Modelling Soil Compaction Based On Soil Moisture, Bulk Density And Other Soil Parameters

Ciarán Flanagan

Student ID: 18336646

A thesis submitted in part fulfilment of the degree of

BSc. (Hons.) in Computer Science with Data Science

Supervisor: Assoc Professor Rem Collier

UCD School of Computer Science
University College Dublin

April 30, 2022

Table of Contents

1 Project Description 4

1.1 Problem Statement 4

1.2 Background 5

1.3 Related Work 5

1.4 Data Sets 5

1.5 Resources Required 6

2 Background Work 7

2.1 What Is Soil Compaction? 7

2.2 Why Is Soil Compaction Problematic? 7

2.3 Prediction of soil compaction due to agricultural machines 8

2.4 Evaluation of soil compaction during sugarcane harvest 9

2.5 Tyre evolution and the reduction in soil compaction 9

2.6 SoilFlex: A model for prediction of soil stresses and soil compaction 9

2.7 soilphysics: An R package for calculating soil water availability 10

2.8 Micro-services For Agricultural Applications 10

2.9 Discussion 11

3 Project Management Plan 12

3.1 Semester 1 12

3.2 Semester 2 12

4 Data Considerations 14

4.1 Dataset 14

4.2 Analysis 19

5 Outline of Approach 23

5.1 Data Cleaning 23

5.2 Model Development 24

5.3 Model Accessibility 29

6 Summary and Conclusions 35
7 Acknowledgements 36

Abstract

The adverse effects of soil compaction can be extremely costly to farmers, not just in Ireland, but the world over. Countless hours and resources have been dedicated to combating the effects of soil compaction. There are also strong links between soil compaction and the level of soil moisture present in the soil. This project aims to assist in the understanding of soil compaction by modelling the relationship between soil compaction, soil moisture and other parameters and their effects on yield and emergence.

Over the course of 2020 and 2021 data was collected on UCD Lyon's farm to investigate and analyse soil compaction and its effects. This project intends to investigate the possibility of building accessible models that can be used to aid agricultural decision making. The primary aim of this project is not to analyse the data available, but instead to investigate the possibility of constructing machine learning models based on the data available. The intention is to make these models as accessible as possible so their use is not limited to those with a technical Computer Science or Data Science background.

The GitLab repository for this project can be found here. The repository includes all the code, data and microservices developed throughout the course of this project. The author of this report can be contacted at ciaran.flanagan1@ucdconnect.ie with any questions, queries or discussions.

Chapter 1: Project Description

This project aims to build on data gathered over the two last years on soil compaction at UCD Lyons Farm to create machine learning models for assessment of soil compaction and soil moisture in order to optimise the plant population for a winter wheat crop. The project aims to build the model in such a way as to be used for both assessments and in practical use cases.

This project is part of the larger CONSUS collaboration research partnership between University College Dublin (UCD) and Origin Enterprises PLC, which has been supported through the Science Foundation Ireland (SFI) Strategic Partnership Program.

This project is primarily a Computer Science and Data Science project, however, it has been guided and advised by Prof. Kevin McDonnell and his team from the School of Agriculture and Food Science in UCD. Their inputs and suggestions have influenced this project. I will refer to them as the Agricultural Experts when referencing their inputs, suggestions and/or ideas throughout this project.

1.1 Problem Statement

Soil Compaction has a massive effect on crop yields and overall crop productivity in agriculture. In 2016 there was an estimated product loss value of €713 million in France and €487 million in Germany [1]. By better understanding and forecasting soil compaction, we can actively prevent and reduce the impact it has on agriculture.

In order to achieve this, this project seeks to:

- Understand the factors that affect soil compaction and identify appropriate data sets needed for forecasting it.
- Assess soil compaction models and evaluate the accuracy of these models.
- Design a micro-services based framework for implementing the soil compaction models.
- Develop an easily accessible Graphical User Interface to enable the use of the tools for decision support.

It is anticipated that the tool will be evaluated by researchers in the UCD School of Agriculture and Food Science who are currently engaged in soil compaction research and by researchers in the UCD School of Computer Science.

1.2 Background

Soil Compaction is the physical consolidation of the soil by an applied force that destroys structure, reduces porosity, limits water and air infiltration, increases resistance to root penetration and often results in reduced crop yields.

For many crop species, there is a correlation between increased soil compaction and reduced crop yields. The soil structure is vital for crop growth. A properly structured soil allows room for crop roots to grow and water and nutrients to penetrate the soil. Soil compaction reduces and eliminates this structure by compressing the open pore spaces due to the added pressure on the soil.

Soil can become compacted in multiple ways, however, this project will only be looking at soil compaction caused by farm machinery. There is a number of reasons for the increased compaction caused by modern farm machinery, including the increased weight of modern machinery and the increase in tractor tyre size.

1.3 Related Work

The two most relevant papers used in the research of this project have been *SoilFlex: A model for prediction of soil stresses and soil compaction due to agricultural field traffic including asynthesis of analytical approaches* [2] and *soilphysics: An R package for calculating soil water availability to plants by different soil physical indices* [3].

The *SoilFlex* paper creates a two-dimensional model for the calculation of soil stresses and soil compaction due to agricultural field traffic. It is written as an Excel spreadsheet which makes it easy to use for research or by agricultural advisors but does not make it very accessible for practical everyday applications. It is also only available via an email request to the original authors.

The *soilphysics* paper aims to address the accessibility of the *SoilFlex* paper by creating an R package for the simulation of compaction by agricultural traffic. However this solution still requires some knowledge and experience with using R.

This project aims to create similar models to that of *soilphysics* using data collected on UCD's Lyons Farm and to also add a graphical user interface portion to further increase the models' accessibility.

1.4 Data Sets

Over the last two harvest seasons at UCD's Lyons Farm the following data sets have been collected and have been made available for use in this project. The datasets include the following attributes.

Harvest 1: Soil Moisture, Bulk Density, Cone Penetrometer, Plant Emergence, Combine Yield, Linear Metre Yield, Thousand Grain Weight, Grain Screenings, Grain Protein and Grain Moisture.

Harvest 2: Soil Moisture, Bulk Density, Cone Penetrometer, Plant Emergence, Combine Yield, Linear Metre Yield, Thousand Grain Weight, Grain Screenings, Grain Protein, Grain Moisture, Leaf

Area Index (LAI) and Chlorophyll Levels.

Soil Moisture
: The moisture content in the soil.

Bulk Density
: A measure of how densely compacted the soil is. Measured in $g\,cm^{-3}$.

Cone Penetrometer
: Method used to determine the geotechnical engineering properties of soils using the penetration depth of a cone.

Plant Emergence
: A measure of the plants' emergence.

Combine Yield
: Yield as measured by the combine.

Linear Metre Yield
: Yield per linear metre.

Thousand Grain Weight (TGW)
: The weight measure of a thousand grains. Measured in g.

Grain Screenings
: Involves passing grain over various screen sizes to get an estimate for the average size of the grains in the yield.

Grain Protein
: A measure of the protein level in the crop.

Grain Moisture
: A measure of the grain moisture in the crop.

Leaf Area Index (LAI)
: A measure of the leaf area cover per metre squared. Measured in m^2.

Chlorophyll Levels
: A measure of the chlorophyll levels in the crop.

1.5 Resources Required

None Required.

Chapter 2: Background Work

2.1 What Is Soil Compaction?

Soil compaction can be defined as "The densification and distortion of soil by which total and air-filled porosity are reduced, causing deterioration or loss of one or more soil functions" [4]. Compaction takes place when soils are subjected to stresses that exceed their strength. This can happen in a number of ways, including via farm machinery and animal traffic. This project will focus solely on the effects of farm machinery traffic and its resulting soil compaction.

There is a trend towards the use of larger and heavier machinery on farms. [5] This is primarily due to the desire to reduce the reliance on the labour force where salaries are generally high. [6] More frequent use of these machines, combined with their increased weight, is leading to soils becoming subjected to higher and higher stresses which they struggle to cope with.

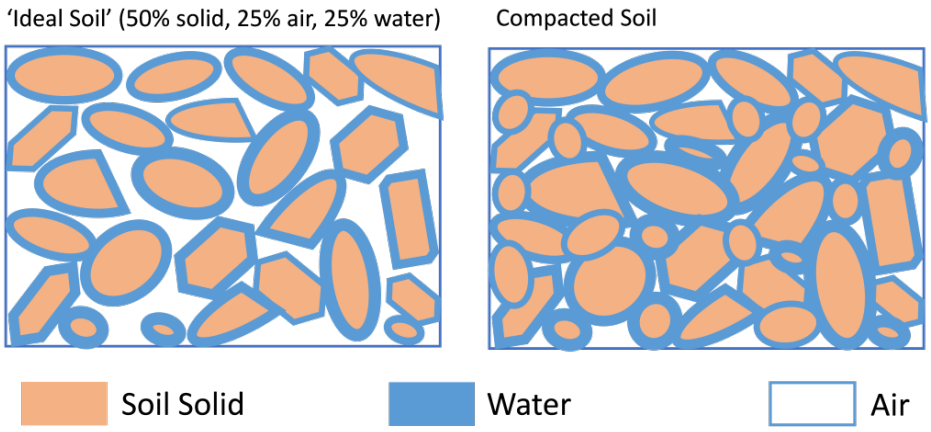

Figure 2.1: Soil compaction causes a reduction in available space for soil, air and water and it limits pathways for crop roots. [7]

There are two main forms of soil compaction, **topsoil** and **subsoil** compaction. Compaction of the topsoil layer (soil layer closest to the surface and the layer where most crop activity occurs) has a significant impact on crop yields. However, this compaction can be elevated somewhat by common tillage methods and natural processes. Compaction of the subsoil layer however is more persistent, accumulates over many seasons and is invisible on the surface.

In the later sections, I examine some of the effects of these changes.

2.2 Why Is Soil Compaction Problematic?

As I'll examine below soil compaction and crop yield are strongly linked. That is to say that an increase in soil compaction results in a decrease in crop yield [8]. Voorhees et al. calculated that in practice this results in losses of about 6% in small grain cereal production, which equates to a

product loss of €97 million in Denmark, €487 million in Germany and €713 million in France. [9]

Soil compaction can also have other undesirable effects on farmland. Compaction reduces the filter capacity of the soil, which in turn increases the risk of surface runoff and soil erosion. Soil compaction is also a long term issue that is likely to persist beyond the use of the land for farming purposes.

An example of the effects of soil compaction can easily be seen at the gateway to many fields. Due to the gateway receiving all traffic going into and out of the field, the area around the gateway often becomes very compacted. This often results in little to no growth around or near the gateway to fields.

2.3 Prediction of soil compaction due to agricultural machines

Modelling the predicted stress on soil and the resulting soil compaction is often beneficial. Wellingthon da Silva et al. [10] found that by modelling operations involved in the sugarcane harvest and crop rotation process they could predict which operations may potentially result in soil compaction and to what severity the soil compaction would occur.

They found that for seven different operations (harrowing, subsoiling, peanut planting, peanut harvesting, tillage, sugarcane planting and sugarcane harvesting), three of the operations were likely to result in severe soil compaction. These three operations were peanut planting, peanut harvesting and sugarcane planting. They found that this was due to the stress volume placed by the equipment used for these operations on the soil. Due to soil compaction forming as a result of the applied stress exceeding the soil strength, the authors found that the operations that were likely to cause compaction were due to the equipment applying too much pressure to the available axles.

For example, the peanut harvester used was a single axle unit pulled behind a tractor unit. It was found that the tractor unit itself did not apply enough stress on the soil to cause severe soil compaction. However, the pressure applied by the harvesting unit itself did.

It was also found that at harvest, the fully loaded trailers used for transporting the sugarcane off the fields were large contributors to soil compaction. This was attributed to their high weight, road-going tyres (higher inflation pressures) and their tendency to traverse the same area of ground multiple times.

Along with this, it was also found that the soil management practices, what operations had been carried out on the soil before, could also play a role in the level of soil compaction. This is mainly due to how hard the soil has been left after an operation.

2.4 Evaluation of soil compaction during sugarcane harvest

A paper by N. Lozano et al. [11] investigated the resultant soil compaction after a sugarcane harvest in Brazil in 2010. During the sugarcane harvest farmers in Brazil commonly use haul-out trucks with high-pressure road-going tyres to transport the sugarcane from the field to the processing plant. N. Lozano et al. [11] tested the soil compaction after various configurations of the haul-out truck had been used. These included a single haul-out truck with a single front axle and dual rear axles, a haul-out truck towing a haul-out trailer and a tractor and haul-out trailer combination.

N. Lozano et al. [11] found that the tractor-trailer combination produced the least overall compaction, however, all configurations caused compaction of some sort. They found that as a soil became wetter, its ability to width stand compaction decreased. They found that the topsoil (0cm to 20cm deep) was at a high risk of compaction during harvest and that field traffic should be restricted when the soil moisture content is above 16%.

Overall N. Lozano et al. [11] also found that all forms of haul-out resulted in some level of compaction and that new forms of haul-out were needed.

2.5 Tyre evolution and the reduction in soil compaction

There has been a focus in agricultural tyre evolution on the reduction of soil compaction. However it was found (Loraine ten Damme et al. [12]) that although the construction of the tyre has evolved, it is the increased width and lower inflation pressures of modern tyres that contribute the most to the decrease in soil compaction.

Loraine ten Damme et al. [12] found that for tyres of different construction, but with identical width and inflation pressure, there was no measurable difference in the resulting soil compaction. This led them to conclude that overall tyre width and inflation pressure is more important to prevent soil compaction than the construction or make-up of the tyre.

2.6 SoilFlex: A model for prediction of soil stresses and soil compaction

Keller et al. [2] has created a two-dimensional model for the calculation of soil stresses and soil compaction due to agricultural field traffic. The model is written as an Excel spreadsheet which makes it easy to use for research or by agricultural advisors but does not make it very accessible for practical everyday applications. The Excel spreadsheet is available via email request to the original authors only. Various papers have attempted to adapt this model into other forms, some of which are discussed below.

The model is feature-rich, it makes it possible to form realistic predictions of the contact area and stress distribution from readily available tyre parameters. The model also allows for calculations including multiple numbers of tyres, rather than just the one tyre as is common in other models.

2.7 soilphysics: An R package for calculating soil water availability

soilphysics is an R package that can be used for calculating and modelling various aspects of soil compaction. de Lima et al. [3], found that existing soil compaction models were not easily accessible and believed that by creating a freely distributed R package they could make these models more accessible. For example, the SoilFlex [2] model, while easy to use must be requested by email. The authors of the soilphysics paper have also created a web-based version of their package, called PredComp (available here).

The R package created by de Lima et al. [3] includes functions for modelling soil compaction such as *stressTraffic()*, *soilDeformation()* and *soilStrength()*. These functions can be used to model various aspects of soil compaction. For example the *stressTraffic()* produces a visual representation of the modelled contact stress and stress propagation.

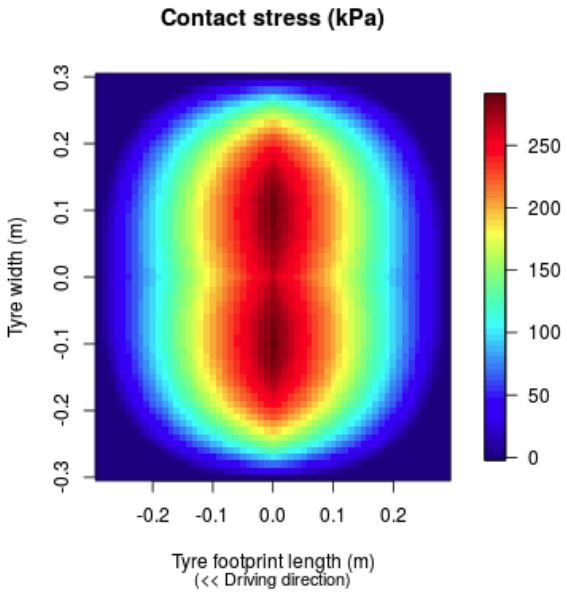

Figure 2.2: Visual Representation of Contact Stress produced by the *stressTraffic()* function in soilphysics. [3] As generated on the PredComp web version.

2.8 Micro-services For Agricultural Applications

This project aims to implement the various soil compaction models using a micro-services framework. A micro-services framework was chosen for its numerous advantages and because of my familiarity with the technology. While working on the background research for this project I wanted to see if a micro-services framework had been deployed in an agricultural setting before and how

it had been deployed. This was to validate the suitability of using a micro-services framework in an agricultural project such as this one.

Taneja et al. [13] utilised a micro-services framework when working on the SmartHerd system. Although the paper mostly focuses on the advantages and implementation of micro-services in fog computing (a computing paradigm that extends cloud computing and services to the edge of the network [14]) I believe the advantages the authors found with using a micro-services framework will also apply to this project.

Taneja et al. chose to use a micro-services framework because it provided them with a number of advantages. For this reason, I also chose to utilise a micro-services framework in this project. Some of the advantages of a micro-services framework include;

- Lightweight communication between services
- Independent control and deployment of each service
- Technology in-dependency and ability to be language agnostic
- Isolation support, an outage on one service will not affect the other services

2.9 Discussion

From the background work above I have formulated a plan that aims to achieve the objectives set out in the Problem Statement in the Project Description.

The first portion of the project will be dedicated to preparing data. After which this data will be used to assess other soil compaction models and then implement my own soil compaction models. This will involve comparing and contrasting different models to each other and to the models I implement.

Once the models have been implemented I will tackle the accessibility aspect of the project. The project aims to make the models implemented accessible and easy to use. This will be achieved by deploying the models via a micro-services framework and an easy to use Graphical User Interface (GUI).

Implementing the models via a micro-services framework was chosen due to the relative ease and flexibility such a framework offers. A micro-services framework allows for models of different types to be easily integrated. It also allows for integration with a front-end GUI which is vital to achieving the accessibility aims of this project.

Chapter 3: Project Management Plan

Below I've outlined a Gantt Chart for both semesters. A large portion of Semester 1 was dedicated to preparing for the main work of the project. I scheduled work in Semester 2 to begin two weeks before the resumption of the teaching term on the 17th of January 2022.

3.1 Semester 1

Figure 3.1: Gantt Chart For Semester 1.

Figure 3.1 outlines the Gantt Chart for Semester 1. The main tasks for this semester were *Background & Project Description* and *Literature Review & Background Reading*. The semester then finished with a short project presentation which is timetabled at the end of the Gantt Chart.

The *Background & Project Description* task involved some background reading on the topic and an introduction to the team from the UCD School of Agricultural Science and Food Science that I would be working with. I had a basic understanding of the topic as I had studied Agricultural Science for a few months previously and was raised on a working farm. This time was also used to understand the project and the objectives I was aiming to achieve.

A large majority of the semester was spent on the *Literature Review & Background Reading* task. This involved in-depth reading and analysis of the topic, which included literature reviews and background readings. These are detailed above in the Background Work section.

3.2 Semester 2

The Semester 2 phase started on the 3rd of January 2022 and ran until the 24th of April 2022. The 24th of April is the end of the Spring Teaching Term. This period is a total of 16 weeks. The various tasks to be completed have been laid out above in Figure 3.2.

I started the phase by working on the data, this included data considerations and data prepossessing. These were relatively short tasks and only took a week each to complete. The next major task was to assess the existing soil compaction models and to find any insights or learnings that could

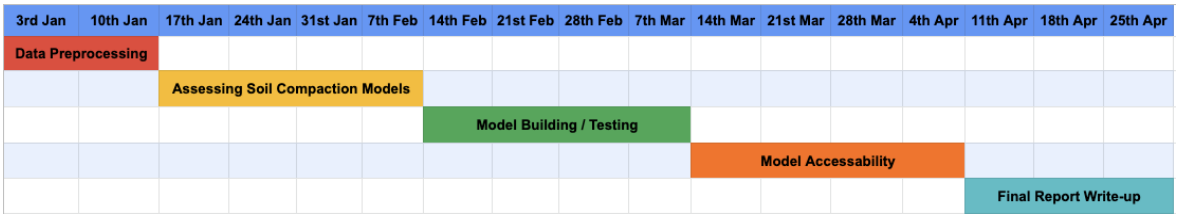

Figure 3.2: Gantt Chart For Semester 2.

be taken from them. This took about four weeks and was followed by the implementation of my own soil compaction models via a micro-services framework.

The implementation of the soil compaction models and the micro-services framework is another large task that also took about four weeks to complete. Following on from this I started developing the GUI interface for accessing the models which was completed in just under three weeks.

The final task was the completion of the report which was carried out at the end of the semester. The report summarises all work done in the project, evaluates what has been achieved and makes suggestions on what could have been improved or what would have been done differently if the project was to be done again.

Chapter 4: Data Considerations

4.1 Dataset

The dataset used for this project was collected over the course of two years by UCD PhD student Mark Ronan. It includes data collected over the course of 2020 and 2021.

The data collected for both years included Soil Moisture, Bulk Density, Cone Penetrometer, Plant Emergence, Combine Yield, Linear Metre Yield, Thousand Grain Weight, Grain Screenings, Grain Protein and Grain Moisture. As well as these, Leaf Area Index (LAI) and Chlorophyll Level data were collected in 2021.

Below I discuss each data attribute in more detail. **Note:** while all of the data attributes have been explained and discussed below, not all attributes were used in the process of building the models for this project.

4.1.1 Soil Moisture

Soil Moisture is a measure of the water content in the soil. It is measured as a percentage of the total soil weight. Readings can be obtained by weighing a set volume of soil, drying the soil out and then weighing the soil again. The weight of the fresh soil less the weight of the dry soil divided by the weight of the dry soil gives the water moisture percentage of the soil.

Alternatively a soil moisture meter may also be used to obtain the data.

Figure 4.1: A commercially available soil moisture meter. [15]

4.1.2 Bulk Density

Bulk Density is a measure of how densely compacted the soil is. It's a measure of the weight of the soil per cm^{-3}. It is measured in $g\,cm^{-3}$.

Bulk density is measured by obtaining a known volume of soil and measuring the dry weight of the soil within that known volume. The bulk density can then be calculated by dividing the dry soil weight by the known soil volume.

4.1.3 Cone Penetrometer

The Cone Penetrometer data is data generated after conducting a Cone Penetrometer test on a soil sample. The Cone Penetrometer test provides a profile of the soil's shear strength over a series of different depths. It does this by recording the pressure values at 2.5cm interval depths.

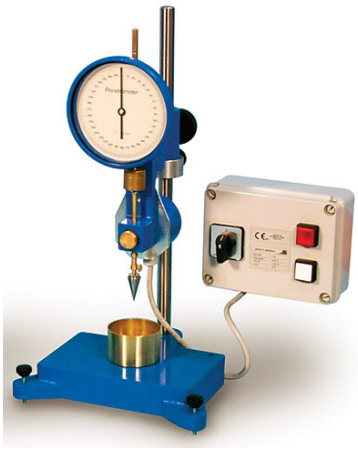

Figure 4.2: A Cone Penetrometer is used to provide a profile of soil stratification. [16]

4.1.4 Plant Emergence

The Plant Emergence data is a count of the number of plants that have emerged over a given area. Three random samples of the plot are taken. A count of the number of plants that have emerged is then taken. An average is then calculated for the plot as a whole.

4.1.5 Combine Yield

The Combine Yield is the yield as measured by the combine at harvest. It is measured in metric tonnes per acre. This data can be used to understand how a crop has performed. The combine yield can be affected by several different factors, such as crop type, variety type, seeding rate, soil type, weather / climate, etc.

Figure 4.3: An example of recording the plant emergence data. [17]

4.1.6 Linear Metre Yield

The Linear Metre Yield is another form of yield measurement. It measures the yield per linear metre. This data was not used in the project. The combine yield is a better representation of the overall yield of the field which is why it was used as one of the target values for this project.

4.1.7 Thousand Grain Weight

The Thousand Grain Weight data is a weight measurement of one thousand grains of a given crop. The Thousand Grain Weight (T.G.W) is often used in determining seeding application, as knowing the weight of one thousand grains will allow us to work out what volume/weight of seed will be needed depending on the desired seeding rate.

For example, the Thousand Grain Weight for some common Irish cereal crops are [18]

- **Winter Barley:** 40 –> 55 grams
- **Winter Wheat:** 40 –> 55 grams
- **Winter Oats:** 30 –> 40 grams

Figure 4.4: A commercially available T.G.W Weight Calculator. [19]

The Thousand Grain Weight can be obtained by counting out one thousand grains and weighing the grains. For a more accurate result, this process should be carried out several times and the results averaged. Alternatively, there are commercial kits available to make this process more efficient.

4.1.8 Grain Screenings

Grain Screenings is a measure of how large the grain kernels are. It involves passing the grain harvest over a series of screens with differing mess sizes. The screen size starts off small and increases in size in a set interval. This dataset uses screen sizes of *2cm, 2.25cm, 2.5cm* and *> 2.5cm*.

The data is measured as a percentage of the total number of grains. For example, 76.64% of grain kernels are larger than 2.5cm.

Figure 4.5: A commercially available grain screening kit. [20]

4.1.9 Grain Protein

Grain Protein is measured as a percentage of the overall mass of the grain. For example, Durum wheat, which is commonly used for making pasta, has a relatively high protein percentage between **14% - 16%**. Lower protein grains, commonly used in animal feeds, have a protein percentage of less than **12%**. [21]

Grain protein can be measured in the lab using methods such as the Kjeldahl method [23]. Farmers can also use commercially available, handheld grain protein metres such as the one above.

4.1.10 Grain Moisture

Grain Moisture is a measure of the moisture level in the harvested grain. It is measured as a percentage of the total grain weight. The data can be gathered in much the same way as the *Soil Moisture* data is. Commercial grain moisture testers are also available.

Grain Moisture is an important measure as it plays an important role in the quality and ability to store the grain. For example, a moisture content of **13.5%** or less is recommended for Barley so it

Figure 4.6: A commercially available, handheld grain protein measurement device. [22]

can be stored safely without the need for further artificial drying. [24] Grain that is too wet when it is stored can lead to spoilage. It is important to ensure all grain stored is sufficiently dry, as high moisture grain can create areas of high temperature rapidly. This can result in serious grain damage, damage to grain bins and ideal conditions for moulds and pests. [25]

Figure 4.7: A commercially available grain moisture meter. [16]

4.1.11 Leaf Area Index

Leaf Area Index is a measure used to describe how much of a given area contains leaf cover. It is directly related to the amount of light that can be used by the plants.

Note: The LAI in the dataset used was measured on the 15th of June, the 12th of July and the 6th of August 2021. There is no LAI data for 2020 in the datasets used.

4.1.12 Chlorophyll Levels

The Chlorophyll data is a measure of the concentration of chlorophyll in the plant. Chlorophyll is the pigment that gives plants their green colour. It is also a vital part of the photosynthesis process carried out by plants.

Note: The chlorophyll levels were measured on the 28th of May, the 23rd of June, the 12th of July and the 5th of August 2021. There is no chlorophyll data for 2020 in the datasets used.

4.1.13 Treatments

One of the attributes in the dataset is *treatment*, which is a summary of the various soil operations or work that was carried out on the plot. It is an alphabetic encoding of five main operations undertaken on the soil. This data was not collected as is the case for the data above. It is generated from the other data attributes and serves as a summary.

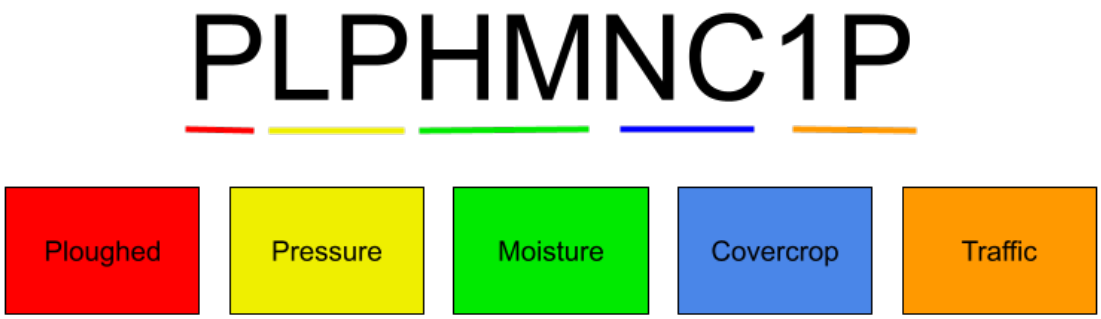

Figure 4.8: Breakdown of Treatment Encoding

Figure 4.8 shows the breakdown of the treatment encoding. Later I will use these encodings to build some of the more promising models. The encodings were taken and converted into binary values that were then used as training data for the models.

The first letter of the encoding corresponds to the soil operation that was undertaken. In the dataset provided this was either ploughing or tilling. The effect of these operations has been explored in more detail in the Contrasting Ploughed & Tilled Soil section.

The second and third letters refer to the tyre pressure of the machinery used. *LP* denotes "low pressure" and *HP* denotes "high pressure". The fourth and fifth letters refer to the soil moisture level in the same manner. *HM* being "high moisture" and *LM* coinciding with "low moisture".

The sixth and seventh letters describe the presence or absence of a cover crop. *NC* indicates that there was "no cover crop" and *CC* when a cover crop is present.

The final two letters refer to the traffic level the plot experienced. *1P* denotes the plot experienced a single pass and *3P* denotes that there were three passes taken over the plot. For our purposes, we take a single pass to mean "low traffic" and three passes to be "high traffic".

4.2 Analysis

Before any model building was undertaken I felt it was important to first gather a clear and deep understanding of the data that was available for the project. Much time was spent analysing and comparing the analysis to previous analyses that had been carried out on the datasets.

Two main areas of focus were analysed. The difference in soil moisture and the contrast between

ploughed and tilled soil.

4.2.1 Soil Moisture

Soil moisture is measured by the percentage of moisture in the soil. In the collected data the moisture percentage was measured at three different depths, 0cm to 10cm, 10cm to 20cm and 20cm to 30cm. It's important to note here that these depth ranges were chosen as the bulk density measurements were also taken at these depth intervals. Having a consistent depth range across the two measurements makes comparing and working with the data easier as there is no conversion between the two needed.

The soil moisture is measured using a commercially available moisture metre. It can also be measured by first getting a weight measurement for a set volume of soil, then drying the soil out and reweighing the set volume of soil. The difference in weight can then be used to calculate the percentage of moisture in the soil.

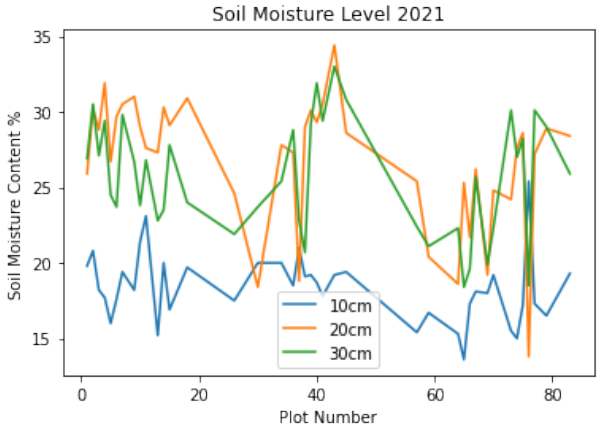

Figure 4.9: Soil moisture % per plot in 2021.

Figure 4.9 shows the soil moisture content for the year 2021, while Figure 4.10 shows the soil moisture content for the year 2020. Some plots are missing moisture content measurements in 2020, which explains the coarse nature of the graph. When we compare the two years of soil moisture data available, we can see that in 2021 the deeper layers of the soil were the wettest, however, in 2020 this trend is reversed. 2020 sees the upper layers, 10cm and 20cm, contain a higher percentage of moisture.

These differences could be explained by 2021 seeing more rainfall than 2020 or it may be due to the weather on the given day the moisture readings were taken. Without further weather analysis, we can't draw a definitive conclusion here. Correlating historical weather data could potentially reveal some insights into the cause of the discrepancy between the two years, however, this is beyond the scope of this project.

The large difference in soil moisture values between the two years was taken into consideration when designing and developing the models later on in the project. Ideally, there would be enough data available to be able to account for these differences in moisture content over the years, however, this is not the case here.

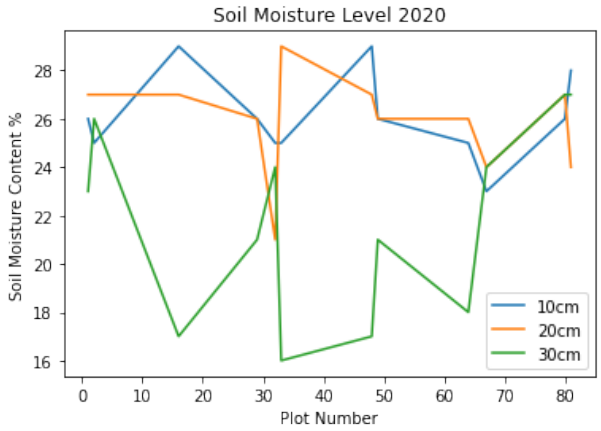

Figure 4.10: Soil moisture % per plot in 2020.

4.2.2 Contrasting Ploughed & Tilled Soil

Ploughing and tilling were both used to prepare the seedbed in the available data. It is important to note here, that the same seedbed operation was carried out on the same plot in both years. For example, if a plot was ploughed in 2020 it was also ploughed in 2021.

Ploughing is a process whereby a plough is drawn over the soil to prepare a seedbed for planting. It involves turning over the uppermost section of the soil. This in turn brings fresh nutrients to the surface while also simultaneously burying any weeds. It is typically a slower process than tilling, however, it typically reaches deeper into the soil than tilling.

Tilling is a seedbed operation that involves the agitation of the top layer of soil. This can be done in a variety of different ways including but not limited to, digging, stirring and overturning. Tilling is typically quicker, carried out at a higher speed than ploughing, but typically does not reach as deep as ploughing.

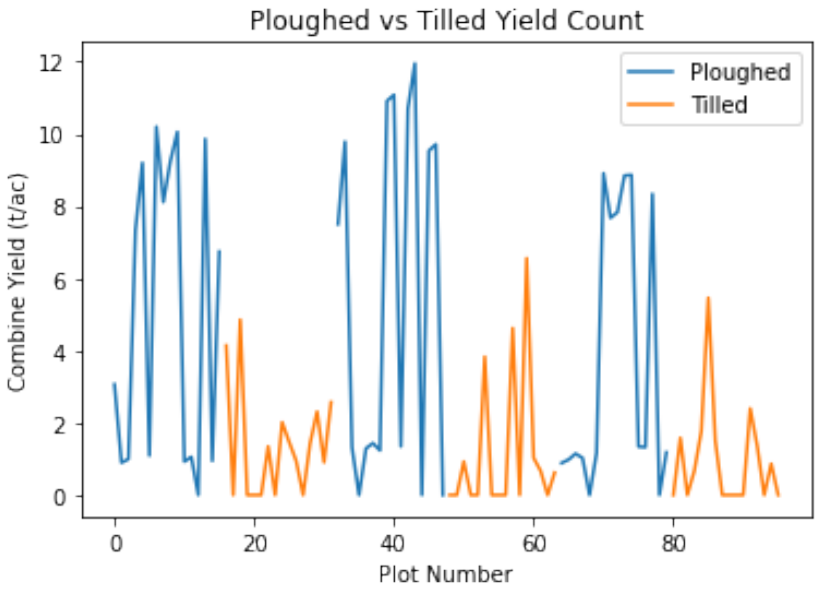

Figure 4.11: Ploughed vs Tilled Combine Yield data averages.

Once the data has been cleaned, over the two years of data available there were 82 ploughed plots and 54 tilled plots. Below I have averaged both the yield and emergence of each plot and compared them based on whether the plot was ploughed or tilled. Blue indicates the plot has been

ploughed and orange indicates the plot was tilled.

Once averaged out it becomes clear from Figure 4.11 and Figure 4.12 that the plots which were ploughed result in a higher overall combine yields and a higher emergence count respectively. Due to the connection between emergence count and yield, this is not surprising. Due to the ploughed plots having a higher yield and greater emergence count in every plot we can confidently say that ploughing is preferable to tilling in our case. This may be explained by the greater ability of ploughing to breakup and disturb soil compaction due to it's deeper reach.

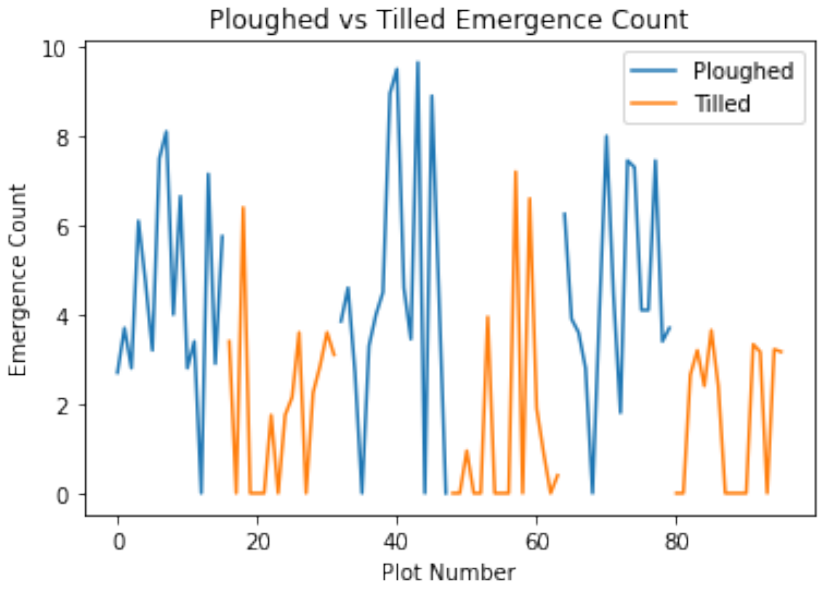

Figure 4.12: Ploughed vs Tilled Emergence data averages.

Chapter 5: Outline of Approach

In this section, I will look at the various aspects of the project undertaken. I have divided the chapter into three sections. These sections cover the main tasks, from data cleaning to model development and then finally to covering the work involved in making the models accessible.

Below I have included some code snippets where relevant. The code snippets are used to further explain and demonstrate the actions carried out. The code for the entire project can be found in this GitLab code repository. Every effort has been made to make the code easy to understand and run. The repository's accompanying *README.md* explains how to run the various aspects of the project.

5.1 Data Cleaning

Data cleaning is the process in which data is cleaned and prepared for use. Cleaning data involves removing any unwanted, incorrect, poorly formatted or irrelevant data from the dataset.

The data for this project was delivered in a clean and almost ready to use nature. As a result, there was only a small amount of data cleaning required. The most time and effort was spent on the formatting of the data.

The data provided was in the form of two distinct years, 2020 and 2021. Both datasets were formatted differently, therefore the first step was to align the formats of the two years. This was done by taking the available data and condensing it into one large spreadsheet, which could later be easily read and manipulated by my models. The format conversion was largely completed using Excel.

Any missing values were also removed and replaced with *0*. It was also necessary to convert certain data attributes to other formats. For example, the data for *Combine Yield* in 2020 was collected in *tonnes/hectare* and in 2021 the data was measured in *tonnes/acre*. Therefore I choose to convert the yield data in 2020 to *tonnes/acre* so the two years could be compared. This conversion was done by multiplying the *t/hc* yield by **2.471**, as 1 hectare is equivalent to 2.471 acres.

Operation	Example
Missing Values	"" -> 0
Stripped Strings	" string " -> "string"
Formatting Numbers	24% -> 24
tonnes/hectare -> tonnes/acre	(2.021 * **2.471**) -> 4.994

Table 5.1: Data Cleaning Operations

Certain data attributes were also cleaned to make working with the data easier. This involved removing extra characters, white space and general formatting. For example, the data for the moisture in the soil was measured as a percentage and the data contained the **%** symbol to represent this. As part of my data cleaning process, these characters were removed.

5.2 Model Development

Arguably the most important aspect of this project was the model development portion. It was vital that the models were as accurate as possible and that they would be of use to farmers and agricultural advisors. Therefore a lot of thought and work was dedicated to understanding the potential inputs available to us and the outputs that would be most useful to our target audience. Below I discuss in more detail the work undertaken when building these models to reach these targets.

5.2.1 Initial Models

The main objective for the models was to predict both the emergence and overall yield given the input data. Both yield and emergence were chosen early on as targets as they provided the most useful result to any potential users of the models. It is worth noting that over the course of the model testing, the yield has shown to be more of a useful target than the emergence. The emergence values that were being predicted by the models varied little and the link between the emergence and overall yield was weak. The yield was also found to be a more practical metric for use by farmers and agricultural advisors.

There are two main models used in this project, Linear Regression and Support Vector Regression (SVR). Each model type is split into its own micro-service. This allows for the future addition of more model types by adding more microservices to the system. The design of this is discussed further in the Model Accessibility section. Each model type has several variations built onto it. These variations include a *Soil Treatments* model, *Soil Treatments with Granular Moisture* model, *Moisture Percentage* model and a *Bulk Density & Moisture* model. Each of these variations is available in models based on Linear Regression and Support Vector Regression.

Each model variation is trained in a similar manner. The available data is split into a test and training set. This approach is taken as it allows us to verify and test the models using the test data. The data is usually split using a 77%/33% ratio.

```
# Splitting data for training & testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33,
    random_state=42)
```

Code Snippet 5.1: Splitting Data

The first model variation developed was a very basic moisture percentage model, referred to as the *Moisture Percentage* model variation. This variation was chosen as it is a relatively basic model with only a few inputs. It would also be an easy model to use as it would only require the user to input the soil moisture percentage at three given depths to make a prediction.

The moisture percentage data available is split into three different depths. These depths are 0cm-10cm, 10cm-20cm and 20cm-30cm. This data was then used to train and test the model. Unfortunately, the results from this model were not very promising. I believe this is down to a lack of data and a wide variation in the moisture data. The soil moisture percentage in 2020 was very different to the soil moisture percentage in 2021, as investigated above in the Data Analysis section. The model's performance is discussed in more detail below when it is compared to the other models developed later in the project.

The second variation I explored was a bulk density and soil moisture model, referred to as the *Bulk Density & Moisture* model variation. The idea behind this model was to investigate if we could predict any noticeable change in emergence or yield based on the compaction of the soil and how

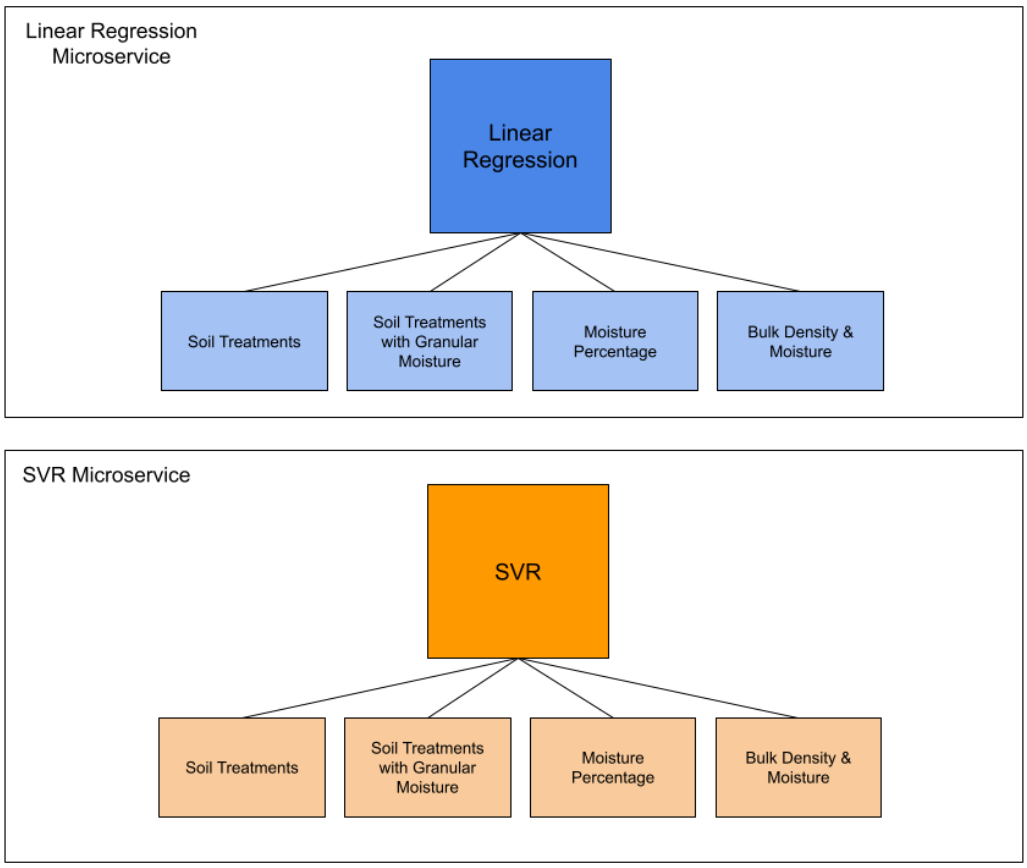

Figure 5.1: Microservice and Model Structure

dry the soil is. A wet and compacted soil is usually avoided due to the damage they cause. This model variation aimed to see if this theory could be modelled using the data we have available.

Much like the soil moisture percentage data the soil's bulk density is also measured at the same three depths. However, unlike the soil moisture percentage, the bulk density data is recorded in $g\ cm^{-3}$.

The accuracy of these two models is discussed in further detail below. The accuracy of the two models was found to be poor, especially when the data from both 2021 and 2020 was used. I also found that once we presented our models to the Agricultural Experts involved in the project, they found that although the *Moisture Percentage* and *Bulk Density & Moisture* model variations were useful, they were not as promising as the soil treatment models. Therefore these two models were not investigated and developed to the same levels as the soil treatment models below were.

5.2.2 Final Models

The final set of model variations developed was based on the soil treatments. After discussions with the Agricultural Experts, it was decided to focus most of the remaining time and energy on this set of models as they proved to be the most promising and most useful. I also developed a variation on these soil treatment models which included a granular soil moisture input, rather than a binary input as the original model contained. Had time permitted a third soil treatments model variations was planned. This variation would have built on the granular moisture model and also included a granular option for the bulk density. This granular option would be used in-place of a binary high / low traffic value.

As previously described above in the analysis of the soil treatments data [4.1.13], the soil treatment is a summary of the various soil operations and activities and a selection of the important soil attributes. The soil treatment attribute is a nine letter acronym which covers the soil operations, the farm machinery tyre pressure, soil moisture, the presence of a cover crop and the traffic level the soil experienced (which is used to replicate a bulk density measure). These five parameters are then converted into a binary value, in a similar manner to that of one-hot encoding. Encoding the treatments as a range of binary attributes allows for the creation of a model that is easy to work with. These binary inputs are intuitive to a user who is not familiar with machine learning or data science. For example, converting the presence of a cover crop parameter into a binary option is straightforward for a user to understand when they want to use the model to make a prediction. Their soil either has a cover crop present or it does not.

As discussed above in the analysis of the soil moisture data [4.2.1], the soil moisture data varied wildly over the two years of data that was made available for this project. As a result, this had a detrimental effect on the performance of the models that were built using the soil moisture data. For that reason, the models built and analysed in this section only use the data from 2021. The final models included in the source code for this project are built using both years of data. However, for this report and for presenting to the Agricultural Experts involved in the project, I decided it was more beneficial to present and discuss the accuracy and potential of the models using just the data collected in 2021. Due to the nature of farming, yields, emergence, rainfall and many other parameters change significantly from year to year. As a result the accuracy of any models built with a very small number of years of data, in this case only two years, will suffer. The models in the source code are intentionally developed to use all the years of data available as the longer-term aim of the CONUS work is to gather further data in the coming years. The more years of data added, the more potential and the more accurate the models will become. However, we can get an idea of how accurate and how useful the models will be by only using the data from 2021 as a demonstration.

Arguably the most important and most useful accuracy metric used to measure the models' performance is *Mean Absolute Error (MAE)*. The MAE is a measure of the error in the models' performance on the same scale as the data which is being measured. The MAE uses the same scale as the data being measured, which allows us to get a "real world" estimate of the error involved in the models. For example, a model with a MAE score of *0.75* would mean that the model in question has an error of *0.75 tonnes/acre*. This makes the MAE a very useful metric to measure the error rate of the models as it returns a value that is easy to understand to a user who may not be familiar with machine learning models. The MAE was a vital metric that was used when presenting my model findings to the Agricultural Experts.

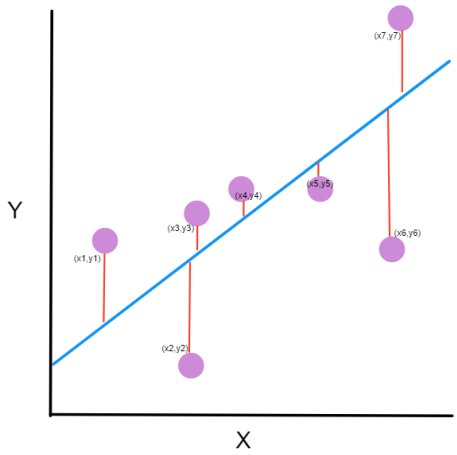

Figure 5.2: A visualisation of how the MSE is calculated [26]

The *Mean Squared Error (MSE)* is derived from the average of the squares of errors. It is also

sometimes called the *Mean Squared Deviation (MSD)*. It is a useful measure as it can be used to get an idea of how close or how far away the model is from its intended prediction. For this reason, it is said to be a measure of the quality of the estimator or prediction. It is always a positive number which approaches zero as the error diminishes. It is derived from the square of the Euclidean distance.

R^2 is a goodness-of-fit measure for linear regression models. R^2 is used to determine how well the model fits the data. R^2 is only used for linear regression models, because of this I haven't included a R^2 score for the SVR models in Table 5.3. A linear regression model is said to fit well with the data if the differences between the observations and the predicted values are small, the R^2 value measures this fit.

Linear Regression and Support Vector Regression (SVR) are both popular classification models that ultimately aim to solve the same problem. On a macro level, the models developed in this project aim to take various inputs and predict an emergence value and a yield value as accurately as possible. Linear Regression models and SVR are both used in this project to attempt to achieve this aim.

Although Linear Regression and SVR are similar, they differ slightly. In simple terms, a linear regression model can be thought of as trying to find a "line of best fit" through the various input data fed into the model. It is often considered to be one of the more basic regression models, but at the same time is one of the easier models to understand. In this project, I've used the linear regression models as a baseline and I have then used the SVR models in an attempt to increase the accuracy of the models.

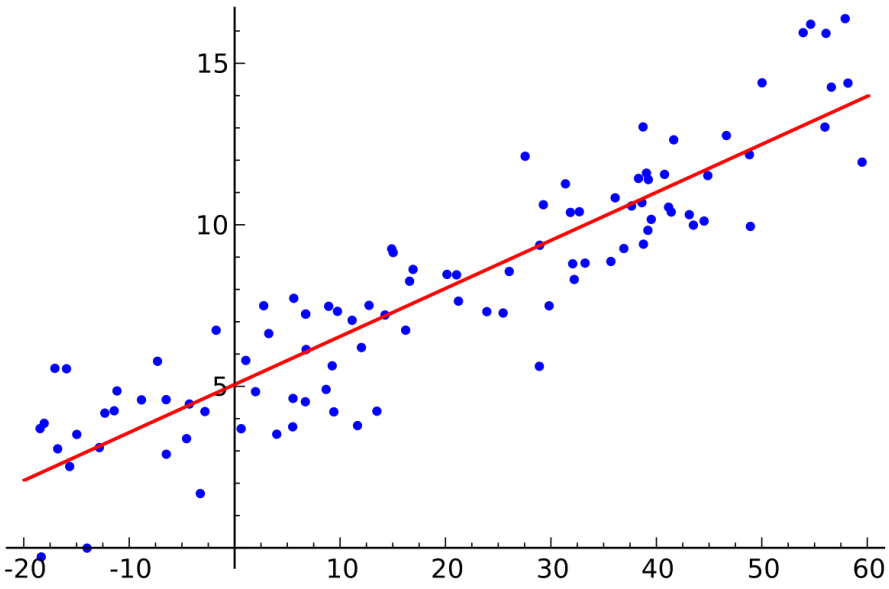

Figure 5.3: An example of a visualisation of a linear regression model [27]

Support Vector Regression (SVR) is a regression-based variant of the widely popular Support Vector Machines (SVM). Support Vector Machines are a supervised learning model that analyses data for classification and regression problems. SVM aims to separate the data into classes with as big a margin between the classes as possible. The most useful advantage SVM has over Linear Regression in this project, is a technique called the *"Kernal trick"*, which allows SVM to perform non-linear classification by mapping the inputs into higher dimensions. [28]

Due to the R^2 metric only being applicable to linear regression models we can only compare the R^2 scores between the model variations in Table 5.2. The R^2 score can not be used as to evaluate the SVR model variations in Table 5.3. One important thing the R^2 score illustrates here is the

similarity between the two soil treatment models. After modifying the original *Soil Treatments* model variation, there is very little change in the R^2 score for the *Soil Treatments with Granular Moisture* model. This is encouraging as the accuracy of the model is not negatively effected, which was one of the initial fears when investigating the viability of adding a granular moisture element to the models.

Model Name	MAE	MSE	R^2
Soil Treatments	0.41812	0.39258	0.51721
Soil Treatments with Granular Moisture	0.39860	0.38697	0.52410
Moisture Percentage	0.35584	0.22085	-1.06536
Bulk Density & Moisture	0.62004	0.72541	0.10789

Table 5.2: Comparing 2021 Yield Model Accuracy - Linear Regression

Interestingly the R^2 score for the *Moisture Percentage* model variation is negative. This may suggest that the model is over-fitting the training data. When a model is over-fitting, it is very strongly correlated to the training data and doesn't perform very well with unseen or test data. The negative R^2 score for the *Moisture Percentage* model variation is very likely to be a result of the relatively small amount of data available to build the model with.

Table 5.2 outlines the MAE, MSE and R^2 for the various linear regression model variations when predicting the yield. Table 5.3 outlines the same three accuracy parameters for the model variations built on SVR. When comparing the MAE of both the linear regression and SVR models, the MAE of the SVR models is slightly better. This translates to a smaller error in the emergence and yield predictions.

The Mean Squared Error (MSE) is a good quality of estimation error as it measures the average of the squares of errors. And again, when comparing the Linear Regression models and the SVR models I found that the SVR models had a smaller MSE. This would indicate that there is less error from the SVR models.

Model Name	MAE	MSE
Soil Treatments	0.38988	0.29582
Soil Treatments with Granular Moisture	0.36385	0.30301
Moisture Percentage	0.35614	0.21045
Bulk Density & Moisture	0.57095	0.62840

Table 5.3: Comparing 2021 Yield Model Accuracy - SVR

Comparing the two tables also shows us that both the *Soil Treatment* models, which were the most promising from an agricultural point of view, are also the most promising from an accuracy point of view. This makes me confident that I made the correct choice in focusing more effort on the *Soil Treatment* models instead of the original *Bulk Density Moisture* and *Moisture Percentage* models.

After discussing the four model variations with the Agricultural experts it was made clear that the *Soil Treatments* and *Soil Treatments with Granular Moisture* models would be of most use to them. The overall accuracy between the two variations isn't very big. For example, when I compare the Mean Absolute Error of the two SVR treatment models I find that the difference is only *0.02603* or about 6%. This result is encouraging, as using a granular soil moisture input is more useful to the user than a binary value. The *Soil Treatments with Granular Moisture* model variation allows the user to do this without sacrificing any accuracy in the model predictions.

5.2.3 Adding Future Data

From the beginning of the project, it was clear that future-proofing the project was going to be needed. As discussed above, the results of the models are somewhat affected by the volume of data available. This was known early on in the project and it was decided that to mitigate this issue somewhat, the models would be developed in such a way as to allow additional data (data generated and collected in future years for example) to be easily added to the models.

When deciding on how best to go about this task several different solutions were explored. One such solution was to divide the datasets into directories corresponding to their respective years and to develop a system for the automatic detection and inclusion of this data. This system could also be accompanied by a user friendly GUI for uploading the data. The system was deemed to be the best system from a user experience point of view, however, it had the drawback of being time-intensive to develop. More on the opportunity cost and decision not to pursue this system is discussed in the Summary and Conclusions section.

As a result, a more streamlined and basic system is used to allow users to add future data. A dataset template CSV file is included in the GitLab repository along with the code for this project. This template lays out the format the data must be in for the models to read and make use of the dataset. Any new data can be converted into this format and then appended to the existing data. Figure 5.4 illustrates part of the format the data will be required to be in.

Plot	Treatment	BD 0-10cm	BD 10-20cm	BD 20-30cm	BD Moisture 0-10cm	BD Moisture 10-20 cm	BD Moisture 20-30 cm	Emergence	Chlorophyll	28th May	Chlorophyll	23rd June
1	PLPHMNCLA	1.11	1.29	1.61	21.2	31.0	34.3	7.4	39.3		41.6	
2	PHPLMCCHA	1.16	1.42	1.58	19.8	25.9	26.9	7.4	38.0		43.9	
3	PLPHMNCHA	1.15	1.55	1.66	20.8	30.2	30.5	5.6	45.9		45.5	
4	PHPLMNCLA	1.05	1.47	1.47	18.2	28.8	27.1	4.8	44.7		46.3	
5	PLPLMNCHA	1.14	1.21	1.51	17.7	31.9	29.4	5.0	36.7		48.3	
6	PHPLMNCHA	0.90	1.19	1.34	16.0	26.7	24.5	6.4	41.8		41.7	
7	PLPLMNCLA	1.17	1.37	1.51	17.6	29.7	23.7	7.2	41.2		40.9	
8	PHPHMNCLA	1.26	1.49	1.53	19.4	30.5	29.8	6.4	38.8		45.7	

Figure 5.4: An example of part of the data format

5.3 Model Accessibility

One of the core objectives of the project was to not only develop models based on the data available but also to make these models available and easily accessible. As mentioned above in the Problem Statement, a microservices based framework and a user-friendly front-end GUI was settled on as being the best way to achieve this objective. Below I discuss the approach taken and the process involved in making the models accessible.

Git was used extensively throughout the project for code management. Although version control is often used in a team environment, its use can still provide value when working on an individual project such as this. Figure 5.5 shows an example of how branches were used for code management. The two branches used in this example, were responsible for the initial frontend setup using Vue and the initial backend setup using Flask. The branches were worked on independently of each other and once completed were merged back into the main *master* branch.

Pull requests were also a core Git feature used in the project. Due to the project being an individual undertaking there were no code reviews in the traditional sense, however, I found that using pull requests provided a second look over the code before merging which enabled me to spot and fix bugs quicker than would have otherwise been possible.

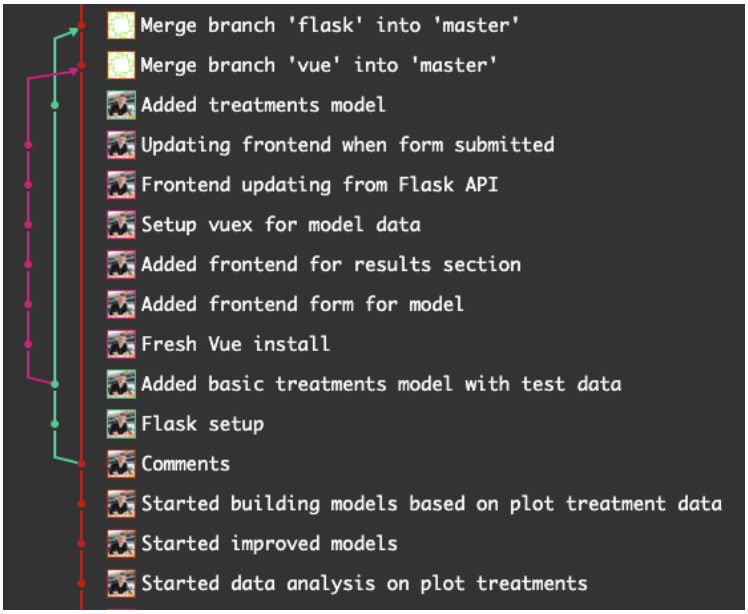

Figure 5.5: An example of Git branches used for code management.

5.3.1 Microservices API

The back-end micro-services for the project are written in Python and use the Flask framework [29]. Flask was chosen as it is a relatively lightweight Python web framework. My models were designed and implemented in Python, therefore using a Python-based framework was the obvious choice for the back-end. Alternatively, I also investigated the Django framework [30] as a possible solution, but ultimately decided that some of the features offered by the Django framework were not needed and a lighter-weight framework such as Flask would be more suitable.

One of the main tasks involved with the setup of the Flask back-end was to convert the models, which were developed in Jupyter notebooks into vanilla Python scripts so they could be accessed by Flask. Jupyter notebooks [31] is an interactive web-based UI that allows for easy data representation and display. It is therefore a common tool used for Data Science applications. Fig 5.6 shows the process that was taken to convert the models so they could be accessed by Flask.

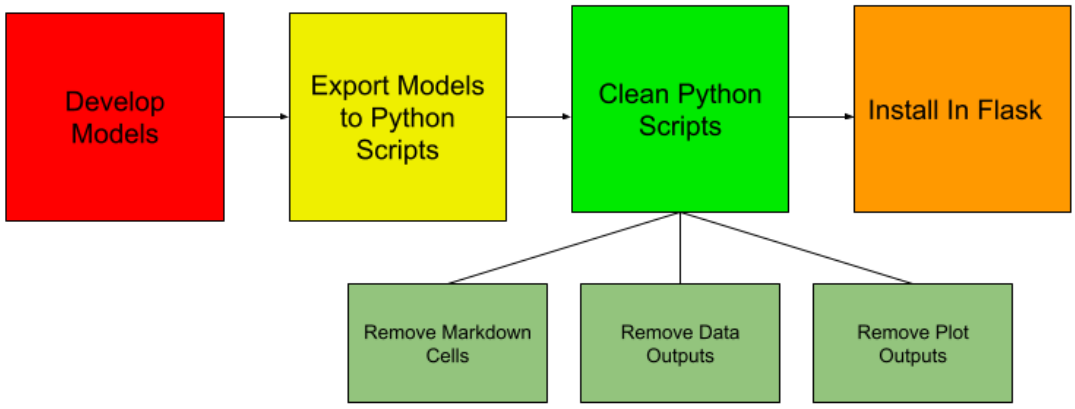

Figure 5.6: Process For Exporting Models Built-in Jupyter Notebooks to Python Script Files

Jupyter notebooks are stored in a JSON representation and rendered with HTML. This makes them incompatible with standard Python script files unless the Python code is first exported. Once the

Python code has been exported there is some cleaning required to remove some of the code that is specific to Jupyter notebooks. For example, Jupyter notebooks make rendering inline plots very easy. However, Python script files do not support this inline rendering. As a result, these lines of code need to be removed before the models can successfully be installed into the Flask application. Jupyter notebooks generally also take advantage of specific markdown cells, which again can be removed during the cleaning process. The final step in the model export process is to install the cleaned model code into the Flask application.

Once installed and run the model's results are translated into a JSON format, which enables seamless communication with the frontend Vue application and the Flask backend application. This JSON format contains the model name and two objects. A yield and an emergence object, both of which contain the result returned by the model. Listing 5.2 illustrates an example of a return for the *Treatments* model.

```
{
  "model":"treatments",
  "yield":{
    "result":2.6167494486998626
  },
  "emergence":{
    "result":7.115124180143208
  }
}
```

Code Snippet 5.2: Example JSON Return from API Endpoint

This format was chosen as it makes comparing the results from two models very easy. After discussions with the Agricultural Experts, it became obvious that being able to easily compare two models with slightly different inputs would be a very useful accessibility feature. For example, a user might want to compare the difference in predicted yield using high pressure versus low-pressure tyres. The chosen JSON format allows me to return the data for model comparison in a very similar way to that of a single model result. Listing 5.3 illustrates the JSON structure when two models are being compared. *result2* is the result of the compared model, while *difference* is the percentage change in the result of the two models. This percentage change can either be positive or negative.

```
{
  "model":"treatments compare",
  "yield":{
    "result":2.6161234859462446,
    "result2":2.6167494486998626,
    "difference":0.023927110359304303
  },
  "emergence":{
    "result":7.042192236627212,
    "result2":7.115124180143208,
    "difference":1.0356426105022898
  }
}
```

Code Snippet 5.3: Example JSON Return For Comparing Models

The microservices are also designed to return relevant HTTP response codes. HTTP response codes are an integral part of any REST API as they indicate whether a specific request has been successfully completed, has run into an error or has some other response. For example, if any of the required get parameters are not included in the request the microservices respond with a **422** error. A **422** error indicates that the microservice understands the content type and the request,

but it is unable to process it.[32] If the request is successful, the microservices return a **200** success response code. [33]

Initially, I wanted to host the microservices/models and the frontend GUI online using a service such as Heroku.[34] However, due to time constraints it was decided to focus on further model development at the expense of deploying the models. Instructions on how to run the microservices and frontend GUI locally are included in the Git repository. The choice to not deploy the models is discussed further in the Summary and Conclusions section.

5.3.2 Front-end GUI

The front-end for the project was built using Vue 3 [35] and written in TypeScript [36] (TypeScript is a strongly typed programming language that builds on JavaScript). The frontend also utilises Bootstrap CSS [37] which is a frontend CSS library that allows for rapid prototyping and removes a lot of the setup process involved with vanilla CSS. The frontend application uses the built-in JavaScript fetch API [38] to communicate with the back-end API developed from the Flask application.

Vue was chosen as a frontend library due to my familiarity with the framework. Vue also excels at handling dynamic single-page applications (SPAs) which leads to a better user experience. One of the main focuses of the frontend for this project was to develop an easy to use, no-frills GUI. Vue also comes with Vuex [39], a state management pattern and library for Vue. The main use for Vuex is as a state management library. It assists in the management and sharing of data across multiple Vue components and pages. Although Vuex is primarily used in larger applications with multiple pages, it can also be used on SPAs to assist with data management and structure. It is, for this reason, I have decided to use it alongside Vue to assist in managing the data sent back and forth to the Flask backend microservices.

The Vue frontend uses the JavaScript fetch API to communicate with my API to retrieve the results from the microservices. It then updates the frontend GUI in real-time, without the need for the user to refresh the page manually. This instant updating of the frontend data was another reason for choosing the Vue.JS framework.

Vue is a component-driven JavaScript framework. This involves segregating the frontend structure into various reusable components. This makes for rapid prototyping and code management and was another important consideration when it came to choosing a front-end JavaScript library.

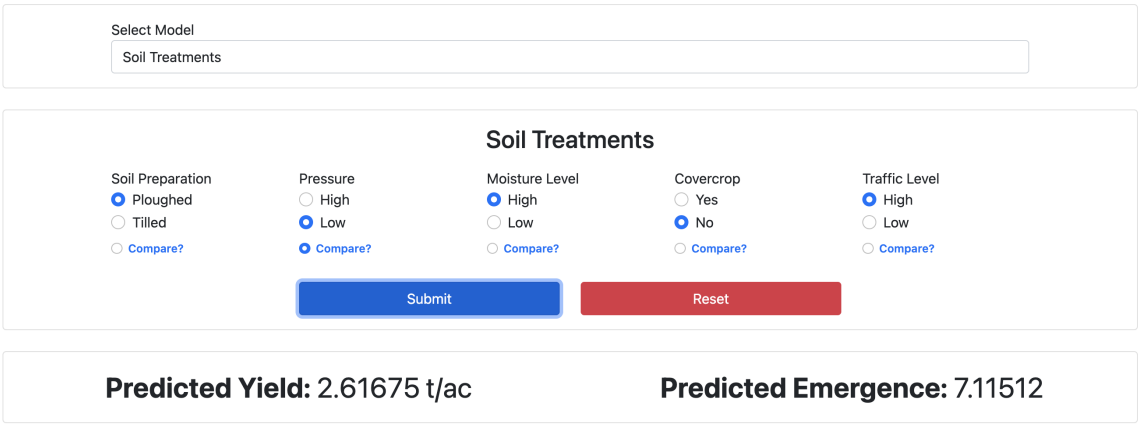

Figure 5.7: Rendered Vue Components

The GUI is broken down into three main components, the model selector component (*ModelSelect*),

the models themselves and the results component (*Results*). The *ModelSelect* component is responsible for controlling which of the model components is displayed to the user. It contains a single HTML select element which updates a Vuex variable upon selection. The model components are then displayed based on the value of the Vuex variable set in the *ModelSelect* component. The *Results* component, as the name implies, is responsible for displaying the predicted yield and predicted emergence that is returned by the models. It also displays other information relating to the models and the model comparisons. The *Results* component displays this information by accessing the data stored in Vuex via a computed method. A computed method in Vue is a method that automatically updates when the data it references is changed. It is these computed methods that allow for instant updating of the frontend without the need for a browser refresh.

```
/**
 * Returns models predicted yield
 */
predicted_yield() {
    // Data in Vuex stored at this.$store.state.models.yield.result
    return this.$store.state.models.yield.result;
}
```

Code Snippet 5.4: A Computed Method In Vue

5.3.3 Application Run-Through

The following section shows a brief run-through of the frontend GUI that was developed to increase the accessibility of the models.

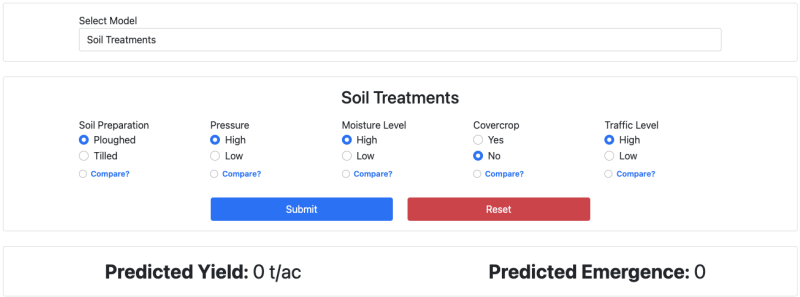

Figure 5.8: GUI Landing Page

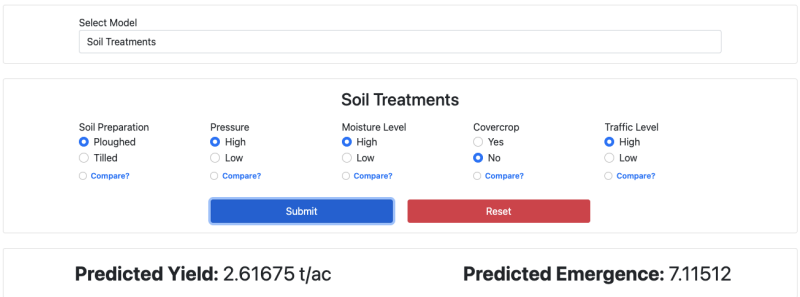

Figure 5.9: Results From A Model

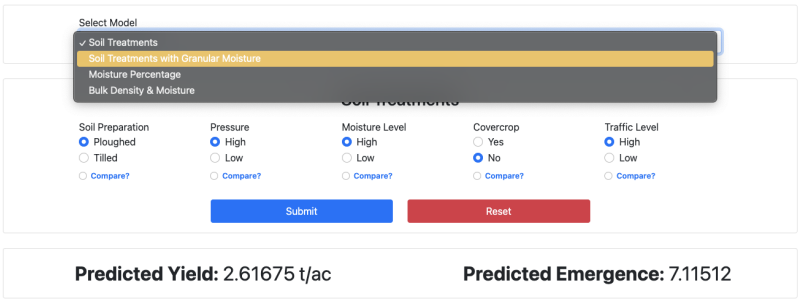

Figure 5.10: Selecting A Model Variation

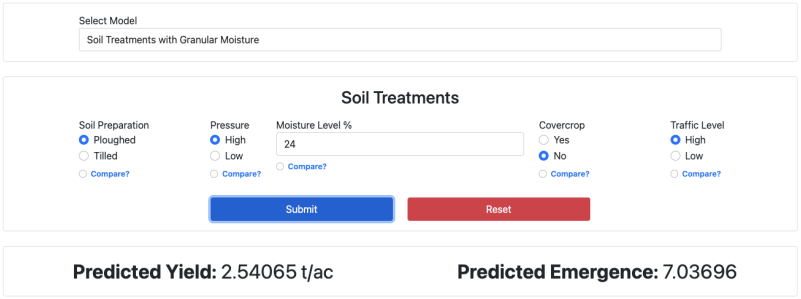

Figure 5.11: A Second Model It's Results

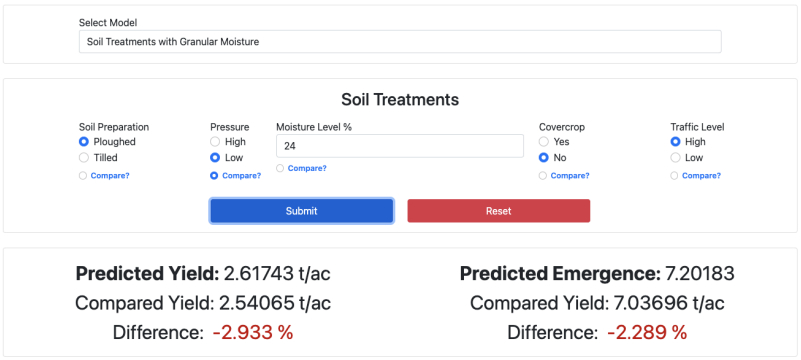

Figure 5.12: Comparing Two Model's Results

Chapter 6: **Summary and Conclusions**

The purpose of this project was to investigate whether it was possible to model the predicted yield and emergence based on a range of different input data and make these models accessible for use. My results show that while it is possible to build meaningful, useful models to predict yield and emergence, the lack of data affected the overall quality and accuracy of these models. This report also shows that it was possible to develop a system around these models to make them as accessible as possible.

Ultimately the project set out to build accurate, accessible models that could be used to inform agricultural decision making. These aims have largely been achieved. In particular, I believe that the aim to make the models accessible to all, including those without any computer science, data science or machine learning experience has been achieved. When compared to the accessibility of other similar agricultural models such as *SoilFlex* [2] and *soilphysics* [3], the accessibility of the models developed in this project is far greater. The implementation of these models in a micro-services based framework also allows the project to be expanded on in the future, which again increases its accessibility to others.

Although my results show that the models developed in this project have an acceptable level of accuracy for use, this accuracy is only found in the models when using data from a single year. In practice, this is not ideal. However, I believe that the poor accuracy levels found in the models using multi-year data are because of the small amount of data available. Due to the nature of farming, an uncountable number of variables affect every year's harvest. This was briefly explored when comparing the soil moisture levels between the two years of data in 4.2.1. Therefore with enough data, I feel that these models could be of use.

The results show that the project's aim to build models that could be used to assist agricultural decision making has largely been achieved. However, it is also important to mention that there are some limitations to these models that have not been discussed in the report. The data used to train the models in this report was collected at UCD Lyons Farm in Co. Kildare. Therefore some of the geographic-specific data will only be relevant to the UCD Lyons Farm locality. For example, a farm found to have a different soil type to that of UCD Lyons Farm will not benefit as much from these models as a farm with the same soil type. This is something that could be explored and improved with future work. Soil type, average climate and many other factors could be included when training these models to achieve more accurate results for a wider variety of farms.

With more time and resources I think there is great potential to expand on this project. One of the objectives that weren't met in this project due to time constraints was the desire to have the models hosted online. At present the models and GUI can only be run locally, which although not difficult to do, is still more involved than a hosted solution would be. Another area that fell victim to a lack of time was the future adding of data. Although this project implemented a valid and easy to use system for adding data in the future, there are better solutions available. This was discussed more in Adding Future Data.

In conclusion, while the models developed during the course of this project have limitations, I feel that they are still of use and could be improved significantly with a larger dataset. I feel that the aim to make the models accessible has been achieved and the foundations have been laid for the project to be expanded on in the future.

Chapter 7: Acknowledgements

Throughout the writing of this report, I have received a great deal of support and assistance from a number of different individuals. This report would not have been possible without their help and support.

I would first like to thank my supervisor, Associate Professor Rem Collier, whose expertise and direction have been invaluable in executing the project plan. Your insightful experience and direction, in particular when it came to the structure of the microservices and frontend GUI have been invaluable. Our weekly Zoom meetings have not only been informative and productive but also very enjoyable.

I would also like to thank Prof. Kevin McDonnell and the rest of the CONUS team at the UCD School of Agriculture and Food Science, Mark Ronan, Dr Mary Harty, Dr Gary Gillespie and Dr Saoirse Tracy. Your help and kindness in explaining the agricultural side of the project has been of immense value. I would have often found myself lost without your help.

In addition, I would like to thank Dr Aonghus Lawlor for your work in coordinating and directing the Final Year Project over the past few months. Your lectures, examples and templates have been incredibly valuable. They have made the writing of this report much easier than it would have been otherwise.

Bibliography

1. *POLICY BRIEF SUBSOIL COMPACTION -A THREAT TO SUSTAINABLE FOOD PRODUCTION AND SOIL ECOSYSTEM SERVICES* https://www.ecologic.eu/sites/default/files/publication/2018/2730_recare_subsoil-compaction_web.pdf.
2. Keller, T., Défossez, P., Weisskopf, P., Arvidsson, J. & Richard, G. SoilFlex: A model for prediction of soil stresses and soil compaction due to agricultural field traffic including a synthesis of analytical approaches. *Soil and Tillage Research* **93,** 391–411. ISSN: 0167-1987. https://www.sciencedirect.com/science/article/pii/S0167198706001413 (2007).
3. de Lima, R., da Silva, A., da Silva, A., Leão, T. & Mosaddeghi, M. soilphysics: An R package for calculating soil water availability to plants by different soil physical indices. *Computers and Electronics in Agriculture* **120,** 63–71. ISSN: 0168-1699. https://www.sciencedirect.com/science/article/pii/S0168169915003403 (2016).
4. Schjønning, P. *et al.* in (ed Sparks, D. L.) 183–237 (Academic Press, 2015). https://www.sciencedirect.com/science/article/pii/S0065211315001108.
5. Keller, T., Sandin, M., Colombi, T., Horn, R. & Or, D. Historical increase in agricultural machinery weights enhanced soil stress levels and adversely affected soil functioning. *Soil and Tillage Research* **194,** 104293. ISSN: 0167-1987. https://www.sciencedirect.com/science/article/pii/S016719871930131X (2019).
6. *Soil threats in Europe* https://esdac.jrc.ec.europa.eu/public_path/shared_folder/doc_pub/EUR27607.pdf.
7. Linda.francis & Linda.francis. *Wet Weather + Field Traffic = More Soil Compaction, Reduced Nutrient Use Efficiency and Yield* Nov. 2015. https://agrilife.org/texasrowcrops/2015/11/04/wet-weather-field-traffic-more-soil-compaction-reduced-nutrient-use-efficiency-and-yield/.
8. Duiker, S. W. *Effects of soil compaction* Nov. 2021. https://extension.psu.edu/effects-of-soil-compaction.
9. Voorhees, W. *et al.* Long-term effect of subsoil compaction on yield of maize. *Advances in GeoEcology,* 331–338 (2000).
10. Da Silva Guimarães Júnnyor, W. *et al.* Prediction of soil stresses and compaction due to agricultural machines in sugarcane cultivation systems with and without crop rotation. *Science of The Total Environment* **681,** 424–434. ISSN: 0048-9697. https://www.sciencedirect.com/science/article/pii/S0048969719320200 (2019).
11. Lozano, N., Rolim, M., Oliveira, V., Tavares, U. & Pedrosa, E. Evaluation of soil compaction by modeling field vehicle traffic with SoilFlex during sugarcane harvest. *Soil and Tillage Research* **129,** 61–68. ISSN: 0167-1987. https://www.sciencedirect.com/science/article/pii/S0167198713000329 (2013).
12. ten Damme, L. *et al.* The contribution of tyre evolution to the reduction of soil compaction risks. *Soil and Tillage Research* **194,** 104283. ISSN: 0167-1987. https://www.sciencedirect.com/science/article/pii/S0167198719300923 (2019).
13. Taneja, M., Jalodia, N., Byabazaire, J., Davy, A. & Olariu, C. SmartHerd management: A microservices-based fog computing–assisted IoT platform towards data-driven smart dairy farming. *Software: practice and experience* **49,** 1055–1078 (2019).
14. IoT, P. *et al. IOT, from Cloud to Fog Computing* June 2021. https://blogs.cisco.com/perspectives/iot-from-cloud-to-fog-computing.

15. *PH Moisture Meter* https://www.quickcrop.ie/product/light-moisture-meter.
16. *Grain Moisture Meter* https://agrarzone.com/grain-moisture-meter-gmm-mini.
17. *Evaluating The Stand: Canola Encyclopedia* June 2021. https://www.canolacouncil.org/canola-encyclopedia/plant-establishment/evaluating-the-stand/.
18. Teagasc. *Sowing and varieties - Teagasc: Agriculture and Food Development Authority* https://www.teagasc.ie/crops/crops/cereal-crops/winter-cereals/sowing-and-varieties/#:~:text=we%20then%20need,30%20%E2%80%93%2040%20grams.
19. *Culverthorpe 1000 grain weight calculator* https://www.pragdirect.co.uk/culverthorpe-1000-grain-weight-calculator.
20. *Grain Testing* https://www.agracom.com.au/services/grain-testing.
21. Parry, O. *Options for farmers to test grain protein in a field: Food testing equipment* Apr. 2021. https://www.calibrecontrol.com/news-blog/2019/7/19/options-to-test-grain-protein-in-field.
22. *Grainsense hand-held grain protein and moisture meter* https://www.calibrecontrol.com/main-product-list/grainsense-hand-held-grain-protein-analyser.
23. *Kjeldahl Method* https://www.sciencedirect.com/topics/earth-and-planetary-sciences/kjeldahl-method.
24. *Harvesting, Drying and Storing Malting Barley* https://ambainc.org/wp-content/uploads/2019/10/Harvesting.pdf.
25. *Strategies for managing high moisture grain at harvest* https://grdc.com.au/news-and-media/news-and-media-releases/south/2020/november/strategies-for-managing-high-moisture-grain-at-harvest#:~:text=Mr%5C%20Warrick%5C%20says%5C%20high%5C%20moisture,for%5C%20moulds%5C%20and%5C%20insect%5C%20pests.&text=Ideally%5C%2C%5C%20use%5C%20a%5C%20silo%5C%20with,air%5C%20distribution%5C%20through%5C%20wet%5C%20grain..
26. *Machine Learning: an introduction to mean squared error and regression lines* https://www.freecodecamp.org/news/machine-learning-mean-squared-error-regression-line-c7dde9a26b93/.
27. *Introduction to Machine Learning Algorithms: Linear Regression* https://towardsdatascience.com/introduction-to-machine-learning-algorithms-linear-regression-14c4e325882a.
28. Wilimitis, D. *The Kernel Trick* Feb. 2019. https://towardsdatascience.com/the-kernel-trick-c98cdbcaeb3f.
29. *Flask Documentation (2.0.x)* https://flask.palletsprojects.com/en/2.0.x/.
30. *The web framework for perfectionists with deadlines | Django* https://www.djangoproject.com/.
31. *Project Jupyter* https://jupyter.org/.
32. *422 unprocessable entity - http: MDN* https://developer.mozilla.org/en-US/docs/Web/HTTP/Status/422.
33. *200 OK - http: MDN* https://developer.mozilla.org/en-US/docs/Web/HTTP/Status/200.
34. *Heroku - Cloud Application Platform* https://www.heroku.com/.
35. *Vue.js - The Progression JavaScript Framework* https://vuejs.org/.
36. *TypeScript: JavaScript With Syntax For Types* https://www.typescriptlang.org/.
37. *Bootstrap · The most popular HTML, CSS, and JS library in the world* https://getbootstrap.com/.
38. *Using the Fetch API - Web APIs | MDN* https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch.

39. *Vue.js - Vuex Data Store* https://vuex.vuejs.org/.