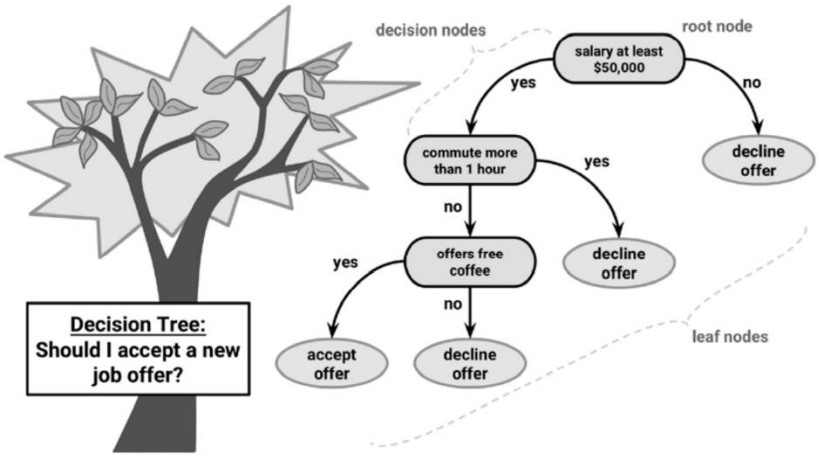
MODULE 3

Lecture Notes

APARNA S BALAN
DEPARTMENT OF COMPUTER APPLICATIONS
Vidya Academy of Science & Technology

Concept of Decision Tree

Decision tree learners are powerful classifiers, which utilize a tree structure to model the relationships among the features and the potential outcomes. A decision tree classifier uses a structure of branching decisions, which channel examples into a final predicted class value.



This tree, predicts whether a job offer should be accepted. A job offer to be considered begins at the **root node**, where it is then passed through decision nodes that require choices to be made based on the attributes of the job. These choices split the data across **branches** that indicate potential outcomes of a decision, depicted here as yes or no outcomes, though in some cases there may be more than two possibilities. In the case a final decision can be made, the tree is terminated by **leaf nodes** (**terminal nodes**) that denote the action to be taken as the result of the series of decisions.

The benefit of decision tree algorithms is that it outputs the resulting structure in a human-readable format. This makes decision trees appropriate for applications where the results need to be shared with others in order to inform future business practices. Decision trees are used for credit scoring models, marketing studies of customer behaviour, diagnosis of medical conditions etc. Decision trees can't be used for problems where the data has a large number of nominal features with many levels or it has a large number of numeric features. These cases may result in a very large number of decisions and an overly complex tree.

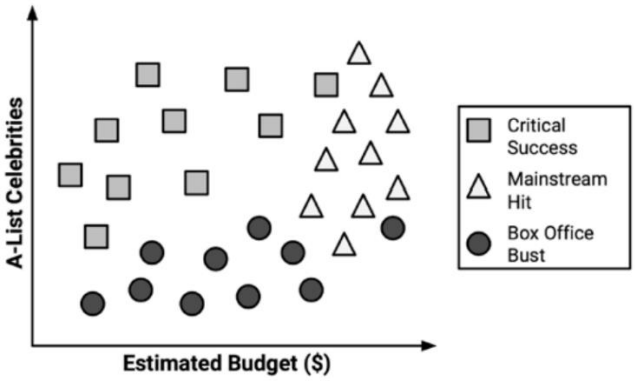
Divide And Conquer Approach

Decision trees are built using a heuristic called **recursive partitioning**. This approach is also known as **divide and conquer** because it splits the data into subsets, which are then split repeatedly into even smaller subsets, and so on and until the process stops when the algorithm determines the data within the subsets are sufficiently homogenous, or another stopping criterion has been met.

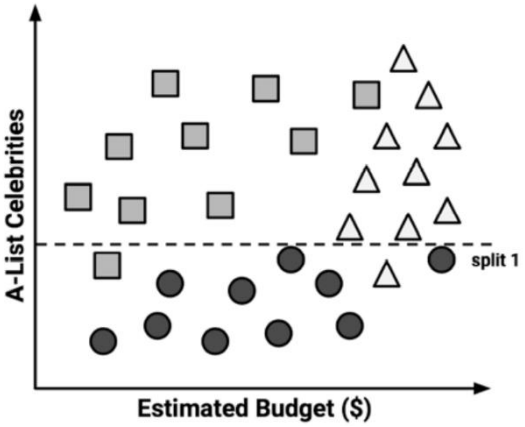
At first, the root node represents the entire dataset, since no splitting has transpired. Next, the decision tree algorithm must choose a feature to split upon; ideally, it chooses the

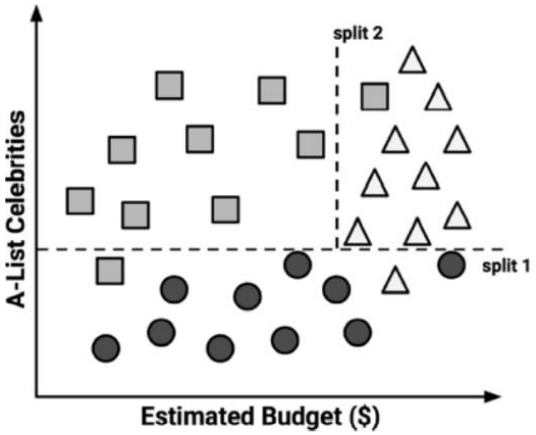
feature most predictive of the target class. The examples are then partitioned into groups according to the distinct values of this feature, and the first set of tree branches are formed.

Working down each branch, the algorithm continues to divide and conquer the data, choosing the best candidate feature each time to create another decision node, until a stopping criterion is reached. Divide and conquer might stop at a node in a case that: all (or nearly all) of the examples at the node have the same class, there are no remaining features to distinguish among the examples, the tree has grown to a predefined size limit.



Consider a situation where the decision tree algorithm is used to predict whether a potential movie would fall into one of three categories: Critical Success, Mainstream Hit, or Box Office Bust. By analyzing the past records, it was found that the film's estimated shooting budget is related to the number of A-list celebrities lined up for starring roles and the level of success. This scatter plot shows this relationship.

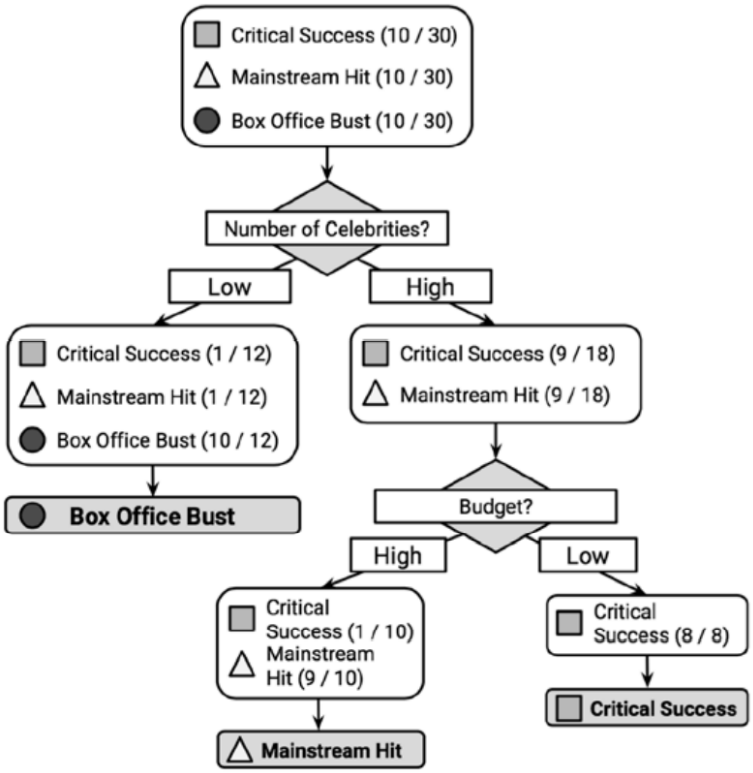




To build a simple decision tree from the above data, create the tree's root node by splitting the feature indicating the number of celebrities and partitioning the movies into groups with and without a significant number of A-list stars. Next, among the group of movies with a larger number of celebrities, make another split between movies with and without a high budget. Now the data is partitioned into three groups.

The group at the top-left corner of the diagram is distinguished by a high number of celebrities and a relatively low budget. At the top-right corner, majority of movies are box office hits with high budgets and a large number of celebrities. The final group, which has little star

power but budgets ranging from small to large, contains the flops. In order to avoid overfitting, the splitting process is stopped if more than 80 percent of the examples in each group are from a single class.



The model for predicting the future success of movies can be represented in a simple tree, as shown in the diagram. To evaluate a script, follow the branches through each decision until the script's success or failure has been predicted. Totally 30 records were used to build the tree. It has three leaf nodes and two features were used for branching purpose. Each of the intermediate node has two branches. The branches are labeled with the categorical values.

C5.0 Decision Tree Algorithm

The most well-known implementation of decision tree is the C5.0 algorithm. This algorithm was developed by computer scientist J. Ross Quinlan as an improved version of his prior algorithm, C4.5, which itself is an improvement over his Iterative Dichotomiser 3 (ID3) algorithm.

Strengths

- An all-purpose classifier that does well on most problems
- Highly automatic learning process, which can handle numeric or nominal features, as well as missing data
- Excludes unimportant features
- Can be used on both small and large datasets
- Results in a model that can be interpreted without a mathematical background (for relatively small trees)
- More efficient than other complex models

Weaknesses

- Decision tree models are often biased toward splits on features having a large number of levels
- It is easy to overfit or underfit the model
- Can have trouble modeling some relationships due to reliance on axis-parallel splits
- Small changes in the training data can result in large changes to decision logic
- Large trees can be difficult to interpret and the decisions they make may seem counterintuitive

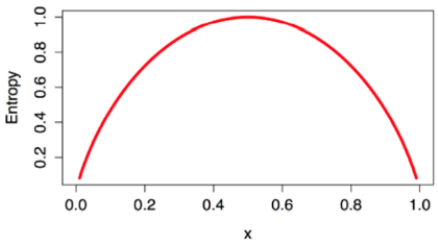
Choosing The Best Split

At each stage the decision tree has to identify a feature, which is used to split the data. The split has to be in such a way that the resulting partitions contains examples primarily of a single class. The degree to which a subset of examples contains only a single class is known as **purity**, and any subset composed of only a single class is called **pure**.

C5.0 uses **entropy** that quantifies the randomness, or disorder, within a set of class values, to find the best split. Sets with high entropy are very diverse and provide little information about other items that may also belong in the set, as there is no apparent commonality. The decision tree tries to find splits that reduce entropy, ultimately increasing homogeneity within the groups. Entropy is measured in bits. If there are only two possible classes, entropy values can range from 0 to 1. For n classes, entropy ranges from 0 to $\log_2(n)$. The minimum value indicates that the sample is completely homogenous, while the maximum value indicates that the data are as diverse as possible and no group. The entropy can be computed as

$$Entropy(S) = \sum_{i=1}^{C} -p_i \, log_2(p_i)$$

In this formula, S represent the dataset, the term c refers to the number of class levels and p_i refers to the proportion of values falling into class level i (i.e., number of data points belonging to one class divided by total number of data points). Consider an example 100 data points where the data points belong to two classes i.e. red (60 records) and white (40 records). The for the given dataset entropy can be calculated as $-\frac{60}{100} \times log_2\left(\frac{60}{100}\right) - \frac{40}{100} \times log_2\left(\frac{40}{100}\right) = 0.9709$.



If the proportion of examples in one class is x, then the proportion in the other class is (1 – x). If entropy for all the possible values of x is plotted, a curve will be obtained as shown in the figure. The peek of entropy is at x=0.50, a 50-50

split results in maximum entropy. As one class increasingly dominates the other, the entropy reduces to zero.

To use entropy to determine the optimal feature to split upon, the algorithm calculates information gain, which is the change in homogeneity that would result from a split on each possible feature. The information gain for a feature F is calculated as the difference between the entropy in the segment before the split (S) and the partitions resulting from the split (S_2)

$$InfoGain(F) = Entropy(S) - Entropy(S_2)$$

After a split, the data is divided into more than one partition. Therefore, the function to calculate Entropy(S_2) needs to consider the total entropy across all of the partitions. The following function can be used to calculate entropy of S_2. The total entropy resulting from a split is the sum of the entropy of each of the n partitions weighted by the proportion of examples falling in the partition (w_i).

$$Entropy(S_2) = \sum_{i=1}^{n} w_i \, Entropy(P_i)$$

The higher the information gain, the better a feature is at creating homogeneous groups after a split on this feature. If the information gain is zero, there is no reduction in entropy for splitting on this feature.

Pruning The Decision Tree

If the tree grows overly large, many of the decisions it makes will be overly specific and the model will be overfitted to the training data. The process of **pruning** a decision tree involves reducing its size such that it generalizes better to unseen data.

One solution to this problem is to stop the tree from growing once it reaches a certain number of decisions or when the decision nodes contain only a small number of examples. This is called **early stopping** or **pre-pruning** the decision tree. An alternative, called **post-pruning**(used by C5.0 algorithm) involves growing a tree that is intentionally too large and pruning leaf nodes to reduce the size of the tree to a more appropriate level. This is often a more effective approach than pre-pruning, because it is quite difficult to determine the optimal depth of a decision tree without growing it first.

In some cases, entire branches are moved further up the tree or replaced by simpler decisions. These processes of grafting branches are known as **subtree raising** and **subtree replacement**, respectively.

Classification Rules Learning

Concept Of Classification Rules

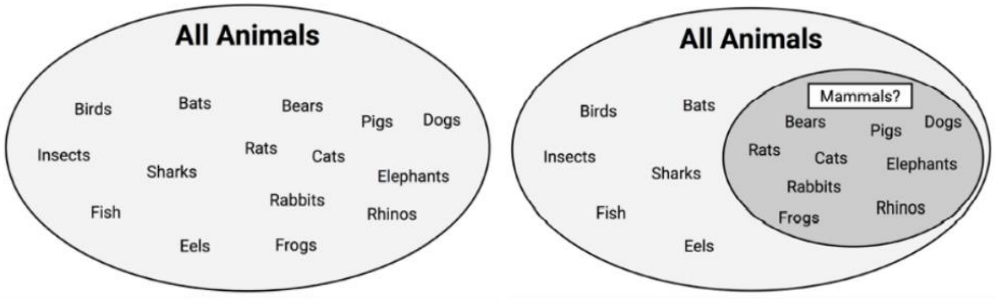
Classification rules represent knowledge in the form of logical if-else statements that assign a class to unlabeled examples. They are specified in terms of an antecedent and a consequent; these form a hypothesis stating that "if this happens, then that happens." The **antecedent** comprises certain combinations of feature values, while the **consequent** specifies the class value to assign when the rule's conditions are met.

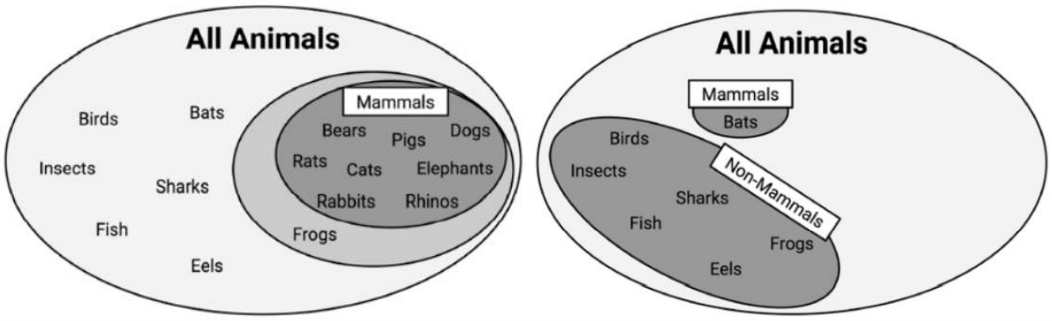
Rule learners are generally applied to problems where the features are primarily or entirely nominal. They do well at identifying rare events, even if the rare event occurs only for a very specific interaction among feature values.

Separate And Conquer Approach

Classification rule learning algorithms utilize a heuristic known as separate and conquer. The process involves identifying a rule that covers a subset of examples in the training data, and then separating this partition from the remaining data. As the rules are added, additional subsets of the data are separated until the entire dataset has been covered and no more examples remain.

Consider an example where rules are created to identify whether or not an animal is a mammal. The set of all animals can be depicted as large space, as shown in the following initial diagram. Homogeneous groups can be found using the available features. For example, using a feature that indicates whether the species travels via land, sea, or air, the first rule might suggest that any land-based animals are mammals, which is shown on the second image. But here frogs are classified as mammals but actually they are amphibians. Therefore, the rule needs to be a bit more specific by suggesting that mammals must walk on land and have a tail, which is depicted in the third figure.





An additional rule can be defined to separate out the bats, the only remaining mammal. Thus, this subset can be separated from the other data. A potential feature distinguishing bats from the other remaining animals would be the presence of fur. By using a rule built around this feature, bats can be correctly identified. At this point the rule learning process would stop since all of the training instances have been classified. So a total of three rules: 1) Animals that walk on land and have tails are mammals 2) If the animal does not have fur, it is not a mammal 3) Otherwise, the animal is a mammal

As the rules seem to cover portions of the data, separate and conquer algorithms are also known as covering algorithms, and the resulting rules are called covering rules.

The 1R Algorithm

The 1R algorithm (One Rule or OneR), constructs a classifier by selecting a single rule. Although this may seem overly simplistic, in empirical studies, the accuracy of this algorithm can approach that of much more sophisticated algorithms for many real-world tasks.

For each feature, 1R divides the data into groups based on similar values of the feature. Then, for each segment, the algorithm predicts the majority class. The error rate for the rule based on each feature is calculated and the rule with the fewest errors is chosen as the one rule.

For the Travels By feature, the dataset was divided into three groups: Air, Land, and Sea. Animals in the Air and Sea groups were predicted to be non-mammal, while animals in the Land group were predicted to be mammals. This resulted in two errors: bats and frogs. The Has Fur feature divided animals into two groups. Those with fur were predicted to be mammals, while those without fur were not predicted to be mammals. Three errors were counted: pigs, elephants, and rhinos. As the Travels By feature results in fewer errors, the 1R algorithm will return the following "one rule" based on Travels By:

1. If the animal travels by air, it is not a mammal
2. If the animal travels by land, it is a mammal
3. If the animal travels by sea, it is not a mammal

Animal	Travels By	Has Fur	Mammal
Bats	Air	Yes	Yes
Bears	Land	Yes	Yes
Birds	Air	No	No
Cats	Land	Yes	Yes
Dogs	Land	Yes	Yes
Eels	Sea	No	No
Elephants	Land	No	Yes
Fish	Sea	No	No
Frogs	Land	No	No
Insects	Air	No	No
Pigs	Land	No	Yes
Rabbits	Land	Yes	Yes
Rats	Land	Yes	Yes
Rhinos	Land	No	Yes
Sharks	Sea	No	No

Full Dataset

Travels By	Predicted	Mammal	
Air	No	Yes	✖
Air	No	No	
Air	No	No	
Land	Yes	Yes	
Land	Yes	Yes	
Land	Yes	Yes	
Land	Yes	Yes	
Land	Yes	No	✖
Land	Yes	Yes	
Land	Yes	Yes	
Land	Yes	Yes	
Land	Yes	Yes	
Sea	No	No	
Sea	No	No	
Sea	No	No	

Rule for "Travels By"
Error Rate = 2 / 15

Has Fur	Predicted	Mammal	
No	No	No	
No	No	No	
No	No	Yes	✖
No	No	No	
No	No	No	
No	No	No	
No	No	Yes	✖
No	No	Yes	✖
No	No	No	
Yes	Yes	Yes	
Yes	Yes	Yes	
Yes	Yes	Yes	
Yes	Yes	Yes	
Yes	Yes	Yes	
Yes	Yes	Yes	

Rule for "Has Fur"
Error Rate = 3 / 15

Strengths

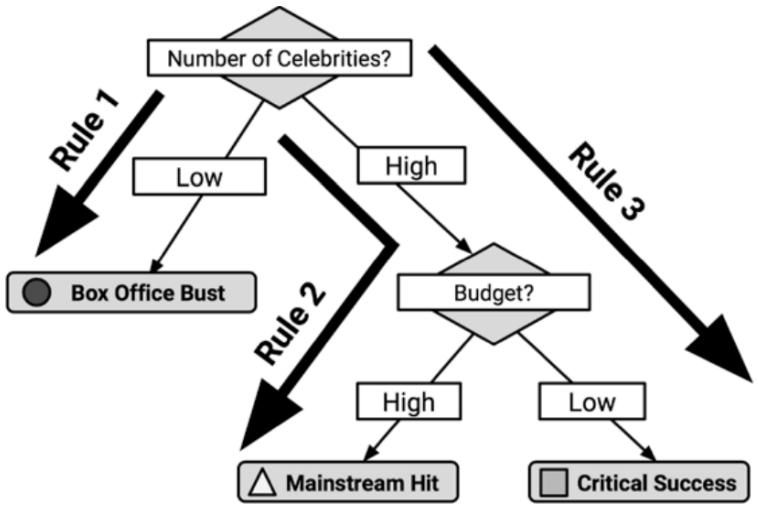
- Generates a single, easy-to-understand, human-readable rule of thumb
- Often performs surprisingly well
- Can serve as a benchmark for more complex algorithms

Weaknesses

- Uses only a single feature
- Probably overly simplistic

Rules from Decision Trees

Classification rules can also be obtained directly from decision trees. A series of decisions can be obtained by traversing the tree beginning at a leaf node and following the branches back to the root. These decisions can be combined into a single rule.



This figure shows how rules could be constructed from the decision tree to predict movie success. Following the paths from the root node down to each leaf, the rules would be:

1. If the number of celebrities is low, then the movie will be a Box Office Bust.

2. If the number of celebrities is high and the budget is high, then the movie will be a Mainstream Hit.
3. If the number of celebrities is high and the budget is low, then the movie will be a Critical Success.

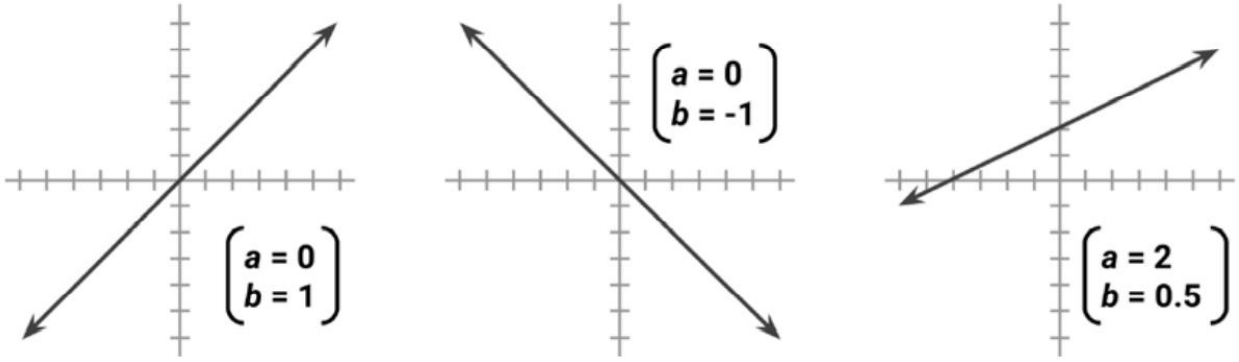
The main disadvantage of using a decision tree to generate rules is that the resulting rules are often more complex than those learned by a rule learning algorithm. But it is sometimes more computationally efficient to generate rules from trees.

Regression Methods

Regression includes techniques for estimating relationships among numeric data. It models the size and the strength of numeric relationships.

Concept of Regression

Regression is concerned with specifying the relationship between a single numeric dependent variable (the value to be predicted) and one or more numeric independent variables (the predictors). The dependent variable depends upon the value of the independent variable or variables.



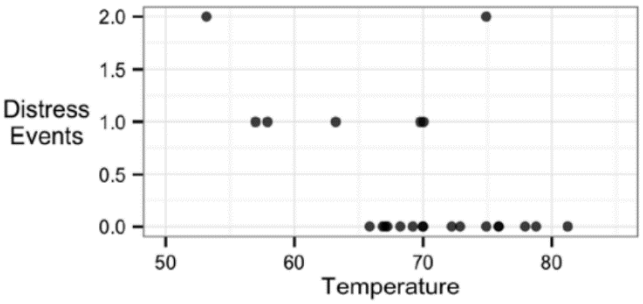
The simplest forms of regression assume that the relationship between the independent and dependent variables follows a straight line. In basic algebra lines can be defined in a **slope-intercept** form similar to $y = a + bx$. In this form, the letter y indicates the dependent variable and x indicates the independent variable. The **slope** term b specifies how much the line rises for each increase in x. Positive values define lines that slope upward while negative values define lines that slope downward. The term a is known as the **intercept** because it specifies the point where the line crosses, or intercepts, the vertical y axis. It indicates the value of y when $x = 0$.

Regression equations model data using a similar slope-intercept format. It tries to identify values of a and b so that the specified line is best able to relate the supplied x values to the values of y. There may not always be a single function that perfectly relates the values, so there must be some way to quantify the margin of error.

Regression analysis is commonly used for modeling complex relationships among data elements, estimating the impact of a treatment on an outcome, and extrapolating into the future. This method can be applied to cases like: examining how populations and individuals vary by their measured characteristics, quantifying the causal relationship between an event and the response, identifying patterns that can be used to forecast future behaviour given known criteria. Regression methods are also used for statistical hypothesis testing, which determines whether a premise is likely to be true or false in light of the observed data.

The **linear regression** models use straight lines. When there is only a single independent variable it is known as **simple linear regression**. If there are two or more independent variables, this is known as **multiple linear regression**, or simply "multiple regression". Both of these techniques assume that the dependent variable is measured on a continuous scale. **Logistic regression** is used to model a binary categorical outcome, while **Poisson regression** models integer count data. The method known as **multinomial logistic regression** models a categorical outcome; thus, it can be used for classification.

Simple Linear Regression



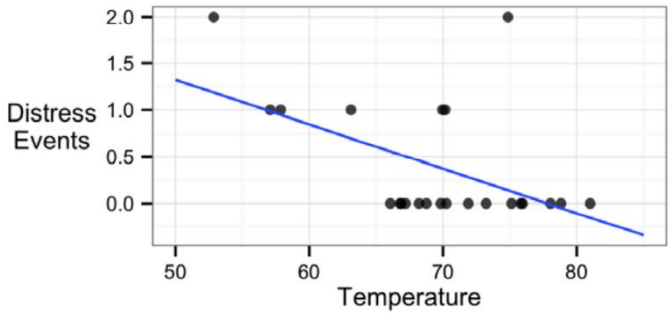
The following regression model demonstrates a link between temperature and O-ring (responsible for sealing the rocket joints) failure, and could forecast the chance of failure given the expected temperature at launch. The cold temperatures make the components more brittle and less able to seal properly, which would result in a higher chance of a dangerous fuel leak. To build the regression model, the data on launch temperature and component distresses from 23 successful shuttle launches are collected. The given scatterplot shows a plot of primary O-ring distresses detected for the 23 launches, as compared to the temperature at launch. The plot shows that launches occurring at higher temperatures tend to have fewer O-ring distress events.

A simple linear regression model defines the relationship between a dependent variable and a single independent predictor variable using a line defined by an equation in the following form: $\boldsymbol{y = \alpha + \beta x}$. The **intercept**, α (alpha), describes where the line crosses the y axis, while the

slope, β (beta), describes the change in y given an increase of x. The value of α and β can be obtained using the following formulae.

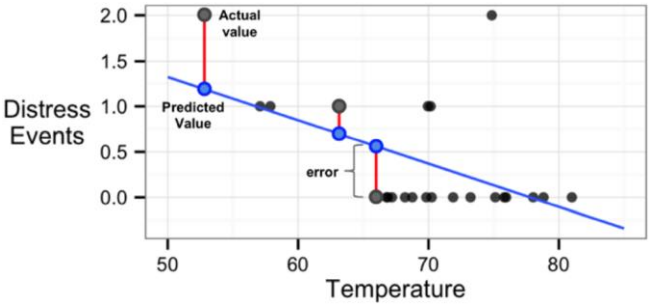
$$\beta = \frac{n\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i \sum_{i=1}^{n} y_i}{n\sum_{i=1}^{n} x_i^2 - (\sum_{i=1}^{n} x_i)^2}$$

$$\alpha = \frac{1}{n}\left(\sum_{i=1}^{n} y_i - \beta \sum_{i=1}^{n} x_i\right)$$



Suppose that the estimated regression parameters in the equation for the shuttle launch data are: α = 3.70 and β = -0.048. Hence, the full linear equation is $y = 3.70 - 0.048x$. A line can be plotted on the scatterplot like the given image. As the line shows, at 60 degrees Fahrenheit, number of distresses is just under one O-ring. At 70 degrees Fahrenheit, the failures are around 0.3. The model can be extrapolated all the way to 31 degrees—the forecasted temperature for the Challenger launch— can be expected about $3.70 - 0.048 * 31 = 2.212$ O-ring distress events. Assuming that each O-ring failure is equally likely to cause a catastrophic fuel leak means that the Challenger launch at 31 degrees was nearly three times riskier than the typical launch at 60 degrees, and over eight times riskier than a launch at 70 degrees.

Ordinary Least Squares Estimation



Ordinary Least Squares (OLS) is used to determine the optimal estimates of α and β. In OLS regression, the slope and intercept are chosen so that they minimize the sum of the squared errors, i.e., the vertical distance between the predicted y value and the actual y value. These errors are known as residuals which are illustrated for several points in the given diagram. The goal of OLS regression is to minimize the following equation which defines e (the error) as the difference between the actual y value and the predicted y value.

$$\sum (y_i - \hat{y}_i)^2 = \sum e_i^2$$

The solution for α depends on the value of β. It can be obtained using the following formula $\alpha = \hat{y} - \beta\hat{x}$. The value of β which results in the minimum squared error is $\beta =$

$\frac{\sum(x_i-\bar{x})(y_i-\bar{y})}{\sum(x_i-\bar{x})^2} = \frac{Cov(xy)}{Var(x)}$. Here $Cov(xy)$ denotes the covariance function for x and y and $Var(x)$ denotes the variance of x.

Correlations

The correlation between two variables is a number that indicates how closely their relationship follows a straight line. The default correlation type is Pearson's correlation coefficient. The correlation ranges between -1 and +1. The extreme values indicate a perfectly linear relationship, while a correlation close to zero indicates the absence of a linear relationship. The following formula defines Pearson's correlation. The ρ (rho) is used to denote the Pearson correlation statistic and σ (sigma) indicate the standard deviation of x or y.

$$\rho_{x,y} = Corr(x,y) = \frac{Cov(x,y)}{\sigma_x \sigma_y}$$

The correlation between the launch temperature and the number of O-ring distress events can be calculated using this formula. The computed value of correlation is - 0.51111. The negative correlation implies that increases in temperature are related to decreases in the number of distressed O-rings. The correlation also tells us about the relative strength of the relationship between temperature and O-ring distress. Because -0.51 is halfway to the maximum negative correlation of -1, this implies that there is a moderately strong negative linear association. As a rules of thumb, the value of correlation can be interpreted in the following way. Assign a status of "weak" to values between 0.1 and 0.3, "moderate" to the range of 0.3 to 0.5, and "strong" to values above 0.5 (these also apply to similar ranges of negative correlations).

Multiple Linear Regression

Multiple linear regression is used for most numeric prediction tasks because most real-world analyses have more than one independent variable. The goal is to find values of beta coefficients that minimize the prediction error of a linear equation. Here additional terms for additional independent variables are added to the previous equation of simple linear regression.

$$y = \alpha + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_i x_i + \varepsilon$$

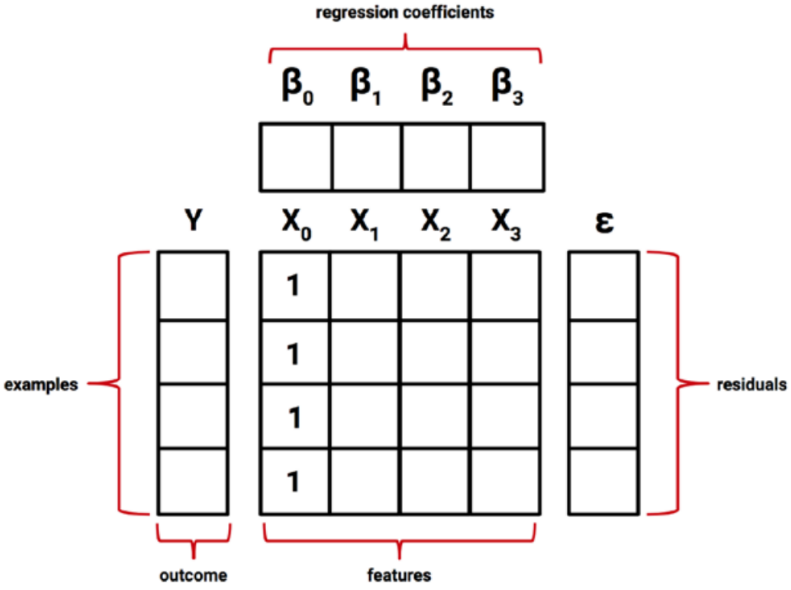
The dependent variable y is specified as the sum of an intercept term α plus the product of the estimated β value and the x values for each of the i features. An error term (ε - epsilon) has been added here as a reminder that the predictions are not perfect. The value of y changes by the amount β_i for each unit increase in x_i. The intercept α is then the expected value of y when the independent variables are all zero.

Since the intercept term α is no different than any other regression parameter, it is also sometimes denoted as β_0.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_i x_i + \varepsilon$$

The intercept is unrelated to any of the independent x variables. Now imagine β_0 as if it were being multiplied by a term x_0 , which is a constant with the value 1.

$$y = \beta_0 x_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_i x_i + \varepsilon$$



To estimate the values of the regression parameters, each observed value of the dependent variable y must be related to the observed values of the independent x variables using the regression equation.

The many rows and columns of data in the figure can be described in a condensed formulation using bold font matrix notation to indicate that each of the terms represents multiple values: $\boldsymbol{Y} = \boldsymbol{\beta X} + \boldsymbol{\varepsilon}$. The dependent variable is now a vector, Y, with a row for every example. The independent variables have been combined into a matrix, X, with a column for each feature plus an additional column of '1' values for the intercept term. Each column has a row for every example. The regression coefficients β and residual errors ε are also now vectors.

The aim is to solve for β, the vector of regression coefficients that minimizes the sum of the squared errors between the predicted and actual Y values. The best estimate of the vector β can be computed as $\widehat{\boldsymbol{\beta}} = (\boldsymbol{X^T X})^{-1} \boldsymbol{X^T Y}$, where the T indicates the transpose of matrix X, while the negative exponent indicates the matrix inverse.

Strengths

- By far the most common approach for modeling numeric data
- Can be adapted to model almost any modeling task
- Provides estimates of both the strength and size of the relationships among features and the outcome

Weaknesses

- Makes strong assumptions about the data
- The model's form must be specified by the user in advance
- Does not handle missing data
- Only works with numeric features, so categorical data requires extra processing
- Requires some knowledge of statistics to understand the model