DATA599 CAPSTONE

PROJECT REPORT

Agriculture and Agri-Food Canada - Exploring Optimal Machine Learning Models for Predicting Crop Yield at Township Level

Mayukha Bheemavarapu

Mehul Bhargava

Navdeep Singh Saini

Val Veeramani

Table of Contents

Agriculture and Agri-Food Canada - Exploring Optimal Machine Learning Models for Predicting Crop Yield at Township Level

Team Members:

Mayukha Bheemavarapu, Mehul Bhargava, Navdeep Singh Saini, Val Veeramani

INTRODUCTION

Aim:

The project aims to explore optimal machine learning models for predicting crop yield based on growing patterns at the township level. Using the Canadian Prairie domain as the study area, we seek to test the performance of various Machine Learning algorithms in the prediction of Canola yield and compare the results with the observed values.

The project will use the Python programming language with its various libraries and packages for applying Machine Learning methods and algorithms.

Background and Focus:

Agriculture and Agri-Food Canada is a department of the government of Canada that focuses on the regulatory, production, marketing, and policy aspects of all farms and Agri-based goods. AAFC works to grow Canada's exports, while providing leadership in the expansion and development of a competitive, innovative, and sustainable Canadian agriculture and agri-food sector.

Crop yields are crucial to meet the demands of millions in Canada. Global warming and climate change has a significant impact on food security for a lot of communities. Predicting crop yields would go a long way in decreasing the negative effects of climate change. It will also help our fellow farmers and government plan accordingly. Therefore, our goal is to provide AAFC with an optimal model to predict future crop yields.

Summary of methods used:

1. **Regularization Techniques:** Regularization is a form of regression that shrinks the coefficient estimates towards zero. Regularization works by adding a penalty or shrinkage term with Residual Sum of Squares (RSS) to the complex model.
    a. **Ridge Regression:** This algorithm minimizes RSS along with the sum of squares of the magnitude of weights.

b. **LASSO:** This algorithm minimizes the RSS plus the sum of absolute value of the magnitude of weights.

2. **Dimensionality Reduction:** Used Principal component analysis (PCA) to simplify the complexity in high-dimensional data while retaining trends and patterns by transforming the data into fewer dimensions called Principal Components (PCs).

   a. **Principal Component Regression:** This algorithm feeds the PCs obtained from PCA as predictors to the linear regression model.
   b. **PCA + Random Forest:** This algorithm feeds the PCs obtained from PCA as predictors to a Random Forest model.

3. **Deep learning algorithms:**
   a. **Feed Forward Neural Network:** In this neural network, the data passes through multiple nodes but it moves only in one direction and never backwards.
   b. **MLP Regressor:** A multilayer perceptron is a special case of a feedforward neural network where every layer is a fully connected layer.

Summary and Results:

- Higher the number of townships per eco district, higher the accuracy and lower the test MSE.
- Zone of high accuracy of models corresponds to the zone of high density of Canola.
- Machine learning seems to outperform the deep learning methods when it comes to modeling with a low number of records.

RELATED WORK:

- The importance and potential benefits of crop forecasting varies according to a range of criteria, namely, relevance, reliability, stake- holder engagement, holism and accuracy (Hammer et al., 2001; McIntosh et al., 2007). But Nurturing crop growth isn't straightforward. There are a multitude of factors that affect the yield including temperature and precipitation of weather during the growing season. Other factors include (Guo and Xue, 2012; Qian et al., 2009), soil conditions (Alvarez, 2009; Qian et al., 2009), disease, and anthropogenic factors like irrigation or fertilizers (Prasad et al., 2006). Even though

meteorological data are very accessible, some of the variables and factors are often difficult to quantify.

- Two modern technologies called the Normalized Difference Vegetation Index(NDVI) and the Enhanced Vegetation Index (EVI) which comes from the Moderate-resolution Imaging Spectroradiometer (MODIS) on the Terra satellite are often used to foresee crop conditions in Canada as well as several other parts of the world.
- Since our dataset relates to the Prairie regions of Canada, Mkhabela et al. (2011) shows us that MODIS-NDVI could be used to effectively predict crop yields one to two months before harvest specifically in those regions in Canada.
- Bolton and Friedl (2013) used both MODIS- EVI and NDVI to forecast maize and soybean yields in parts of central United States where he found EVI being more of a better predictor than NDVI was. But it really depends on the region. Some methods seem to work better in some areas where others seem to falter.

DATA

The project involves predicting the crop yield at a township level for the canola crop for the provinces of Manitoba and Saskatchewan. The total agricultural area is divided into individual entities of 10x10 sq.km and each of such individual units is called a township.

Data Sources:

1. **Crop yield data:**

- It consists of recorded crop yield data provided by provincial crop insurance agencies of Saskatchewan, and Manitoba. Originally, this data was obtained at quarter section level and now have been rescaled to township level using geostatistical techniques.

2. **Earth Observation predictor variables:**

- This consists of satellite derived predictor variables which include:

a. Normalized Difference Vegetation Index (NDVI) from MODIS satellite platform.
b. Surface soil moisture from active and passive microwave sensors.
c. The evaporative stress index from thermal-optical data.
d. Agro-climate variables derived from station-based weather observation inputs.

3. High resolution modeled weather data

- This is obtained from the Canadian Meteorological at 2.5km and 10km of radius. These data sets have been aggregated at township level for at least 20 years.

Data Overview:

- The data is for the prairies region. This type of area has moderate temperatures, rainfall, and other climatic conditions. It has fewer trees, flat or gently sloped land, and it's ideal for growing crops. In Canada, the western provinces of Manitoba, Saskatchewan, and Alberta are considered part of the Canadian Prairies. The data we have is for Manitoba and Saskatchewan.
- Each predictor type in the data is subdivided into 3 week running means as separate predictors for the 18th week (May) to the 40th week (September). The time period from May to September is the growing season for the canola crop.
- The area in the data is in hierarchical order with eco zones on the top, then eco region, eco province, eco district, and townships (10x10 sq.km), the smallest area. In this project, the crop yield is predicted at a township level for each eco district. Refer to Figure [] for eco districts and Figure [] for townships.
- The data is for the years 2010 to 2020 (11 years).
- There are four different datasets; each of them has township ID and year as variables (based on which we have joined them).

Below are the descriptions of the four data sets that we have:

1. Crop yield data:

It consists of recorded canola crop yield (in Kg/acre) at township level for 2010 to 2021 data. It is the target dataset, based on which we have joined all the four datasets. The key factors in joining the datasets are the year and the Township ID. It also consists of columns for eco zone IDs, eco region IDs, eco province IDs, and eco district IDs.

2. Climate data:

It consists of recorded climatic conditions data for the years 1985 to 2021. The data in this file had been preprocessed into crop relevant variables and 3-week running mean before being provided for the project.

The dataset has six types of predictors defined below:

- **Precipitation (Pcpn):** Total precipitation (rainfall) in millimeters. The values are in a 3-week running average form.

- **Growing Degree Days (EGDD_C):** effective accumulative growing degree days calculated using a 5-degree threshold (equation 1). Heat index is used for cool season crops like wheat, canola, barley, and oats. The values are a 3-week running average.

  $$GDD = \Sigma\,(Tavg - Tbase) \quad \#(1)$$

  where,

  T_{avg} = 0.5* (T_{max}+T_{min}), average daily temperature

  T_{base} = 5 ◦C for cool season crops

- **Heat stress days (HeatD):** the number of days in which the Tmax (maximum temperature per day) exceeds 30 degrees Celsius. The values are in a 3-week running average form.
- **Frost risk days (FrostD):** the number of days with a Tmin (minimum daily temperature) of less than 2°C. The values are in a 3-week running average form.
- **Water Stress Index (SI):** defined as SI = 1-AET/PET, where AET and PET are the actual and potential evapotranspiration, respectively. It relates to the availability of water and the amount of water required. If the required water level is like the available water, the index value increases. The values are in a 3-week running average form.
- **Available Water Holding Capacity (PrcnAWHC):** soil water content expressed as a percentage of the plant's available soil water holding capacity. In other words, it's the soil's ability to hold water to make it available for the plant. The values are in a 3-week running average form.

3. Soil Moisture Data (ECV_SMOS): This dataset contains weekly data for remote sensing of surface (5 cm) soil moisture by township from 2010-2020 for the entire prairies region. This dataset required further processing of data into a 3-week running mean to be used for modeling. Satellites or aircraft are used to get the soil moisture for a large area. The values are in the form of the number of pixels of the remotely sensed images of the reflected emitted radiation from the earth's surface.

4. NDVI MODIS:

Normalized Difference Vegetation Index is a graphical indicator that is used to analyze the green patches on land. MODIS stands for Moderate Resolution Imaging Spectroradiometer, a satellite-

based sensor used to measure earth and climate measurements. MODIS vegetation indices give an idea if the crops are healthy in that area. The values are in a 3-week running average form. Its value ranges from -1.0 to 1.0, where negative values indicate clouds and water, positive values near zero indicate bare soil or area with rocks, and positive values near one indicate dense green vegetation.

Data Wrangling:

Starting with data loading and reading, we explored the datasets by looking at their shape, data types, null values, and negative values. All of the datasets have different ways to handle and impute null and negative values, as all of them have different meanings.

Data wrangling for each dataset:

1. Crop Yield dataset:

Firstly, we had several yield values with negative (as shown in Table 1) & null metrics. As this is something that logically wouldn't make sense to appear in our dataset, we decided to convert all the remaining negative values into null values and drop all of them. Also, we found that there are 4429 township IDs and 144 unique eco district IDs.

Table 1: Overall description of data

	YieldAcre2010	YieldAcre2011
Count	4404	4404
Mean	667.90	661.96
Std	201.13	194.20
min	61.70	**-325.36**
25%	538.50	527.58
50%	686.07	672.15
75%	815.20	805.61
Max	1420.71	1229.87

There were 25 null records every year from 2010 to 2021 which we had to handle.
Secondly, this dataset had several demographic stratification measures that we decided not to employ for our future modeling purposes. So, we first went ahead and removed them from this dataset.

2. Climate dataset:

We noticed that there were some negative Average Stress Index values (as seen in Table 2). It doesn't make sense for Stress Index values to be negative. Based on the advice of our client, we decided to convert these negative values to 0.

Table 2: Negative values in Climate dataset

AvgSI25_27	AvgSI26_28
252864	252864
0.297030	0.264344
0.150335	0.167118
-0.002857	**-0.043810**
0.179048	0.138571

We then went on to filter the data for the years we're interested in [2010-2020].

3. ECV_SMOS:

This table is structured in a way that is different from the other predictor tables(as seen in Table 3). We don't have information in a three-week moving average format.

Table 3: First two rows of the Soil Moisture dataset

TWP_ID	VALID_ COUNT	MIN	MAX	RANGE	MEAN	TOTAL_ COUNT	YEAR	DOY	WEEK
00101E1	98	32.69	32.69	0	32.69	98	2011	129	19
00101W1	95	30.03	32.69	2.66	30.84	95	2011	129	19

There are null values as well in the dataset. But there are no negative values.

Lastly, the year 2010 starts from week 23 while the other years start from 18. We sorted the data by the year 2010 and ascending weekly. And we noticed something similar to what is shown in Table 4.

Table 4: Sorted Soil Moisture dataset

TWP_ID	VALID_ COUNT	MIN	MAX	RANGE	MEAN	TOTAL_ COUNT	YEAR	DOY	WEEK

00101E1	98	29.63	29.63	0	29.63	98	2010	158	23
00101W1	95	28.36	29.63	1.27	28.74	95	2010	158	23

For this dataset, we know that the number of valid data pixels in the township (VALID_COUNT), if its ratio with total TOTAL_COUNT is too low (less than 50%), the data quality for this week is poor. We will remove those rows with poor data quality. We will also drop some columns which are not required as well as we have to convert the Week and mean column to 3 weeks running average for 18th to 40th weeks and melt it in such a way so that it can be joined with other datasets.

So, we have removed **6394** rows in this process

Process to transform the table into the format our other tables are in:

Due to the year 2010 not having weeks 18-22, we decided to drop these weeks for all the rest of the years from 2011-2020.

We used pivoting. The mean column is what we're interested in, so we calculated those for every year by the 3-week moving average.

Then we arrive at something like what we see in Table 5.

Table 5: Intermediate table

YEAR	SoulMoisture 23_25	SoilMoisturee 24_26
2010	22.46	20.81
2011	24.77	24.61

Now we will use a for loop to be able to connect each township and year with their corresponding 3-week average soil moisture.

Finally, the dataset is ready to use now. Table 6 shows the final form of this dataset.

Table 6: Finalized Soil Moisture table

TWP_ID	YEAR	SoilMoisture23_25	SoilMoisture24_26
00101E1	2010	29.47	26.04
00101E1	2011	24.60	29.47

4. NDVI_MODIS:

There were no null values but there were some negative values as we can see in Table 7.

Table 7: Negative values in NDVI dataset

	YEAR	NDVI23_25	NDVI24_26
Count	135324	135324	135324
Mean	2010	0.514	0.576
Std	6.06	4.708	4.708
min	2000	**-999**	**-999**
25%	2005	0.453	0.525
50%	2010	0.538	0.609
75%	2015	0.623	0.681
Max	2020	0.875	0.884

So, we converted these negative values to 0 as requested by our client.

We also did not need the NDVI_MAX column so we removed that as well.

Finally, we filtered our data to the years we're interested in.

Data joining process:

- We had to join all datasets in order to easily run our modeling algorithms. Before that we melted the yield dataset (converting it from wide to long). We then proceeded to remove the year 2021 from it as we are only interested in the years 2010-2020.
- We ensured that there were no remaining negative values in the dataset. Now, we started to join each of the predictor datasets with the yield dataset.
- We performed an inner join to combine the yield with the climate dataset using the Township ID and Year columns. Then we combined both the soil moisture and the NDVI dataset again using inner join.
- Finally, we got our dataset which we used for modeling and further analysis. It has 42007 rows, 165 columns, 137 unique eco districts, and 3819 unique townships.

TOOLS, METHODOLOGY, AND TECHNIQUES:

Problem: Explore and deploy an optimal model to predict the crop yield at Township level.The model in itself comprises sub-models, each corresponding to individual ecodistrict (137 in our current data).

Caveats involved: Since we are stratifying the data based on ecodistricts, the number of records obtained is very low i.e., we are not looking at an ideal scenario where n (number of records) >> p (number of predictor variables).

Our approach:

Keeping the end goal and caveats in mind, we went ahead with three approaches. The reasoning behind the approach, the description of techniques used is as follows:

Approach 1- Regularization Techniques:

Our initial approach was to use regularization techniques as these algorithms penalize the models in accordance with the number and weighting of the features. These models also provide us with the features involved in fitting the model and the corresponding importance.

Below are the regularization techniques used:

1. **Ridge Regression:**

   In general, least squares estimation minimizes the Residual Sum of Squares whereas Ridge regression includes a penalty in the estimation process. This algorithm minimizes RSS along with the sum of squares of the magnitude of weights as shown in equation 2. The penalty shrinks coefficient estimates close to zero.

$$Cost\ (W) = RSS(W) + \lambda * (sum\ of\ squares\ of\ weights)$$

$$= \sum_{i=1}^{N} \left\{ y_i - \sum_{j=0}^{M} w_j\, x_{ij} \right\}^2 + \lambda \sum_{j=0}^{M} w_j^2 \qquad (2)$$

   - RSS - Residual Sum of Squares
   - λ - Tuning parameter
   - x: the matrix of input features
   - y: the actual outcome variable
   - w: the weights or the coefficients

2. **LASSO:**

LASSO stands for Least Absolute Shrinkage and Selection Operator and is similar to Ridge Regression except for the fact that it minimizes the RSS plus the sum of absolute value of the magnitude of weights as shown in equation 3. In LASSO, the penalty forces some coefficients to zero whereas Ridge Regression forces them 'close' to zero.

$$Cost\ (W) = RSS(W) + \lambda * (sum\ of\ absolute\ value\ of\ weights)$$

$$= \sum_{i=1}^{N} \left\{ y_i - \sum_{j=0}^{M} w_j\, x_{ij} \right\}^2 + \lambda \sum_{j=0}^{M} |w_j| \qquad (3)$$

- RSS - Residual Sum of Squares
- λ - Tuning parameter
- x: the matrix of input features
- y: the actual outcome variable
- w: the weights or the coefficients

Approach 2- Dimensionality Reduction:

Since the number of records we have is low and has a higher number of dimensions, we went ahead with Principal Component Analysis to obtain dimensionality reduction.

Principal component analysis (PCA) simplifies the complexity in high-dimensional data while retaining trends and patterns by transforming the data into fewer dimensions. These fewer dimensions or principal components obtained act as summaries of features. We can also figure out the proportion of variance explained by each of the components based on which we can decide how many principal components that we would like to retain for further analysis.

Below are the two algorithms implemented involving PCA:

1. Principal Component Regression:

In PCA, we rotated our original predictors and mapped it in a new dimensional space and tossed out the rotations that do not appear to contain important information.

In Principal Component Regression, we use this reduced space as predictor variables for our regression problem.

Following are the steps involved in Principal Component Regression:

a. Generate Principal Components from the original predictor dataset
b. Obtain the cut off for the number of PCs (k) to be used by looking at the explained cumulative variance (here, 95% threshold is considered)

Ex: Top 10 PCs retain 93% of total variance explained in the original dataset whereas the top 12 PCs retain 95% of total variance. Depending on my threshold, I would pick 12 PCs for further analysis

c. Fit a linear regression model (using ordinary least squares) on these k principal components

2. Principal Component Analysis + Random Forest:

Here, post obtaining the PCs corresponding to the desired threshold of cumulative variance, the PCs are fed as predictor variables to Random Forest regressor.

Random forest in general doesn't perform well when we have more features than samples which is the case for multiple Eco districts in our data. By using PCA before fitting the random forest model, we are eliminating this caveat.

Following are the steps involved in PCA + Random Forest Regression:

a. Generate Principal Components from the original predictor dataset
b. Obtain the cut off for the number of PCs (k) to be used by looking at the explained cumulative variance (here, 98% threshold is considered)

Ex: Top 10 PCs retain 93% of total variance explained in the original dataset whereas the top 12 PCs retain 95% of total variance. Depending on my threshold, I would pick 12 PCs for further analysis

c. Fit a Random Forest model on these k principal components

Approach 3- Deep Learning:

One goal of our project is to see whether it would be possible to use Deep learning methods to predict the crop yield using the data at hand. We have implemented the following methods to perform regression using deep learning methods:

1. Feed Forward Neural Network:

A Feed Forward Neural Network is an artificial neural network where the connections between nodes does not form a cycle. In this neural network, the data passes through multiple nodes but it moves only in one direction and never backwards.

Train - Validation - test split used in our model: 60 - 20 - 20
Activation function: ReLu
Optimizer: Adam
Layers: 10

2. Multi-Layer Perceptron:

A multilayer perceptron is a special case of a feedforward neural network where every layer is a fully connected layer.

Train - Validation - test split used in our model: 70 - 10 - 20
Activation function: ReLu
Optimizer: Adam
Hidden layers: 1

Rectified Linear Activation Function (ReLu):
ReLu is a piecewise linear function. It provides an output of zero if the input is negative and outputs the input as-is if it is positive.

Adam optimizer:
Adam involves a combination of 2 gradient descent methodologies: RMSprop and Stochastic Gradient Descent with momentum. It uses the squared gradients to scale the learning rate like RMSprop and it takes advantage of momentum by using the moving average of the gradient instead of the gradient itself like SGD with momentum.

Metrics used for model evaluation:

1. **Mean Squared Error (MSE):**

MSE is calculated by taking the average of squared difference between the predicted values and the actual value (equation 4).

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2 \quad (4)$$

We are calculating both MSE Train as well as MSE Test in our project.

2. **Mean Absolute Error (MAE):**

Here, we calculate the absolute difference between the predicted values and the actual value (equation 5).

$$MAE = \frac{1}{N}\sum_{i=1}^{N}|y_i - \hat{y}_i| \quad (5)$$

3. **Accuracy:**

Accuracy here is defined as 100 – mean (MAPE)

MAPE stands for the **mean absolute percentage error** (MAPE), and it calculates the mean of the absolute percentage errors of forecasts (equation 6).

$$MAPE = \frac{1}{N}\Sigma_i^N \left| \frac{y_i - \hat{y}_i}{y_i} \right| * 100 \quad (6)$$

4. R - Squared:

R - Squared describes how much variance of the response variable is explained by the predictor variables (equation 7).

$$R^2 = 1 - \frac{\Sigma_i^N (y_i - \hat{y}_i)^2}{\Sigma_i^N (y_i - \bar{y})^2} \quad (7)$$

We are calculating both R - Squared Train as well as R - Squared Test in our project.

Table 8: Tasks and the respective tools involved

Task	Tools (or) Python libraries involved
Data Wrangling	● Pandas ● Numpy
Resampling and Model Selection	● Sklearn ○ Train_test_split ○ StandardScaler ○ Lasso ○ LassoCV ○ Mean_squared_error ○ R2_score ○ scale ○ MLPRegressor ○ Metrics ○ Cross_val_score ○ LinearRegression

	○ RandomForestRegressor ○ Ridge ○ RidgeCV ● Tensorflow ● Keras ○ Sequential ○ Dense ○ Adam ○ EarlyStopping
Data Visualization	● Excel ● Altair
Project Collaboration	● Github ● Zoom ● Slack
Quality Checks	Excel

Methodology:

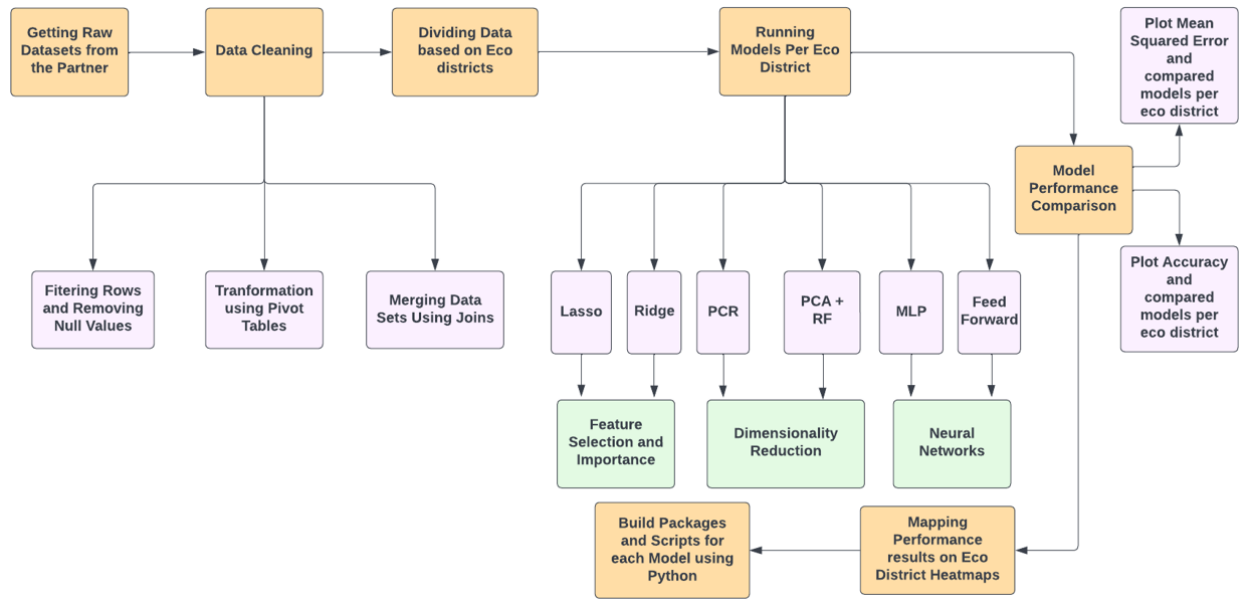

Figure 1: Flow Chart and Architecture

The above Figure 1 describes the end process of our whole capstone for finding the optimal models which can be used in the future for the crop yield prediction. It started with the data collection from different data sets and sources and cleaning it involving various stages of filtering, removing null values, Transforming columns and rows using pivot tables and finally merging the datasets via joining. Once getting the wrangled data, it is further subdivided into each eco district followed by running various models highlighted in the figure per eco district. Post that in the model comparison, metrics such as Mean Square Error and accuracy were plotted for each eco district area and their number of townships for all the models and a thorough comparison was performed as far as these two indicators are concerned. In the final stages, the results were also mapped into the eco district regions via GIS software from the partner end, just to confirm and find expected patterns. Finally as part of end deliverables, we build python scripts and packages for the customer which they can use to run each model and find key performance validation metrics in the future as well.

ANALYSIS:

Feature Selection:

Before predicting the crop yield, the goal is to list down an important set of features, in terms of those corresponding to the relationship with the response variable. In order to work on this requirement, some of the modeling techniques, namely LASSO and Ridge Regression, were implemented with the following results for the top and bottom most features for one of the EcoDistrict **735**:

Top 5 features:

LASSO:

Table 9: Top 5 features by LASSO

Features	Importance
SumPcpn22_24	59.0418080931581
SumEGDD_C28_30	52.0962092477598

SoilMoisture24_26	48.5481592334986
SoilMoisture33_35	40.4867152952536
SumPcpn38_40	26.6604914705978

Ridge:

Table 10: Top 5 features by Ridge

Features	Importance
NDVI34_36	87.3768809325791
NDVI30_32	75.4859935739852
NDVI19_21	73.8345888253101
NDVI25_27	72.6061541748538
NDVI31_33	72.2839960135458

Interpretation:

Based on the above results, it can be inferred that mid and later part of the growing season are the key factors for the crop yield production. Based on the Lasso Model (Table 9), especially in the later period of the growing season, soil moisture plays a very significant role, so if we don't have the moisture content during August-September, it can affect the crop yield to a great extent. The similar thing is applicable for the vegetative index factor in the later part of growing in case of ridge modeling is concerned for extracting important predictors. The net implications are that if there is not sufficient records in the mid or later half of the season, it can have severe implications in the yield predictions.

Apart from the above, there is no specific trend or pattern among all the other predictors when compared with the results of LASSO and Ridge Regression. It is possible that the growing degree and precipitation may play a crucial part, but that is not getting reflected from the ridge regression. However, the results may vary from one eco district to another.

Model Performance:

The accuracy and MSE measures, though similar for several of the models, still have their fair share of differences when we compare them to specific Eco District IDs. The difference is attributed to the number of records and unique townships present in each of these ecodistricts, with certain models performing better while still working with fewer records.

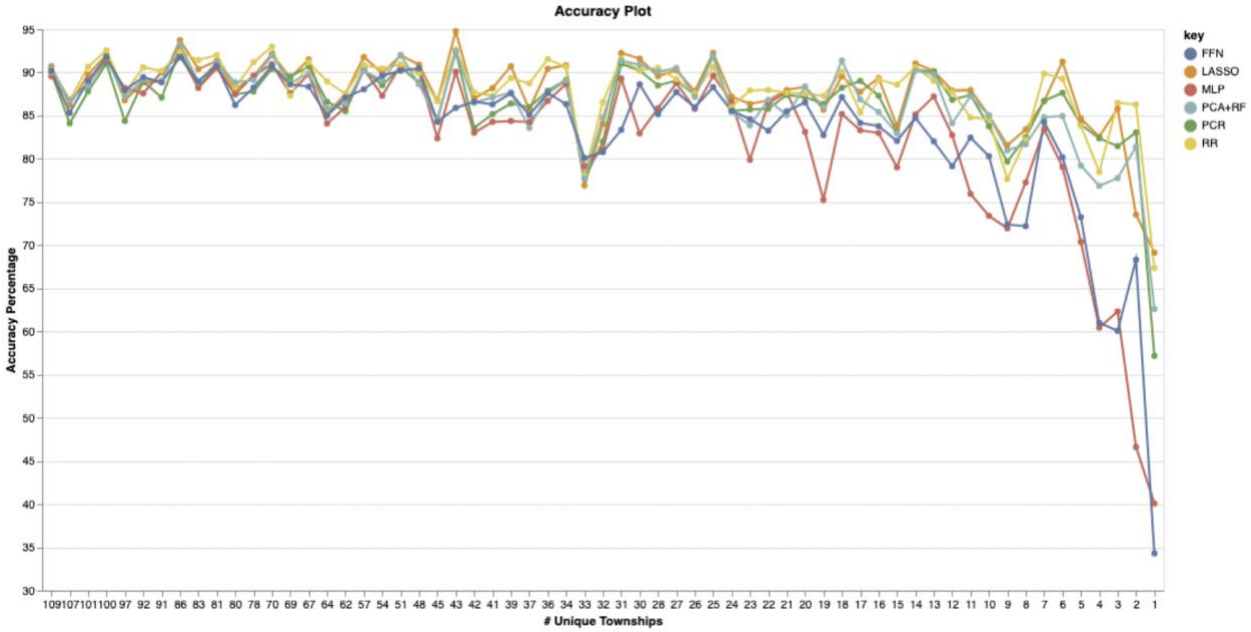

Figure 2: Accuracy Plot

When looking at the accuracy plot (Figure 2), the accuracies seem to be fairly close between all the models when the unique townships are the highest. It is clear to us that PCR performs the poorest during these early stages, while Ridge Regression performs the best. There are several situations where Ridge isn't the best performing model. This seems to occur when the unique number of townships seems to be around the 40s, where Lasso seems to have higher accuracy metrics. In some unique situations, PCR also seems to be the best performing model. This occurred when the number of unique townships was in its teens. PCA with Random Forest has a couple of moments where it shines. These occur when the number of unique townships is close to 50 as well as when it is close to 18.

But as the number of unique townships slowly decreases, there is a big drop off with the deep learning models, MLP and Feed Forward Neural Network, especially when the number of unique townships goes below the value of 20, while Ridge regression still maintains the best overall accuracy even while having fewer records to work with. But if we were to pick one of the deep

learning models for reference, it seems like we should choose the Feed Forward Neural Network since it seems to considerably outperform its peer, the MLP, in several situations.

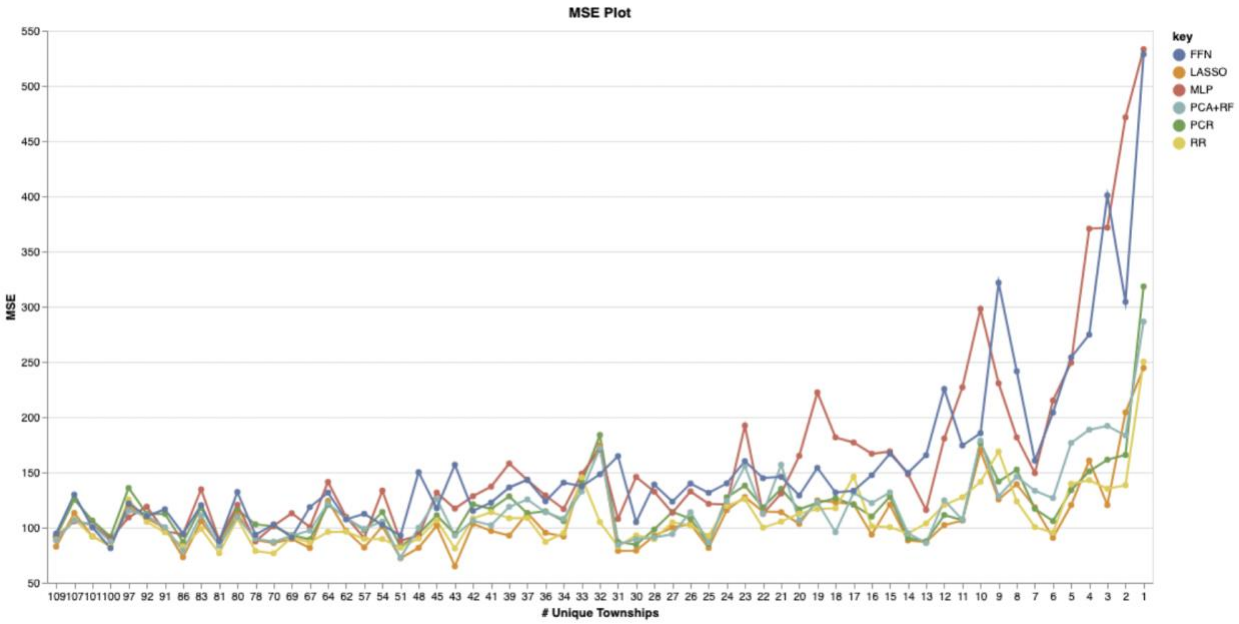

Figure 3: MSE Plot

The MSE plot also sheds a similar story to the accuracy plot in some ways (Figure 3). Most of the models have MSE values that are very similar to each other when we have a high number of unique townships. And this doesn't seem to pan out in the latter, where the deep learning models perform considerably worse than the machine learning models. With a high number of records, ridge regression seems to have the lowest MSE values, while FFN seems to have the highest values. With a low number of records, MLP seems to be the model with the highest MSE values.

In terms of having to pick the best overall model, we would have to settle on Ridge Regression. But in a similar vein to the accuracy plot, there are situations where it isn't the best model to choose. Again, we could notice that when the number of unique townships is in its 40s, Lasso seems to shine through with the lowest MSE values out of all the other models. The same case applies to PCR when the unique township values are in their teens.

A similar pattern is observed in the case where all the above performance metrics are being mapped to the eco district heat map via GIS software from the partner's end (Figure 4 and Figure 5). The accuracy is here in the central regions having more samples and vice versa with the Mean Squared Error. In the case of fringe regions, where there is less data, it's totally the opposite.

Below is one of the results for MSE and Accuracy from the Ridge Regression, which is performing the best overall:

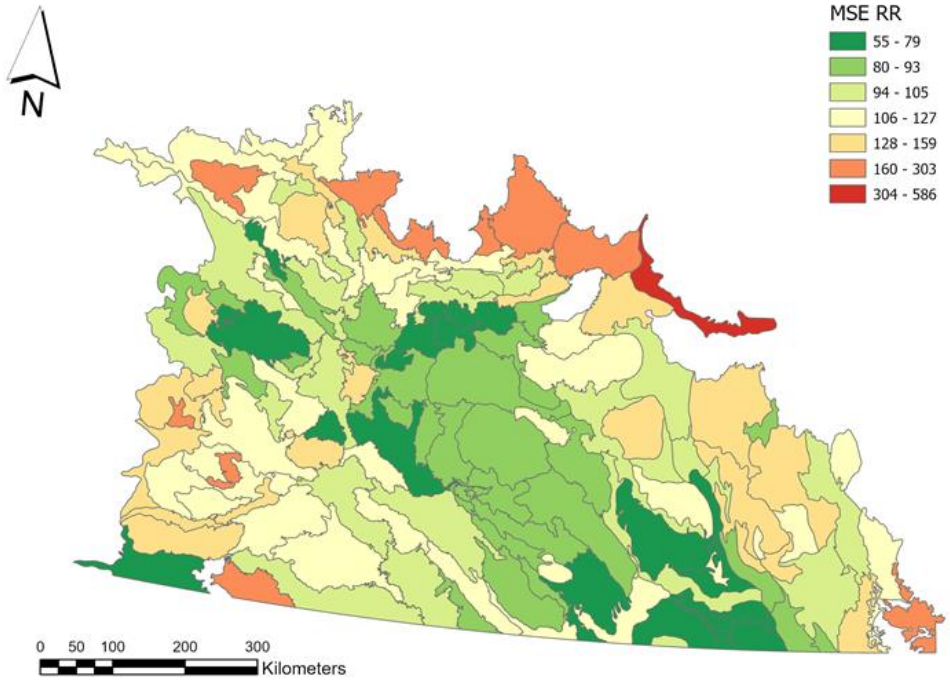

Figure 4: MSE Heatmap for Ridge Regression

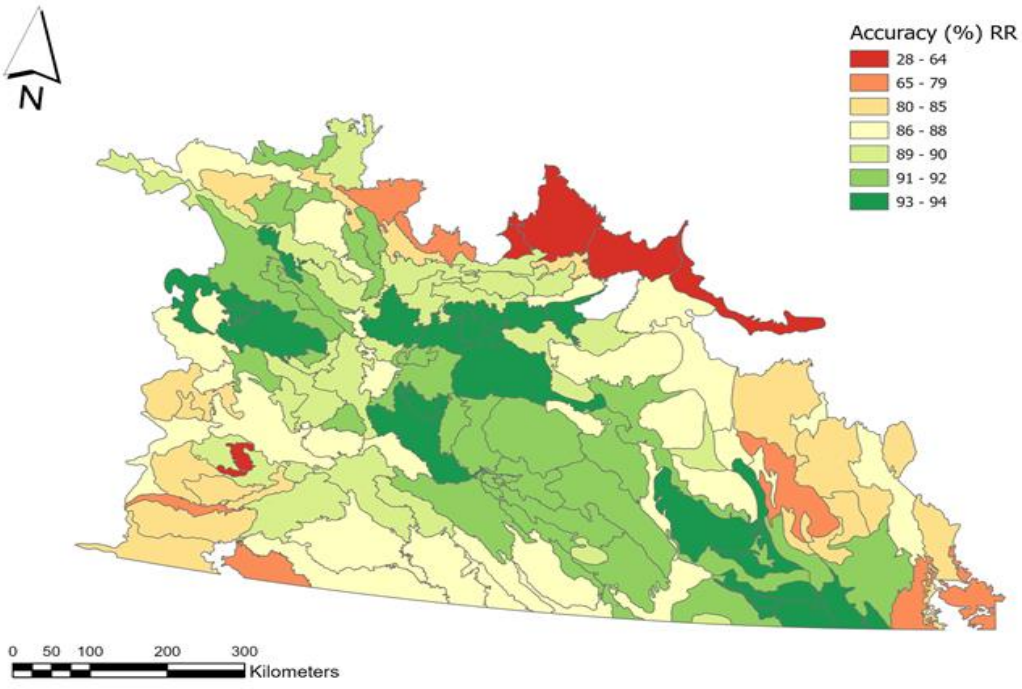

Figure 5: Accuracy Heatmap for Ridge Regression

In terms of the crop density, high density regions are also related to high accuracy models. Below are some of the canola spatial-temporal density maps (Figure 6). The regions in these figures having a good number of crop yields per area amounts to better accuracy as well in terms of fitting models and lower MSE as a result.

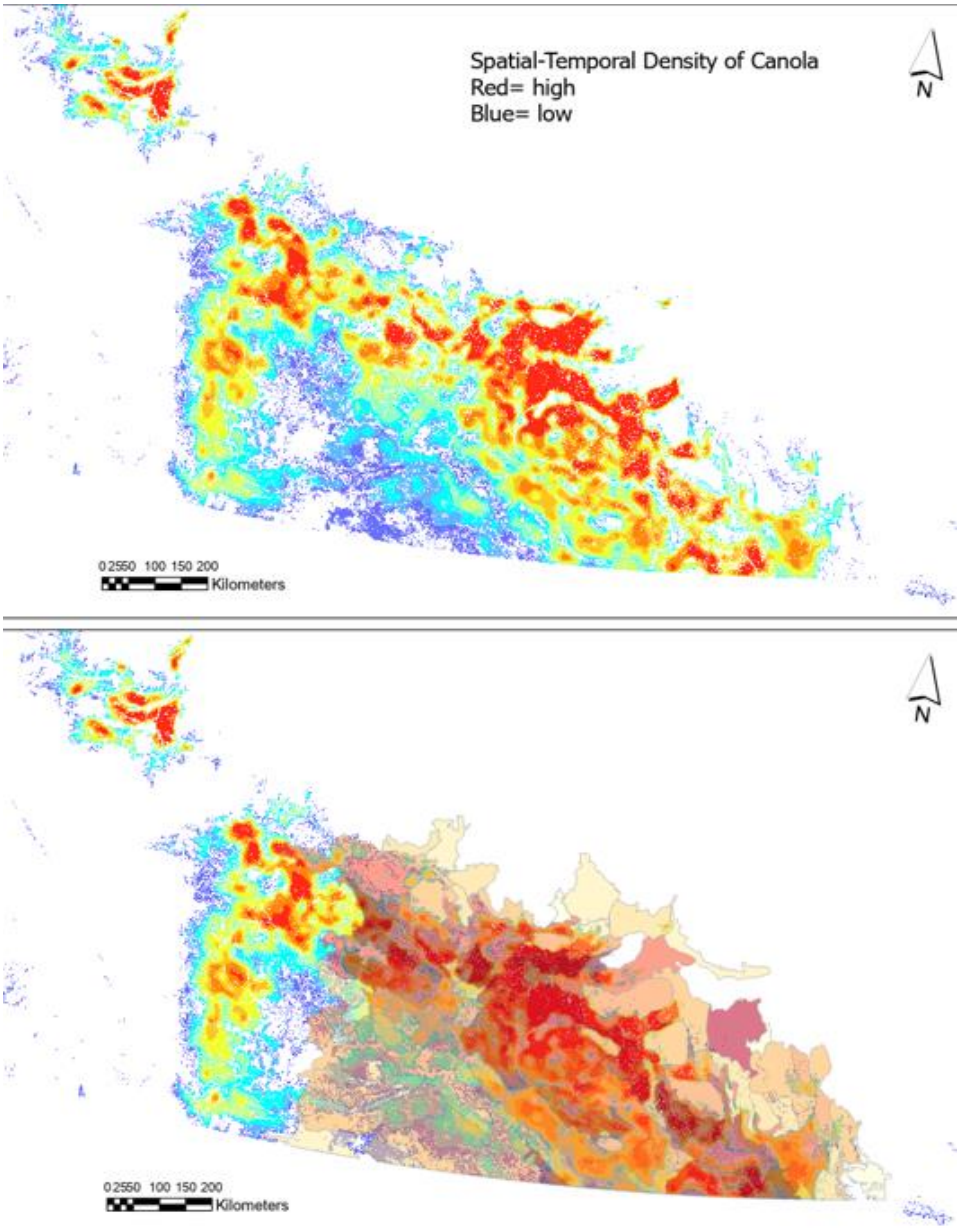

Figure 6: Crop Yield Density Heatmap for Different Eco Districts

INTERPRETATION:

There is a common trend that we notice when it comes to all the models for both the accuracy and MSE plots. This is because as the number of unique townships decreases, the accuracy starts going down while the MSE value starts going up. This is something that we should expect to see when it comes to modeling. But since we have employed both machine learning and deep learning models, the effectiveness of the models with a low number of records is different between those two techniques.

Machine learning seems to by far outperform deep learning methods when it comes to modeling with a low number of records. What this tells us is that our data is not particularly suited for deep learning models. Deep learning models tend to need an overly large amount of data in order to be able to perform at their best. We don't have that luxury with our datasets here. So, it is best that we stick to machine learning methods and pick the model for our specific situations based on the strengths of each model.

As far as the implications are concerned, smaller regions won't be able to predict crop yield with higher precision as compared to larger districts. But in the case of large or highly dense crop areas, there are many models, including neural networks, predicting the necessary output with the least errors. So overall, there is a need to have crop yields in the bigger eco districts.

CONCLUSION:

In conclusion, we believe that it is difficult to definitively say that there is one model that is best for all circumstances. But we are sure that it is between the machine learning methods. If we had to pick an overall best model, it would be Ridge Regression, but there are certain townships where the other models seem to outperform it. So it is in our best interest to mix and match our techniques in order for us to come up with the most accurate crop yield value prediction.

REFERENCES:

1. Hammer, G., Hansen, J., Phillips, J., Mjelde, J., Hill, H., Love, A., et al. (2001). Advances in application of climate prediction in agriculture. Agric. Syst. 70, 515–553. doi: 10.1016/S0308-521X(01)00058-0
2. McIntosh, P., Poo, M., Risbey, J., Lisson, S., and Rebbeck, M. (2007). Seasonal climate forecasts for agriculture: towards better understanding and value. Field Crops Res. 104, 130–138. doi: 10.1016/j.fcr.2007.03.019
3. Guo, W.W., Xue, H., 2012. An incorporative statistic and neural approach for crop yield modeling and forecasting. Neural Comput. Appl. 21, 109–117.
4. Qian, B., De Jong, R., Warren, R., Chipanshi, A., Hill, H., 2009. Statistical spring wheat yield forecasting for the Canadian Prairie provinces. Agric. Forest Meteorol. 149, 1022–1031.
5. Alvarez, R., 2009. Predicting average regional yield and production of wheat in the Argentine Pampas by an artificial neural network approach. Eur. J. Agron. 30, 70–77.
6. Prasad, A.K., Chai, L., Singh, R.P., Kafatos, M., 2006. Crop yield estimation model for Iowa using remote sensing and surface parameters. Int. J. Appl. Earth Observ. Geoinform. 8, 26–33.
7. Bolton, D.K., Friedl, M.A., 2013. Forecasting crop yield using remotely sensed vegetation indices and crop phenology metrics. Agric. Forest Meteorol. 173, 74–84.
8. Mkhabela, M.S., Bullock, P., Raj, S., Wang, S., Yang, Y., 2011. Crop yield forecasting on the Canadian Prairies using MODIS NDVI data. Agric. Forest Meteorol. 151, 385–393.
9. Machine Learning Mastery. (2020). *How to Handle Big-p, Little-n (p >> n) in Machine Learning*: https://machinelearningmastery.com/how-to-handle-big-p-little-n-p-n-in-machine-learning
10. Analytics Vidhya. (2016). *A Complete Tutorial on Ridge and Lasso Regression in Python*: https://www.analyticsvidhya.com/blog/2016/01/ridge-lasso-regression-python-complete-tutorial/
11. Deep AI. *Feed Forward Neural Network*: https://deepai.org/machine-learning-glossary-and-terms/feed-forward-neural-network

12. Towards Data Science. (2018). *Adam — latest trends in deep learning optimization*: https://towardsdatascience.com/adam-latest-trends-in-deep-learning-optimization-6be9a291375c
13. Enjoy algorithms. *How to Handle Big-p, Little-n (p >> n) in Machine Learning*: https://www.enjoyalgorithms.com/blog/evaluation-metrics-regression-models
14. Kirenz. (2021). *Lasso Regression with Python*: https://www.kirenz.com/post/2019-08-12-python-lasso-regression-auto/
15. Machine Learning Mastery. (July 24, 2019). Deep Learning in Python with Keras: https://machinelearningmastery.com/tutorial-first-neural-network-python-keras/
16. Machine Learning Mastery. (October 11, 2020). Ridge Regression Models in Python: https://machinelearningmastery.com/ridge-regression-with-python/
17. Zach. (Nov 16,2020). Principal Components Regression in Python (Step-by-Step): https://www.statology.org/principal-components-regression-in-python/
18. Kenneth Leung. (April 6,2022). Principal Component Regression: Clearly Explained and Implemented: https://towardsdatascience.com/principal-component-regression-clearly-explained-and-implemented-608471530a2f
19. Jake VanderPlas. In Depth: Principal Component Analysis: https://jakevdp.github.io/PythonDataScienceHandbook/05.09-principal-component-analysis.html
20. Michael Galarnyk. (Dec 4,2017). PCA using Python (scikit-learn): https://towardsdatascience.com/pca-using-python-scikit-learn-e653f8989e60
21. Usman Malik. Principal Component Analysis (PCA) in Python using Scikit-Learn: https://stackabuse.com/implementing-pca-in-python-with-scikit-learn/
22. Zach. (Sept 18,2021). How to Create a Scree Plot in Python (Step-by-Step): https://www.statology.org/scree-plot-python/
23. Renesh Bedre. Principal component analysis (PCA) and visualization using Python (Detailed guide with example): https://www.reneshbedre.com/blog/principal-component-analysis.html
24. Thomas J.Fan. (Feb 19,2022). Multi-layer Perceptron Regressor https://scikit-learn.org/stable/modules/generated/sklearn.neural_network.MLPRegressor.html
25. Neural networks models (Supervised) https://scikit-learn.org/stable/modules/neural_networks_supervised.html

26. Kaushik Choudhury. (Aug 31,2020). Deep Neural Multilayer Perceptron (MLP) with Scikit-learn https://towardsdatascience.com/deep-neural-multilayer-perceptron-mlp-with-scikit-learn-2698e77155e
27. (May 31,2022). How to use MLP Classifier and Regressor in Python? https://www.projectpro.io/recipes/use-mlp-classifier-and-regressor-in-python
28. Mejbah Ahammad. Building a Regression Multi-Layer Perceptron (MLP) https://towardsdatascience.com/deep-neural-multilayer-perceptron-mlp-with-scikit-learn-2698e77155e

APPENDIX:

Usage Document - Exploring optimal machine learning models for predicting crop yield at township level:

Folder Structure:

Below mentioned are the important folders and the corresponding files contained within them:

1. **Data:**
    a. 'EDA_AAFC.ipynb' - Contains the code for data wrangling i.e., the steps performed to obtain the final dataset. It includes data exploration, data cleaning, data filtering, data transformation, and the final joining of all the four datasets.
    b. 'Model_Evaluation.xlsx' - Contains the evaluation metrics for all the six models corresponding to each Eco District
    c. 'Accuracy and MSE Plots.ipynb' – Contains Python visualizations of 'Count of Unique Townships' plotted against 'Accuracy' and 'Test MSE' respectively

2. **Models:**

The folder *'Models'* contains the following subfolders corresponding to the respective algorithms:

a. FF - *Feed Forward Neural Network*
b. LASSO - *Least Absolute Shrinkage and Selection Operator*
c. MLP - *Multi Layer Perceptron*
d. PCR - *Principal Component Regression*
e. PCA + RF - *Principal component Analysis* + *Random Forest*
f. RR - *Ridge Regression*

Each folder contains:

a. 'aafc_data.csv': The dataset obtained post data wrangling. This serves as the training data for our model.
b. 'scoring_test_df.csv': New data set (not the train/test data) on which prediction is to be done
c. 'test_script.py': The .py script which trains the respective model and performs the following functions:

i. validation_metrics(): Displays Mean Squared Error Train, Mean Squared Error Test, Mean Absolute Error and Accuracy of the fitted model
ii. predicted_train_dataset(): Displays the train dataset along with the predicted yield values
iii. predicted_test_dataset(): Displays the test dataset along with the predicted yield values
iv. feature_importance(): Displays the features used in the fitted model and the respective importance
    Note: This function is present only in LASSO and Ridge Regression
v. number_principal_components(): Number of principal components used for fitting the model
    Note: This function is present only in PCR and PCA + RF
vi. cumulative_explained_variance(): Displays what %of variance of the train data set is explained by the chosen principal components
    Note: This function is present only in PCR and PCA + RF

vii. score(): Outputs a data frame containing predicted yield for a new data set - *'scoring_test_df.csv'*

d. A subfolder of the template 'folder/folder_aafc' which contains a .py script, which is the base script that contains code for each model and the above-described functions.
    i. PCR - *Models/PCR/pcr_aafc/PCR.py*
    ii. RR - *Models/RR/ridge_regression_aafc/RidgeRegression.py*
    iii. PCA_RF - *Models/PCA_RF/pca_rf_aafc/PCA_RF.py*
    iv. MLP - *Models/MLP/mlp_aafc/MLP.py*
    v. LASSO - *Models/LASSO/lasso_aafc/LASSO.py*
    vi. FF - *Models/FF/feed_forward_aafc/FeedForwardNN.py*

    Note: We have also included the crude .ipynb files based on which we made the final .py scripts in the same location.
    The .ipynb files are just for reference and do not contain thorough documentation. Please refer to .py files for final scripts.

e. 'Outputs' folder that contains:
    i. 'train_predicted_df.csv' - Train dataset along with the predicted yield values

ii. ‘test_predicted_df.csv’ – Test dataset along with the predicted yield values
iii. ‘new_data_predicted_df.csv’ - Predicted yield for the scoring data set
iv. ‘features_importance_df.csv’ - Used for giving important features post executing Ridge and Lasso regularization models

3. **docs:**
    a. ‘final-report/ Final Report (AAFC).pdf’ – Contains the final report containing end-to-end details of the entire project
    b. ‘proposal/ AAFC_Proposal.pdf’ – Contains the project proposal that was initially approved
    c. ‘final-presentation/Final_Presentation_AAFC.pptx’ – Contains the final power point deck for the project

Execution:

Below are the steps to be followed to run the Ridge Regression script (Similar steps are to be taken to run the rest of the 5 models):

1. **Checks:**
    a. Install necessary libraries: pandas, numpy, tensorflow, keras, sklearn, altair
    b. Navigate to Models/RR/ and check if the ‘aafc_data.csv’ file is the latest and final wrangled dataset – Replace it with a new training dataset if required
    c. Check if ‘scoring_test_df.csv’ is updated and is a new dataset for which you want to predict the yield
2. On terminal, navigate to Models/RR/ and run the command *‘python test_script.py’*
3. Enter a valid Ecodistrict ID
   For example: ‘748’
4. It prints the validation metrics along with generating the following outputs in the ‘Outputs’ folder:
    a. train_predicted_df.csv
    b. test_predicted_df.csv
    c. new_data_predicted_df.csv
    d. features_importance_df.csv (Applicable for Ridge and Lasso only)