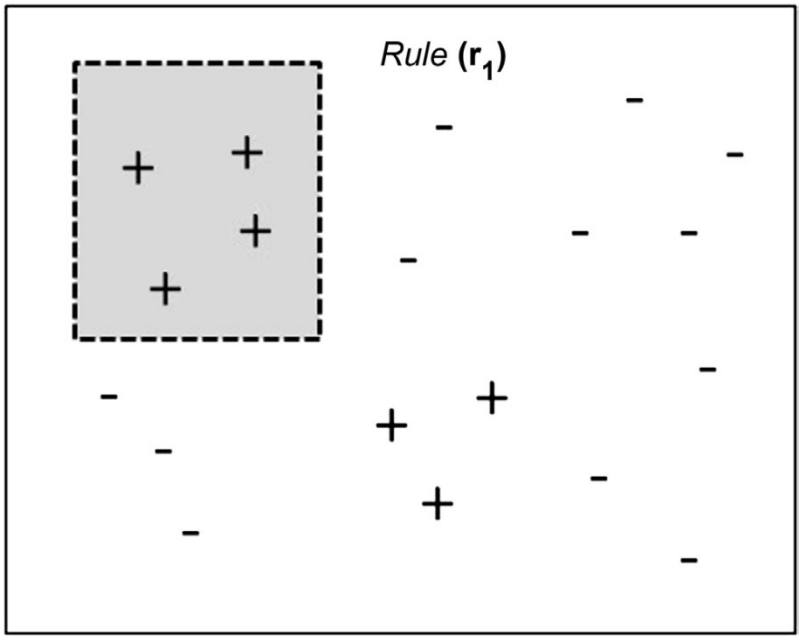


Rule Induction: Sequential Covering

Step 2: Rule Development

The objective in this step is to cover all "+" data points using classification rules with none or as few "–" as possible. In this example, **rule r_1** identifies the area of four "+" in the top left corner. Since this rule is based on simple logic operators in conjuncts, the boundary is rectilinear.

Once **rule r_1** is formed, the entire data points covered by **rule r_1** are eliminated and the next best rule is found from data sets. The algorithm grows in a *greedy fashion* using a technique called **Learn-One-Rule**. One of the outcomes of greedy algorithms that start with initial configuration is that they yield *local optima instead of a global optimum.*



Rule Induction: Sequential Covering

Step 3: Learn-One-Rule

Each **rule r_i** is grown by the learn-one-rule approach. Each rule starts with an empty rule set and conjuncts are added one by one to increase the rule accuracy.
Rule accuracy is the ratio of amount of "+" covered by the rule to all records covered by the rule:

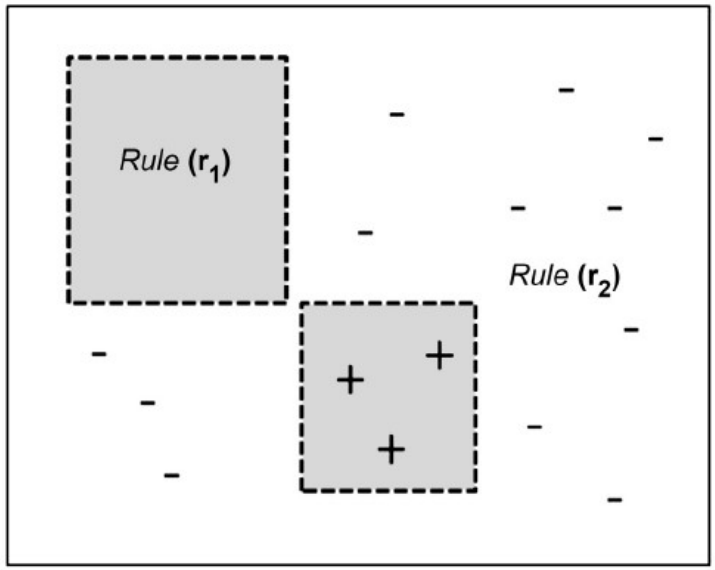
$$\text{Rule accuracy A}(r_i) = \frac{\text{Correct records covered by rule}}{\text{All records covered by the rule}}$$

Learn-one-rule starts with an empty rule set: if {} then class = "+". Obviously the accuracy of this rule is the same as the proportion of "+" data points in the data set. Then the algorithm greedily adds conjuncts until the accuracy reaches 100%.
If the addition of a conjunct decreases the accuracy, then the algorithm looks for other conjuncts or stops and starts the iteration of the next rule.

Rule Induction: Sequential Covering

Step 4: Next Rule

After a rule is developed, then all the data points covered by the rule are eliminated from the data set. The above steps are repeated for the next rule to cover the rest of the "+" data points. In the figure, **rule r_2** is developed after the data points covered by **r_1** are eliminated.



Step 5: Development of Rule Set

After the rule set is developed to identify all "+" data points, the rule model is evaluated with a data set used for pruning to reduce generalization errors. The **metric** used to evaluate the need for pruning is **$(p - n)/(p + n)$**, where p is the number of positive records covered by the rule and n is the number of negative records covered by the rule. The conjunct is iteratively removed, if it improves the metric. All rules to identify "+" data points are aggregated to form a rule group. **In multi-class problem, the previous steps are repeated with for next class label** (Saian & Ku-Mahamud, 2011).

Naïve Bayesian

Naïve Bayesian algorithms calculate the probability for each value of the class variable for given values of input variables. With the help of conditional probabilities, for a given unknown record, the model calculates the outcome of all values of target classes and comes up with a predicted winner.

The naïve Bayesian algorithm basically leverages the probabilistic relationship between the factors (attributes) and the class label (outcome). The algorithm makes a strong and sometimes naïve assumption of independence between the attributes, thus its name.

The independence assumption doesn't hold true always, but the simplicity and robustness of the algorithm offsets the limitation introduced by the independence assumption.

Naïve Bayesian: Bayes' Theorem

Bayes' theorem (Reverend Thomas Bayes, 1763 "Essay Towards Solving a Problem in the Doctrine of Chances"):

It provides a mathematical expression for how a degree of subjective belief changes to account for new evidence.

The naïve Bayesian algorithm is built on Bayes' theorem.

$$P(Y|\mathbf{X}) = \frac{P(Y) * P(\mathbf{X}|Y)}{P(\mathbf{X})}$$

P(X): *prob. of the evidence*
P(Y): *prior probability*
P(X/Y): *class conditional probability*
P(Y/X): *conditional (posterior) probability*

Calculating ***posterior probability*** is the objective of predictive analytics using Bayes' theorem. This is the likelihood of an outcome as we learn the values of the input attributes.

$$P(Y|\mathbf{X}) = \frac{P(Y) * \prod_{i=1}^{n} P(\mathbf{X_i}|Y)}{P(\mathbf{X})}$$

Naïve Bayesian

Ex. Golf Data Set

No.	Temperature (X1)	Humidity (X2)	Outlook (X3)	Wind (X4)	Play (Class label) Y
1	high	med	sunny	FALSE	no
2	high	high	sunny	TRUE	no
3	low	low	rain	TRUE	no
4	med	high	sunny	FALSE	no
5	low	med	rain	TRUE	no
6	high	med	overcast	FALSE	yes
7	low	high	rain	FALSE	yes
8	low	med	rain	FALSE	yes
9	low	low	overcast	TRUE	yes
10	low	low	sunny	FALSE	yes
11	med	med	rain	FALSE	yes
12	med	low	sunny	TRUE	yes
13	med	high	overcast	TRUE	yes
14	high	low	overcast	FALSE	yes

P(Y = no) = 5/14
P(Y = yes) = 9/14

Class Conditional Probability: P(X/Y)

Temperature (X1)	P(X1/Y = NO)	P(X1/Y = YES)
High	2/5	2/9
Medium	1/5	3/9
Low	2/5	4/9
Humidity (X2)	**P(X2/Y = NO)**	**P(X2/Y = YES)**
High	2/5	2/9
Medium	1/5	4/9
Low	2/5	3/9
Outlook (X3)	**P(X3/Y = NO)**	**P(X3/Y = YES)**
Overcast	0/5	4/9
Rainy	2/5	3/9
Sunny	3/5	2/9
Wind (X4)	**P(X4/Y = NO)**	**P(X4/Y = YES)**
FALSE	2/5	6/9
TRUE	3/5	3/9

Q. If a new, unlabeled test record has the attribute values Temperature= *high*, Humidity = *low*, Outlook = *sunny*, and Wind = *false*, what would be the class label prediction?

Naïve Bayesian

Q. If a new, unlabeled test record has the attribute values Temperature= *high*, Humidity = *low*, Outlook = *sunny*, and Wind = *false*, what would be the class label prediction?

$$P(Y = yes|\mathbf{X}) = \frac{P(Y) * \prod_{i=1}^{n} P(Xi|Y)}{P(X)}$$

$$= P(Y = yes) * \{P(Temp = high|Y = yes) * P(Humidity = low|Y = yes) * P(Outlook = sunny|\ Y = yes) * P(Wind = false|Y = yes)\}/P(X)$$

$$= 9/14 * \{2/9 * 4/9 * 2/9 * 6/9\}/P(X)$$

$$= 0.0094/P(X)$$

$$P(Y = no|\mathbf{X}) = 5/14 * \{2/5 * 4/5 * 3/5 * 2/5\}$$

$$= 0.0274/P(X)$$

After normalizing both the estimates by dividing both by (0.0094 + 0.027) we get

$$\text{Likelihood of (Play = yes)} = \frac{0.0094}{0.0274 + 0.0094} = 26\%$$

$$\text{Likelihood of (Play = no)} = \frac{0.0094}{0.0274 + 0.0094} = 74\%$$

In this case P(Y = yes | X) < P(Y = no | X), hence the prediction for the unlabeled test record will be Play = no.

Naïve Bayesian: Robustness & Limitations

Bayesian modeling is quite **robust** in handling missing values. If the test example set does not contain a value (say, *Temperature* is not calculated in the example set), the Bayesian model simply omits the corresponding class conditional probability for the outcomes.

Even though the naïve Bayes algorithm is quite robust to missing attributes, it does have a few limitations.

Issue-1: Incomplete Training Set: Use Laplace correction

Issue-2: Continuous Attributes: use discretization or pdf

Issue-3: Attribute Independence: Test for independence

Naïve Bayesian: Limitations

Issue-1: Incomplete Training Set

Problems arise when an attribute value in the testing record has no example in the training record.

In the **Golf dataset**, if a test example consists of the attribute value **Outlook = overcast**, then **P(Outlook = overcast | Y = no) = 0**.

Even if one of the attribute's class conditional probabilities is zero, by nature of the Bayesian equation, the entire posterior probability will be zero.

$$
\begin{aligned}
&P(Y = no|X) = P(Y = No) * \{P(Temp = high|Y = no) * P(Humidity = \\
&low|\ Y = no) * P(Outlook = overcast|Y = no) * P(Wind = false|Y = no)\}/P(X) \\
&= 5/14 * \{2/5 * 1/5 * 0 * 2/5\}/P(X) \\
&= 0
\end{aligned}
$$

In this case, P(Y = yes|**X**) > P(Y = no|**X**), so the test example will be classified as **Play = yes**.

If there are no training records for any other attribute value, like **Temperature = low** for **outcome *yes***, then probability of both outcomes, P(Y = no|**X**) and P(Y = yes|**X**), will also be zero, and an arbitrary prediction shall be made because of the dilemma!

Naïve Bayesian: Limitations

Issue-1: Incomplete Training Set : Laplace Correction

To mitigate this problem, assign small default probabilities for the missing records instead of zero. Then, the absence of an attribute value doesn't wipe out the value of P(**X**|Y), although it will reduce the probability to small number. This technique is called **Laplace correction**. This correction adds a controlled error in all class conditional probabilities.

In the **Golf data set**, if the example set contains **Outlook = overcast**, then **P(X|Y = no) = 0**. The *class conditional probability* for all the three values for **Outlook is 0/5, 2/5, and 3/5, where Y = no**. We can add controlled error by adding 1 to all numerators and 3 for all denominators, so the *class conditional probabilities* are 1/8, 3/8 and 4/8 (sum of all the class conditional probabilities is still 1).

Generically, the Laplace correction is given by

$$Corrected\ Probability\ P(X_i|Y) = \frac{0+\mu p_1}{5+p_1}, \frac{2+\mu p_2}{5+\mu}, \frac{3+\mu p_3}{5+\mu}$$

where $p_1 + p_2 + p_3 = 1$ and μ is the correction.

Naïve Bayesian: Limitations

Issue-2: Continuous Attributes

If an attribute has continuous numeric values instead of nominal values, convert the continuous values to nominal values by discretization and use the same approach as discussed. But discretization requires exercising subjective judgment on the bucketing range, leading to loss of information.
Instead, we can preserve the continuous values as such and use the probability density function [pdf] (Normal, Poisoon, Gamma etc.).

In the **Golf data set**, **temperature** and **humidity** are continuous attributes. So, compute the mean and s.d. for both class labels (Play = yes & Play = no) for temperature and humidity.
Now, If an unlabeled test record has a **Humidity value of 78**, we can compute the **probability density** using the **pdf** of Humidity, for both outcomes.

Naïve Bayesian: Limitations

Issue-2: Continuous Attributes

Consider **Golf data set**:

Play Value		Humidity (X2)	Temperature (X3)
Y = no	Mean	74.6	84
	Sd	7.89	9.62
Y = yes	Mean	73	78.22
	sd	6.16	9.88

Unlabeled test record (***x***): **Humidity value is 78**

Consider: Humidity follows *N*(μ, *σ*)**

Then, using the value *x* = 78,

P(temperature = 78|Y = yes) = 0.04, *if* **Play = yes**
P(temperature = 78|Y = no) = 0.05, *if* **Play = no**

Naïve Bayesian: Limitations

Issue-3: Attributes Independence

One of the fundamental assumptions in the naïve Bayesian model is *attribute independence*. Bayes' theorem is guaranteed only for independent attributes. This is why the technique is called "naïve" Bayesian, because it assumes attributes independence. However, in practice the naïve Bayesian model works fine with slightly correlated features (Rish, 2001).

How to tackle: Before applying the naïve Bayesian algorithm, it makes sense to remove strongly correlated attributes.

For numeric attributes, this can be achieved by computing a weighted correlation matrix. An advanced application of Bayes' theorem, called a Bayesian belief network, is designed to handle data sets with attribute dependencies.

For nominal attributes, the independence of two categorical (nominal) attributes can be tested by the *chi-square* (χ^2) test for independence.

Naïve Bayesian: Conclusion

- The Bayesian algorithm provides a probabilistic framework for a classification problem.
- It has a simple and sound foundation for modeling the data and is quite robust to outliers and missing values.
- This algorithm is deployed widely in text mining and document classification where the application has a large set of attributes and attribute values to compute.
- The naïve Bayesian classifier is very workable as an initial model in the proof of concept (POC) stage for an analytics project.
- It also serves as a good benchmark for comparison to other models.
- One major limitation of the model is the assumption of independent attributes, which can be mitigated by advanced modeling or decreasing the dependence across the attributes through pre-processing.
- The uniqueness of the technique is that it leverages new information as it arrives and tries to make a best prediction considering new evidence.