**Decision Trees**

Decision Trees are built using the approach known as

**Divide and Conquer**.

Feature values are used to split the data into smaller and smaller subsets of similar cases.

Rules

You can think of a rule as a path through a tree to a decision.

Classification Rules use the approach called

Separate and Conquer.

According to the author. . .

The process involves identifying a rule that covers a subset of the examples in the training data, and then

separates this partition from the remaining data. As rules are added, additional subsets of data are separated

until the entire dataset has been covered or no more examples remain.

Rules

Rules based learners usually use

nominal features

Greedy learners

Both

• Divide-and-conquer

• Separate-and-conquer

algorithms are known as greedy learners because data is used on a first-come, first serve basis.

• from Wikipedia greedy algorithms

One Rule algorithm

ZeroR decide to pick the highest probability outcome.

OneR develop a rule with each feature, use the one rule that has the best performance.

RIPPER algorithm

RIPPER

Repeated Incremental Pruning to Produce Error Reduction

1. Grow

2. Prune

3. Optimize

The information gain criterion is used to identify the next splitting attribute. When increasing rule’s specificity

no longer reduces entropy, then rule is immediately pruned.

RIPPER algorithm

For further details about the RIPPER Algorithm, see Cohen’s paper and the following presentation.

• Fast Effective Rule Induction

• A Ripper presentation

Uses best one feature rule that reduces entropy

Artificial Neural Networks

black box

Neural Networks are considered a black box process.

ANNs are based on complex mathematical systems.

But not a zero node NN is an alternative representation of the simple linear regression model.

y = mx + b

y(x) = w1x + w21

y(x) = f(w1x + w21)

artificial neurons

• ANNs are versatile learners that can be applied to nearly any learning task: classification, numeric

prediction, and even unsupervised pattern recognition.

• ANNs are best applied to problems where the input data and the output data are well-understood

or at least fairly simple, yet the process that relates the input to the output is extremely complex.

artificial neurons

ANNs are designed as conceptual models of human brain activity.

• incoming signals received by a cell’s dendrites

• signal transmitted through the axon

• synapse

• see page 207/221 and 208/222.

• activation function

1

artificial neurons

An artificial neuron with n input dendrites, with weights w on the inputs x, the activation function f,

and the resulting signal y is the output axon.

y(x) = f (Pn

i=1 wixi)

Activation functions

In a biological sense, the activation function could be imagined as a process that involves summing the

total input sinal and determining whether it meets the firing threshold.

If so, the neuron passes the signal on. Otherwise, it does nothing.

Activation functions

• threshold activation function

• unit step activation function

• sigmoid activation function - differentiable

• linear activation function

• Gaussian activation function - Radial Basis Function (RBF) network

• relu activation function

Activation functions

For many of the activation functions, the range of input values that affect the output signal is relatively

narrow.

The compression of the sinal results in a saturated signal at the high and low ends of very dynamic inputs.

When this occurs, the activation function is called a squashing function.

The solution to this is to use standardization/normalization of the features.

Network topology

The capacity of a neural network to learn is rooted in its topology, or the paterns and structures of

interconnected neurons.

Number of layers

A set of neurons called input nodes receive unprocessed signals directly from the input data. Each input

node is responsible for processing a single feature in the dataset.

The feature’s value is transformed by the node’s activation function. The signals resulting from the input

nodes are received by the output node, which uses its own activation function to generate a final prediction.

• single-layer network

• multilayer network

• hidden layers / deep learning

Direction of infomation travel

• feedforeward networks - commonly used

• feedback networks - theortical - not used

When people talk about applying ANNs they are most likely talking about using the multilayer preceptron

(MLP) topology.

Number of nodes in each layer

The number of input nodes is predetermined by the number of features in the input data.

The number of output nodes is predetermined by the number of outcomes to be modeled or the number of

class level in the outcome.

The number of hidden nodes is left to the user to decide prior to training the model.

More complex network topologies with a greater number of network connections allow the learning of

more complex problems.

But run the risk of overfitting.

Number of nodes in each layer

A best practice is to use the fewest nodes that result in adequate performance on a validation dataset.

It has been proven that a neural network with at least one hidden layer of suffiently many neurons is a

universal function approximator.

Training ANNs

Learning by experience.

The network’s connection weights reflect the patterns observed over time.

Training ANNs by adjusting connection weights is very computationally intensive.

An efficent method of traning an ANN was discovered, called backpropagation.

weights

How does the algorithm determine how much (or whether) a weight should be changed?

gradient descent

derivative of each activation function.

Modeling the strength of concrete

The author gives as an example of the use of ANNs.

The analysis of the concrete dataset.

**Itemset:** Set of items that occur together

**Association Rule:**Probability that particular items are purchased together.

Association Rules

Introduction

Do you know what an SKU is?

Answer: Stock Keeping Unit

Introduction

In this chapter we will learn about methods for identifying associations among items in transactional data.

This is known as market basket analysis.

Understanding association rules

The result of market basket analysis is a set of association rules.

For example,

{peanut butter,jelly} -> {bread}

Association rules are learned from subsets of itemsets.

Understanding association rules

Assocation rules were developed in the context of Big Data and database science and data mining for

knowledge discovery (KDD).

Looking for the needle in the haystack.

Assocation rules are unsupervised, so there is no need for the algorithm to be trained.

And there is no objective measure of performance for such rule learners.

Apriori algorithm

The completity of transactional data is what makes association rule mining a challenging task.

Transactional datasets are typically extremely large, both in terms of the number of transactions

and the number of features or items for sale.

The potential itemsets grows with the number of items for sale.

The good thing is that many itemsets are rare.

By igoring rare itemsets, it is possible to limit the seach for rules.

Apriori algorithm

The most widely used algorithm is the Apriori algorithm.

It employs a simple a priori belief as a guideline for reducing the association rule space, all subsets of a

frequent itemset must also be frequent. This is the Apriori property.

See the paper

Fast algorithms for mining association rules, Agrawal and Srikant (1994).

Or

A comparison of association rule discovery and bayesian network causal inference algorithms, Bowes, et. al.

Measuring rule interest

Whether or not an association rule is deemed interesting is determined by two statistical measures:

• support

P(X)

• confidence

P(Y |X)

Measuring rule interest

By providing minimum thresholds for each of these meterics and applying the Apriori principle, it is easy

to limit the number of rules reported.

Measuring rule interest - support

The support of an itemset measures how frequently it occurs in the data.

support(X) = count(X)

N

where N is the number of transactions in the database and count(X) is the number of transactions that

contain the itemset X.

Measuring rule interest - confidence

A rule’s confidence is a measurement of its predictive power or accuracy.

It is defined as the support of the itemset containing both X and Y divided by the support of the itemset

containing only X.

confidence(X ! Y ) = support(X,Y )

support(X)

The confidence tells us the proportion of transactions where the presence of item or itemset X results in the

presence of item or itemset Y .

Measuring rule interest - confidence

Note X ! Y is not the same as Y ! X.

Rules that have high support and high confidence are referred to as strong rules.

Building a set of rules with the Apriori principle

The Apriori principle states that all subsets of a frequent itemset must also be frequent.

The Apriori algorithm uses the Apriori principle to exclude potential association rules prior to actually

evaluating them.

The process of creating rules occurs in two phases:

• find all itemsets that meet a minimum support threshold

• create rules from these itemsets that meet a minimum confidence threshold

Example

The author gives an example of the use of Market Basket Analysis using transaction data to identify

frequently purchased groceries with association rules.

The example uses:

• unstructured data

• nosql

• sparse matrix

We will use the R packages

• arules

• arulesViz

Example

Lift is a metric used to measure how much more likely one item is to be purchased relative to its typical

purchase rate, given that you know another item has been purchased.

lif t(X ! Y ) = confidence(X!Y )

support(Y )

Here

lif t(X ! Y ) = lif t(Y ! X)

Clustering

Clustering

There are no labels in the dataset.

Clustering produces labels for similar groups in the data.

Applications of Clustering

• segmenting customers

• identifying patterns that fall outside of known clusters

• simplify larger datasets

Clustering

Unlabeled examples are given a cluster label and inferred entirely from the relationships within the data.

Clustering is unsupervised classification

Clustering produces ‘’new data”

Clustering

A problem with clustering is that the class labels produced do not have meaning.

Clustering will tell you which groups of examples are closely related but it is up to you to apply meaning to

the labels.

Semi-Supervised Learning

If we begin with unlabelled data, we can use clustering to create class labels.

From there, we could apply a supervised learner such as decision trees to find the most important

predictors of these classes.

k-means algorithm

The k-means algorithm is perhaps the most commonly used clustering method.

See the CRAN Task View: Cluster Analysis & Finite Mixture Models for a list of all the packages R has

related to Clustering and beyond.

k-means algorithm

k-means is not kNN

The similarity is that you need to specify a k.

The goal is to minimize the differences within each cluster and to maximize the differences between clusters.

k-means algorithm

The algorithm:

• starts with k random selected centers/centroids.

• assigns examples to an initial set of k clusters.

• it updates the assignments by adjusting the cluster boundaries according to the examples that fall into

the cluster.

• the process of updating and assigning occurs several times until making changes no longer improves the

cluster fit.

When using using k-means it is a good idea to run the algorithm more than once to check the robustness of

your findings.

Using distance

As with kNN, k-means treats feature values as coordinates in a multidimentional feature space.

Euclidean distance is used

dist(x, y) =

pPn

i=1(xi − yi)2

Using this distance function, we find the distance between each example and each cluster center.

The example is then assigned to the nearest cluster center.

Using distance

Because we are again using a distance measure, we need

• numeric features

• to normalize the features

Choosing the appropriate number of clusters

We need to balance the number of clusters k, try not to overfit the data.

Rule-of-thumb is to set k equal to

p

n/2.

Or use the elbow method

• homogeneity within clusters is expected to increase as additional clusters are added.

• heterogenity will decrease with more clusters.

Pick k at the elbow.

Other Clustering methods

There are many algorithms that can be used to cluster data:

• k-means kmeans

• Model based clustering Mclust

• Hierarchical clustering hclust

• density based clustering dbscan

Ensembles

caret

The caret package in R gives a unified interface to most of the packages in R that are used for data mining,

machine learning and statistical learning.

It also provides a very nice function for splitting data.

It provides functions for evaluating the results.

Helps with working with ensembles.

And and it gives parallel processing when appropriate.

zelig

On a totally unrelated package, there is the zelig package for basic statistics.

caret

The caret package is very useful for tuning models.

When tuning models, many different similar models need to be fit. This is a perfect situation for parallel

processing.

The caret package can parallel process with the parallel package in R.

Meta-learning

The idea is to combine several models to form a powerful team.

Build a strong team of weak learners.

Ensembles

All ensemble methods are based on the idea that by combining multiple weaker learners, a stronger

learner is created.

Bagging

bootstrap aggreating

Here sampling without replacement is used to produce the traning data, the other examples are said to

be out-of-bag, they are used to validation/testing.

Boosting

Ensembles of models are trained on randomly resampled data and voting (weighted) is used to determine the

final prediction used.

We implemented boosting eariler with C5.0

Random forests

Ensembles of decision trees are produces and voting is used again.

Here not only is the data resampled, but now the features are also sampled.

Decision tree forests.

Example

The author uses random forests to look at the credit data again.

Validation

We will discuss:

• holdout method: training, validation, test data

• cross-validation

• bootstrapping and the 0.632 bootstrap

• the R package caret

Estimating future performance

Split the data further. We have been using training and test datasets. We should add a third dataset,

validation.

Develop the model(s) on the training dataset and then validate on the validation dataset. Finally, after

choosing the final model(s) use the test dataset to see how the model(s) perform on unseen data.

Sampling

We have been using random sampling.

The caret package can be used to perform stratified sampling, which may blance the datasets better.

library(caret)

in\_train <- createDataPartition(credit$default,

p = 0.75)

Final Model

The author suggests:

Since the models trained on larger datasets generally perform better, a common practice is to retain the

model on the full set of data after the final model has been selected and evaluated, allowing the model

maximum use of available data.

Repeated holdout

To further evaluate the model, one can repeatly sample the training data and fit the model.

The final model would result from “averaging” over all of the models fit.

This process is referred to at repeated holdout.

Cross-validation

A formalization of the repeated holdout method is k-fold cross-validation.

Here k folds are randomly selected and the model is trained on each k-1 subsets and validated on the

remaining fold.

The final model would again result from “averaging” over all of the models fit.

Aside: this is similar to the leave-one-out method or jackknifing.

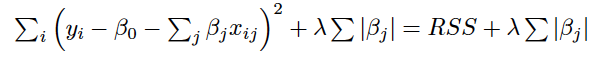
Bootstrapping

An alternative to k-fold cross-validation is bootstrap sampling.

Here the training and test datasets are created by sampling with replacement. The nonselected examples

make up the test datasets.

When using bootstrapping the process is repeated many times and the results “averaged” at the end.



Tuning Parameters and Ensembles

• Tuning parameters

• Ensembles

Tuning parameters can be used to improve the performance of a single model.

Ensembles can be used to build of team of learners that may have better performance than a single model.

Tuning Parameters

We know a little about tuning parameters from the C5.0 algorithm where we introducted Boosting.

And from kNN where tried different values of k.

Suppose we set a range of values for a tuning parameter and then fit the model for each of the values of

the tuning parameter, keeping a measure of performance. Then we can pick the best value of the tuning

parameter and the model produced.

caret package

The caret package in R gives functions that make tuning a model easy.

The functions train(), trainControl() and expand.grid()

The kappa value can be used to optimize.

Running experiments

In Machine Learning tuning over a grid is called running an experiment.

Ensembles

The author discusses meta-learners.

The technique of combinding and managing the predictions of multiple models falls within a wider set of

meta-learning methods that broadly encompass any technique that involves learning how to learn.

These may include:

• gradually improved performance by automatically iterating over design decisions

• self-modifying and adapting to learning tests

Ensembles

All ensemble methods are based on the idea that by combining multiple weaker learners, a stronger learner

is created.

Use a team of models or a committee of models.

Bagging

Boostrap aggregating or Bagging

A number of training datasets are generated by boostrap sampling the original training data. Boostrap

sampling is sampling the same number of rows as there are in the training data, with replacement.

These datasets are used to generate a set of models using a single learning algorithm.

The models’ predictions are combined using voting (for classification) or averaging (for prediction).

Bagging neads unstable learners. So bagging is often used with decision trees.

Boosting

Boosting uses ensembles of models trained on resampled data (re-weighted datasets) and a vote to

determine the final classification or average for a prediction.

The resampled datasets in boosting are constructed specifically to generate complementary learners, and

the vote is weighted based on each model’s performance rather than giving each an equal vote.

AdaBoost

AdaBoost or adaptive boosting.

The algorithm is based on the idea that generating weak learners that iteratively learn a larger portion of the

difficult-to-classify examples in the training data by paying more attention (that is giving more weight) to

the often misclassified examples.

Other Boosting Algorithms

• Gradient Boosting Machines

• XGBoost

Random Forests

This methods combines the base principles of Bagging with random feature selection to add additional

diversity to the decision tree models.

After the ensemble of trees is generated, the model uses a vote to combine the trees’ predictions.

Because random forests use only a small, random portion of the full feature set, it can handle extremely

large datasets.

Regression

The main idea with Regression is to model the relationship between a dependent variable and an independent

variable(s).

To make numeric predictions.

Dummy Variables

In R the lm() function that is used to fit linear regression models it knows about dummy variables. There is

no extra work that is need to include categorical variables into a regression model. This is because when a

categorical variable is a factor in R, the lm() function knows the dummy variables to use.

Understanding Regression Trees and Model Trees

Last Chapter, Trees were used for Classification.

This Chapter, Trees are used for Numeric Prediction.

CART

One type of tree for prediction is CART, Classification and Regression Trees.

This is a bit of a misnomer, Linear Regresion methods are not used. Predictions are made based on the

average value of examples that reach a leaf.

Model Trees

A second type of tree for prediction is known as Model Trees.

These were developed later, are less widely used but may be more powerful.

A multiple linear regression model is built from the exmples reaching that node.

Trees are an alternative to Regression Modeling

Trees can make predictions and can be considered as an alternative to regression modeling.

How are Trees built

The data are partitioned using a divide-and-conquer strategy according to the feature that will result in

the greatest increase in homogeneity in the outcome after a split is performed.

For Classification Trees entropy is used.

For Numeric Decision Trees statistics such as standard deviation are used.

Logistic Regression

Why not linear regression?

The author applies linear regresion to the launch data. In this data set the dependent variable is distress\_ct.

This variable has only 3 categories.

And making predictions less than 0 or greater than 3 does not make much sense.

Logistics Regression or Multinomial Regression would make more sense.

These are Generalized Linear Models (GLMs).

An excellent introduction to Logisitic Regression

Here is a link to an R-bloggers post How to perform a Logistic Regression in R.

The author of the post creates training and test data sets. And introduces the use of the ROC to evaluate

the fitted model.

Logisitic Regression

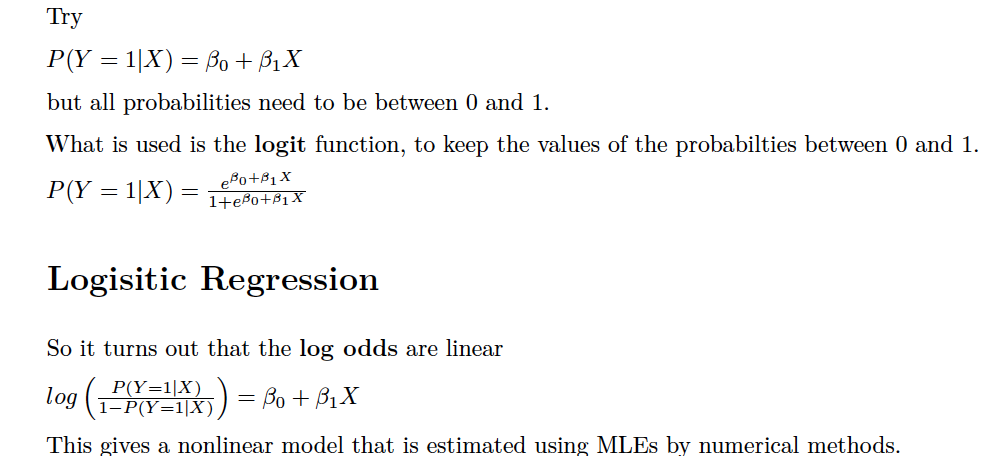
A logistic regression model, models a binary dependent variable

Y = 1 or Yes

or

Y = 0 or No

where P(Y = 1|X) is modeled in terms of the predictors X.



Multiple Logisitic Regression

Multiple Logistic Regression can be used when there is more than one predictor variable.

Categorical or Numeric variables can be used as predictors.

Evaluations

The AIC is used to compare models.

The ROC curve is used to compare models.

The Area under the ROC is commonly used to evaluate and compare models.

Logisitic Regression

Try Logistic Regression with the launch data and the credit data.

Evaluation

We have primarily talked about Classification methods, such as, kNN, Naive Bayes, C5.0, RIPPER,

CART, Logistic Regression, etc.

In the Classification setting we have used Accuracy/Success Rate to Evaluate the “usefulness” of an

algorithm.

Accuracy = TP+TN

TP+TN+FP+FN

So we have looked at the Confussion Matrix.

acc <- mean( pred == testy )

Introduction

We have started to look at Prediction methods, such as, Linear Regression, Multiple Linear Regression,

etc.

So we looked at “Accuracy” as the correlation between the test values of the response and the predicted/

fitted values from the model.

When using Prediction methods a quantitative response is predicted.

Question:

But Logistic Regression is used for Classification, right?

Answer:

Yes, but it uses the predicted probabilties.

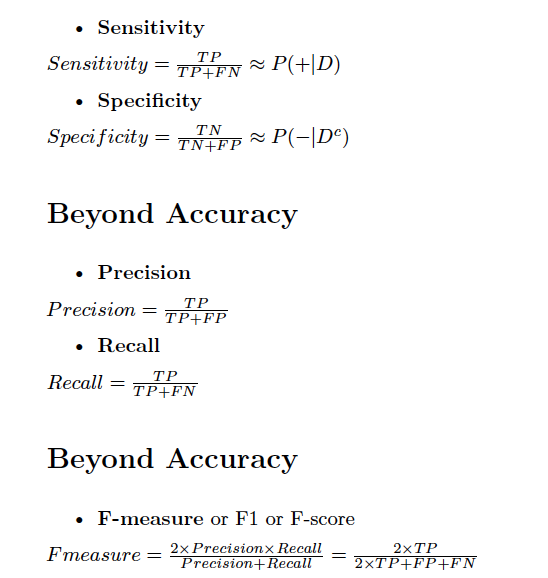
In R we can classify using the ifelse() function to convert the probabilities into 0 and 1.

ifelse(prob < 0.5, 0, 1)

Beyond Accuracy

• Kappa - adjusts accuracy by accounting for the possibility of a correct prediction by chance alone. So

should be a bit smaller than what we have discussed as Accuracy.



Visualizing Performance Tradeoffs - ROC

Visualizations can be very helpful for understanding how the performance of learning algorithms differ.

Useful for comparing two or more learners side-by-side.

The Receiver Operating characteristic (ROC) is commonly used.

To use the ROC we need:

1. the class values/labels

2. the predicted probabilities of the positive class

ROC - Sensitivity/Specificity plot

The ROC plots the Sensitivity versus 1 - Specificity.

For the MS Statistics students this is:

• True Positive Rate versus False Positive Rate

or

• Power versus \_.

ROC - Sensitivity/Specificity plot

No predictive value, 45 degree line

Perfect predictive value, up and across. 100% true positives with no errors.

ROC - AUC

The Area Under the Curve (AUC) is commonly used to compare Classifiers.

Holdout Method

• Training

• Validation

• Testing

Repeated Holdout

Cross-Validation

k-fold cross validation

10-fold cross validation

Train on 9 of the folds and test on the last. Average the accuracy measure.

Bootstrap sampling

Random sample with replacement. Train on the sample and test on the remaining examples.

error = 0.632 Å~ errortest + 0.368 Å~ errortrain