Covenant College

PHY 492: Senior Integration Paper

Introduction to Machine Learning and Stewardship

Author
Roy Makkar Gabriel

Instructor
Dr. Curtis Stern

May 05, 2021

Abstract

The aim of this study is to provide a biblical perspective to the world of machine learning and an introduction to its uses. This study will also provide templates to build each algorithm and explain the mathematical derivation behind each model. It is important to understand that machine learning is a tool that has and will continue to drastically improve solving complex problems. However, it can easily be used with an sinful mindset to harmfully manipulate data and abuse its power.

Contents

List of Algorithms

1 Introduction

All models were built in python 3.8. All code templates can be accessed here: Prince's Sip. [51]

1.1 Biblical Mandate of Stewardship

Based on the PCA Historical Center, "A Biblical basis for stewardship must begin with die affirmation that God is sovereign and that we recognize Him as exercising divine ownership over all that there is (Ex. 19:5, Ps. 24:1, Ps. 50:10, Hag. 2:8)." [56] Stewardship is personal as we live life in conformity with God's declared will. We are obligated to be Christ centered in all that we do and in all that we are. This means we need to become servants. "As Jesus Christ served our deepest needs, so our stewardship is to serve the deepest needs of others." In other words, one way to show love to our neighbors is by using our God given talents and self-developed skills and being stewards of his creation. The bible says, "As each has received a gift, use it to serve one another, as good stewards of God's varied grace." (1 Pet 4:10) All that is ours is for the Lord and is to be used for His glory. In other words, as we pursue our calling, it is important to be biblically based.

1.2 Technology and the Creation Mandate

How should Christians interact with Technology? Is it truly the soul-consuming demon some alarmists make it out to be? In John Dyer's book *From the Garden to the City: The Redeeming and Corrupting Power of Technology*, the "garden" refers to the Garden of Eden in Genesis 2, while the "city" refers to the New Jerusalem in Revelation 21. [26] In other words, the book ambitiously covers the Bible's narrative from beginning to end with the goal of helping readers understand what the Bible might have to say about technology.

Dyer defines technology broadly (anytime we use tools to change the world) that it includes almost anything, including language, large language groups, or nearly any physical object (even the Lord's supper). He believes that "technology is one of the chief means by which humans attempt to create a world without God". Yet on the same page he writes, "God not only approves of but even helps with our technological development." Dyer never resolves this contradiction but simply accepts it as a fundamental paradox.

He chalks it up as a mystery surrounding God's relationship with the created order. He says, "the physical world, and what we make of it, is so important to God that he graciously chooses to use what we make in his plan of redemption." Using the predestination versus free will "contradiction", we can understand Dyer's point. Just as God knows what our next choice will be and does not interfere with the outcome, God provides us the free will to develop our technology knowing that some will drift away from him and some will realize the beauty in God's order. This brings us to the question, "is technology neutral?"

Technology, or the tools we use to change the world, are neutral by nature but a set of values emerge in its usage or function. The computer, for example, is a beautiful technology that allows us to do all sorts of things one can imagine or fail to imagine. Yet, we must realize that "our work machines and our porn machines are now the same machine." The computer itself does no harm unless we lead it into harm. It is not the machine that is embedded with a "negative tendency of usage" but society and the environment around us.

Thus, as Christian Industrial Engineers who see the world as different sets of systems who await optimization and flourishing, it is important to realize that those systems are not inherently evil by design unless its function or usage was for evil purposes. Thus, as the technology of Artificial Intelligence emerges, we must be stewards of this new creation and point it towards God's glory and not our own.

1.3 Introduction to Artificial Intelligence

In *Artificial Intelligence: A Modern Approach* by Stuart Russell and Peter Norvig artificial intelligence (AI) is defined as: "the designing and building of intelligent agents that receive percepts from the environment and take actions that affect that environment." [52]. Others may define AI as technology that is programmed to self-develop and mimic human behavior. If the tasks carried by AI agent is single tasked, it is considered to be Weak AI, otherwise a complex and human-like task is considered to be Strong AI.

1.3.1 Introduction to Machine Learning

Machine Learning is a subfield of AI and it is concerned with algorithm development that is optimized and learnable as the agent is exposed to more data. It is formally defined by the Analytics Software & Solutions (SAS) as the "method of data analysis that automates analytical model building. It is a branch of artificial intelligence based on the idea that systems can learn from data, identify patterns and make decisions with minimal human intervention." [39] It is important to note that all Machine Learning algorithms are denoted as AI algorithms. However, not all AI algorithms are denoted as Machine Learning algorithms.

Machine Learning is mainly divided into three categories: Supervised Learning, Unsupervised Learning, and Reinforcement Learning. Supervised Learning trains data based on input data and compares output based on true output values. Unsupervised Learning trains data based solely on input data with no labels. It attempts to discover patterns on its own. Reinforcement Learning improves on its action based on a reward system.

It is important to understand how this system works in order to develop it in a Godly manner. As seen from the above definitions, AI and Machine Learning are learnable technologies and thus output what they are taught. If they are taught to manipulate data and breach security, they will. Similarly, if they are taught to minimize costs and maximize efficiency, they will. All in all, as Christians, we must be aware of this fact and use this development as we further God's kingdom.

In order to build a Machine Learning algorithm, one must think of it like a system with a certain procedure to follow. It is time to place our engineering goggles and isolate this system.

1.3.2 Data Mining

First, we obtain the data from several sources and choose the ones that most closely relate to the problem the user or the client is attempting to solve. Data Mining is defined as the process of turning "raw data into useful information." [23] It is our job as engineers to look and use this data that is available to us. Sometimes, we must also experiment and create this data in a manner for others to use, develop, and analyze.

1.3.3 Data Pre-processing

Second, we process, or clean, the data that we mined. If the data was categorical or included text, it must be formatted in a way that the computer understands, such as binary or One Hot Encoding. If the data was numerical, taking care of missing values, anomalies, and outliers is important. Once the data is processed in a way such that the machine can understand it with no missing gaps in between, it is important to split the data into training and testing. The training data is used to train the machine learning agent to perform a certain task, such as predicting an output based on some input. During training, an objective function and a loss function are needed. The objective function is used to determine how the model or algorithm will predict the output and the loss function is the error metric that the model minimizes as it optimizes itself. The testing data is used to validate the performance of the model.

It is important to note that learning the parameters of a prediction function and testing it on the same data is a methodological mistake. A model that would just repeat the labels of the samples that it has just seen would have a perfect score but would fail to predict anything useful on unseen data. This situation is called overfitting. To further avoid ovefitting, yet another part of the dataset can be held out as a “validation set”. Training proceeds on the training set, after which evaluation is done on the validation set, and when the experiment seems to be successful, final evaluation is done on the test set. Underfitting refers to a "model that can neither model the training data nor generalize to new data." [44]

This stage can be easily manipulated because if a person were to clean the data in such a way that the model would learn to output manipulative or discriminative data. The Apostle Paul says, "The point is this: whoever sows sparingly will also reap sparingly, and whoever sows bountifully will also reap bountifully." (2 Cor 9:6) If the model is fed trash, the output will be trash; if the model is fed treasure, the output will be treasure.

1.3.4 Building the Model

Third, the user builds the model using an estimator. Estimators, or machine learning models include Regression models, Classification models, Clustering models, and several others. The user chooses the hyperparameters based on the goal of the objective function and trains the estimator on the training set, while analyzing the loss during training. The lower the loss the better. However, there must still exist some error to ensure no over-fitting is taking place. Loss functions include Mean Squared Error, Binary Cross Entropy, and several others.

1.3.5 Evaluating the Model

Finally, we evaluate the model on the testing set and analyze how well the model performed. If a regression model was used, continuous error functions are used such as Mean Squared Error or Adjusted R^2. If a classification model was used, a discrete error function is used such as Binary Cross Entropy or a truth table. Clustering algorithms can use both metrics.

This is one of the most crucial elements in developing a model. During evaluation, we need to understand how the model helps further the Kingdom and better the livelihood of our neighbors through technology. There are always costs to building a model. "For which of you, desiring to build a tower, does not first sit down and count the cost, whether he has enough to complete it?" (Luke 14:28) Is the model worth the cost? For example, say an engineer developed an AI algorithm for a self-driving car. As the car drives the speed limit in a friendly neighborhood, a few children jump out in the streets as they are playing. Assume that the car has one of two choices: Slam the breaks but not fast enough to save the children's lives or change direction and move onto the second lane where a head-on car collision takes place and kills the driver (and maybe others in the car as well). How should an engineer account for such an event? More importantly, why should an engineer think of such scenarios? It is the "how" that drives us engineers. However, it is just, if not more important, to spend some time understanding the "why" behind complex Machine Learning models that will be used to change the world one step at a time. It is important to have our "why" Christ centered in His values and in His perspective in order to ensure successful technological growth.

2 The Algorithms

2.1 Regression

We are called as Industrial Engineers to understand the design and structure of how systems work. If machine learning algorithms were seen as an isolated system, we are called to understand how these systems work. It is our duty to be able to build such models and explain their purpose in the industry for which they are used.

Regression is a statistical method to predict a single dependent value or multiple dependent values from a continuous independent variable or variables. The goal of the regression model is to build a mathematical equation that defines the dependent variable y in terms of the independent variable(s) x. [50]

2.1.1 Simple Linear Regression

Intuition

The Simple Linear Regression model is the most common method in regression. The linear model is used to predict a quantitative outcome variable y_i from one independent variable x_i. [37] The Simple Linear Regression equation follows the form of

$$y_i = b_0 + \beta x_i + \epsilon_i \tag{1}$$

where y_i is the dependent variable, x_i is the independent variable, β is the coefficient of the independent variable (also the slope of the trend line), ϵ_i is the error constant, and b_0 is the constant or intercept. The error constant will be ignored for simplification purposes.

The linear trend line (the best fit line) is created using an Ordinary Least Square algorithm from a scattered plot. Let y_i be an actual data point from the dataset and let $\hat{y_i}$ be a predicted value from the model. The Ordinary Least Square algorithm calculates the minimum of the sum of the square of the difference between y_i and $\hat{y_i}$ (see Equation 2 and Figure 1).

$$\arg\min_{\hat{y_i}} \sum_{i=1}^{n} (y_i - \hat{y_i})^2 \tag{2}$$

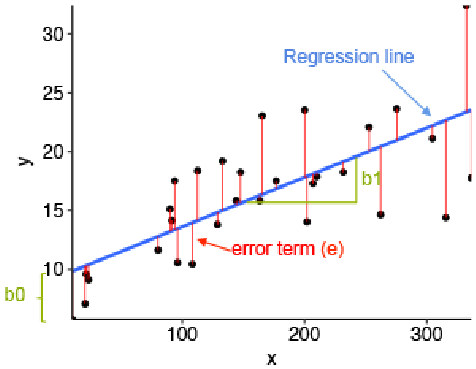

Figure 1: A Visualization of the Linear Regression Algorithm (Taken from [37]).

Building the Simple Linear Regression Model

```
Algorithm 1: Simple Linear Regression Model
  Data: [Real Estate Dataset 18]
  Result: A simple linear equation that predicts a median house
          value based on median income
  initialization;
  slice dataset to one dependent variable and one independent variable;
  split dataset into training set and testing set;
  train the simple linear regression model on the training set;
  predict the test set results;
  visualize the training and test results;
  evaluate the model;
```

2.1.2 Multiple Linear Regression

Intuition

Multiple linear regression is used to predict a quantitative outcome variable y_i from multiple independent variables x_1, x_2, ..., x_n (see Equation 3). Thus, there are multiple coefficients b_1, b_2, ..., b_n associated with each independent variable. The model is attempts to calculate the coefficients and the intercept value to create a linear equation associating the dependent variable to the independent variables. [41] The error constant will be ignored for simplification purposes.

$$y_i = b_0 + b_1x_1 + b_2x_2 + ... + b_nx_n + \epsilon_i \tag{3}$$

There are multiple assumptions associated with this type of regression: Linearity, homoscedasticity, multivariate normality, and lack of multicollinearity. Linearity refers to the mathematical relationship that can be represented as a straight line. Homoscedasticity refers to the error term in the relationship between the independent and dependent variables being the same across all values of the independent variables. [31] Multivariate normality assumes that the variables are normally distributed. Independence of errors Lack of multicollinearity assumes that the "independent variables are not highly correlated with each other." [17] Generally, correlation values below 0.8 are accepted.

In order to build this model, the p-value or significance value must be determined. The p-value is the "the evidence against a null hypothesis. The smaller the p-value, the stronger the evidence that you should reject the null hypothesis." [45] Generally, a p-value of less than of equal to 0.05 rejects the null hypothesis, which states that there is no difference between the variables of a dataset. The more critical the data, such as medical and health data, the less the p-value should be to ensure the model predicts with an accuracy score higher than 95%.

A three-dimensional multiple linear regression example can be seen in Figure 2.

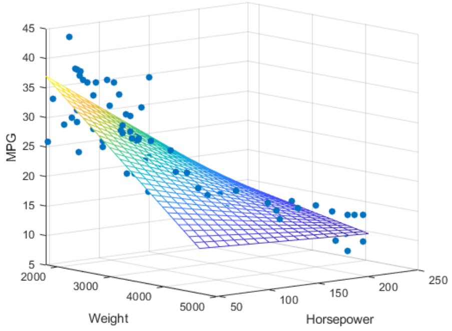

Figure 2: A Visualization of the Multiple Linear Regression Algorithm (Taken from [29]).

Building the Multiple Linear Regression Model

There are multiple ways in building this type of regression model: All in, backward elimination, forward selection, bidirectional elimination, and score comparison. The "All-in" case refers to using all the independent variables to predict the dependent variable. Backward elimination uses a series of steps to get rid of the independent variables predicting the dependent variable with the highest p-value. This will continually be done until all predictors have a p-value less than what was originally set (usually 0.05). Forward selection performs a series of simple linear regression for each independent variable individually with the dependent variable. Afterward, the predictor with the lowest p-value is kept and another predictor is added on as long as the new p-value is less than what was originally set (usually 0.05). Bidirectional elimination combines the backward elimination logic and the forward selection logic. Thus, two p-value significance levels must be chosen; one to keep the independent variable and one to remove the independent variable from the model. Finally, score comparison uses all strategies mentioned earlier and measures the goodness of fit using a chosen criteria such as the Akaike criterion or R^2. There will be $2^n - 1$ different combinations where n is the number of independent variables. The model with the optimal score is chosen.

For the sake of time and the computer machine being used, the Multiple

Linear Regression model will be built using backward elimination.

Algorithm 2: Multiple Linear Regression Model

```
Data: [Real Estate Dataset 18]
Result: A multiple linear equation that predicts a median house
        value based on median income
initialization;
select a significance level to stay in the model (e.g SL = 0.05);
fit the model with all the independent variables (the predictors);
determine the independent variable with the maximum p-value score;
$predictor = max\_ind\_var$;
while $predictor > SL$ do
    drop $predictor$ from dataset;
    fit the model with the new independent variables;
    determine the independent variable with the maximum p-value
      score;
    $predictor = new\_max\_ind\_var$;
end
predict the test set results;
evaluate the model;
```

2.1.3 Polynomial Regression

Intuition

Polynomial Regression is a form of a regression model where the relationship between the independent variables and the dependent variables is represented by a nth degree polynomial equation. See Equation 4 for more details.

$$y_i = b_0 + b_1 x + b_2 x^2 + ... + b_n x^n + \epsilon_i \tag{4}$$

The model fits an equation between the independent and dependent variables. As the degree of the polynomial changes, so does the accuracy of the curve. This means that the model can fit a wide range of curvature that approximates the dependent variable. However, there are risks of over-fitting and the data being skewed due to outliers in the data. This is important when deciding the optimal degree of the polynomial. [46] A visualization example can be seen in Figure 3.

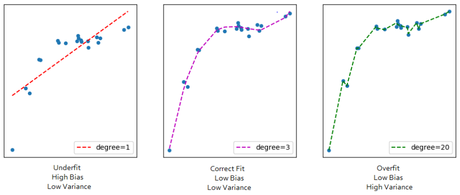

Figure 3: A Visualization of the Polynomial Regression Algorithm (Taken from [46]).

Building the Polynomial Regression Model

```
Algorithm 3: Polynomial Regression Model
  Data: [Real Estate Dataset 18]
  Result: A polynomial equation that predicts a median house value
          based on median income
  initialization;
  pre-process categorical data and drop NaN values;
  slice dataset to one dependent variable and multiple independent
    variables;
  split dataset into training set and testing set;
  train the polynomial regression model on the training set;
  $degree = 2$;
  while $degree < 5$ do
      fit the model with degree parameter = $degree$;
      predict the test set results;
      calculate $R^2$ score of the regression curve on the test set;
      $degree$ += 1;
  end
  evaluate the model by choosing optimal degree based on $R^2$ score;
```

2.1.4 Support Vector for Regression (SVR)

Intuition

The Support Vector for Regression model is a "non-parametric technique because it relies on kernel functions". [57] In ϵ-SV regression, where ϵ is the error constant mentioned in earlier regression models, the goal is to find a function that has at most ϵ deviation from the dependent variable targets y_i for all the training data and is simultaneously as flat as possible. [14] The model provides adaptability to define the extent to which an error is acceptable in the model and will find an "an appropriate line (or hyperplane in higher dimensions) to fit the data." [16] It is represented by equations Equation 5,

$$y = f(x) =< w, x > +b = \sum_{i=1}^{n} w_i x_i + b, y, b \in \mathbb{R}, x, w \in \mathbb{R}^n \tag{5}$$

where w is the coefficient vector that has been normalized using L2-Normalization, w_i are the elements from the corresponding normalized coefficient vector, x_i are the elements of the independent variables, and b is the intercept. [12]

In contrast to the previous regression models, where Ordinary Least Square (OLS) method was used as a loss function, the objective function of the SVR is to minimize the L2-Normalization of the coefficient vector. L2-Normalization normalizes the dataset values in where each row has its sum of squares add up to one. See Equation 6 for more details.

$$\|v\| = \sqrt{\sum_{i=1}^{n} |x_i|^2} \tag{6}$$

In addition, the model sets the absolute error less than or equal to the maximum error, ϵ, which can be tuned to optimize the accuracy of the model. Thus, the objective function of the model is to minimize the coefficient vector using L2-Normalization (Equation 7 with constraints specified in Equation 8. For any value that exceeds the error ϵ, the deviation from the margin is denoted by ξ. The kernel function associated with the SVR is the tuning parameter that maps the data into a higher dimensional space to achieve a higher accuracy.

$$\min_{w} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{n} |\xi_i| \tag{7}$$

$$|y_i - w_i x_i| \leq \epsilon + |\xi_i| \tag{8}$$

Note that w in Equation 6 is the coefficient vector that has been normalized using L2-Normalization, y_i are the elements of the dependent variable, w_i are the elements from the corresponding normalized coefficient vector, and x_i are the elements of the independent variables. A visual illustration can be seen in Figure 4. Furthermore, C in Equation 7 is another hyperparameter that can be tuned where an increase in C causes an increase in the tolerance for points outside of the margins of the model.

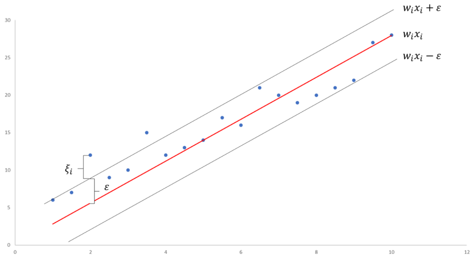

Figure 4: SVR Illustration (from [Sharp 16]).

Building the SVR Model

The SVR kernel being used is the rbf kernel (see Equation 9). Note that $\mathbf{x}$ and $\mathbf{x}'$ are two feature vectors. (More details on rbf kernel in subsubsection 2.2.3).

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{\|\mathbf{x} - \mathbf{x}'\|^2}{2\sigma^2}\right) \tag{9}$$

This kernel is being used to map the features to a higher dimension and increase the accuracy of the model by building a non-linear Support Vector

Regression model. The hyperparameter C was set to 1.

```
Algorithm 4: Support Vector Regression Model
  Data: [Real Estate Dataset 18]
  Result: An SVR model that predicts a median house value based
          on median income
  initialization;
  pre-process categorical data and drop NaN values;
  slice dataset to one dependent variable and multiple independent
    variables;
  split dataset into training set and testing set;
  feature scale the the training and testing sets (X and y separately);
  train the SVR model on the training set using a specified kernel;
  predict the test set results;
  calculate $R^2$ score on the predicted test set;
```

2.1.5 Decision Tree Regression

Intuition

A Decision Tree Regression model is a "supervised machine learning model used to predict a target by learning decision rules from features." [20] The model uses the process of Inductive Reasoning [32] by making decisions through asking a series of questions. In simple terms, the algorithm splits the data using lines for two-dimensional data, planes for three-dimensional data and so on. Once the data is split, the algorithm uses information entropy to calculate the optimum amount and location of splits to minimize the entropy of the data. In other words, the "space spanned by all predictor variables, is recursively partitioned into a set of rectangular areas." This sequence is called recursive partitioning, where it starts at the root node (known as the parent) and splits it to child nodes, which can also be further split, resulting in children nodes. [15] Once the nodes have been split to minimize the entropy of the data, the algorithm calculates the mean in each sub-group (or rectangular area for two-dimensional data) and that would be the value used to predict the dependent variable. The sequences in making decision forms a tree-like structure, hence the name Decision Tree Regression.

In Decision Tree Regression models, the mean squared error (MSE) is usually used as the default criterion to split the nodes and minimize randomness. This is because the data being used is of the continuous-type. If the data being used was of the discrete-type, entropy would have been used as a criterion under the Decision Tree Classification model (see subsubsection 2.2.5). The smaller the MSE, the better the model will predict the dependent variable. It is worth noting that there are other methods besides MSE that can be used such as Friedman's MSE, Mean Absolute Error, and Poisson deviance. The MSE is calculated using Equation 10. Note that Y denotes the vector of observed values of the dependent variable, $\hat{Y}\hat{Y}$ denotes the predicted values and n denotes the number of training samples.

$$\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{10}$$

As seen in Figure 5, the decision boundary captures the overall trend of the data and assumes a constant continuous output between different values of the independent variable(s).

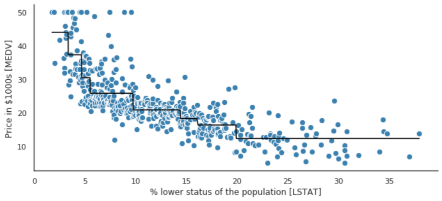

Figure 5: Decision Tree Regression Illustration (from [20]).

Building the Decision Tree Regression Model

No change from last regression model except the model being used (see algorithm 5).

```
Algorithm 5: Decision Tree Regression Model
  Data: [Real Estate Dataset 18]
  Result: A Decision Tree Regression model that predicts a median
          house value based on median income
  initialization;
  pre-process categorical data and drop NaN values;
  slice dataset to one dependent variable and multiple independent
    variables;
  split dataset into training set and testing set;
  feature scale the the training and testing sets (X and y separately);
  train the DTR model on the training set using a specified criteria;
  predict the test set results;
  calculate $R^2$ score on the predicted test set;
```

2.1.6 Random Forest Regression

Intuition

A Random Forest Regression "is an ensemble technique capable of performing both regression and classification tasks with the use of multiple decision trees and a technique called Bootstrap Aggregation, commonly known as bagging." [13] The ensemble technique "combines the predictions from multiple machine learning algorithms together to make more accurate predictions than any individual model." [49] Bootstrap Aggregation or bagging is a type of ensemble learning that allows the user to better understand the bias and the variance with the dataset. Thus, as the model using the bagging technique reduces the variance for each decision tree independently and aggregates the outputs without bias to any model (see Figure 6).

The Random Forest Regression algorithm starts by picking at random K data points from the dataset. It then builds the Decision Tree associated with these K data points. When building the model, the number of estimators, $n_estimators$, parameters would be defined and used to build the model, repeating the first two steps. It the utilizes each decision tree to predict the dependent variable for a new data point in question, and assigns the data point the mean across all of the predicted dependent variable values (See Figure 6).

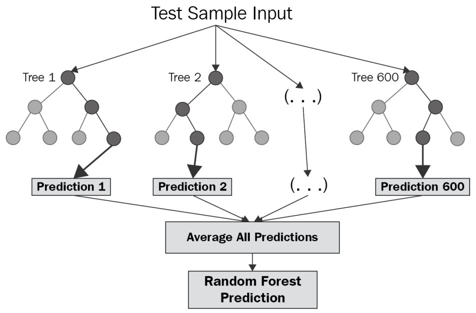

Figure 6: Random Forest Diagram (Regression) Diagram (from [20]).

Building the Random Forest Regression Model

No change from last regression model except the model being used (see algorithm 6).

```
Algorithm 6: Random Forest Regression Model
  Data: [Real Estate Dataset 18]
  Result: A Random Forest Regression model that predicts a median
          house value based on median income
  initialization;
  pre-process categorical data and drop NaN values;
  slice dataset to one dependent variable and multiple independent
    variables;
  split dataset into training set and testing set;
  feature scale the the training and testing sets (X and y separately);
  train the RFR model on the training set using a specified number of
    estimators;
  predict the test set results;
  calculate $R^2$ score on the predicted test set;
```

2.1.7 Evaluation Between Models For a Given Dataset

In order to evaluate the performance of each model, comparable metrics had to be used: R^2, Adjusted R^2, and the Root Mean Squared Error. The

Table 1: Regression Model Evaluation on Real Estate Data

Model	(Adjusted) R^2	**RMSE**($)
SLR	0.478	83638
MLR	0.652	68210
PLR$_2$	0.700	63251
PLR$_3$	0.708	62469
PLR$_4$	-0.047	118232
SVR	0.759	56664
DTR	0.655	67836
RFR	0.809	50423

adjusted R^2 was used to compare the high dimensional regression models with the simple linear regression model. In other words, the adjusted R^2 accuracy score was used since the number of predictors changed from the simple linear regression model to the other regression models.

The above regression algorithms were implemented on real estate data and the results can be seen in Table 1. Based on the results, the Random Forest Regression model performed best followed by the Support Vector Regression Model. This is because their Adjusted R^2 was closest to 1 and RMSE was relatively the smallest. This can best used to efficiently optimize the optimum models (as opposed to optimizing all the models together) and boost their performance while saving memory and time.

For further classification of the optimum model, the closer the Predicted Values were to the True Values, the closer the trend line was to the $y = x$ line in the figures above (e.g Figure 7). Ideally, all the points should lie on the $y = x$ line. Mathematically, the closer the (Adjusted) R^2 accuracy is to 1, the closer the values get to the $y = x$ line. [48]

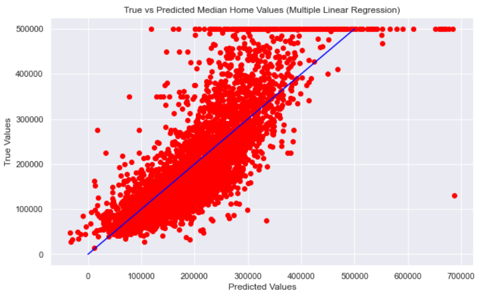

Figure 7: SVR Scatter Plot of Actual vs Predicted Median Home Values (DTR)

Furthermore, due to the negative Adjusted R^2 accuracy score and the highest RMSE in using the polynomial linear regression model with a degree of 4, it is the worst model to choose for this dataset. This is because the model was overtrained on the training dataset and thus was over-fitted, not being able to predict new found data.

Analyzing the real estate data using the different regression models is an example of how we can take good care of our resources. Pursuing one's calling to be an Industrial Engineer, Data Scientist, Consultant, or any similar field, it is important to understand how such models work by design in order to aid clients, businesses, and others on utilizing their resources most efficiently. James says "For as the body without the spirit is dead, so faith without works is dead also." (James 2:26) Furthermore, as the Apostle Paul writes in 1 Corinthians, "even great works of charity are meaningless without Faith, Hope, and Love." [27] It is important as analytical engineers to understand the structure of how such algorithms work in order to help those around us and show love to our neighbors in the most biblical manner possible. This is an example of being steward's of God's creation and how we can show love to our neighbors through efficient analyses and consultation by understanding the fundamentals of such models.

2.2 Classification

A key part of being an engineer is problem solving. An engineer observes "a need in society and using specialist knowledge creates an artefact to fulfil that need, thus improving the quality of life." [27] This includes making wise decisions and choices. What if a machine can help us make those decisions but we have the final say in implementation?

Classification requires the use of "machine learning algorithms that learn how to assign a class label to examples from the problem domain". [10] In other words, is also a "data mining (machine learning) technique used to predict group membership for data instances."[40] For example, a classification algorithm could classify emails as "spam" or "not spam". This is different from regression as classification is usually used with categorical, non-continuous data.

There are mainly two types of classifiers: Generative and discriminative. Generative classifiers assume some function form for $P(X|Y)$ and $P(X)$, where X and Y are events $\forall\ f : X \rightarrow Y$ or $P(Y|X)$ and estimate them directly from the given data. In other words, $P(Y|X)$ is known as the posterior probability. $P(X|Y)$ is known as the likelihood probability. $P(Y)$ is known as the prior probability and $P(X)$ is known as the normalization constant. To ease calculations, the algorithms use Bayes Rule to calculate the posterior probabilities (See Equation 33).

$$P(Y \mid X) = \frac{P(X \mid Y)P(Y)}{P(X)} \tag{11}$$

Discriminative classifiers directly estimates the posterior probabilities without modelling underlying prior and likelihood distributions.

2.2.1 Logistic Regression

Logistic Regression is a form of discriminative classification where the algorithm uses threshold values to classify data points as different classes using the same fundamentals found in linear regression. It uses the Sigmoid function to map any value into a value between 0 and 1 as seen in Equation 37.

$$\text{logistic}(\eta) = \frac{1}{1 + exp(-\eta)} \tag{12}$$

and it looks like:

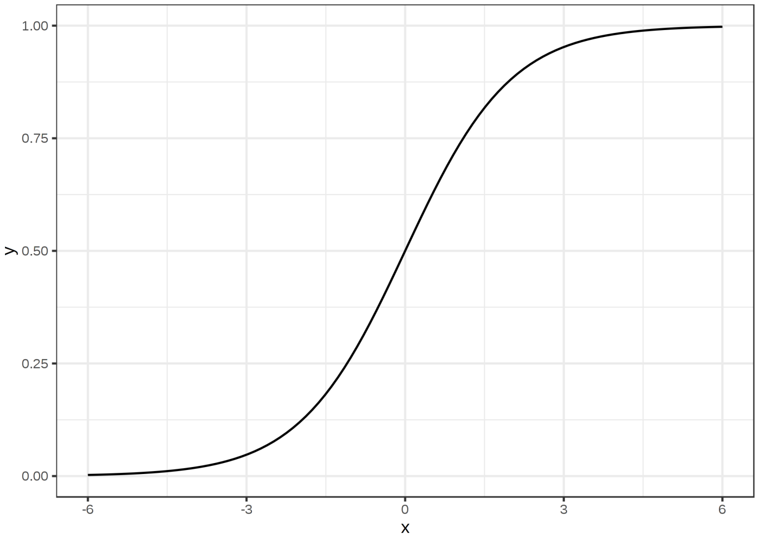

Figure 8: The Logistic (Sigmoid) Function (Taken from [11]).

Recall from linear regression the equation

$$\hat{y}^{(i)} = \beta_0 + \beta_1 x_1^{(i)} + \ldots + \beta_p x_p^{(i)} \tag{13}$$

Logistic regression calculates the probability of data points using the Sigmoid function:

$$P(y^{(i)} = 1) = \frac{1}{1 + \exp(-(\beta_0 + \beta_1 x_1^{(i)} + \ldots + \beta_p x_p^{(i)}))} \tag{14}$$

This forces values to be between 0 and 1 and thus a threshold can be chosen to assign classes. For example, all probabilities greater than 0.5 will be assigned to class a, else assign to class b. This is known as soft classification.

There are several types of logistic regression algorithms including Binary Logistic Regression (only two outcomes), Multinomial Logistic Regression (three or more categorical outcomes), Ordinal Logistic Regression (three or more order categorical data).

2.2.2 K-Nearest Neighbors (KNN)

K-nearest-neighbor (KNN) classification was "developed from the need to perform discriminant analysis when reliable parametric estimates of probability densities are unknown or difficult to determine." [34] The k-nearest-neighbor classifier is commonly based on the Euclidean distance between a test sample and the specified training samples denoted by

$$d(p,q) = \sqrt{(p_1 - q_1)^2 + (p_2 - q_2)^2 + \cdots + (p_i - q_i)^2 + \cdots + (p_n - q_n)^2}. \tag{15}$$

where p and q denotes an input samples with i features and a total of n samples. The KNN algorithm "assumes that similar things exist in close proximity." [38] The algorithm calculates the Euclidean distance between each instance of the given data groups data points together that are in close proximity to one another. The number of groups to classify is determined by the user and it is referred to as kernels, K. The algorithm works in such a way as to minimize the Euclidean distance between the kernels. Typically the distance is calculated from the center of each Kernel. Once a testing data point is passed through the algorithm, the data point is classified as a certain class based on the closest Kernel using Euclidean Distance.

As the number of kernels increase, the decision boundary becomes increasingly smoother until all data points are classified each in their own class. However, this is not ideal as it would cause the model to over fit on the training data and provide high variance and low bias in the testing data. As the number of kernels decrease, the predictions based on the created decision boundary become less stable. This implies that there is a range for which the number of clusters K is optimal. This is done by a method known as the Elbow Method where the number of kernels K is chosen based on the point where the plot of the MSE from the cluster centers versus number of Kernels K begins to flatten.

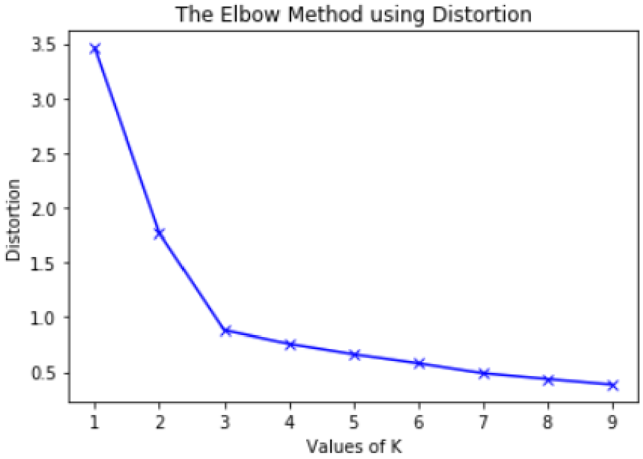

Figure 9: In this example, the optimal kernel number would be $K = 3$

An example of what the KNN algorithm would look like with $K = 3$ is shown in Figure 10.

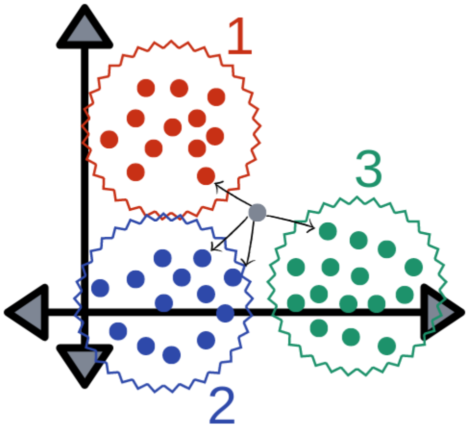

Figure 10: A visualization of how KNN works with $K = 3$ (Taken from [35]).

2.2.3 Support Vector Machine (SVM)

Support Vector Machine (SVM) is a "supervised machine learning model that uses classification algorithms for two-group classification problems." [16] Its objective is to "find a hyperplane in an N-dimensional space that distinctly classifies the data points." [54] As seen from Figure 11 and Figure 12, the SVM algorithm iterates through several possible hyperplanes until the distance between data points of different classes is maximized. This is referred to as Maximum Margin and the equation to find the Maximum Margin is seen in Equation 16.

$$
\begin{aligned}
&\min_{\alpha} \frac{1}{2}||c-d|| \\
&\qquad c = \sum_{y_i \in class=1} \alpha_i x_i, \qquad d = \sum_{y_i \in class=-1} \alpha_i \\
&\qquad \text{s.t.} \sum_{y_i \in class=1} \alpha_i = 1, \qquad \sum_{y_i \in class=-1} \alpha_i = 1 \\
&\qquad \alpha_i \geq 0 \qquad\qquad i = 1, \cdots, m
\end{aligned}
\tag{16}
$$

Support Vectors are the data points that influence the decision boundary position due to the Maximum Margin concept. They would be the data points closest to the hyperplane and are used to maximize the margin of the classifier.

The algorithm that helps maximize the margin is the hinge loss denoted by

$$\ell(y) = \max(0, 1 - t \cdot y) \tag{17}$$

for some intended output $t = \pm 1$. A more advanced equation of the one shown previously can be seen in Equation 18. The cost is zero if the predicted value and the actual value are of the same sign. Usually a regularization is added to the loss function for optimization purposes and (stochastic) gradient descent is used to converge the regularized loss function to a minimum so that it maximizes the margin (See Equation 19).

$$L = \frac{1}{N} \sum_i \sum_{j \neq y_i} [\max(0, f(x_i; W)_j - f(x_i; W)_{y_i} + 1)] + \alpha R(W) \tag{18}$$

$$\nabla_{w_j} L_i = \mathbb{1}(w_j^T x_i - w_{y_i}^T x_i + \Delta > 0) x_i \tag{19}$$

Note that in Equation 19, $\mathbb{1}$ denotes the indicator function that is one if the condition inside the function is true or zero otherwise and Δ denotes some bias term.

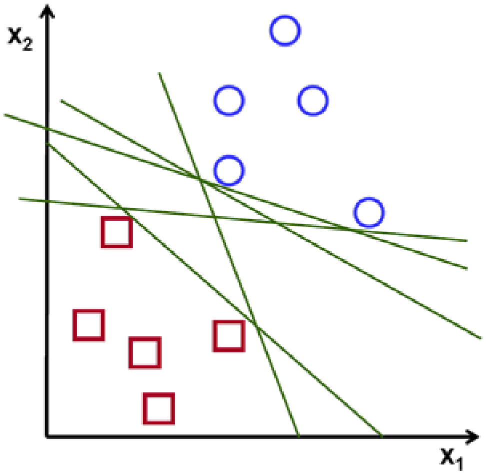

Figure 11: A visualization of SVM iterating to find the optimal hyperplane (Taken from [54]).

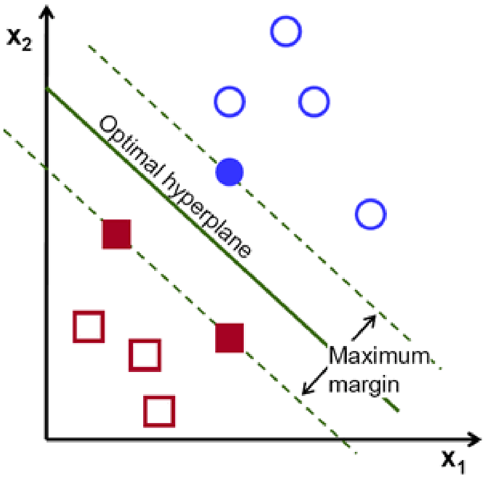

Figure 12: A visualization of the optimal hyperplane using SVM (Taken from [54]).

One of the advantages of SVMs is that the algorithm is also effective in higher dimensions than two dimensions. Usually, the data tends to be spread non-linearly. To visualize an example, Figure 13 is two-dimensional scatter plot of non-linearly spread data. In other words, it is difficult for a linear decision boundary to separate the two classes from each other. Thus, the SVM algorithm expands the dimension of the given dataset and a slice of that three-dimensional space can be visualized in Figure 14. The SVM has the opportunity to obtain a linear decision boundary in this dimensionality, which can be seen in Figure 15. Finally, the SVM algorithm maps the decision boundary from the higher dimension to the lower dimension, which in this example becomes a circle as seen in Figure 16.

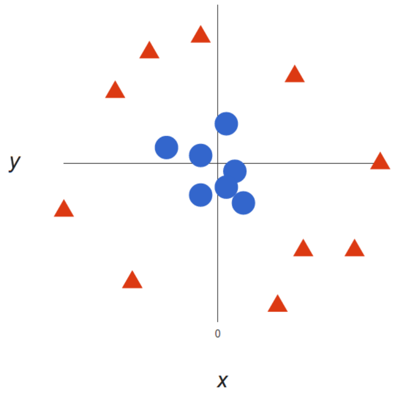

Figure 13: A visualization of non-linearly spread data (Taken from [16]).

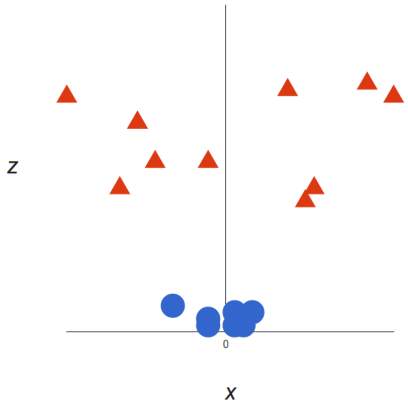

Figure 14: A visualization of a slice in the three-dimensional space (Taken from [16]).

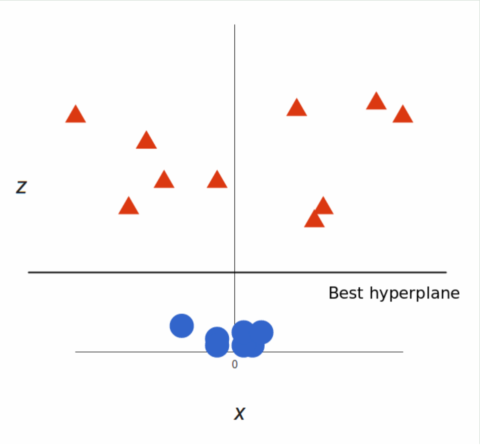

Figure 15: A visualization of the optimal decision boundary in the third dimension (Taken from [16]).

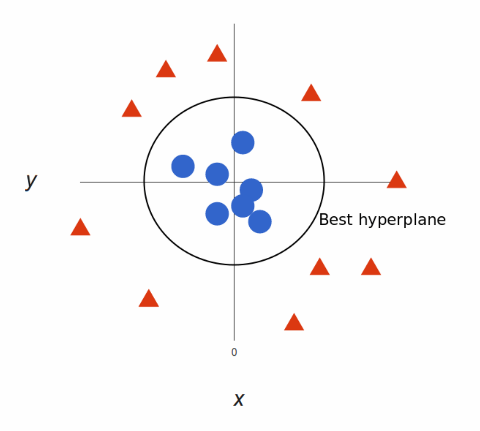

Figure 16: A visualization of the optimal decision boundary in the second dimension (Taken from [16]).

Mathematically, the above mentioned procedure is done using Equation 20 $\forall\ \theta(x) : \mathbf{R^m} \to \mathbf{R^n}$ s.t. $n \geq m$, $i, j, \cdots, k \in \mathbf{N}$.

$$f(x) = sign(\omega\theta(x) - b) = sign(\omega_1 x_1^i + \omega_2 x_2^j + \cdots + \omega_n x_n^k - b) \qquad (20)$$

In other words, at some dimension n, the data that was once non-linear in dimension m has now become linear with weights ω. If the sign of the linear equation is positive, it belongs to class 1, else it would belong to class -1. [9]

SVMs are effective in high dimensional spaces as well as in cases where the number of dimensions is greater than the number of samples. This is where the Kernel Trick is used.

Kernel SVM

Kernel SVMs were introduced to increase the accuracy of traditional SVM algorithms. Hilbert-Schmidt Kernels are one of the most popular kernel tricks applied to SVM to increase their accuracy and they can be seen in Table 2.

Table 2: Examples of Kernel Functions used in SVM models.

$\theta(x)$	$\mathbb{K}(x, v)$
Degree **d** polynomial	$(xv + 1)^d$
Radial Basis Function Machine	$\exp(-\frac{\|\|x-v\|\|^2}{2\sigma})$
Two Layer Neural Network	$(\eta(xv) + c)$

The reason Kernel functions are applied is because mapping to higher dimensions can be computationally and mathematically expensive. Instead of figuring out what each vector value is in each new dimension, the Kernel function just calculates the dot product between the vectors in the N-dimensional space instantly. In other words, it is possible to obtain a non-linear classifier using one of the Kernel Functions without actually having to map to higher dimensions. Once this is done, the SVM algorithm feeds this information to the loss function to calculate the decision boundary.

2.2.4 Naïve Bayes

Naïve Bayes is a generative, probabilistic classification algorithm. Its fundamental calculations are derived from Bayes Theorem (See Equation 21).

$$P(A|B) = \frac{P(A,B)}{P(B)} = \frac{P(B|A)P(A)}{P(B)} \tag{21}$$

In Bayesian classification, the interest lies in the probability of a label given some observed feature vector $x = [x_1, \cdots, x_d]$ with d features. This can be rewritten as $P(y \mid x_1, \cdots, x_d)$.

Bayes's theorem expresses this in terms of quantities that can be computed more directly:

$$P(y \mid x_1, \cdots, x_d) = \frac{P(x_1, \cdots, x_d \mid y)P(y)}{P(x_1, \cdots, x_d)} \tag{22}$$

The main assumption in Naïve Bayes is that, given the label, the observed features are conditionally independent i.e.

$$P(x_1, \cdots, x_d \mid y) = P(x_1 \mid y) \times \cdots \times P(x_d \mid y) \tag{23}$$

Therefore, Bayes rule can be rewritten as

$$P(y \mid x_1, \cdots, x_d) = \frac{P(x_1 \mid y) \times .. \times P(x_d \mid y)P(y)}{P(x_1, \cdots, x_d)} \tag{24}$$

In order to train a Naïve Bayes classifier, the probability is calculated using a frequentist approach. Given the training data, the frequency of different observations in the training set is calculated given different labels. For example,

$$P(x_1 = i \mid y = j) = \frac{P(x_1 = i, y = j)}{P(y = j)} \tag{25}$$

denotes the ratio of the number of times in the training data where $x_1 = i$ and $y = j$ to the total number of times in the training data where $y = j$.

In order to test the Bayesian Classifier, the probability of a label given an observed feature vector is estimated. The combined probabilities are computed from the training data to estimate the probability of a given label. For example, in order to decide between two labels y_1 and y_2, the ratio of the posterior probabilities for each label is computed as follows:

$$\begin{aligned}\frac{P(y_1 \mid x_1, \cdots, x_d)}{P(y_2 \mid x_1, \cdots, x_d)} &= \frac{P(x_1, \cdots, x_d \mid y_1)}{P(x_1, \cdots, x_d \mid y_2)} \frac{P(y_1)}{P(y_2)} \\ &= \frac{P(x_1 \mid y_1) \times \cdots \times P(x_d \mid y_1) P(y_1)}{P(x_1 \mid y_2) \times \cdots \times P(x_d \mid y_2) P(y_2)} \end{aligned} \tag{26}$$

All that the Bayesian Classifier calculates are $P(x_1|y_i), .., P(x_d \mid y_i)$ and $P(y_i)$ for each label by plugging in the numbers obtained during training. The label with the highest posterior probabilities is the one that is selected. [2] Note that the normalization constant $P(x_1, \cdots, x_d)$ is a common term between different classes and is thus ignored. This is another reason why this algorithm is of the generative type.

There are several types of Naïve Bayes classifiers: Bernoulli Naïve Bayes, Multinomial Naïve Bayes, Gaussian Naïve Bayes, etc. Each type is used based on the given data. For example, if the data follows a Gaussian distribution, it is best to use the Gaussian Naïve Bayes classifier. Each of the algorithms would use the probability density function based on the respective data distribution.

It is important to emphasize the assumption in Naïve Bayes of observed features being conditionally independent. This is not always the case. Usually, features tend to be correlated to some degree. Thus, feature engineering is required to either get rid of those features or to go along with the model with the independence assumption being made. [42]

2.2.5 Decision Tree Classification

Decision Tree Classification (DTC) is a supervised machine learning algorithm that continuously splits data according to certain parameters. It is similar to Decision Tree Regressors (See section 2.1.5). However, the difference lies in the type of data given and the output from the model. Decision Tree Classifiers tend to behave better than Decision Tree Regressors if the type of data is discrete and/or categorical. Furthermore, if the intended output of the classifier was to be discrete, then Decision Tree Classifiers is the better choice.

The structure of the DTC is the same as that of the DTR. The trees split according binary recursive partition, which splits the data into partitions and further branches until a leaf node (a pure node with the minimum entropy and maximum information gain between parent and child nodes). [24]

Entropy is calculated using Equation 27.

$$E(S) = H = \sum_{i=1}^{n} -p_i log_2(p_i) \tag{27}$$

Note that p_i is the frequentist probability of element of class i in the dataset. For example, there are two classes, a positive and negative class. If entropy is close to 1, it is considered a high entropy, high level of disorder, or low level of purity. Thus, the DTC algorithm uses entropy in its Information Gain calculation (see Equation 28). This equation calculates the difference in entropy between one variable given another variable and the entropy of one variable alone. The greater the reduction in uncertainty (the difference) from one parent node to another, the more the Information Gain ($IG(Y, X)$). In other words, the DTC will maximize information gain when the child node is a pure node (zero entropy). If all samples belong to one class in one node, the entropy is zero.

$$IG(Y, X) = E(Y) - E(Y|X) \tag{28}$$

A visualization of a DTC algorithm partitioning based on certain parameters can be seen in Figure 17.

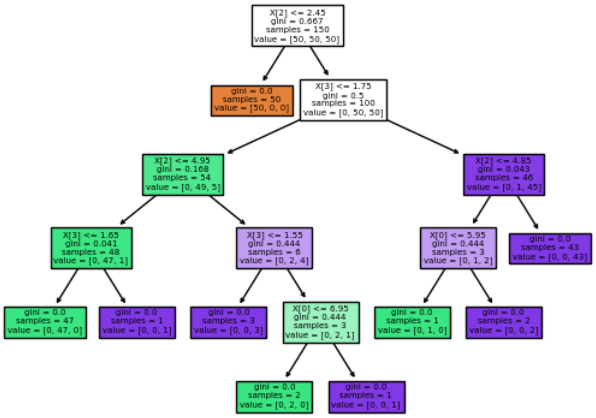

Figure 17: Visualization of DTC Partitioning on certain parameters (Taken from [1]).

It is important to note that DTC can be easily overfit on the training data. The DTC would create low bias and high variance and thus the DTC will not predict well on the testing data. This is why it is essential to set a minimum number of samples at a leaf node or to set a maximum depth of the tree. This is referred to as pruning.

2.2.6 Random Forest Classification

Random Forest Classification follows the same definition as that of section 2.1.6 except that it performs ensemble tree learning on Decision Tree Classifiers rather than Decision Tree Regressors. It "fits a number of decision tree classifiers on various sub-samples of the dataset and uses averaging to improve the predictive accuracy and control over-fitting." [53] Bagging methods such as Bootstrap Aggregation exists in this algorithm as well. This controls whether samples or features are drawn with or without replacement. The goal of this classifier is to reduce the variance and increase the bias in the results of the Decision Tree Classifier. This is done by introducing randomization and ensemble learning. It is worth noting the difference between Random Forest Classifiers and Decision Tree classifiers.

Although Random Forest Classifiers tend to be a collection of Decision Tree Classifiers, Random Forest Classifiers "randomly selects observations and features to build several decision trees and then averages the results." [55] In contrast, the Decision Tree Classifier generates a certain set of rules to predict which class data belongs. Furthermore, Random Forest Classifiers overcome the problem of overfitting as it aggregates and combines several Decision Tree Classifiers together. However, there is a limitation to this. The more trees the Random Forest aggregates and combines, the more computationally expensive the procedure.

2.2.7 Evaluation Between Models For a Given Data Set

A breast cancer dataset that was imported from the *sklearn* library was analyzed. The precision, recall, and f1-scores of predicting malignant cells can be seen in Equation 2.2.7. Note that error analysis could have been conducted on classifying benign cells. However, a false negative, a type II error, or in other words, having a model classify a person as not having cancerous cells when in fact they do is a very important issue and must not be taken lightly. Thus, malignant cells need to be classified with higher importance than benign ones. Furthermore, the overall accuracy of the model can also be seen in Equation 2.2.7.

A model's precision score is the fraction of relevant instances among the retrieved instance, recall, or sensitivity, is the fraction of relevant instances that were retrieved:

$$\text{precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{29}$$

$$\text{recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{30}$$

The $\mathbf{F_1}$**Score** is a combination of precision and recall and it is a measure of the model's accuracy:

$$\text{F1 Score} = \frac{\text{True Positives}}{\text{True Positives} + \frac{1}{2}\Big(\text{False Positives} + \text{False Negatives}\Big)} \tag{31}$$

Accuracy is the percentage of how correctly the model performed on a given dataset:

$$\text{Accuracy} = \frac{\text{True Positives} + \text{True Positives}}{\text{True Positives} + \text{True Negatives} + \text{False Positives} + \text{False Negatives}} \tag{32}$$

Note that accuracy can be misleading on imbalanced data. For example, if the data had 97 negative samples and 3 positive samples, the model can predict that all sames are negative and achieve a 97% accuracy score even though the model would not be reliable.

Table 3: Classification Model Evaluation on Breast Cancer Data

Model	**Precision**	**Recall**	**$\mathbf{F_1}$Score**	**Accuracy**	**FalsePositives**
LR	0.94	0.94	0.94	0.96	3
KNN	0.98	0.89	0.93	0.95	6
SVM	0.93	0.96	0.94	0.96	2
NB	0.93	0.96	0.94	0.96	2
DTC	0.93	0.96	0.94	0.96	2
RFC	0.95	0.98	0.96	0.97	1

Note that the largest **$\mathbf{F_1}$Score** and the lowest **FalsePositives** values occurred using the Random Forest Classification algorithm. The model used 20 estimators, that is 20 Decision Tree Classifiers. All models performed well achieving greater than 90% accuracy rate. However, in a sensitive industry such as the classifying whether a person or not has cancerous cells, it is important to value accuracy over interpretability. Although the Random Forest Classification model cannot be visualized due to its high dimensionality, it performs very well and consistently, which a Doctor would value in times of crises.

Classification algorithms are very useful to make decisions as we can use the computational savvy of machines to make complex choices more efficiently. The power of classification algorithms can be used for machine vision purposes. For example, it can be used to understand whether an image is a dog or a cat. However, this can be placed on a morally ambiguous seat where the model can classify different faces. Is it okay for engineers to design machine vision algorithms that detect faces for the military or for Apple's new "iGlasses"? This is, to some degree, an invasion of privacy. However, it does, however, solve the problem of finding individuals, such as terrorists, who wreak havoc in the world. Furthermore, such algorithms can be used for a Cyber technological war in the future. As difficult a decision can be in such an environment, it is important to design such machines that fail or missclassify information very rarely to reduce causalities or errant decision making.

With regards to consumerism, it is important to not take advantage of others in developing such algorithms. Classification requires a plethora of data, and at the rate at which the availability of data is growing, it has become more difficult to sustain the privacy and security of individuals. Should engineers knowingly use such data, where consumers might not be aware that their private information is being used for big institutions to sell? Christian stewardship has been defined by Charles Bugg as: "Utilising and managing all resources God provides for the glory of God and the betterment of His creation." The drive for efficiency is at the heart of engineers, and sits comfortably with Christian stewardship of land, natural resources and talents. However, "there is also pressure on the engineer to design for failure, so that products will need to be purchased on a continual basis by consumers, rather than on a once-for-all basis. The disposable culture in which we live presents a challenge for Christian engineers." [27]

2.3 Clustering

Due to our finitude, we cannot perform tasks as efficiently and sometimes anywhere close to how today's computers can. Through clustering algorithms, engineers are able to use machines to identify patterns in data that they would not have otherwise been able to.

Clustering is an unsupervised machine learning task that involves the machine identifying natural groupings in the data. [3] In other words, "Clustering techniques apply when there is no class to be predicted but rather when the instances are to be divided into natural groups." [30] This is has decreased the difficulty in identifying practical groupings that users would not have otherwise thought of. This process of clustering in an unsupervised machine learning sense is known as pattern discovery or knowledge recognition.

Clustering is important for exploring data. [21] It can be used during the pre-processing phase to identify homogeneous groupings on which to build supervised algorithms. Furthermore, it can be used for anomaly and outlier detection, where anomalies and outliers are data points that do not belong to any structure.

There are several different methods in computing clusters including but not limited to Hierarchical, Partitioning, and Grid-based. Hierarchical clustering groups data based on a hierarchical structure either using a top-down approach or a bottom-up approach. It uses different distance algorithms to to measure the similarity between clusters. Partitioning clustering partitions data objects into clusters based on a pre-set optimization objective such as distance. Grid-based clustering "divides the input space into hyper-rectangular cells, discards the low-density cells, and then combines adjacent high-density cells to form clusters." [43] It is important to note that the centroid of a cluster is the mid-point of a group and it represents the most typical case in a cluster. However, a centroid is not necessarily a data point in a given dataset.

2.3.1 K-Means Clustering

K-Means algorithm is an "iterative algorithm that tries to partition the dataset into K-pre-defined distinct non-overlapping subgroups (clusters) where each data point belongs to only one group." [33] Its goal is to create clusters, where their centroids are at maximum distance from each other, while data points in a certain cluster are the most similar.

The process of K-means appears to give "partitions which are reasonably efficient in the sense of within-class variance." [47] The process in which K-means solves the clustering problem in a given dataset is called Expectation Maximization. The E-step is assigning the data points to the closest cluster and the M-step is computing the centroid of each cluster. Let $x_1, x_2, \cdots, x_n$ be a set of oberservations where each observation x_i is d-dimensional. K-Means clustering aims to partition the n observations into k sets $S = S_1, S_2, \cdots, S_k \; \forall \; k \geq n$. This is done in order to minimize the within-cluster sum of squares (i.e. variance). In mathematical terms, the objective is to find

$$\underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \left\| \mathbf{x} - \boldsymbol{\mu}_i \right\|^2 = \underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{k} |S_i| \operatorname{Var} S_i \tag{33}$$

where μ_i is the mean of points in S_i, which is also the centroid of a cluster. This is equivalent to minimizing the pairwise squared deviations of points in the same cluster:

$$\underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{k} \frac{1}{2|S_i|} \sum_{\mathbf{x}, \mathbf{y} \in S_i} \left\| \mathbf{x} - \mathbf{y} \right\|^2 \tag{34}$$

The equivalence can be deduced from the identity

$$\sum_{\mathbf{x} \in S_i} \left\| \mathbf{x} - \boldsymbol{\mu}_i \right\|^2 = \sum_{\mathbf{x} \neq \mathbf{y} \in S_i} (\mathbf{x} - \boldsymbol{\mu}_i)^T (\boldsymbol{\mu}_i - \mathbf{y}) \tag{35}$$

Because the total variance is constant, this is equivalent to maximizing the sum of squared deviations between points in different clusters, which follows from the law of total variance. [36]

In other words, for an initialized set of k means, the algorithm iterates through the data and assigns each observation x_i to the cluster with the nearest mean (which was initialized randomly) using the least squared Euclidean distance from the cluster's centroid. This partitions the observations into S_i:

$$S_i^{(t)} = \left\{ x_p : \left\| x_p - m_i^{(t)} \right\|^2 \leq \left\| x_p - m_j^{(t)} \right\|^2 \ \forall j, 1 \leq j \leq k \right\}, \tag{36}$$

where each x_p is assigned to exactly one $S^(t)$, even if it could be assigned to two or more of them. This is known as the Voronoi Diagram. Then, the algorithm recalculates the centroid by calculating the means for observations assigned to each cluster until the assignments no longer change (see Equation 37).

$$m_i^{(t+1)} = \frac{1}{\left| S_i^{(t)} \right|} \sum_{x_j \in S_i^{(t)}} x_j \tag{37}$$

A visualization of the K-Means algorithm in action can be seen in Figure 18.

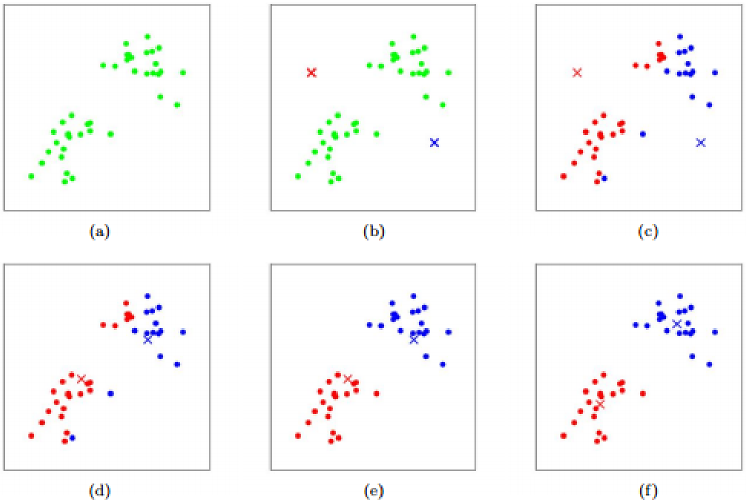

Figure 18: K-means algorithm. Training examples are shown as dots, and cluster centroids are shown as crosses. (a) Original dataset. (b) Random initial cluster centroids. (c-f) Illustration of running two iterations of k-means. In each iteration, we assign each training example to the closest cluster centroid (shown by "painting" the training examples the same color as the cluster centroid to which is assigned); then we move each cluster centroid to the mean of the points assigned to it. Images courtesy of Michael Jordan. (As taken from [22]).

As seen in Figure 19, the K-Means algorithm will produce unintuitive and possibly unexpected clusters. "In the first three plots, the input data does not conform to some implicit assumption that K-Means makes and undesirable clusters are produced as a result. In the last plot, K-Means returns intuitive clusters despite unevenly sized blobs." [25]

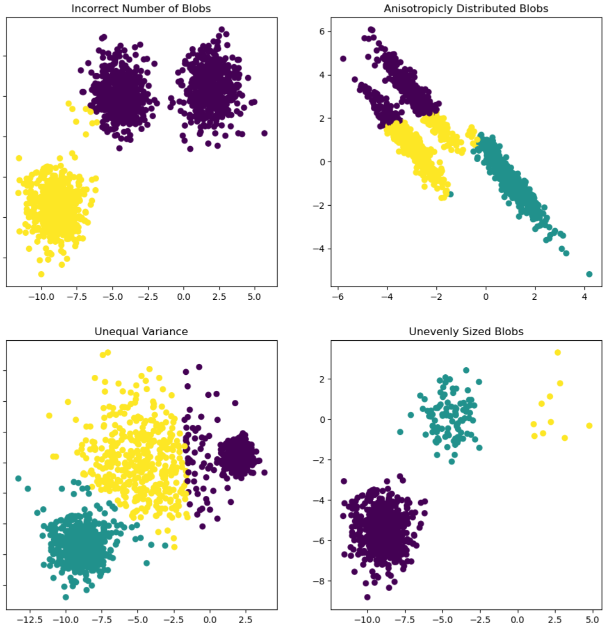

Figure 19: A visualization of the K-Means algorithm identifying clusters (Taken from [25]).

It is important to emphasize that the data must be standardized to have a mean of 0 and a standard deviation of 1 because the K-Means algorithm uses distance based measurements to determine the similarity between data points. For example, given a dataset with features height, weight, and age, where each feature has a different unit of measurement, their standardized values would allow the K-Means algorithm to perform distance calculations with relation to the other features.

Furthermore, due to the iterative process in the algorithm and the randomness in initializing the centroids, different initialization of centroids may lead to different clusters. This is known as the random initialization trap, where the K-Means algorithm converges to a local optimum and never converges to a global optimum.The solution to the random initialization trap is using the K-Means++ algorithm, which initializes the centroids far away from each other.

Choosing the right number of clusters follows the same logic as that of choosing the number in the KNN algorithm (See Figure 9). However, the difference lies in the y-axis, where instead of plotting Distortion versus Values of K, the elbow method is used on a plot of Sum of Squared Distances or (WCSS) versus Values of K.

2.3.2 Hierarchical Clustering

Intuition

Hierarchical clustering, also known as hierarchical cluster analysis, is an unsupervised machine learning algorithm that groups data into a hierarchy of clusters based on Agglomerative and Divisive techniques. [58] In other words, the algorithm "build[s] nested clusters by merging or splitting them successively." [5] The Agglomerative approach is a bottom-up approach, where each observation x_i in the data is initilialized as its own individual cluster. The Divisive approach is a top-down approach, where all observations start as one cluster.

Agglomerative Hierarchical clustering Technique

The Agglomerative Hierarchical clustering Technique initializes all data points as individual clusters. At each iteration, the algorithm identifies the nearest cluster (this would imply two data points are similar) and merges to become one cluster until K clusters are formed. The linkage criteria determine the the metric used for the merge strategy (see Table 4).

Table 4: Different Linkage Criteria. Note that $X_1, X_2, \cdots, X_n$ are observations of one cluster, $Y_1, Y_2, \cdots, Y_n$ are observations of another cluster, and $d(x, y)$ is the distance between a subject with observation vector x and a subject with observation vector y.

Merge Strategy	**Approach**
Ward	$d_{12} = \|\mathbf{X}_i, \mathbf{Y}_j)\|^2$
Maximum (or Complete Linkage)	$d_{12} = \max_{i,j} \; d(\mathbf{X}_i, \mathbf{X}_j)$
Average Linkage	$d_{12} = \frac{1}{kl} \sum_{i=1}^{k} \sum_{j=1}^{l} d(\mathbf{X}_i, \mathbf{Y}_j)$
Single Linkage	$d_{12} = \min_{i,j} \; d(\mathbf{X}_i, \mathbf{Y}_j)$

The Hierarchical clustering Technique can be visualized using a Dendrogram: A tree-like diagram that records the sequences of merges or splits. The y-axis measures the distance between clusters based on the linkage criteria and the x-axis displays the observations of given data (see Figure 20).

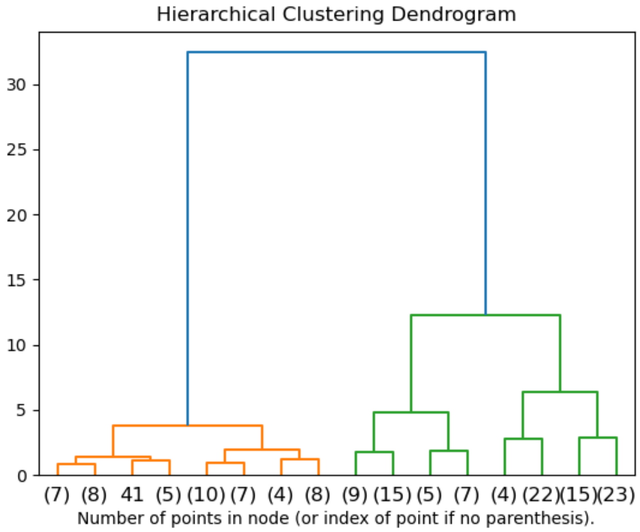

Figure 20: A Dendogram splitting data observations $A, B, \cdots, F$ (Taken from [58]).

The number of clusters is chosen using horizontal levels and setting thresholds based on the largest change in distance between consecutive number of clusters. In Figure 20, the number optimal number of clusters would be $K = 2$. The maximum change in distance between consecutive number of clusters is anywhere in the range of $y = [12.5, 32.5]$. A horizontal level crosses the Dendogram twice at any of those levels and thus the optimal number of clusters would be two.

Divisive Hierarchical clustering Technique

The Divisive Hierarchical clustering Technique is exactly the opposite to the Agglomerative Hierarchical clustering Technique. All observations begin as a single cluster and each iteration, the observations are split from their respective clusters based on the same formulas used in Table 4. The procedure to choose the number of clusters depends on the Dendogram, similar to that of the Agglomerative Hierarchical clustering Technique.

This algorithm is not recommended as much as the Agglomerative Hierarchical clustering Technique due to the exponential complexity of the algorithm ($O(2^n)$).

2.3.3 Evaluation Between Models For a Given Data Set

Clustering algorithms were implemented on a dataset that represents information of customers entering and leaving a mall. It is vital that clustering algorithms are interpretable and inferrable. In this example, it would be most important to understand which customers are the regular ones, and which type of customer to spend more advertising expense on as to increase that demographic further.

After using the elbow method and a dendogram, 5 clusters was the optimal cluster to choose. Furthermore, using different random states, the models performed similarly consistently. This is a sign the model is reproducible in the field on unseen data. As seen in Figure 21 and Figure 22, the models clustered the data very similarly with slight differences due to the nature of how the algorithms converge.

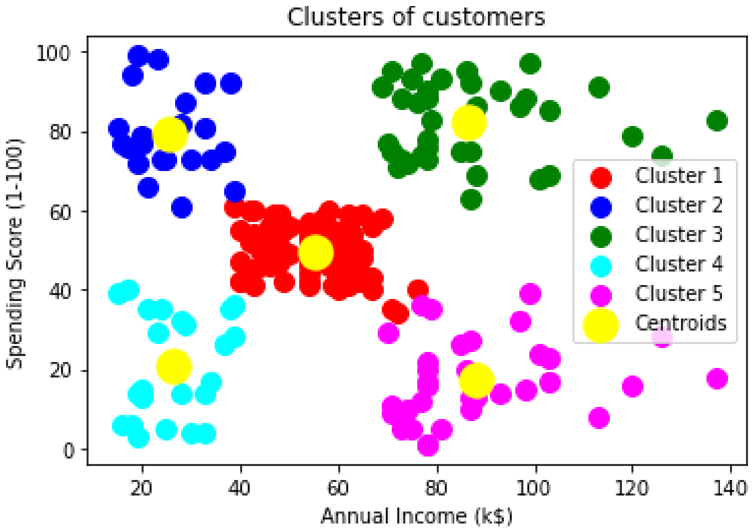

Figure 21: K Means Implementation on Mall Customers Dataset

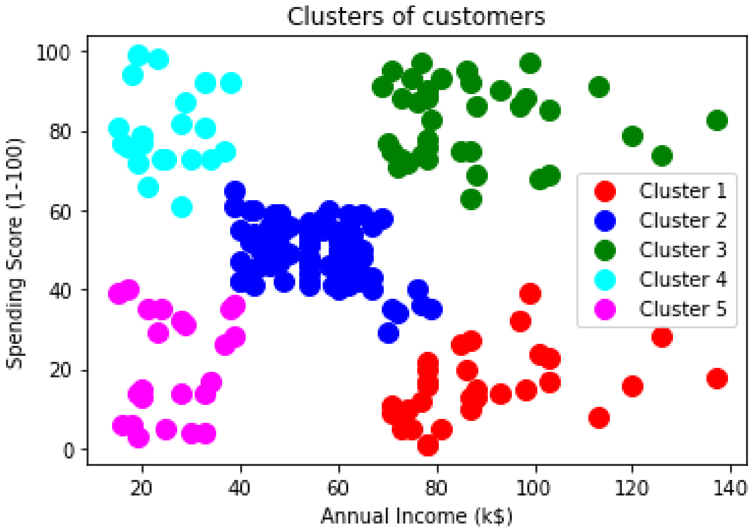

Figure 22: Hierarchical Clustering Implementation on Mall Customers Dataset

One way to interpret this data would be to understand that there are those who have high income and do not spend much at the mall. Since there is a cluster where several high income earners have high spending scores, it would be worthwhile to invest in advertising and increase that demographic as customers. Clustering algorithms are very important to interpret and can be applied in several different industries.

What if we could use clustering algorithms to identify patterns in the Bible that we have not yet identified? This could open new doors in biblical studies. However, it is important to be aware that the patterns identified by such algorithms may be fallable and erroneous due to the machine not inherently understanding the themes of the Bible. Can we quantify such themes? How can we explain to a machine the value of the coming of the second Adam, Jesus, to take the penalty of our sings and defeat Death in order that all who believe in Him and hear His word shall have eternal life? It is one thing for machines to be able to do tasks that we cannot, but it is more important for engineers to be able to understand and reflect on the identified patterns and decide their value and the cost in moving forward with such decisions.

2.4 Model Selection

It is important to understand that each machine learning model is used as a tool to make educated decisions. That being said, it is important to understand which model to use and how to optimize its parameters. "Model selection strategies for machine learning algorithms typically involve the numerical optimisation of an appropriate model selection criterion, often based on an estimator of generalisation performance", such as grid search and k-fold cross validation. [19] The loss or error associated with such optimization techniques consists of bias and variance components. The ideal model would consist of little to no bias and little to no variance in its predictions. However, this can also lead to over-fitting during training and perform badly on testing sets. Thus, several techniques are used to iterate through different parameters on the training set, and sometimes a validation set, to optimize the parameters of a machine learning model and avoid over-fitting.

2.4.1 K-Fold Cross Validation

In machine learning, "the standard measure of accuracy for trained models is the prediction error (PE), i.e. the expected loss on future examples." [28] The expected value of prediction error (EPE) is the error associated with the performance of learning algorithms over training sets. Cross validation is a statistical technique to provide an unbiased estimate of the skill of machine learning models, i.e. their EPE. However, its variance may be large and thus cross validation techniques provide confidence intervals on PE and EPE that will enable users to test the significance of the performance between algorithms. The structure of machine learning algorithms should follow a similar schematic as that seen in Figure 23.

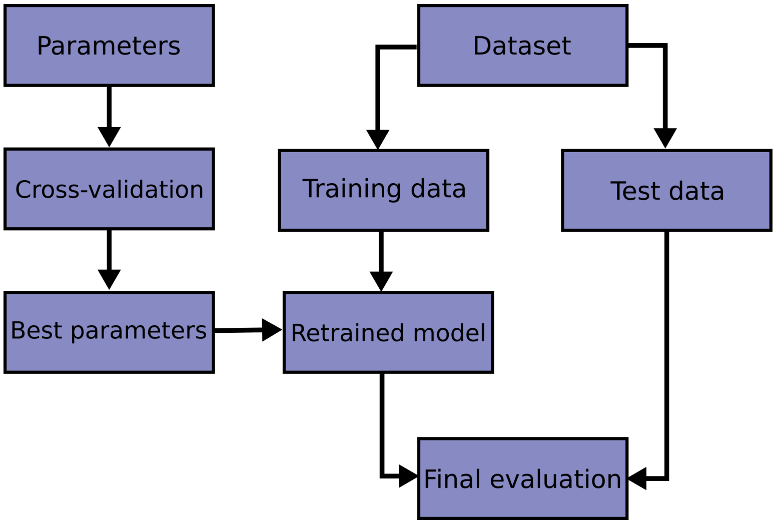

Figure 23: A general structure of correctly tuned machine learning models (Taken from [7]).

During K-Fold Cross Validation, a test set would be held out for final evaluation as usual, and the remaining training set is split into k smaller sets (called K-Folds). The learning model would train on the $k - 1$ folds, and validate its performance on the remaining data. This is done iteratively until at least all the subsets of the training set have been validated. The performance measure of the K-Fold Cross Validation is the average of the error values computed during the iterated training technique (See Figure 24 for a visualization example). Note that this process can be computationally expensive, and it is up to the user to decide the importance of performing cross-validation on a given model. If the value of obtaining an interpretable model with a high performance metric outweighs the cost of operational machinery and time complexity, it is worth doing K-Fold Cross Validation.

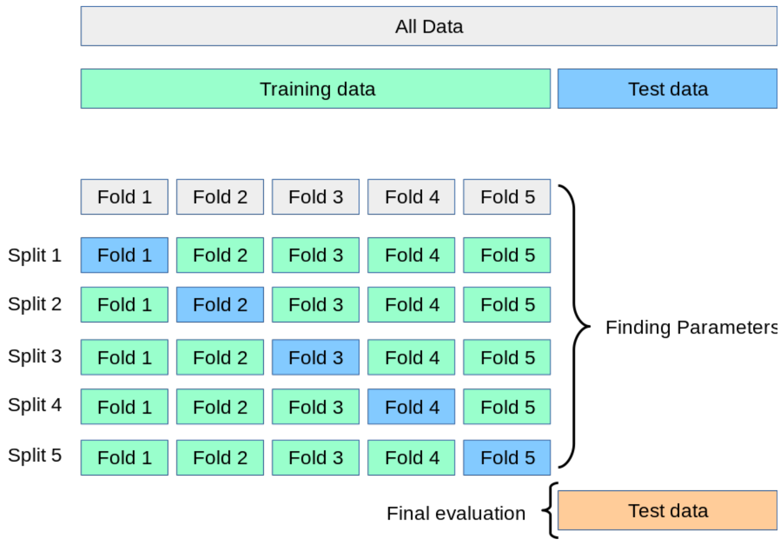

Figure 24: An example of a K-Fold Cross Validation (Taken from [7]).

2.4.2 Grid Search

In combination with cross validation techniques, grid search can be used to optimize the parameters of a model. A search consists of an estimator (the machine learning model), a parameter space, a method for searching and sampling candidates, a cross-validation scheme, and a score function. [8] Given a grid of parameters, where the grid consists of different possible values for certain parameters to take, the grid search algorithm exhaustively iterates through every combination passed and outputs best possible parameter combination based on the score (RMSE, recall, etc.). This can be very computationally expensive, and thus there are different approaches that can be taken, such as randomizing the parameters and successively halving parameters based on the score of the model.

2.4.3 Importance of Interpretability

It is important to compare the performance as well as the ability to interpret the model itself. Some models may perform very well on new testing data, but may not be easily interpretable. It is up to the user and their client to decide whether "what is predicted" (i.e. the power of the model) is more important than the "why" (i.e. how does the model work in achieving such results). It is said that the most powerful predictive models are the least interpretable. In other words, the fewer the parameters, the less complex the algorithm, the easier it can be explained. This is more of a business decision than an engineering decision. If the structure of the model needs to be visually represented (say for investors), an interpretable model would have a higher value (e.g. Decision Trees would be better than Random Forests). Furthermore, if the model is interpretable, it can be easier to debug and understand faults in its system as it is being optimized.

On the other hand, interpretability can be ignored if the project is for personal reasons or the problem being solved has been well studied. "Some applications have been sufficiently well studied so that there is enough practical experience with the model and problems with the model have been solved over time." [4] An example of this would be character recognition. There have been numerous machine learning algorithms that are able to classify the image as a certain character and as of right now, there has been no need to gain further insight on this topic.

Unfortunately, some take this for granted and are able to manipulate the system through providing a complex algorithm that cannot be easily interpreted. Problems with users who deceive a system result from a mismatch between the goals of the creator and the user of a model. For example, bank XYZ provides loans to applicants based on a credit score that is predicted using a regression machine learning algorithm. The algorithm includes the number of credit cards the applicant has as one of its features and it is determined that the more credit cards, the worse the credit score. Thus, an applicant can get rid of their credit cards to obtain a better credit score on the model the bank uses, but the risk of repaying the loan given by the bank is still the same. "The system can only be gamed if the inputs are proxies for a causal feature, but do not actually cause the outcome. Whenever possible, proxy features should be avoided as they make models gameable." [4]

3 Conclusion

3.0.1 Christianity and The Future of Machine Learning

Machine Learning continues to grow exponentially. As Christians who have access to important data and have the means to develop advanced machine learning algorithms, it is important to be in the forefront of such an advancing technology and reflect the light of the Lord continuously so that others may follow. It is also important not to abuse such power because just as it is said in Proverbs 21:20, "Precious treasure and oil are in a wise man's dwelling, but a foolish man devours it." In addition, it is our duty to develop such technology for the good and love of our neighbors and for the Kingdom of the Lord. Just as Jesus taught through the Parables of the Three Servants, "to those who use well what they are given, even more will be given, and they will have an abundance. But from those who do nothing, even what little they have will be taken away." (Mat 25:29-30) It is our mandate to be God's stewards in developing this technology with biblical foundation. The technology itself is not evil but how we use it depends vastly on our "why" and our biblical mandate. There are consequences to our actions, and as fallen beings, it is easy to fall in the trap of abusing such self-learning technology. Our purpose is to glorify God, know Him, make Him known, and enjoy Him forever. If our calling includes optimizing technological systems that behave in a self-learning environment, then we must remain firm in our faith and continue to challenge others to do the same. "Technology must be redeemed for the glory of God, your good, and the good of others." [6]

References

[1] *1.10. Decision Trees — scikit-learn 0.24.1 documentation.* https://scikit-learn.org/stable/modules/tree.html. (Accessed on 03/10/2021).

[2] *1.9. Naive Bayes — scikit-learn 0.24.1 documentation.* https://scikit-learn.org/stable/modules/naive_bayes.html. (Accessed on 03/10/2021).

[3] *10 Clustering Algorithms With Python.* https://machinelearningmastery.com/clustering-algorithms-with-python. (Accessed on 03/12/2021).

[4] *2.1 Importance of Interpretability | Interpretable Machine Learning.* https://christophm.github.io/interpretable-ml-book/interpretability-importance.html. (Accessed on 03/24/2021).

[5] *2.3. Clustering — scikit-learn 0.24.1 documentation.* https://scikit-learn.org/stable/modules/clustering.html#hierarchical-clustering. (Accessed on 03/12/2021).

[6] *26 Bible Verses About Technology | Tithe.ly.* https://get.tithe.ly/blog/technology-in-the-bible. (Accessed on 04/06/2021).

[7] *3.1. Cross-validation: evaluating estimator performance — scikit-learn 0.24.1 documentation.* https://scikit-learn.org/stable/modules/cross_validation.html. (Accessed on 03/24/2021).

[8] *3.2. Tuning the hyper-parameters of an estimator — scikit-learn 0.24.1 documentation.* https://scikit-learn.org/stable/modules/grid_search.html#grid-search. (Accessed on 03/24/2021).

[9] *380995.380999.* https://dl.acm.org/doi/pdf/10.1145/380995.380999. (Accessed on 03/08/2021).

[10] *4 Types of Classification Tasks in Machine Learning.* https://machinelearningmastery.com/types-of-classification-in-machine-learning/. (Accessed on 03/07/2021).

[11] *4.2 Logistic Regression | Interpretable Machine Learning.* https://christophm.github.io/interpretable-ml-book/logistic.html. (Accessed on 03/07/2021).

[12] *81523322.pdf.* https://core.ac.uk/download/pdf/81523322.pdf. (Accessed on 01/21/2021).

[13] *A Beginners Guide to Random Forest Regression | by Krishni | Data Driven Investor | Medium.* https://medium.com/datadriveninvestor/random-forest-regression-9871bc9a25eb. (Accessed on 01/25/2021).

[14] *A tutorial on support vector regression | SpringerLink.* https://link.springer.com/article/10.1023/B:STCO.0000035301.49549.88. (Accessed on 01/21/2021).

[15] *An Introduction to Recursive Partitioning: Rationale, Application and Characteristics of Classification and Regression Trees, Bagging and Random Forests.* https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2927982/. (Accessed on 01/23/2021).

[16] *An Introduction to Support Vector Regression (SVR) | by Tom Sharp [U+FFFD][U+FFFD] | Towards Data Science.* https://towardsdatascience.com/an-introduction-to-support-vector-regression-svr-a3ebc1672c2. (Accessed on 01/21/2021).

[17] *Assumptions of Multiple Linear Regression - Statistics Solutions.* https://www.statisticssolutions.com/assumptions-of-multiple-linear-regression/. (Accessed on 01/18/2021).

[18] *California Housing Prices | Kaggle.* https://www.kaggle.com/abhilashanil/california-housing-prices. (Accessed on 01/17/2021).

[19] *cawley10a.dvi.* https://www.jmlr.org/papers/volume11/cawley10a/cawley10a. (Accessed on 03/24/2021).

[20] *Classification and Regression Analysis with Decision Trees | by Lorraine Li | Towards Data Science.* https://towardsdatascience.com/https-medium-com-lorrli-classification-and-regression-analysis-with-decision-trees-c43cdbc58054. (Accessed on 01/23/2021).

[21] *Clustering.* https://docs.oracle.com/cd/B28359_01/datamine.111/b28129/clustering.htm#i1005728. (Accessed on 03/12/2021).

[22] *CS221.* https://stanford.edu/~cpiech/cs221/handouts/kmeans.html. (Accessed on 03/12/2021).

[23] *Data Mining Definition.* https://www.investopedia.com/terms/d/datamining.asp. (Accessed on 04/05/2021).

[24] *Decision Tree Classification. A Decision Tree is a simple. . . | by Afroz Chakure | The Startup | Medium.* https://medium.com/swlh/decision-tree-classification-de64fc4d5aac. (Accessed on 03/10/2021).

[25] *Demonstration of k-means assumptions — scikit-learn 0.24.1 documentation.* https://scikit-learn.org/stable/auto_examples/cluster/plot_kmeans_assumptions.html. (Accessed on 03/12/2021).

[26] John Dyer. *From the Garden to the City. The Redeeming and Corrupting Power of Technology.* Kregel Publications, 2011.

[27] *Engineering_text_only.pdf.* https://www.cis.org.uk/upload/Resources/Students/Engineering_text_only.pdf. (Accessed on 04/06/2021).

[28] *grandvalet04a.dvi.* https://www.jmlr.org/papers/volume5/grandvalet04a/grandvalet04a.pdf. (Accessed on 03/24/2021).

[29] *Graphs and ML: Multiple Linear Regression | by Lauren Shin | Towards Data Science.* https://towardsdatascience.com/graphs-and-ml-multiple-linear-regression-c6920a1f2e70. (Accessed on 04/06/2021).

[30] Ian H. Witton. *Data Mining: Practical Machine Learning Tools and Techniques (Morgan Kaufmann Series in Data Management Systems.* Morgan Kaufmann, 2016.

[31] *Homoscedasticity - Statistics Solutions.* https://www.statisticssolutions.com/homoscedasticity/. (Accessed on 01/18/2021).

[32] *Inductive Reasoning - an overview | ScienceDirect Topics.* https://www.sciencedirect.com/topics/computer-science/inductive-reasoning. (Accessed on 01/23/2021).

[33] *K-means Clustering: Algorithm, Applications, Evaluation Methods, and Drawbacks | by Imad Dabbura | Towards Data Science.* https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-methods-and-drawbacks-aa03e644b48a. (Accessed on 03/12/2021).

[34] *K-nearest neighbor - Scholarpedia.* http://scholarpedia.org/article/K-nearest_neighbor. (Accessed on 03/07/2021).

[35] *K-Nearest Neighbors (KNN) Algorithm for Machine Learning | by Madison Schott | Capital One Tech | Medium.* https://medium.com/capital-one-tech/k-nearest-neighbors-knn-algorithm-for-machine-learning-e883219c8f26. (Accessed on 03/07/2021).

[36] *Knowledge and information systems : KAIS. (eJournal / eMagazine, 1999) [WorldCat.org]*. https://www.worldcat.org/title/knowledge-and-information-systems-kais/oclc/45478778?referer=di&ht=edition. (Accessed on 03/12/2021).

[37] *Linear Regression Essentials in R - Articles - STHDA*. http://www.sthda.com/english/articles/40-regression-analysis/165-linear-regression-essentials-in-r/. (Accessed on 01/17/2021).

[38] *Machine Learning Basics with the K-Nearest Neighbors Algorithm | by Onel Harrison | Towards Data Science*. https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761. (Accessed on 03/07/2021).

[39] *Machine Learning: What it is and why it matters | SAS*. https://www.sas.com/en_us/insights/analytics/machine-learning.html. (Accessed on 04/05/2021).

[40] *Microsoft Word - document.doc*. https://arxiv.org/pdf/1011.0628.pdf#:~:text=Classification. (Accessed on 03/07/2021).

[41] *Multiple Linear Regression*. http://www.stat.yale.edu/Courses/1997-98/101/linmult.htm. (Accessed on 01/18/2021).

[42] *Naive Bayes Classifier. What is a classifier? | by Rohith Gandhi | Towards Data Science*. https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c. (Accessed on 03/10/2021).

[43] *Oracle Data Mining*. https://www.oracle.com/database/technologies/advanced-analytics/odm.html. (Accessed on 03/12/2021).

[44] *Overfitting and Underfitting With Machine Learning Algorithms*. https://machinelearningmastery.com/overfitting-and-underfitting-with-machine-learning-algorithms/. (Accessed on 04/05/2021).

[45] *P-Value in Statistical Hypothesis Tests: What is it? - Statistics How To*. https://www.statisticshowto.com/probability-and-statistics/statistics-definitions/p-value/. (Accessed on 01/18/2021).

[46] *Polynomial Regression. This is my third blog in the Machine... | by Animesh Agarwal | Towards Data Science*. https://towardsdatascience.com/polynomial-regression-bbe8b9d97491. (Accessed on 01/18/2021).

[47] *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and ... - Google Books*. https://books.google.com/books?hl=en&lr=&id=IC4Ku_7dBFUC&oi=fnd&pg=PA281&ots=nPTgH1IcuN&sig=OONp6g8VyWXiLKQPms0-eFXuc1E#v=onepage&q&f=false. (Accessed on 03/12/2021).

[48] *r - what does an actual vs fitted graph tell us? - Cross Validated*. https://stats.stackexchange.com/questions/104622/what-does-an-actual-vs-fitted-graph-tell-us. (Accessed on 01/20/2021).

[49] *Random Forest Regression. In this blog we'll try to understand... | by Afroz Chakure | The Startup | Medium*. https://medium.com/swlh/random-forest-and-its-implementation-71824ced454f. (Accessed on 01/25/2021).

[50] *Regression Analysis Essentials For Machine Learning - Easy Guides - Wiki - STHDA*. http://www.sthda.com/english/wiki/regression-analysis-essentials-for-machine-learning. (Accessed on 01/17/2021).

[51] *roymgabriel/SIP*. https://github.com/roymgabriel/SIP. (Accessed on 04/09/2021).

[52] Stuart Russell and Peter Norvig. *Artificial Intelligence. A Modern Approach*. Prentice Hall Press, 2009.

[53] *sklearn.ensemble.RandomForestClassifier — scikit-learn 0.24.1 documentation*. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html. (Accessed on 03/10/2021).

[54] *Support Vector Machine — Introduction to Machine Learning Algorithms | by Rohith Gandhi | Towards Data Science*. https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47. (Accessed on 03/07/2021).

[55] *The Random Forest Algorithm: A Complete Guide | Built In*. https://builtin.com/data-science/random-forest-algorithm. (Accessed on 03/10/2021).

[56] *Theology of Stewardship (1981)*. https://pcahistory.org/pca/digest/studies/3-457.html. (Accessed on 04/04/2021).

[57] *Understanding Support Vector Machine Regression - MATLAB & Simulink*. https://www.mathworks.com/help/stats/understanding-support-vector-machine-regression.html. (Accessed on 01/21/2021).

[58] *Understanding the concept of Hierarchical clustering Technique | by Chaitanya Reddy Patlolla | Towards Data Science.* https://towardsdatascience.com/understanding-the-concept-of-hierarchical-clustering-technique-c6e8243758ec. (Accessed on 03/12/2021).