VICTORIA UNIVERSITY OF WELLINGTON
Te Whare Wānanga o te Ūpoko o te Ika a Māui

School of Engineering and Computer Science
Te Kura Mātai Pūkaha, Pūrorohiko

PO Box 600
Wellington
New Zealand

Tel: +64 4 463 5341
Fax: +64 4 463 5045
Internet: office@ecs.vuw.ac.nz

Identification of Irrigated Land Using Machine Learning Techniques

Harsh Panchal

Supervisors: Andrew Lensen, and Harith Al-Sahaf

Submitted in partial fulfilment of the requirements for
Master of Computer Science.

Abstract

The analysis of remote-sensing data using machine learning and or deep learning techniques is becoming popular. Accurate mapping and identification of agricultural land use are essential to many agricultural applications such as identification of irrigated land and identification of what crops are being grown on a particular piece of land. Previously, implementations using remote sensing and machine learning for the agricultural domain have been carried out. There have been efforts to solve challenges in the agricultural domain such as identification of the quality of soil, seed quality and crop disease detection. This research aims to utilise the considerable Sentinel 1 and Sentinel 2 dataset which is open-sourced, there also exists an open-source dataset that provides demarcated irrigated land provided by the Ministry for the Environment (MFE). In practice these datasets are created using a desktop-analysis method which is a labour-intensive task. This research aims to identify irrigated land from Sentinel imagery which could be used to deconstruct the patterns and effects of consumption of water use by these landowners. Likewise, other useful statistics could also be derived from such identification. This research aims at utilising machine learning techniques to accurately identify irrigated land thus making the process much more automated and less manual.

Contents

Figures

Chapter 1

Introduction

Irrigation is an important part of agriculture, when irrigation occurs and how it changes over time and is essential to understanding, managing resources and can aid in the derivation of other useful statistics such as water usage and soil salinity. Most large scale maps of irrigation have been produced at low resolutions or rely on remote sensing data combined with national statistics. The identification and mapping of irrigated land can be cumbersome and time-consuming when done manually using Geographic Information System (GIS) software by experts. These maps can also be biased and not capture high resolution trends due to labor-intensive collection of data. The low resolution of maps previously generated were developed on large scales due to the extensive resource requirements to process satellite imagery. However, an increase in the availability of remote sensing imagery and image classification methods present a cost-effective and accurate means to identify and map irrigated land by automating the process. This research aims to identify irrigated land from non-irrigated land using machine learning techniques such as classification from sentinel imagery which has become a vital source of information in land resource management and also has applications for agricultural land..

Classification can be defined as a systematic arrangement of data in groups or categories according to an established criteria. A simple example could be categorizing hurricanes in category one to category five based on a scale measuring the severity [1]. A supervised classification task such as image classification in the machine learning domain is concerned with learning a function that maps an input to different classes or categories based on example input and output pairs provided to the model during the training or learning phase based on features of the input image. Image classification is an important task in the computer vision domain.

There are various algorithms that can be used as classifier such as Random Forest [16], Support Vector Machines [42], K-Nearest Neighbors and Multi-layer perceptrons [34]. These algorithms provide us with different techniques for classifying pixel values from images which can be extracted and assigned a class or label this is then used to train the model to recognize patterns in the data which can be used to further make predictions on images unseen by the model. Random forest is the most commonly selected algorithm of choice for a classification task involving remote sensing and agricultural machine learning tasks such as classification of irrigated land from non-irrigated land and soil evaluation [15, 33].

1.1 Motivation

Irrigated agriculture contributes to approximately 40% of the total food produced worldwide, but accounts only for 20% of total cultivated land [2]. 85-90% of all freshwater that

can be consumed globally is used by irrigation. The Food and Agriculture Organization forecasts that by 2030 the world would have increased irrigated land by 34% [2]. Irrigation is necessary for agriculture since it stimulates crop growth under unfavourable climatic conditions. However, it also has an impact on the environment, such as draining wetlands, changes in the quality of the soil, decreased streamflow, and in particular, water consumption. A study conducted by AquaLinc in 2020 found that New Zealand has a total of 903,465 hectares of irrigated land out of this Canterbury had a total of 546,205 hectares of irrigated land accounting for 64.05% of the total irrigated land area. Many studies for identification of irrigated land has been carried out in the past in various countries, to the best knowledge at the time of writing this thesis no such work exists for New Zealand except the desktop analysis approach utilized by AquaLinc [18] which is a manual task, time consuming and is carried approximately once every three years.

It is of utmost importance to not only evaluate soil when it comes to agriculture, but evaluation of land is also necessary this helps in making decisions regarding land use development. Benefits of land evaluation for irrigated agriculture is to predict future conditions after development and to foresee benefits to the farmers and national economy and whether these can be sustained without damage to the environment or the economy of the nation. Although this research does not focus on land evaluation, identification of irrigated and non-irrigated land could be considered the first step towards evaluation of land. An automated process to identify irrigated or non-irrigated land from available sentinel imagery can help reduce the time required to map them using a desktop-analysis method or surveys. According to a report by Stats NZ, there has been a 91% increase in irrigated agricultural land in New Zealand between 2002 and 2019 [13]. If we do not track the growth of irrigated land it could also be problematic to other resources such as water. An increase in irrigated land means more water abstraction and increased pressure on river flows, freshwater habitats, mahinga kai(traditional value of food resources and their ecosystems), and well-being of species such as tuna, kākahi (freshwater mussels), kōura (freshwater crayfish), and īnanga (whitebait). These are important species to the people of Aotearoa and valued taonga (treasure).

A convolution neural network approach proposed by Colligan et al. [17] used an ensemble of convolution neural networks which rely on reflectance information from Landsat imagery to classify image pixels corresponding to irrigated land. This method does not depend on exhaustive feature engineering or require high computational resources. Ambika et al. [15] proposed utilizing Moderate Resolution Imaging Spectroradiometer (MODIS) data for identification of irrigated land in India using a decision tree classification algorithm where Normalized Difference Vegetation Index (NDVI) was used to identify a threshold for individual class cluster.

Qi et al. [33] proposed utilizing a non-machine learning approach, where images from summer and winter seasons are used. Along with these ground references from 139 countries were also used and applied masks to collected signatures which could be out of 3 threshold ranges and the ground truth were laid on these threshold levels and the method with accurate pixels matching ground truth is utilized.

This research project aims at building a binary classifier for landcover classification to identify irrigated from non-irrigated land which can be achieved by employing a machine learning approach such as Random Forest and other algorithms to the sentinel 2 data. There have been previous attempts to use random forest algorithm [25], but with different data sources. This project proposes to narrow its focus to the Canterbury region of New Zealand because of the highest percentage of irrigated land is in this region [18]. The different bands in the Sentinel imagery and various indexes calculated based on these bands can further help in identifying whether the target land plot has live vegetation or not. The contrast between

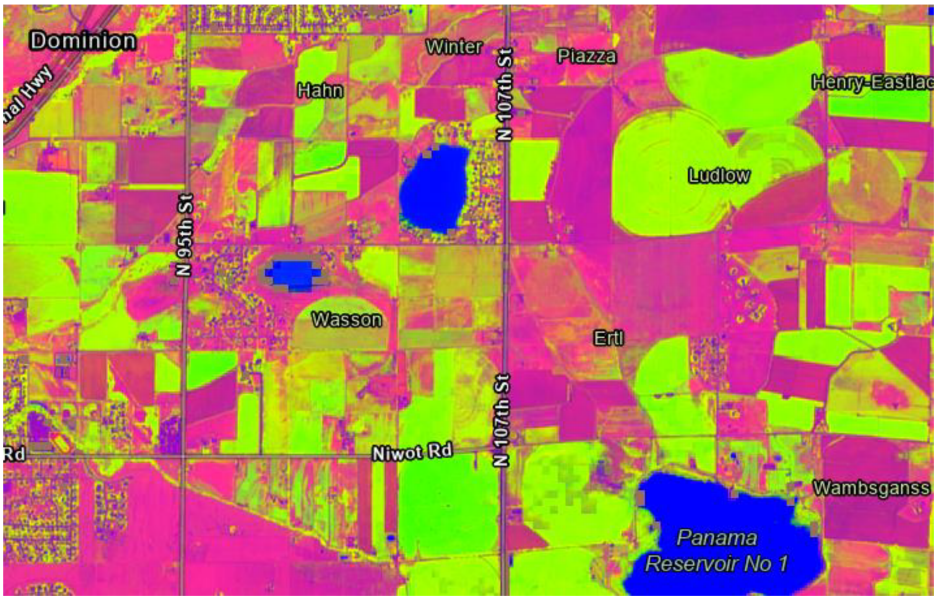

Figure 1.1: Sentinel 2 imagery over Colorado.

irrigated and non-irrigated land in the NDVI and aerial imagery varies between regions. The study conducted by AquaLinc[18], states that the Canterbury region has an extreme contrast between irrigated and non-irrigated land and that this contrast is the highest in the region of New Zealand. The imagery in Figure 1.1 illustrates how indicators such as NDVI and NDWI can be used to classify irrigated land and non-irrigated land.

- Red – Barren earth / soil
- Green - Normalised Difference Vegetation Index
- Blue – Normalised Difference Water Index

1.2 Research Goals

The overall goal of this research is to utilise a machine learning approach that can aid and automate the task of identification of irrigated land from sentinel imagery. This study is carried out in the Canterbury region in New Zealand. This research will help to reduce the labour-intensive task of identifying such land using the desktop analysis method and will enable maps for irrigated and non irrigated land to be developed frequently with high accuracy. Specifically, this study has the following objectives

- This study will investigate the performance of utilising multiple supervised machine learning techniques to classify sentinel image pixels into irrigated and non-irrigated lands.
- Comparing the performance of machine learning techniques to that of the desktop analysis method.
- Understanding the number of samples required for training a model for the task at hand.

1.3 Major Contributions

This thesis will contribute to the research previously done for land use using machine learning techniques and artificial intelligence by:

- Showing how to design and build a supervised learning classifier that can classify irrigated land from non-irrigated land using sentinel-2 satellite imagery in New Zealand.
- Comparing and contrasting few classification algorithm that helps understand which algorithm works best.
- This project is the only automated approach made open source specific for the country of New Zealand aiding agriculture using machine learning.

1.4 Organization

The rest of the report is organized as follows. Chapter 2, gives a brief introduction to the field of remote sensing and sentinel-2 imagery, machine learning techniques e.g., supervised learning, unsupervised learning, semi-supervised learning and reinforcement learning, classification algorithms e.g., random forest, k-nearest neighbor and multi-layer perceptrons, concepts of feature selection and finally concepts regarding hyperparameter tuning. The benchmarks, datasets and baseline approaches are discussed in chapter 3. Chapter 4 explains in depth the experiment designs, results and result discussions are discussed. Finally the conclusion along with future work is discussed in chapter 5.

Chapter 2

Background

This chapter discusses and describes remote sensing, sentinel imagery and satellites, basic machine learning terminologies and concepts, focusing on classification problem. The different algorithms explained in this sections are random forest, k-nearest neighbors, support vector machine and multi-layer perceptrons. The multicolinearity problem along with its solution for feature selection is also explained. Finally, a presentation of related work in literature survey closes the discussion of the chapter.

2.1 Remote Sensing and Sentinel Imagery

Remote sensing is the process of acquiring information about the earth's surface by measuring its reflected and emitted radiation without coming directly in contact with the object. The process of remote sensing involves interaction between incident radiation and target of interest. Remote sensing includes various useful electromagnetic radiation such as visible light (VIS), near infra red (NIR), shortwave infrared (SWIR), thermal infrared (TIR) and microwave bands. Remote sensing was first discovered in the 1800's and in the last two decades remote sensing techniques are applied to explore agriculture applications such as crop discrimination, crop estimation, soil survey etc.

There are various satellites that collect remote sensing data of which some are freely available and some are provided through commercial satellites. Some of the most widely used satellite data are Sentinel-2, Sentinel-1, MODIS data and Landsat data. The data used for this research is Sentinel-2 data which delivers high resolution optical imagery for land-monitoring, emergency responses and security services with a common purpose for land-cover and land-change detection maps, monitoring of vegetation and burned areas. There are 13 spectral bands available: 4 visible bands (10m spatial resolution), 6 near infrared (20m) and 3 shortwave infrared (60m). The Sentinel-2 data has 5 days revisit time. This imagery is orthorectified, spatially co-registered data.

The process of a satellite hovering over an area and scanning the Earth surface is known as a "datatake" which means the continuous acquisition of an image from one Sentinel-2 satellite in a given imaging mode. The maximum length of a datatake is 15,000 km. The ground receiving station is incapable of receiving such a huge area and hence "datastrips" are used which is within a given datatake, a portion of image is downlinked during a pass to the station. The maximum length of a datastrip downlinked is 5000 km. When considering 25x23 km sub images these are referred as "granules" and 100x100km sub images are referred as "tiles" as depicted in figure 2.1

The 13 bands of Sentinel-2 data is explained in the Table 2.1

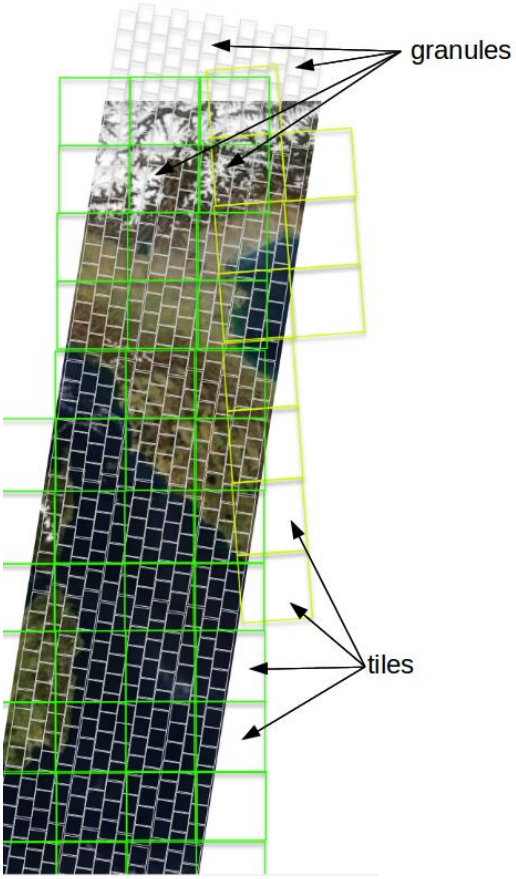

Figure 2.1: Representation of Tiles in Sentinel Imagery.

2.2 Machine Learning Techniques

Computers can be made to learn tasks and get better at it using machine learning which is made up of statistics and computer science. The various algorithms that have been used broadly and in different ways allow the computer to solve a variety of problems some of which could be pre-defined.There are different techniques through which a machine can learn such as supervised learning, unsupervised learning, semi-supervised learning and reinforcement learning.

2.2.1 Supervised Learning

Supervised learning means having an expert judge whether we are able to find the correct answer or not. Similarly, in machine learning supervised learning, means having a fully labelled dataset while training an algorithm [24]. Fully labeled meaning that every example in our training data has an answer attached to it from which the machine can learn. For example, if we are trying to identify if a email is a spam or not, a labelled dataset of emails will tell the model which email is a spam and which is not. Supervised learning techniques is visually explained in Figure 2.2.

2.2.2 Unsupervised Learning

Getting access to labelled data or generating labelled data can be time consuming, labour intensive and also expensive in most cases. The alternative to supervised learning is un-

Table 2.1: This table shows 13 bands of sentinel 2 data

Band Name	Description	Band Number
AOT	Aerosol Optical Thickness	1
B01	Coastal Aerosol	2
B02	Blue	3
B03	Green	4
B04	Red	5
B05	Red Edge 1	6
B06	Red Edge 2	7
B07	Red Edge 3	8
B08	Near Infrared	9
B09	Water Vapour	10
B11	Short Wave Infrared 1	11
B12	Short Wave Infrared 2	12
B8A	Narrow Near Infrared	13

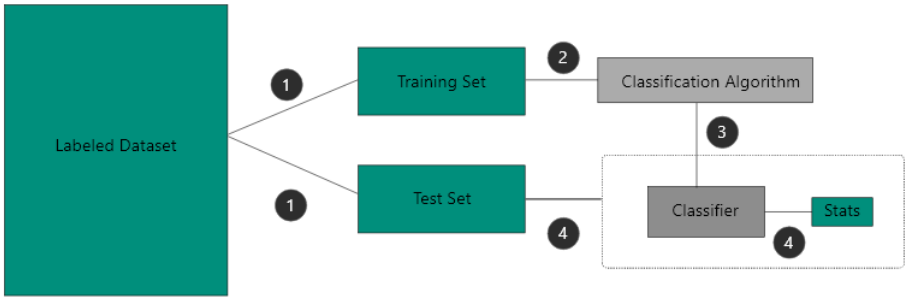

Figure 2.2: Supervised learning techniques example (classification).

supervised learning, where there are no labels or correct answers attached to the training dataset [19]. A simple example could be clustering where points in the dataset are clustered in different groups. The figure Figure 2.3 represents clustering of flowers based on petal width and petal length.

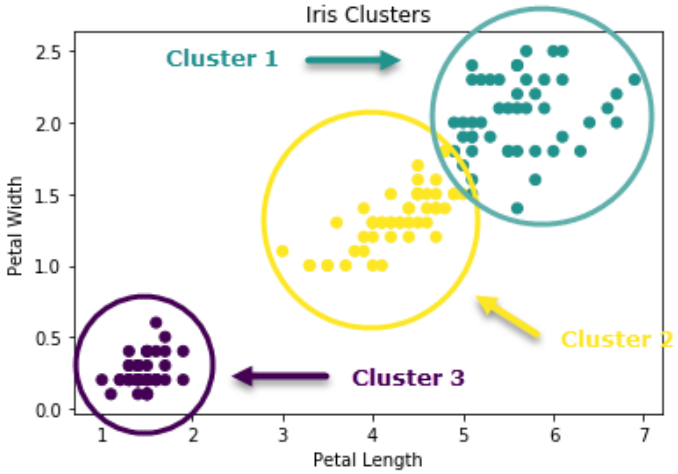

Figure 2.3: Unsupervised learning techniques example (clustering).

2.2.3 Semi-Supervised Learning

Semi-supervised learning is the combination of supervised learning and unsupervised learning techniques. Semi-supervised techniques work with datasets that have both labelled and unlabelled data [20]. Semi-supervised method is particularly useful when it is time intensive to label data and also difficult to extract features from the dataset automatically. A semi-supervised learning can be used in domain such as the medical domain. Images in the medical domain such as CT-Scans and MRI's can be labelled in a small amount by an expert radiologist and this can still benefit the model learning and improve accuracy compared to a unsupervised technique.

2.2.4 Reinforcement Learning

Reinforcement learning operation can be easily understood on the principles of level based games where on completing a level earns a badge or a reward to the player. In fact, video games are the most common testing environment for reinforcement learning. In this learning type the main goal for an agent is to identify next optimal step to reach a goal or optimize performance for a particular task. As the agent takes a step towards the goal receives a reward [22].

To make choices, the agent depends on previous feedback and explores new tactics that may help get a larger payoff. It is an iterative process, higher number of iteration better the performance or strategy of the agent. This can be used to train robots or for autonomous driving cars.

In this research, the supervised machine learning technique is used due to availability of the data with labels "Irrigated" and "Non-Irrigated" . The following section explains the classification task and different algorithms such as Random Forest, K-Nearest Neighbors and Multi-Layer Perceptrons that have been used to solve the problem of identification of irrigated and non-irrigated land.

2.3 Classification and Classification Algorithms

The aim of this section is to discuss the task of performing classification in a dataset. Precisely, this section is concerned with highlighting the problems and difficulties of classification.

Classification means categorizing given set of data into classes based on certain features. Classes can be referred to as target, label or categories. The classification predictive modeling is the task of approximating the mapping function from input variable to discrete output variables [23]. A simple example for classification can be identifying whether an email received is spam or not-spam as demonstrated in Figure 2.4.

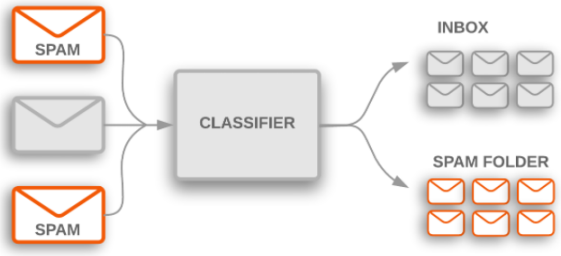

Figure 2.4: Spam email classification [12].

There are a number of issues that increase the difficulty of performing classification by humans as follows:

- High similarity between examples / images of different classes;
- In case of classifying images, large number of images may need to be classified;
- Images in an image classification task can be noisy or unclear, which further increases the difficulty to reveal their identity;
- A domain expert with background knowledge is needed to analyse and discriminate between examples of different classes, which can be expensive and time consuming.

Hence, automating the classification task with the help of machine learning algorithms that have been researched and worked upon previously help perform classification with ease. For this project, classification is done on pixel values obtained from various sentinel imagery bands which are explained in further sections. These extracted pixel values which can also be referred to as features of the sentinel imagery can be fed to various algorithms such as Decision trees (DT), Multi-layer perceptrons (MLP), Support vector machines (SVM), *k*-nearest neighbors (KNN). Even though the classification task is automated a human in the loop of training the model is still required and plays a key component for initial classification of features to classes also known as data labelling which will be fed to the model as training examples for the model to learn and aid the supervised learning.

There are many algorithms that can be used to perform classification. Many machine learning algorithms are stochastic, which means there is certain randomness every time the algorithm is run. In the following sections, the classification algorithms that are used for the identification of irrigated land are random forest, k-nearest neighbor and multi-layer perceptron, out of these classification algorithms, random forest and multi-layer perceptrons are stochastic methods.

2.3.1 Random Forest

Random forest is a machine learning algorithm that produces acceptable results in most cases even with default parameters and without hyper-parameter tuning. Over the last two decades, use of random forest (RF) classifier [16] has received attention due to excellent classification results and speed of processing in remote sensing [32, 31, 36].

Random forest is a supervised learning methodology. The forest build is an ensemble of decision trees [28], usually trained with the "bagging" method [16]. In the simplest form, bagging method builds multiple decision trees and merges them together to get accurate and stable prediction described in Figure 2.5.

To build a single decision tree a subset of the training sample is selected. At each node of the tree randomly subset of features are selected. For each of the feature, different thresholds are tested to see how they split the samples according to a given criterion such as gini impurity or information gain. The feature and the threshold that best splits the data is recorded in a node. When the construction of this single tree ends, depending on factors such as maximum depth or minimum sample number is reached, samples in each leaf are looked and the frequency of the labels are maintained. As a result a tree with partition of training sample according to meaningful samples are created. Since each node is created from randomly selected features, each tree built in this way is different and is more generalized.

During testing phase a sample from the test set will go through each tree, giving a label frequency for each tree. Most represented label is generally the final classification result.

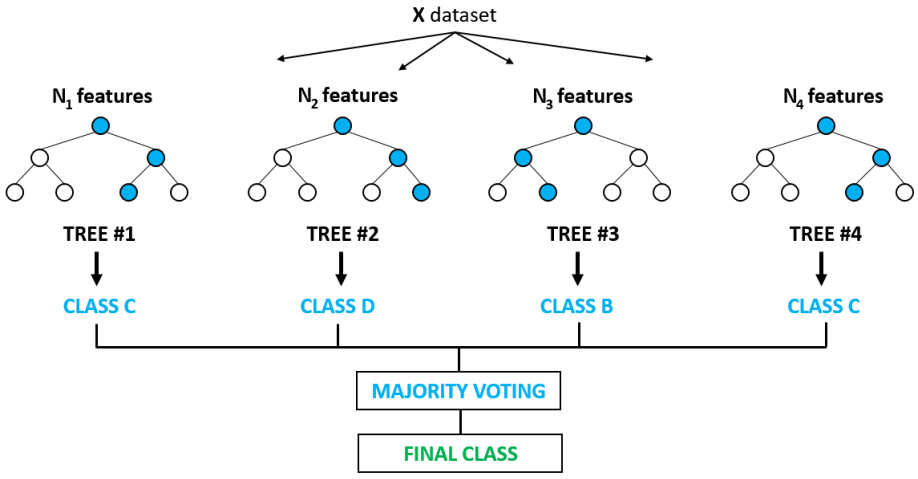

Figure 2.5: Random forest tree example.

2.3.2 K-Nearest Neighbor

K-nearest neighbor (KNN) is also a type of supervised learning algorithm which can be used for classification and regression. In this section KNN will be explained in terms of classification. KNN predicts the label for a new data based on distance between the new data and all the training points. It then selects closest data points from the training set and the respective class labels [43]. Probability of the test data belonging to classes of the training data is calculated and the class with highest probability is selected as the class for the new data point, this is shown in the Figure 2.6.

The distance between the new data point and all training points can be done using distance measures such as Euclidean distance, Manhattan distance, Hamming distance and Brute force.

A few disadvantages of KNN is that it requires high memory to store all the training data, and given that it stores all training, it can be computationally expensive.

2.3.3 Multi Layer Perceptron

The power of neural networks is based from the ability to learn representation in training data and how to relate it to a class or label. In a mathematical sense, neural networks are capable of learning any mapping function. The building block of a multi-layer perceptron is a neuron. These are simple units that have weighted input signals and produce an output signal using an activation function as depicted in Figure 2.7.

These perceptrons when arranged into a network are known as multi layer perceptrons and can be used for classification tasks as well as regression tasks [27]. There have been uses of multi-layer perceptrons in remote sensing classification tasks before returning promising results[35]. Multi-layer perceptrons are made up of three types of layers

- Input
- Hidden layers
- Output

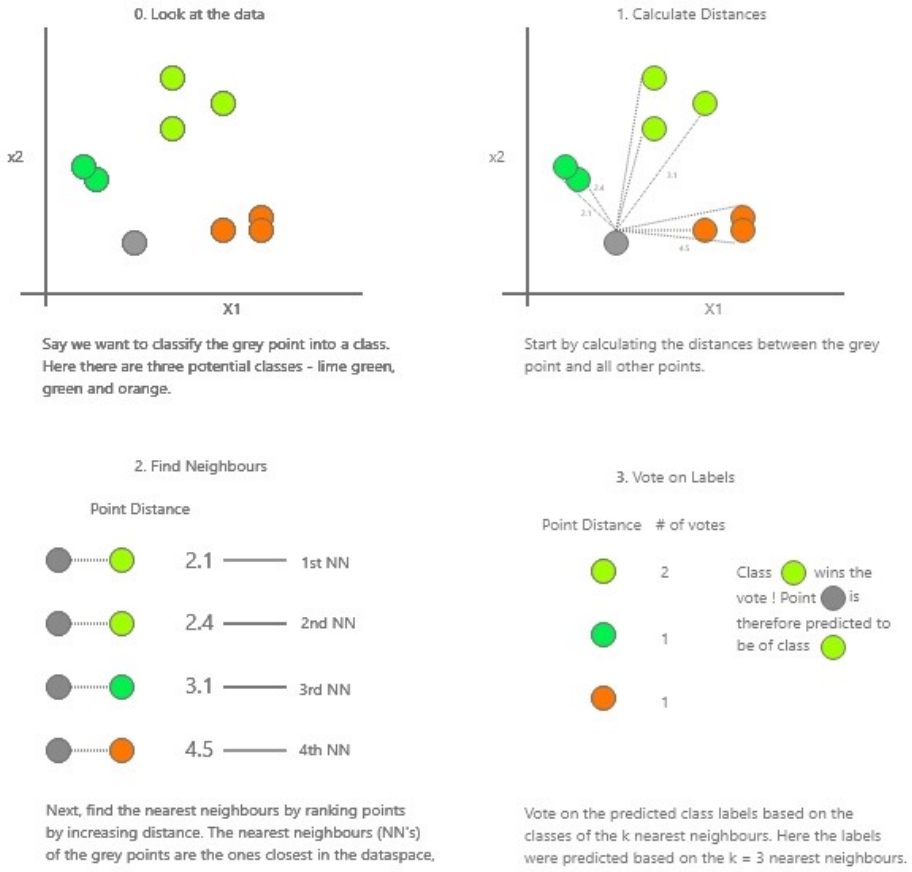

Figure 2.6: K-Nearest neighbour example.

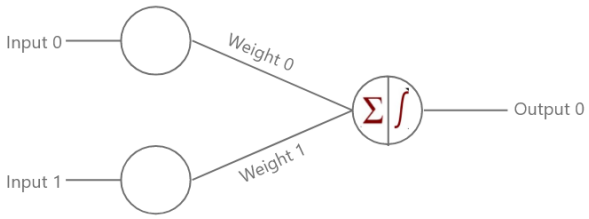

Figure 2.7: Perceptron representation.

The input layer is also called the "Visible" layer because it is responsible to take input from the dataset. Every neuron in this layer depicts per input value or column in the dataset. The hidden layers are not exposed directly and the simplest network would have a single neuron in the hidden layer that predicts the output directly. But with increasing difficulty of problems multiple hidden layers can be present in a network. Finally, the output layer is responsible for outputting a value determining the class or label of the input.

2.4 Dimensionality Reduction

Dimensionality reduction aims at reducing the number of features in a dataset. Dimensionality reduction is performed for the following reasons:

- To increase the accuracy of trained models, by removal of redundant, noisy and irrelevant features;
- Avoiding the curse of dimensionality. Curse of dimensionality means any and all problems rising with a large dataset;
- Avoiding overfitting;
- To simplify the model which in turn can help to reduce time required for training the model.

The aim of using dimensionality reduction in this study is to test whether dimensionality reduction can help to improve the performance on the problem at hand and the experiments and results are discussed in Chapter 4. There are various methods for dimensionality reduction; in this research principal component analysis and feature selection is used to investigate the effect of dimensionality reduction on the performance of identifying irrigated lands.

2.4.1 Principal Component Analysis

Principal component analysis (PCA) is one of the methods to reduce features which is referred to as dimensionality reduction. As the name suggests PCA helps identify principal components of data. It does so by identifying corelations between data and transforms them in a dataset with significantly lower dimensions without loss of information and thus selecting the principal components.

PCA can be done using five steps which are:

- Standardization of data;
- Computing covariance matrix;
- Calculating the eigenvectors and eigenvalues;
- Computing the principal components,
- Reducing the dimensions of the data set.

2.4.1.1 Standardization Of data

Standardization is the process of scaling the data in a manner that all the variables and their values lie within a similar range. Standardization is carried out by subtracting each value in the data from the mean and dividing it with overall deviation in the data set.

$$Z = \frac{Variable\ value - Mean}{Standard\ deviation} \tag{2.1}$$

2.4.1.2 Covariance Matrix

PCA as mentioned identifies correlation and dependencies among the features in the data set.A covariance matrix explains the correlation between the different variables in the dataset.

Mathematically a covariance matrix is a $p \times p$, where p represents dimensions. Each entry represents covariance of corresponding variables. Consider the following example of 2-dimensional data with variables a and b

$$\begin{bmatrix} Cov(a,a) & Cov(a,b) \\ Cov(b,a) & Cov(b,b) \end{bmatrix} \tag{2.2}$$

In the above matrix:

- $Cov(a,a)$ represents the covariance of a variable with itself, which is nothing but the variance of the variable a;
- $Cov(a,b)$ represents the covariance of the variable a with respect to variable b.

2.4.1.3 Calculating the Eigenvectors and Eigenvalues

Calculation of eigenvectors and eigenvalues is another important step towards selection of principal component analysis. Eigenvectors help better understand where in the data is the most variance with the help of covariance matrix calculated in the previous step. For every eigenvector there is a corresponding eigenvalue and hence the eigenvalue simply denote the scalar of the respective eigenvector.

2.4.1.4 Computing Principal Components

After calculation of the eigenvectors and eigenvalues these need to be ordered in an descending order. The highest eigenvalue is most significant and form the first principal component. Principal components of lesser significance are removed to help reduce dimensions.

All data variables that posses maximum information are used to form a matrix known as feature matrix.

2.4.1.5 Reducing the Dimensions of the Data Set

Finally the original data is rearranged with the principal components that best represent information of the data set. To replace the original data axis with new components a multiplication between the transpose of the original data and the transpose of the obtained feature matrix is needed to be performed.

2.4.2 Feature Selection

It is no secret that machine learning methods are highly depended on the quality of the data received as input also most commonly referred as "garbage in, garbage out". The possibility of unclean data, meaning missing values or noisy data being present is more when there is a huge number of features present in the dataset.

Various methods have been proposed in the literature for feature selection that can be used to select a subset of original features of the dataset to help building a better model. Generally, feature selection methods can be categorized into the following groups [3]:

- Filter methods: Filter methods are a pre-processing step, where features are selected independent of any machine learning algorithms and are based on scores from various statistical test such as the correlation of features with the class labels.
- Wrapper methods: In wrapper methods, a subset of features are selected and a model is trained on these features. Based on the performance of the trained model, features are added or removed. This method is usually computationally expensive.
- Embedded methods: Embedded methods combine the qualities of filter methods and wrapper methods, it is implemented by algorithms that have an internal feature selection methods such as genetic algorithm for feature selection.

2.4.2.1 Genetic Algorithms

Genetic algorithms is one of the state of the art techniques for feature selection [30] and falls under the embedded based method of feature selection. Genetic algorithms are based on the natural genetics and biological evaluation. In nature genes tend to evolve over successive generations to better adapt to the enviorment.

The building blocks of genetic algorithms are chromosome, selections, generations, population, fitness function, crossover and mutations. The simplest explanation for a genetic algorithms are that they operate on population of individuals to produce better approximations. At each generation a new population is created by selecting individuals according to the level of fitness in the problem domain and recombining them using operators borrowed from the original genetics. The offspring can also undergo mutation. Figure 2.8 depicts the genetic algorithm process.

The following subsections talk about the process of feature selection used which is based on the basic building blocks of genetic algorithm. Most examples are referred from [4].

Step 1: Initialization First step is to initialize individuals in the population. Since genetic algorithm is a stochastic optimization method the individual genes are usually initialized randomly.

Considering that we have with six features. If we generate population of four individuals, we have four different random features as shown in Figure 2.9. In the diagram above each positive gen means that the corresponding feature is included.

Step 2: Fitness Assignment Every individual in the generated population in Step 1 needs to be assigned a fitness value. An algorithm of choice is then trained with training values and errors are evaluated. High error means low fitness and low error means high fitness. Only individuals with high fitness (low errors) are selected for the next step.

Fitness assignment is done using rank based method where every individual is sorted based on their errors. The formula for the rank-based method is:

$$\Phi(i) = k \cdot R(i), \qquad \text{where } i = 1, 2, \ldots, N. \tag{2.3}$$

The fitness assigned to each individual only depends on its position in the individuals rank and not on the actual error.

As shown in Table 2.2 we have chosen k = 1.5 to calculate the fitness value. The above table can be presented in a pie chart. Area for each individual is proportional to fitness as shown in Figure 2.10.

It is clear from Figure 2.10 that fittest candidate is individual 4 and least fittest is the individual 1.

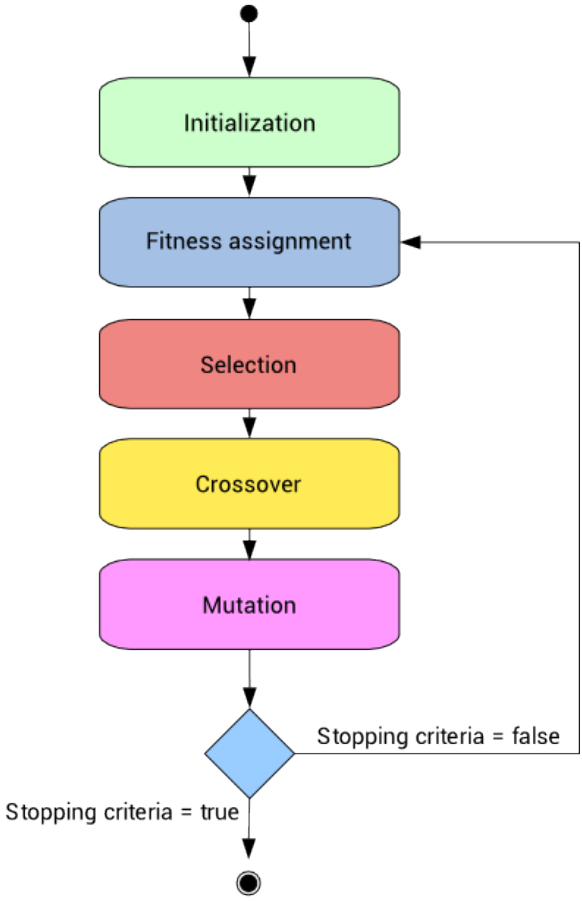

Figure 2.8: Genetic algorithm process. [5]

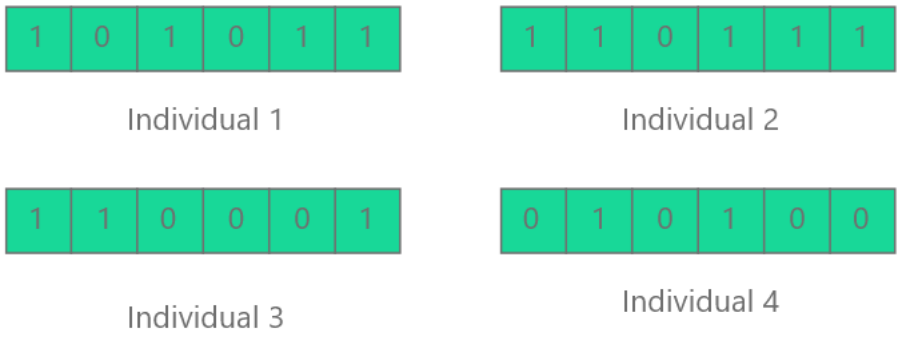

Figure 2.9: Initialization step.

Table 2.2: Fitness Assignment Example

	Selection error	Rank	Fitness
Individual 1	0.9	1	1.5
Individual 2	0.6	3	4.5
Individual 3	0.7	2	3.0
Individual 4	0.5	4	6.0

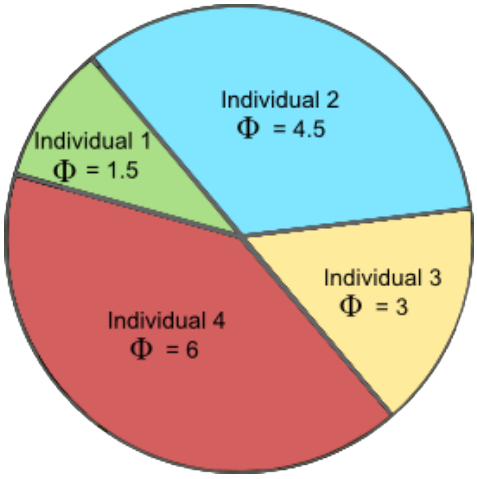

Figure 2.10: Individual fitness pie chart.

Step 3: Selection In the selection phase a selection is done for combining individuals for the next generation.The number of selected individual is N/2 in total at each generation, N being the population size.

One way to approach selection is elitism where the most likely to survive are the fittest individual. Usually the elitism is usually set to a small value.

Another method is the roulette wheel, also known as stochastic sampling with replacement. Individuals in this method are selected at random. And the selected individual along with the one selected in the elitism approach are used for recombination.

Figure 2.11 illustrates the selected individuals.

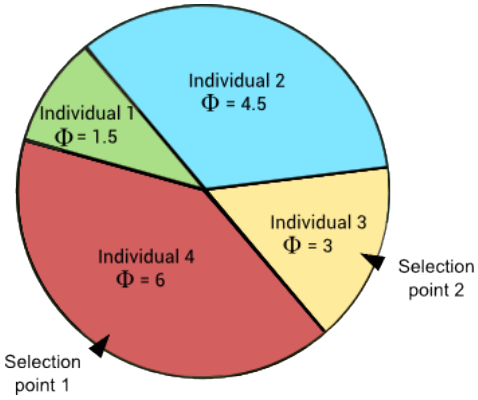

Figure 2.11: Selection pie chart example.

It can be seen in Figure 2.11 that individual 4 has been selected by the elitism method and individual 3 has been selected by the roulette method. Number of individuals selected is always half of the population.

Step 4: Crossover The crossover operation is responsible for recombining the selected individuals to generate a new population. Two individuals at random are selected and combined to get newer off-springs for the new population. This is done till the new population has the same size as the old one.

Figure 2.12 depicts the four off-springs generated here the population size remains constant.

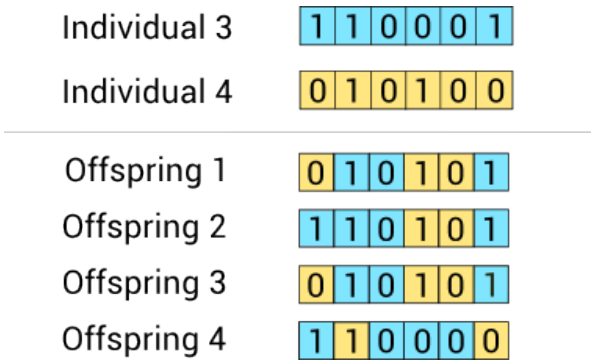

Figure 2.12: Crossover Example.

Step 5: Mutation There is a high probability of crossover generating off-springs similar to parents which may cause the new generation to have little diversification. This is solved by the mutation operation by changing values of certain features in the offspring randomly. This is done using mutation rate where a random number is generated between 0 and 1 and if the number is lower than the mutation rate then the variable is flipped. Figure 2.13 depicts the mutation step and at this stage we have a new generation

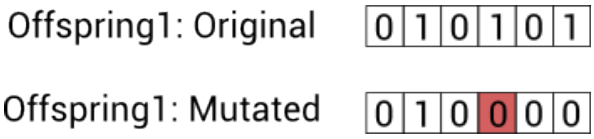

Figure 2.13: Mutation Example.

All of the above steps are repeated until a stopping criterion is reached. Each generation is more adapted to the environment than the old one.

2.5 HyperParameter Tuning

During the training phase of a machine learning model there are numerous parameter settings that need to be set manually to achieve an optimal accuracy and ensure that the model does not overfit. Selecting these parameters initially at random or working with default parameters for the baseline model helps get a feel for the accuracy achieved by the dataset in use.

There is usually a confusion between model parameters and model hyperparameter.

- Model Parameters : These are parameters that are estimated automatically by the model from the given data. Example : Weights of a neural network
- Model HyperParamters : These are parameters that cannot be estimated by the model automatically and need to be set manually. Example : Learning rate in neural networks.

The best way to understand hyperparameters that are best suitable for the model selected to solve the problem at hand is a method called hyperparameter tuning. It helps determine the right combination of hyperparameters and helps maximize model performance and avoids the issue of overfitting.

There are two approaches for hyperparameter tuning:

- Manual hyperparameter tuning : different combinations are selected from various hyperparameters available and are experimented with.
- Automated hyperparameter tuning : optimal hyperparameters are found using an algorithm choice that automates and optimizes the selection of hyperparameters.

There are various automated hyperparameter tuning tools such as

- Scikit-learn
- Hyperpot
- Scikit-Optimize
- Optuna
- Ray.tune

The most common tool is scikit-learn which provides two options the GridSearchCV and the RandomSearchCV method.

2.5.1 GridSearchCV

In this method predefined hyperparameter values of the model being used are looped through and a model is fit on the training set. For example, twenty different parameter values for each of four parameters will require one hundred and sixty thousand trials of cross validation. This would evaluate to one million six hundred thousand model fits and one million six hundred thousand predictions in ten cross validation. In the end, the best parameters from the listed hyperparameters are selected. This is done by defining a dictionary in which hyperparamters are predefined with values it can take, an example of such a dictionary can be seen in Figure 2.14.

```
n_estimators = [int(x) for x in np.linspace(start=200,stop=2000,num=10)]
max_features = ["auto",'sqrt']
max_depth = [int(x) for x in np.linspace(10,110,num=11)]
max_depth.append(None)
min_samples_split = [2,5,10]
min_samples_leaf = [1,2,4]
bootstrap = [True,False]

random_grid = {'n_estimators':n_estimators,
                   'max_features':max_features,
                   'max_depth':max_depth,
                   'min_samples_leaf':min_samples_leaf,
                   'min_samples_split':min_samples_split,
                   'bootstrap':bootstrap}
```

Figure 2.14: Grid parameters example.

GridSearchCV method will try all the possible parameters that are in the dictionary and constatnly evaluates the model for each combination using a cross-validation method. An accuracy/loss for all the combinations of hyperparameters and the one with best performance is selected.

While Scikit Learn offers the GridSearchCV function to simplify the process, it would be an extremely costly execution both in computing power and time.

2.5.2 Randomized GridSearchCV

Similar to gridsearchcv predefined hyperparameter and possible values added to the dictionary and the process of fitting these possible values to a model and calculating the accuracy or loss of the model for the best hyperparameter is similar to that of gridsearchcv but the only and major difference is rather than trying every single combination it randomly selects these hyperparameter values and fits a model and selects the best hyperparameters.

This allows the users to explicitly control the combinations for the number of parameters. The number of iterations is set on time or resources available. Performance of a randomized gridsearchcv will always be lower than that of gridsearchcv in terms of selecting the best hyperparameters, most of the time the randomized gridsearchcv method generates the best hyperparameters list in shorter amount of time than that of the gridsearchcv method.

There can also be instances where the random gridsearchcv method performs better than the gridsearchcv method, this is depicted in Figure 2.15.

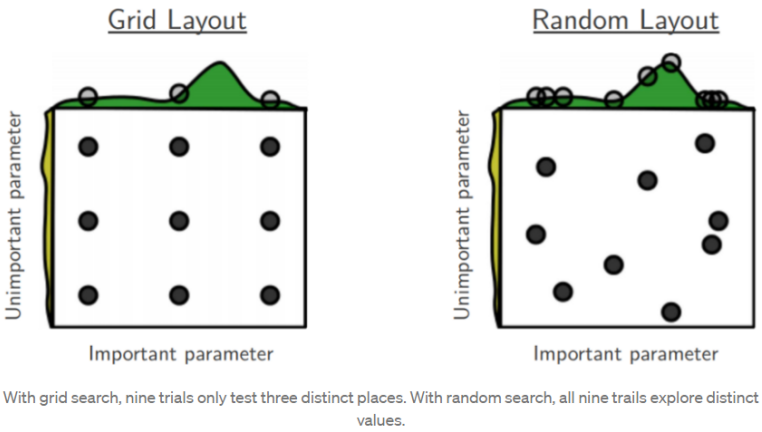

Figure 2.15: Grid parameters example[11].

2.6 Related Work

This section aims to provide related work of using different techniques to solve challenges in agriculture and land evaluation using non-machine learning techniques. The use of, machine learning techniques to solve these challenges. And finally, the use of Convolution Neural Network (CNN) method for identification of irrigated from Sentinel-2 imagery.

Machine learning and remote sensing techniques have been applied before in the agricultural domain as well as in healthcare, sales and space engineering. In the discourse around these techniques, focus is routinely on their diverse application to agricultural land. Research papers study agricultural concerns for example, area wise soil fertility [38], soil organic matter and pH level matter [40], moisture content [26], soil temperature [29], and the color, shape and texture of seeds [21], estimation of land use [41] and monitoring irrigated crop in saline areas [14]. Similar studies and research have been conducted extensively in countries such as India, the United States of America and Syria. Best to our knowledge, there has not been a published study in New Zealand for the identification of irrigated land. This research project could contribute to New Zealand by bridging the literature gap around the application of machine learning and artificial intelligence techniques for the agricultural domain.

2.6.1 Remote Sensing and Non-Machine Learning Approaches

Earlier approaches to identification of irrigated land used more manual approaches [39] and could be complicated and time consuming. Most of these methods also relied on selecting a cut-off threshold value to identify between irrigated and non-irrigated land. Qi et al. [33] conducted a research where the purpose of the research was to classify and map irrigated land in high plains aquifer by using satellite data because in 2002 the only information available for the entire region was 20 years old. The main idea represented was to have a comparison between amount of irrigated land determined in early 1980's and 1992 to understand if irrigated land has increased, decreased or remained the same. All the analysis done was achieved from hard-copy maps and annual reports collected by various offices, other co-operative agencies collected fact sheets with tabular data. The dataset used was satellite imagery from the Landsat Thematic Mapper. 40 summer and 40 spring images were acquired from national cover dataset and processed using band ratio method to enhance signatures. A ground reference from 139 countries equalling to 996 square miles was laid on the classifier and used to determine errors. The paper considers summer and winter imagery separately and divides them into 9 regions post which it applies masks and collects signatures which could be either threshold 1, threshold 2, threshold 3 and then the ground truth data is laid on with each threshold, one of the thresholds with the best percentage of pixels classified correctly is selected and the data is cleaned up. This method is done for both summer and winter imagery separately also known as Leaf on and Leaf Off. After this the datasets for both are merged and percentage for irrigated land is calculated for 4km square cells and then the data is compared to 1980 datasets. The overall weighted percent correct was 77.5 to 79.8 percent. The difference from 1980 (13.7 million acres) to 1992 (13.1 million acres) was found which was reduced. One obvious problem with this approach is that identification and selection of cut-off threshold value between irrigated and non-irrigated land.

Another non machine-learning approach proposed by KrishnaKutty et al. [15] focused on developing annual irrigated area maps at spatial resolution of 250m for 2000–2015 from MODIS data. Several spatial data sets of irrigated area at global scale have been developed such as the USGS Global Land Cover Map using 1km monthly composite of NDVI obtained from Advanced Very High Ratio Radiometer(AVHRR). The methodology was divided into two goals 1) Classifying crop types using spectral similarity along the n-dimensional space vectors and 2) Decision tree model formulated using the Vegetation condition index (VCI). A LULC map derived from the Indian Remote Sensing (IRS) available at 56m resolution was used to map agricultural areas suitable for surface irrigation. Other sources for training and validation data were achieved from the crop calendar, Agroecological zones, Land use and land cover, DES-irrigation, IWIM-Irrigation data and Landsat -7 ETM+. NDVI was used as an indicator for irrigated data. To seperate irrigated and non-irrigated data statistical resolution of 250m and a 16 data temporal composite period was used. Another index called as Leaf Area Index (LAI) also has a positive correlation with NDVI also NDVI saturates when the LAI reaches a level 4 or more which can be further identified as non-crop vegetation. For the decision tree model, the NDVI threshold was applied for individual class clusters. The NDVI threshold was achieved by transforming original NDVI into the VCI. Normalized VCI can be used to identify NDVI differences between irrigated and non-irrigated areas. The classification method was calibrated separately for each region considering regions ecological potential and short-term weather fluctuations. The classified maps were evaluated using the agricultural statistics data from ground survey and compared to previously developed irrigation maps. High resolution irrigated area maps show satisfactory accuracy (R square = 0.95).

2.6.2 Machine Learning Approaches

In a method [44] to examine the potential of using Landsat time-series NDVI data is proposed to differentiate different crop types. MODIS data was used where the dataset was generated from NDVI layers for each image. Data gaps occurring due to scan line errors were filled using a multi-scale segmentation approach to fill missing NDVI data. This study was carried out in PHX MMA in Central Arizona. The stratified random approach and intelligent selection approach was used for the selection of training data for creating a model because the amount of data for various crop types were not huge. A total of nine crops was categorized out of which six single crops and three double crops were available in the dataset. One of the major challenges in the research was that the phenological curves for same crops could vary from one farm to another. The LIBSVM library was used for Support Vector Machine where C-Support vector classification. The 10-fold cross validation approach resulted in the best parameter selection where the accuracy on validation data achieved was 90%. Wheat and Barley were the two crops which were highly misclassified due to the spectral similarity.

2.6.3 Convolution Neural Network Approach

More recent approaches utilise machine learning techniques which help identify and differentiate between irrigated and non-irrigated land. In a study [17] carried out in the state of Montana over the years –2019. The main aim of the study was to identify difference between irrigated, unirrigated and uncultivated land. Landsat imagery was used for the research. Unlike most previous approaches where Random Forest is used with remote sensing imagery this research used an ensemble of convolution networks which work on Landsat Surface Reflectance which measures the fraction of incoming solar radiation reflected from earth's surface to the Landsat sensor. A modified U-Net [37] approach where the model was divided into two parts: a contracting path to extract low level features; and an expanding path which incorporated low level features from contraction path to produce the high resolution predictions. The imagery used in the research consisted of clouds, snow and shadow so that the model would be forced to learn a robust representation of irrigated land. F1-scores were used as an evaluation metric where the U-Net model had an F1-score of 0.86 in identifying irrigated land and 0.97 in identifying non-irrigated land, which outperforms the other methods in the study. Although the accuracy was better than that of the other, one problem of false positive still occurred where the model would misclassify a wetland as an irrigated land.

2.7 General Challenges

One of the primary challenges when it comes to the application of machine learning and Artificial Intelligence to the field of agriculture is that no two environments will be exactly alike. This makes the testing, validation, and successful rollout of applied technologies much more laborious than in most other industries. The weather, soil type, land type is also different depending on the geography. The other major challenges related to the identification of irrigated land are dealing with false-positive areas such as short rotating crops, seasonal variations in irrigation use and differences across different climate zones.The challenge for seasonal variations can be overcome by using the 13-band satellite imagery and Sentinel-1 which could help reveal variations in irrigated land throughout time, this can also help if there are cloud disturbances.

Chapter 3

The Dataset and Baseline Methods

This chapter focuses on the details of the datasets used in the research to evaluate the performance of the baseline models. Along with this the chapter focuses on the study area, the balance accuracy measure used for the experiments and finally the description of the baseline models.

As mentioned in the Section 2.3, since random forest and multi-layer perceptrons are stochastic methods, it is advised that these methods need to be run thirty to fifty times and an average accuracy needs to be calculated in order to have concrete conclusions and rule out the randomness in the results. Due to a few hurdles such as delay in access to high power computation machines, these stochastic tests were done five to eight times and an average result has been reported. It was noticed that the experiments when executed for five to eight times each time produced results close to each other, there was not a huge difference between accuracies.

3.1 The Dataset

The study area is the region of Canterbury, New Zealand. The Canterbury region as mentioned has the highest amount of irrigation land in use . Canterbury has a total of 546,205 hectares of irrigated land accounting for 64.05% of the total irrigated land area in the region of New Zealand [18]. The Canterbury region has an extreme contrast between irrigated and non-irrigated land and this contrast is the highest in the region as shown in Figure 3.1.

The dataset used in the project is based on imagery collected from Sentinel-2 satellite. The sentinel imagery has 13 bands out of which five bands blue (B02), red (B03), green (B04), near infrared red (B08) and short wave infrared (B12) described in table 2.1 are used and two important statistics such as the Normalized Difference Vegetation Index (NDVI) and Normalized Difference Water Index (NDWI) which are based on a set of calculation on the five bands mentioned. Before we dive into understanding the data collection process and data statistics it is necessary to understand the normalized difference vegetation index and normalized difference water index.

3.1.1 Normalized Difference Vegetation Index

The normalized difference vegetation index is often used to monitor drought and forecast agricultural production. NDVI is preferable for vegetation monitoring since it compensates for changes in lighting conditions, exposure and other factors [8].

NDVI is calculated using the near infrared red band (B08) and the red band (B03). The

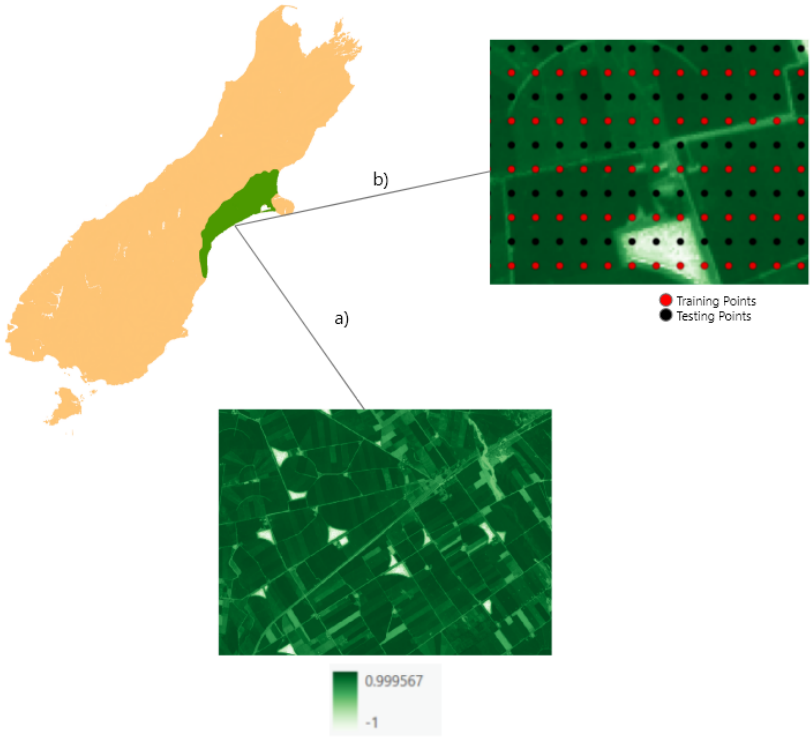

Figure 3.1: Study Area and data used in analysis. Subfigure a) Normalized Difference Vegetation Index. Subfigure b) Training and Testing Points

formula for NDVI is:

$$NDVI = \frac{B08 - B03}{B08 + B03} \tag{3.1}$$

The formula shows that the density of vegetation at a certain point of the image is equal to the difference in the intensities of reflected light in the red and infrared range divided by the sum of these intensities. This index defines values from -1.0 to 1.0, basically representing greens, where negative values are mainly formed from clouds, water and snow, and values close to zero are primarily formed from rocks and bare soil. Very small values (0.1 or less) of the NDVI function correspond to empty areas of rocks, sand or snow. Moderate values (from 0.2 to 0.3) represent shrubs and meadows, while large values (from 0.6 to 0.8) indicate temperate and tropical forests. Crop Monitoring successfully utilizes this scale to show farmers which parts of their fields have dense, moderate, or sparse vegetation at any given moment. An example can be seen in Figure 3.2.

3.1.2 Normalized Difference Water Index

Droughts can cause severe stress on vegetation on Earth. It is necessary to identify these areas in time or this can severely affect the crops. NDWI index helps control irrigation in respect to the project it is helpful to understand wetness in the land which can support learning to understand if a piece of land is irrigated or not [9].

NDWI is calculated using the near infrared red band (B08) and the short wave infrared (B12). NDWI was introduce in the 1996. This is calculated as:

$$NDWI = \frac{B08 - B012}{B08 + B012} \tag{3.2}$$

Figure 3.2: Comparison between true image, NDVI image NDWI image (Left to Right).

The formula shows that instead of using the red range, the reflection intensity in which is determined by the presence of chlorophyll, a short-wave near-infrared (SWIR) is used in which high absorption of light by water occurs. A wider range of 1500–1750 nm is possible. The use of the same near infrared (NIR) as in the case of NDVI is due to the fact that water does not absorb this part of the electromagnetic spectrum, thus the index is resistant to atmospheric effects, distinguishing it from NDVI.

The NDWI product is dimensionless and varies from -1 to $+1$, depending on the content, as well as the type of vegetation and cover. The high NDWI values correspond to high plant water content and coating of high plant fraction. Low NDWI values correspond to low vegetation content and cover with low vegetation. This can be seen in Figure 3.2.

3.1.3 Data Collection and Train-Test Sets

The Sentinel-2 data can be collected from various sources such as the sentinels scientific data hub on the Copernicus website, the Sentinel-hub website and this data is also hosted on the Amazon Web Services storage. The sentinel-data for this project was downloaded using the AWS s3 storage and python script.

A part of the Canterbury region was used in this study. This is usually referred to as a scene in Sentinel-2. The scene utilized is called "NN", this is depicted in Figure 3.3 where CM,DM,EM are scenes from an user product for different regions.

The data is collected for all four seasons: summer (December–February), autumn (March–May), winter (June–August) and spring (September–November). Five images per seasons were collected totalling in twenty images per year. The years targeted were 2018,2019 and 2020, a total of sixty images were collected. The five images included the blue (B02), red (B03), green (B04) and the calculated NDVI and NDWI bands. Each image was made up of 10980×10980 pixels. The dataset has twenty features in total which consists of pixel value at different locations from one of the five sentinel band images and a class label such as "Irrigated" and "Non Irrigated". The labels were collected from polygon shape files provided by Ministry for the Enviorment New Zealand, which are open-sourced and available from the MFE website. The survey was carried out for the years 2017 and a recent survey was carried out in 2020. The 2017 polygon shape file consisted of a total of 13491 polygons for the 2017 survey and the 2020 survey had a total of 16494 polygons.The data for the years 2018 and 2019 were labeled based on the 2017 polygons and the data for the year 2020 was labeled based on the 2020 polygon. There is usually an assumption that land used for irrigation does not change for approximately five years, this is clearly visible from the 2017 and 2020 polygon shape files where all the polygons from 2017 still exist in the 2020 polygon shape file except a addition of a few more polygons can be seen in the 2020 survey. The Figure 3.4 shows that green polygons are common to 2017 and 2020 and the purple polygons are from 2020.

The counts for dataset such as number of irrigated and non-irrigated land in each year

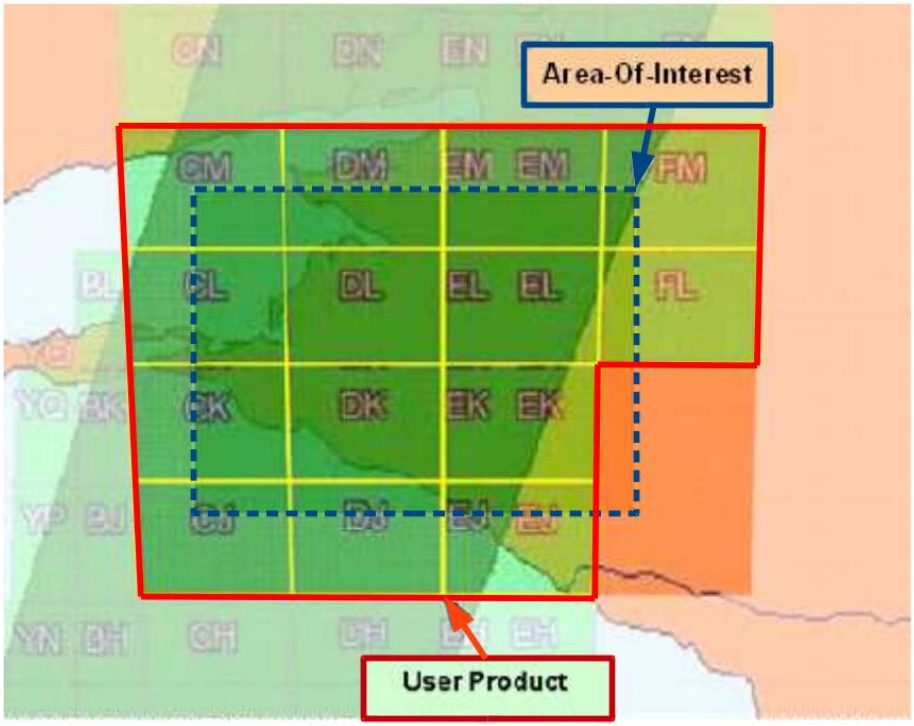

Figure 3.3: Relation between Sentinel-2 data concepts

Table 3.1: Dataset Count

	Training set			Test set			
	Irrigated	Non-irrigated	Total	Irrigated	Non-irrigated	Total	Overall
2018	29.17%	70.82%	581581	29.11%	70.89%	582660	1164241
2019	29.17%	70.82%	581581	29.11%	70.89%	582660	1164241
2020	31.06%	68.93%	581581	31.00%	69.00%	582660	1164241

in training and test set and the total number of counts is mentioned in Table 3.1.

3.2 Balanced Accuracy

The accuracy measure used through out the project is balanced accuracy. Balanced accuracy is good to use in situations where the test set has an imbalance. In our case as shown in Table 3.1 classes are highly imbalanced with non-irrigated classes are much higher than irrigated classes. Balanced accuracy is measured on the basis of sensitivity (true positive rate) which answers "how many positive cases were identified by the model" and specificity (true negative rate) which answers same question but for the negative cases depicted in Figure 3.6. Consider a confusion matrix as shown in Figure 3.5.

Balanced accuracy is the arithmetic mean of the sensitivity and specificity:

$$Balanced\ Accuracy = \frac{Sensitivity + Specificity}{2}. \tag{3.3}$$

If classes are imbalanced and we use accuracy as measure the chances of achieving an accuracy if only negative examples were predicted correctly could be higher than predicting both positive and negative examples. Consider an example shown in Figure 3.7.

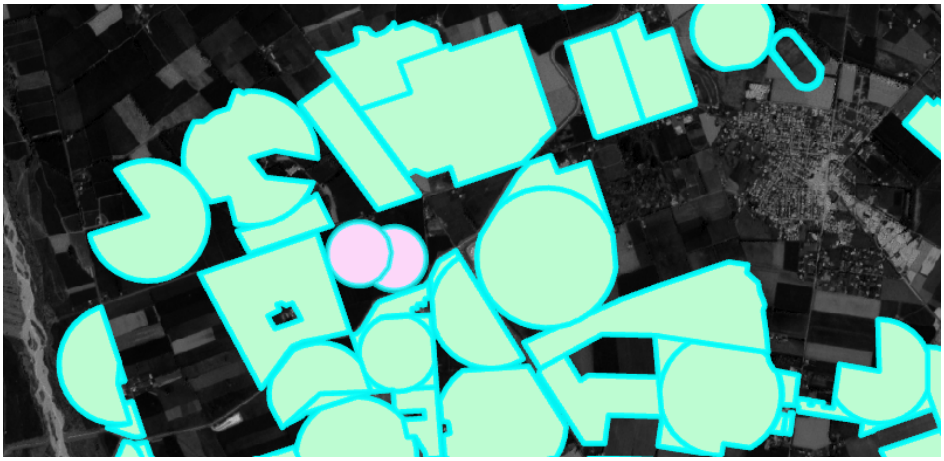

Figure 3.4: Irrigated polygon 2017 (light-green) and 2020 (purple).

	Actual positive (1)	Actual negative (0)
Predicted positive (1)	TP	FP
Predicted negative (0)	FN	TN

Figure 3.5: Confusion Matrix.

For the example if accuracy is calculated for the proportion of correct predictions is

$$\frac{5+10000}{5+50+10+10000} = 99.4\%$$

and if accuracy is calculated only for the negative predictions the accuracy is

$$\frac{0+10050}{0+0+15+10050} = 99.9\%$$

.

Balanced accuracy attempts to account for the imbalance in class. The computation for balanced accuracy for the example would be as follows:

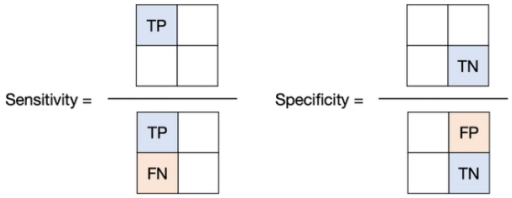

Figure 3.6: Sensitivity and Specificity.

	Actual positive (1)	Actual negative (0)
Predicted positive (1)	5	50
Predicted negative (0)	10	10000

Figure 3.7: Example for classifier with imbalanced classes.

$$
\begin{aligned}
Sensitivity &= \frac{5}{5+10} \\
&= 33.3\%, \\
specificity &= \frac{10000}{50+10000} \\
&= 99.5\%, \\
BalancedAccuracy &= \frac{Sensitivity + specificity}{2}, \\
&= \frac{33.3\% + 99.5\%}{2} \\
&= 66.4\%
\end{aligned}
$$

This shows that the classifier when measured using balance accuracy is better at giving an understanding of the models performance even with an imbalance class issue.

3.3 Baseline Models

This section discusses the baseline approaches used and the performance in terms of balanced accuracy and time taken by the model for training. The dataset used for the baseline model is the year 2018 where the entire dataset is split into 50% training and 50% testing. All methods used in the project are built using the scikit-learn library.

The models were trained on a machine with a i5-9th generation CPU, 8GB of Ram and an Nvidia GTX 1650.

3.3.1 Random Forest

A random forest approach was used for building a baseline model with default parameters from the ensemble module of the scikit-learn library. A list of parameters, values and definition of these parameters used are shown in table 3.2.

Table 3.2: This table shows random forest parameters used

Parameters	Description	value
n_estimators	The number of trees in the forest	100
criterion	Function to measure quality of a split	gini
max_depth	The maximum depth of the tree	None
min_samples_split	The minimum number of samples required to split an internal node	2
min_samples_leaf	The minimum number of samples required to be at a leaf node	1
max_features	The number of features considered when looking for the best split	auto
max_leaf_nodes	Grow trees with max_leaf_nodes in best first fashion	None
bootstrap	Whether bootstrap samples are used when building trees	True

The time taken for training a random forest sample on the dataset is 11 minutes and 24 seconds. The balance accuracy is 85.96%.

3.3.2 K-Nearest Neighbors

The second baseline approach was the k-nearest neighbor with default parameters from the neighbors module of the scikit-library. A list of parameters, values and definition of these parameters used are shown in Table 3.3.

Table 3.3: This table shows k-nearest neighbor parameters used

Parameters	Description	value
n_neighbors	The number of neighbors used	5
weights	Weight function used in prediction	uniform
algorithm	The algorithm used to compute the nearest neighbor	auto
n_jobs	The number of parallel jobs to run for neighbor search	None

The time taken for training a k-nearest neighbor model on the dataset is 2 hours and 21 minutes. The balance accuracy is 85.33%.

3.3.3 Multi-Layer Perceptrons

The final baseline approach was the multi-layer perceptron with random parameters from the neural network module of the scikit library. A list of parameters, values and definition of these parameters used are shown in Table 3.4.

Table 3.4: This table shows multi-layer perceptron parameters used

Parameters	Description	value
hidden_layer_size	The hidden layer represent layer in between the input and output layer	512,256,128,64,32
activation	Activation function for the hidden layer	relu
solver	The solver for weight optimization	adam
batch_size	The mini batch size for stochastic optimizer	auto
learning rate	The initial learning rate	0.001
early_stopping	Whether to use early stopping to terminate training when validation score is not improving	True

The time taken for training a multi-layer perceptron model on the dataset is 41 minutes. The balance accuracy is 86.42%.

3.3.4 Baseline Model Result Discussions

Table 3.5: Baseline models balance accuracy's and time to run model

Model	Balance Accuracy	Run Time
Random Forest	85.96%	11 Minutes 24 Seconds
K-Nearest Neighbor	85.33%	2 Hours and 23 Minutes
Multi-Layer Perceptron	86.42%	41 Minutes

The highest accuracy as can be seen in Table 3.5 is achieved from using multi-layer perceptrons. The accuracy's between random forest and k-nearest neighbors is almost similar. In terms of time taken random forest is the fastest amongst the other baseline methods. The longest time taken is the k-nearest neighbor because instead of learning a discrimitive function from the training data it memorizes the training data. Each time we want to make a prediction, KNN is searching for the nearest neighbor(s) in the entire training set and hence it is time and computationally expensive.

Chapter 4

Experimental Design, Results and Discussion

The focus of this chapter is to discuss in detail the different experiments carried out and discus the results of these various experiments. The chapter first discusses feature selection using techniques such as principal component analysis and genetic algorithm. Post feature selection, the chapter discusses checking and understanding the maximum number of samples required by the model. Finally, the chapter closes with an overall results discussions from this study.

4.1 Dimensionality Reduction Experiments

As mentioned in Section 2.4, dimensionality reduction is usually done for various reasons. In this project, there are twenty one features in total which is not huge and can be all used for training the model. Dimensionality reduction is carried out to rule out the possibility of the consisting of noise or random fluctuations in the training data which affects the performance of the model negatively. When these noise or random fluctuations do not exist in the testing set make the test accuracy largely lower than the train accuracy this is known as overfitting.

Dimensionality reduction were carried out using principal component analysis explained in Section 2.4.1 and genetic algorithm for feature selection explained in Section 2.4.2.

4.1.1 Feature Selection using Principal Component Analysis

Principal component analysis was performed using the scikit-learn API. Every parameter except the number of components (n_components) were set to default. The Table 4.1 shows a description of the parameters used and their value.

Table 4.1: This table shows principal component analysis parameters used

Parameters	Description	value
n_components	Number of most important dimensions or components to be used	16
copy	If set to false data passed to the fit function are overwritten	True
svd_solver	If auto the solver is selected by a default policy	auto
iterated_power	Number of iterations for the power method computed when svd_solver = 'randomized'	auto
learning_rate	The initial learning rate	0.001
random_state	Used to get reproducible results across multiple functions	None

The choice of n_components is based on an approach to evaluate the same model with different number of input features and choose number of features that result in best average

performance. The PCA was evaluated twenty times using the random forest classifier to determine the best choice for the number of dimensions. Post running the PCA method twenty times and averaging the results returned choice of n_components as sixteen. This can be seen in the box plot in Figure 4.1.

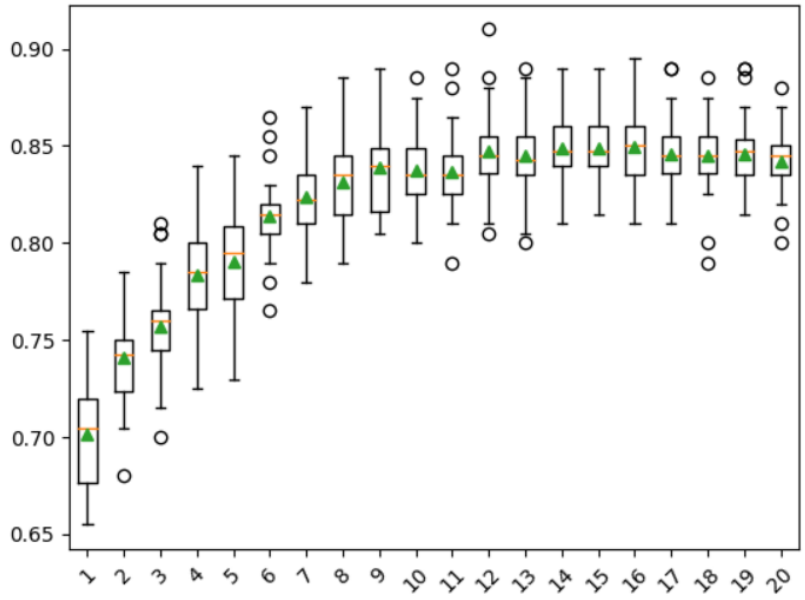

Figure 4.1: Plot for optimal selection of n_components.

The balance accuracy calculated for this methodology is an average of the balance accuracies over five runs for the random forest and the multi-layer perceptrons due to the stochastic nature of the algorithms. The experiments were carried out on the same baseline method dataset of 2018 with a train test split of 50/50 to check the performance.

Table 4.2 shows the results of the principal component analysis performed on the random forest algorithm, k-nearest neighbor and the multi-layer perceptron.

Table 4.2: Principal component analysis results for random forest, k-nearest neighbors and multi-layer perceptrons

Model	Baseline Accuracy	Balanced Accuracy	Run Time
Random Forest	85.96%	85.78%	14 Minutes 01 Seconds
K-Nearest Neighbor	85.33%	85.21%	1 Hour and 96 Minutes
Multi-Layer Perceptron	86.42%	86.41%	1 Hour

It is visible that the accuracies when compared to the baseline model in Table 3.5 the changes are minimal and also the time for each model is higher for random forest, multi-layer perceptron and less for the k-nearest neighbor than the baseline model. With these results it is evident that there is no need for dimensionality reduction and that the twenty features are needed for training the models, since these do not improve the performance time neither the accuracies.

4.1.2 Feature Selection using Genetic Algorithm

Principal component analysis aimed at reduction in dimension by selecting components that best explained the features; however the genetic algorithm approach aims at selecting features. Selecting features helps reduce the number of features by selecting from existing feature set rather than creating new features. The accuracy of the predictive model is considered as the fitness of the solution, where the accuracy of the model is obtained using logistic regression and random forest.Technically, the choice of learning algorithm can be any algorithm but logistic regression is usually faster to train.

The code for feature selection using genetic algorithm is based on the steps mentioned in Figure 2.8. The main call to the genetic algorithm code takes in three positional arguments which are the model, the features and the class labels. The parameters selected for the logistic regression approach are describe in Table 4.3.

Table 4.3: This table shows parameters used for the logistic regression based learner for feature selection using genetic algorithm

Parameters	Description	value
solver	Algorithm to use in the optimization problem	lbfgs
max_iter	Maximum iterations to be taken for solvers to converge	1000
random_state	To ensure reproducible results	7

Since the logistic regression is also a stochastic method these tests were run five to ten times to check the number of features returned changed or not. The parameters for the genetic algorithm such as the number of generation was set to five and the size of population was set to ten and a cross validation split of five was selected after trying a few combinations these parameters returned best results keeping in mind the time taken to select best parameters.

These tests were performed twice with different random_seed returned fifteen features each with different accuracies of 82.42% and 82.23%. The selected features are shown in Figure 4.2.

Accuracy	AU_B02	AU_B03	AU_B04	AU_NDVI	AU_NDWI	SP_B02	SP_B03	SP_B04	SP_NDVI	SP_NDWI	SU_B02	SU_B03	SU_B04	SU_NDVI	SU_NDWI	WI_B02	WI_B03	WI_B04	WI_NDVI	WI_NDWI
82.42%	✔		✔	✔	✔	✔	✔	✔		✔	✔	✔	✔				✔	✔	✔	✔
82.23%	✔	✔	✔	✔	✔		✔	✔	✔		✔	✔		✔		✔	✔	✔		✔

Figure 4.2: Genetic algorithm selected features.

Since the accuracy achieved for both trials is very close and the number of features selected is also same. To test the selected features we will choose the features with the 82.42% accuracy. If there is a case where both the accuracies are similar then the features best representing the problem background are selected.

The Table 4.4 shows the results of the genetic algorithm selected feature analysis performed on the random forest algorithm, k-nearest neighbor and the multi-layer perceptron.

Table 4.4: Genetic algorithm feature selection results for random forest, k-nearest neighbors and multi-layer perceptrons

Model	Baseline Accuracy	Balanced Accuracy	Run Time
Random Forest	85.96%	85.44%	9 Minutes 63 Seconds
K-Nearest Neighbor	85.33%	84.81%	1 Hour and 30 Minutes
Multi-Layer Perceptron	86.42%	86.79%	1 Hour and 13 minutes

It is visible that the accuracies when compared to the baseline model in Table 3.5 and Table 4.2 the changes are minimal and also the time for each model is h higher for k-nearest

neighbor, multi-layer perceptron and less for the random forest than the principal component analysis method. With these results it is evident that all the features that are present in the dataset are necessary as it does not improve performance of the model in terms of time or balance accuracy.

4.2 HyperParameter Tuning Experiments

As mentioned in Section 2.5 there are multiple hyperparamters that can be selected for each of the model. In the baseline model of random forest, k-nearest neighbors and multi-layer perceptrons we worked with default and random parameters to understand the performance of the model on the dataset, these hyperparameters are listed in Section 3.3.

Selection of hyperparameters is important to ensure that we can achieve better accuracy or assure that the model does not end up overfitting. For this experiment we used scikit-learn library and the Randomized GridSearchCV method for hyperparameter tuning because it is much faster as compared to the GridSearchCV method.

Hyperparameter tuning was done for the models based on the same dataset used in the baseline model. Since the total dataset size is 1,164,241 samples we use 500,000 samples to ensure that the time taken to identify these hyperparameters is faster and the accuracy of the hyperparameters selected is acceptable.

4.2.0.1 HyperParameter Tuning Random Forest Model

There are various parameters that can be set in the random forest method of the scikit-learn library. According to the documentation for the random forest method in the scikit learn library there are approximately nineteen different hyperparamters to tune [10]. Out of these nineteen in this experiment selected six hyperparameters to tune. These six hyperparameters are listed in Table 4.5.

Table 4.5: Random Forest HyperParameters used for tuning

HyperParameters	Description
n_estimators	The number of trees in the forest.
max_features	The number of features to consider when looking for the best split.
max_depth	The maximum depth of the tree.
min_samples_split	The minimum number of samples required to split an internal node.
min_sample_leaf	The minimum number of samples required to be at a leaf node.
bootstrap	If false, the whole dataset is used to build each tree.

The time taken for identifying the hyperparameters 46 minutes and 8 seconds. The accuracy achieved for the hyperprameter selection is 82.31%. The value of the hyperparameters tuned are shown in Table 4.6

Table 4.6: Random Forest HyperParameter Values

HyperParameters	Values
n_estimators	1000
max_features	sqrt
max_depth	110
min_samples_split	2
min_sample_leaf	1
bootstrap	True

4.2.0.2 HyperParameter Tuning K-Nearest Neighbor

There are various parameters that can be set in the k-nearest neighbor method of the scikit-learn library. According to the documentation for the KNN method in the scikit-learn library there are approximately eight different hyperparamters to tune [6]. Out of these eight in this experiment selected three hyperparameters to tune. These three hyperparameters are listed in Table 4.7.

Table 4.7: K-Nearest Neighbor hyperparameters used for tuning

HyperParameters	Description
n_neighbors	The number of neighbors to use
weights	Weight function to be used in prediction
algorithm	Algorithm used to compute the nearest neighbors

The time taken for identifying the hyperparameters 37 minutes and 41 seconds. The accuracy achieved for the hyperprameter selection is 80.40%. The value of the hyperparameters tuned are shown in Table 4.8.

Table 4.8: K-Nearest neighbor hyperparameter values

HyperParameters	Values
weights	uniform
n_neighbor	7
algorithm	brute

4.2.0.3 HyperParameter Tuning Multi-Layer Perceptron

Just like random forest and k-nearest neighbors, the multi-layer perceptron has various parameters that can be set in the scikit-learn library. According to the documentation for the MLP method in the scikit-learn library there are approximately twenty-three different hyperparamters to tune [7]. Out of these twenty-three in this experiment selected five hyperparameters to tune. These five hyperparameters are listed in Table 4.9.

Table 4.9: K-Nearest Neighbor HyperParameters used for tuning

HyperParameters	Description
hidden_layer_size	The number of hidden layers in the network.
activation	Activation function for the hidden layer.
solver	Solver for weight optimization.
alpha	L2 penalty parameter.
learning_rate	Learning rate scheduled for weight updates.

The time taken for identifying the hyperparameters 28 minutes and 26 seconds. The accuracy achieved for the hyperprameter selection is 83.56%. The value of the hyperparameters tuned are shown in Table 4.10.

4.3 Sub-Sampling Experiments

As seen in Table 3.1 the number of samples in each data set is approximately close to one million. To train and test a model with a million samples is time consuming and resource intensive. Since the experiments are run on a single machine with specifications mentioned in the Section 3.3. The experiment in this section aims at understanding how much data

Table 4.10: Multi-Layer Perceptron HyperParameter Values

HyperParameters	Values
hidden_layer_size	512,256,128,64,32
activation	relu
solver	sgd
alpha	0.0001
learning_rate	constant

is actually needed to perform identification of irrigated land from sentinel imagery. We have already seen in the results from the previous experiments that all twenty features are important. The question this section tries to answer is whether it is necessary to have approximately one million samples to achieve an optimal accuracy or we could still achieve an optimal accuracy with a small number of training data to train a model?

All experiments conducted up-to now are performed on the 2018 dataset alone. In this section of experiment the training data is combined of 2018 and 2019 and is tested on the 2020 dataset. The reason for selecting only 2018 dataset as training and test set in previous experiments was to keep tests such as hyperparameter tuning unbiased. If performed on the entire dataset the model would not be generalised. From previous experiments we have understood that feature selection does not improve the model and understood the optimal hyperparameter to be used. Here the class value, i.e, irrigated and non-irrigated for the training set are based on the 2017 polygon data and the class values for the test set is based on the 2020 polygon data provided by Ministry for the Enviorment.

The sub-sampling has been done on thousand, five thousand, hundred thousand, five hundred thousand samples selected randomly and the entire dataset which is 2,328,482 samples.

The hyperparameter settings for the experiments mentioned are based on the hyperparameters achieved from Section 4.2. The following Tables 4.11 to 4.13 provide results on the time taken for each sample size and the accuracy achieved.

Table 4.11: Random forest sub-sample results

Sample Size	Accuracy	Time
1000	79.16%	18.79 seconds
5000	81.26%	24.43 seconds
100000	82.86%	1 minute 77 seconds
500000	83.23%	8 minutes 73 seconds
All (2328482)	83.19%	1 hour 8 minutes

Table 4.12: K-Nearest neighbor sub-sample results

Sample Size	Accuracy	Time
1000	77.67%	43.41 seconds
5000	79.37%	1 minute 25 seconds
100000	81.38%	6 minutes 64 seconds
500000	81.85%	18 minutes 22 seconds
All (2328482)	81.72%	37 minutes 06 seconds

When compared to the results of Table 3.5 it is can be observed that the accuracy has dropped this could be because of a few reasons such as combining the 2018 and 2019 dataset for training has more samples that are used for training than the baseline model, this helps better generalise the model.

This section proves that 500,000 samples is an good enough number of samples for either

Table 4.13: Multi-layer perceptron sub-sample results

Sample Size	Accuracy	Time
1000	82.10%	51.09 seconds
5000	80.02%	49.96 seconds
100000	82.43%	4 minutes 35 seconds
500000	82.87%	39 minutes 25 seconds
All (2328482)	81.28%	9 hours 27 minutes

of the three models is sufficient, it achieves a good accuracy and the time taken to train the model is much faster than training on the full set. This is true for each of the three methods and provides a good trade-off between training time and accuracy.

The Figure 4.3 is the region in Canterbury on which the experiments were carried out. The white regions in this image is ocean/water streams. Since the research is focused on identifying irrigated pixels from non-irrigated pixels, water stream or oceans are not detected by this model.

The best results is achieved from the random forest model by using the sub-sampling method of 500000 samples. This can be seen in Figure 4.4, where the white color indicates correctly identified non-irrigated pixels, green color indicates correctly identified irrigated pixels and the red color is the misclassified pixels by the model.

Figure 4.3: Canterbury region used for the model.

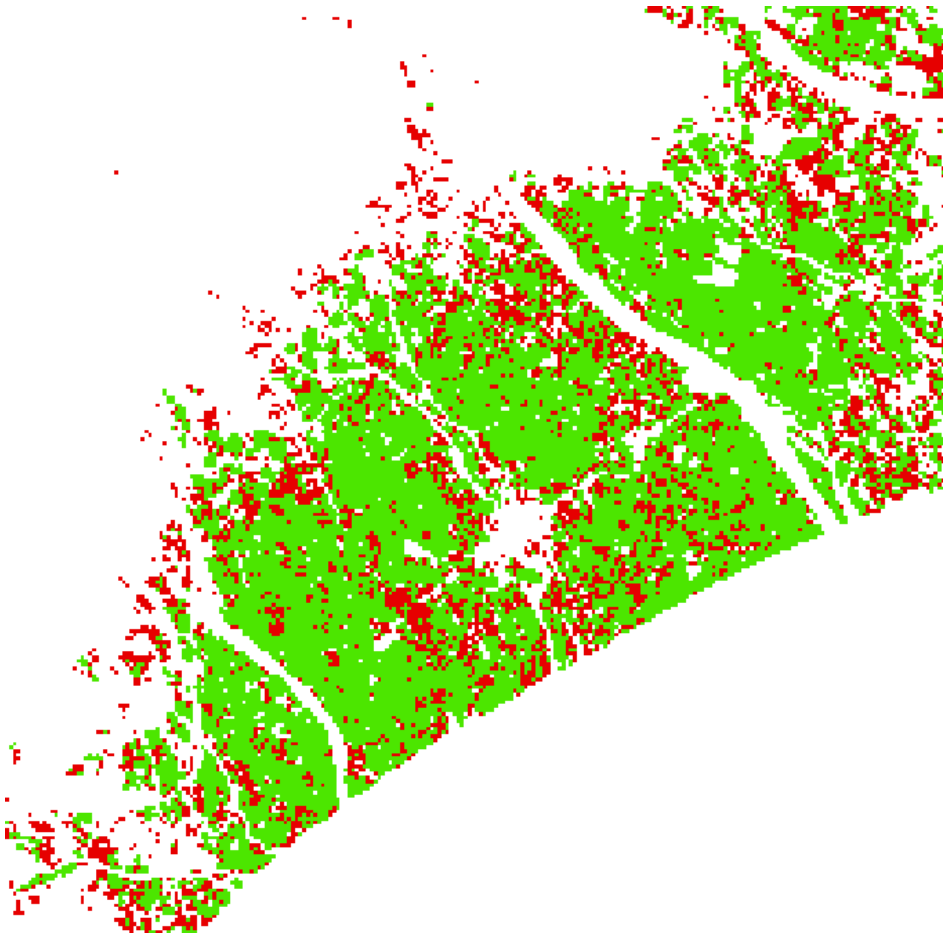

Figure 4.4: Random forest sub-sampling output.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

This chapter summarises the obtained reuslts that are discussed in the baseline model mentioned in Section 3.3 and the experimental design discussed in Chapter 4. This research has successfully proved that the pixel classification of irrigated land from non-irrigated land can be done adopting a machine learning approach over a manual approach which is time consuming and expensive. The research has proposed the use of Sentinel-2 imagery and features generated from this imagery, where all the twenty features are important and needed for the model to achieve good performance. Along with proving that an automatic approach can be used for this task the results obtained form the experiments in Section 4.3 show that only five hundred thousand samples are enough for training a model. The performance of the automated method cannot be compared to the desktop-analysis method being used currently by the ministry for enviorment since the desktop analysis method is based on multiple sources such as previous land ownership documents, surveys [18].

The results in the experiments clearly show that adopting a machine learning technique can be utilized to identify irrigated land from non-irrigated land using traditional machine learning techniques. The automated techniques definitely overpowers the manual method because these results when done automatically provide results much faster than a desktop analysis method, saves cost, requires less human intervention to create the irrigated land results and most importantly can be performed as and when needed.

5.2 Future Work

The overall goals of this research has been achieved successfully. However, there is still scope to improve or build newer applications based on this research in the future. Few directions that can be looked into in the future are as follows:

- Due to the time constraint for the research, points used in training and testing were collected at hundred meter distance and a pixel classification was done to identify irrigated land. Each pixel value in the image can be collected and a better and more detailed identification can be performed.

- This research can be extended to other regions of New Zealand because no two regions would have the same land characteristics and hence this can help in understanding if the same model would work for different regions or new data would have to be added a new model would have to be trained.

- Newer features can be constructed using state of the art techniques such as auto-encoders and it can be used in the model training to understand if these features contribute towards increasing the accuracy of the model.
- Once land is identified as irrigated or not this land could also be measured and water distribution for irrigation based on other statistics could also be controlled.
- If a labeled image dataset for irrigated and non-irrigated land is collected then a possible option to solve this task could be use of convolution-neural networks as mentioned in [17] and segmentation using algorithms such as u-net [37].

Bibliography

[1] Classification example. https://simplicable.com/new/classification. Accessed: 2021-04-16.

[2] FAO food and agriculture organizations role in water. https://www.fao.org/water/en. Accessed: 2021-03-15.

[3] Feature selection methods. https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-v Accessed: 2021-06-04.

[4] Genetic algorithm for feature selection. https://www.neuraldesigner.com/blog/genetic_algorithms_for_feature_selection#InitializationOperator. Accessed: 2021-05-17.

[5] Genetic algorithm steps. https://www.neuraldesigner.com/blog/genetic_algorithms_for_feature_selection. Accessed: 2021-05-10.

[6] K-nearest neighbor documentation scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.KNeighborsClassifier.html. Accessed: 2021-04-28.

[7] Multi-layer perceptron documentation scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPClassifier.html. Accessed: 2021-04-28.

[8] Normalized difference vegetation index. https://eos.com/make-an-analysis/ndvi/. Accessed: 2021-04-16.

[9] Normalized difference vegetation index. https://eos.com/make-an-analysis/ndwi/, note = Accessed: 2021-04-16.

[10] Random forest documentation scikit-learn. https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html. Accessed: 2021-04-28.

[11] Randomized and grid searchcv. https://blog.usejournal.com/a-comparison-of-grid-search-and-randomized-search-using-scikit-learn-29823179bc85. Accessed: 2021-04-16.

[12] Spam classification example. https://medium.com/@naveeen.kumar.k/naive-bayes-spam-detection-7d087cc96d9d, note = Accessed: 2021-04-16.

[13] StatsNZ irrigation in new zealand. https://www.stats.govt.nz/indicators/irrigated-land. Accessed: 2021-04-28.

[14] ABUZAR, M., MCALLISTER, A., AND MORRIS, M. Classification of seasonal images for monitoring irrigated crops in a salinity-affected area of australia. *International Journal of Remote Sensing 22*, 5 (2001), 717–726.

[15] AMBIKA, A. K., WARDLOW, B., AND MISHRA, V. Remotely sensed high resolution irrigated area mapping in india for 2000 to 2015. *Scientific Data 3:160118* (2016).

[16] BREEIMAN, L. Random forest. *Machine Learning 45 5-32* (2001).

[17] COLLIGAN, AND IV, T. H. A deep learning approach to mapping irrigation: U-net IrrMapper. *University of Montana* (2020).

[18] DARK, A. National irrigated land spatial dataset. *Aqualinc Report RD21004* (2020).

[19] GHAHRAMANI, Z. *Unsupervised Learning*. Berlin, Heidelberg, 2004, pp. 72–112.

[20] HADY, M. F. A., AND SCHWENKER, F. *Semi-supervised Learning*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 215–239.

[21] HUANG, S., FAN, X., SUN, L., SHEN, Y., AND SUO, X. Research on classification method of maize seed defect based on machine vision. *Journal of Sensors 2019* (2019), 1–9.

[22] KAELBLING, L. P., LITTMAN, M. L., AND MOORE, A. W. An introduction to reinforcement learning. In *The Biology and Technology of Intelligent Autonomous Agents* (Berlin, Heidelberg, 1995), L. Steels, Ed., pp. 90–127.

[23] KORST, J., PRONK, V., BARBIERI, M., AND CONSOLI, S. *Introduction to Classification Algorithms and Their Performance Analysis Using Medical Examples*. Springer International Publishing, Cham, 2019, pp. 39–73.

[24] KOTSIANTIS, S. B. Supervised machine learning: A review of classification techniques. In *Proceedings of the 2007 Conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in EHealth, HCI, Information Retrieval and Pervasive Technologies* (2007), IOS Press, p. 3–24.

[25] MAGIDI, J., NHAMO, L., MPANDELI, S., AND MABHAUDHI, T. Application of the random forest classifier to map irrigated areas using google earth engine. *Remote Sensing 13* (02 2021).

[26] MORELLOS, A., PANTAZI1, X.-E., MOSHOU, D., ALEXANDRIDIS, T., WHETTON, R., TZIOTZIOS, G., WIEBESOHN, J., BILL, R., AND MOUAZEN, A. Machine learning based prediction of soil total nitrogen, organic carbon and moisture content by using VIS-NIR spectroscopy. *Biosystems Engineering 152* (2016), 104–116.

[27] MURTAGH, F. Multilayer perceptrons for classification and regression. *Neurocomputing 2*, 5 (1991), 183–197.

[28] MYLES, A., FEUDALE, R. N., LIU, Y., WOODY, N., AND BROWN, S. An introduction to decision tree modeling. *Journal of Chemomtrics 18* (2004), 275–285.

[29] NAHVI, B., HABIBI, J., MOHAMMADI, K., SHAMSHIRBAND, S., AND AL RAZGAN, O. S. Using self-adaptive evolutionary algorithm to improve the performance of an extreme learning machine for estimating soil temperature. *Computers and Electronics in Agriculture 124* (2016), 150–160.

[30] ORESKI, S., AND ORESKI, G. Genetic algorithm-based heuristic for feature selection in credit risk assessment. *Expert Systems with Applications 41*, 4, Part 2 (2014), 2052–2064.

[31] PAL, M. Random forest classifier for remote sensing classification. *International Journal of Remote Sensing 26*, 1 (2005), 217–222.

[32] PEIJUN DU, ALIM SAMAT, B. W. S. L. Z. L. Random forest and rotation forest for fully polarized SAR image classification using polarimetric and spatial features. *ISPRS Journal of Photogrammetry and Remote Sensing Volume 105* (2015), 38–53.

[33] QI, S. L., KONDURIS, A., LITKE, D., AND DUPREE, J. Classification of irrigated land using satellite imagery, the high plains aquifer, nominal date 1992. *Water-Resources Investigations Report 02-4236* (2002).

[34] RAMCHOUN, H., IDRISSI, M. A. J., GHANOU, Y., AND ETTAOUIL, M. Multilayer perceptron: Architecture optimization and training with mixed activation functions. In *Proceedings of the 2nd International Conference on Big Data, Cloud and Applications* (2017), Association for Computing Machinery.

[35] R.G. SUMSION, M.S. BRADSHAW, K. H. L. P. S. P. Remote sensing tree classification with a multi-layer perceptron. *PeerJ* (2019).

[36] RODRIGUEZ-GALIANO, V., GHIMIRE, B., ROGAN, J., CHICA-OLMO, M., AND RIGOL-SANCHEZ, J. An assessment of the effectiveness of a random forest classifier for land-cover classification. *ISPRS Journal of Photogrammetry and Remote Sensing 67* (2012), 93–104.

[37] RONNEBERGER, O., FISCHER, P., AND BROX, T. U-net: Convolutional networks for biomedical image segmentation, 2015.

[38] SUCHITHRA, M., AND PAI, M. L. Improving the prediction accuracy of soil nutrient classification by optimizing extreme learning machine parameters. *Information Processing in Agriculture 7*, 1 (2020), 72–82.

[39] WU, W., AND DE PAUW, E.

[40] YANG, M., XU, D., CHEN, S., LI, H., AND SHI, Z. Evaluation of machine learning approaches to predict soil organic matter and pH using vis-NIR spectra. *Sensors 19*, 2 (2019), 1–14.

[41] YANG, P., ZHAO, Q., AND CAI, X. Machine learning based estimation of land productivity in the contiguous US using biophysical predictors. *Environmental Research Letters 15*, 7, 074013.

[42] YUE, S., L. P. . H. P. Svm classification:its contents and challenges. *Applied Math China University 18*, null (2003), 332–342.

[43] ZHANG, S., LI, X., ZONG, M., ZHU, X., AND WANG, R. Efficient knn classification with different numbers of nearest neighbors. *IEEE Transactions on Neural Networks and Learning Systems 29*, 5 (2018), 1774–1785.

[44] ZHENGA, B., W.MYINTA, S., S.THENKABAILB, P., AND M.AGGARWAL, R. A support vector machine to identify irrigated croptypes using time-series landsat ndvi data. *International Journal of Applied Earth Observation and Geo information. 32* (2015), 103–112.