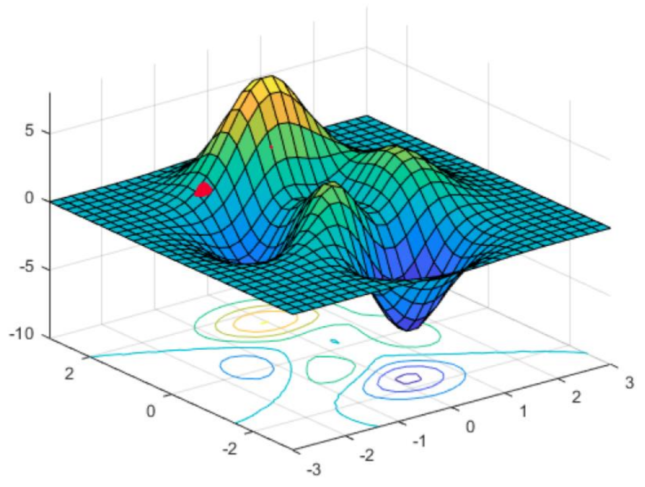

Developing Machine Learning Models for Crop Yield Prediction

Project Report

ABSTRACT

Multiple machine learning models including linear regression, random forest, gaussian process, and k nearest neighbor were developed from scratch. These developed models were then applied to weather data to predict the crop yield. The necessary feature engineering and hyperparameter optimization was performed to improve and finalize the performance of the models. This report contains a discussion of the implementation and results of the models detailed in a series of questions for each section.

Turner

CM50237: Machine Learning 1

Table of Contents

List of Figures

1. What exploration of the data set was conducted?

The basic summary statistics of the dataset were calculated for each feature and the target variable, this included the mean, standard deviation, and quartile ranges. The distribution of the target variable (crop yield) was then plotted with a line separating values over 1 standard deviation above the mean. This can be seen in Figure 1.

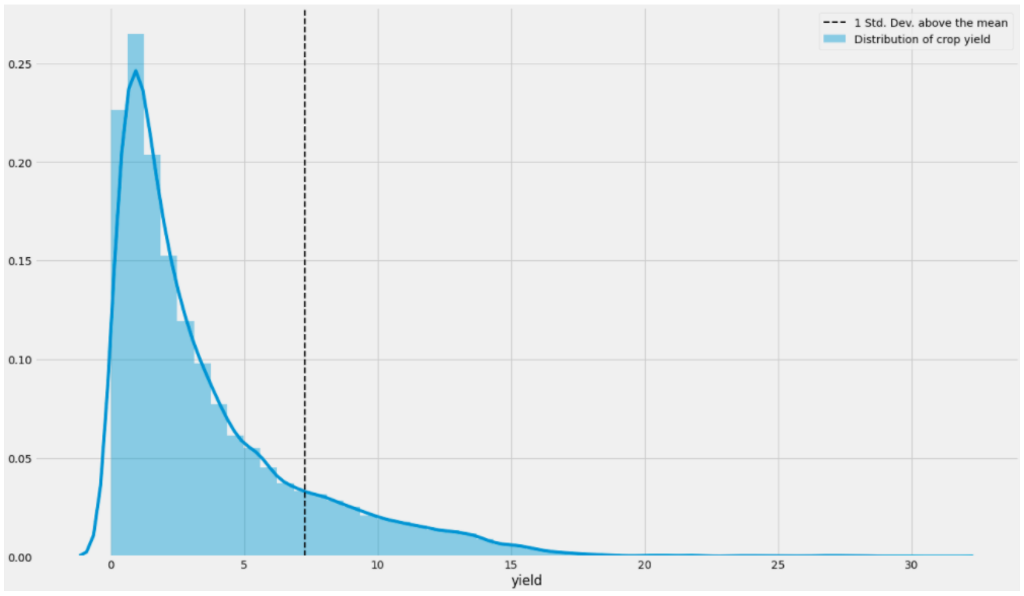

Figure 1. Distribution of crop yield.

First the separability of the target variable was investigated between the winter and summer months with respect to the yield. The target variable was separated into categories, 1 standard deviation was defined as 'High Yield', and below 'Low Yield'. The separation between the density of these categories can be seen for max temperature and rainfall in Figures 2 and 3, respectively.

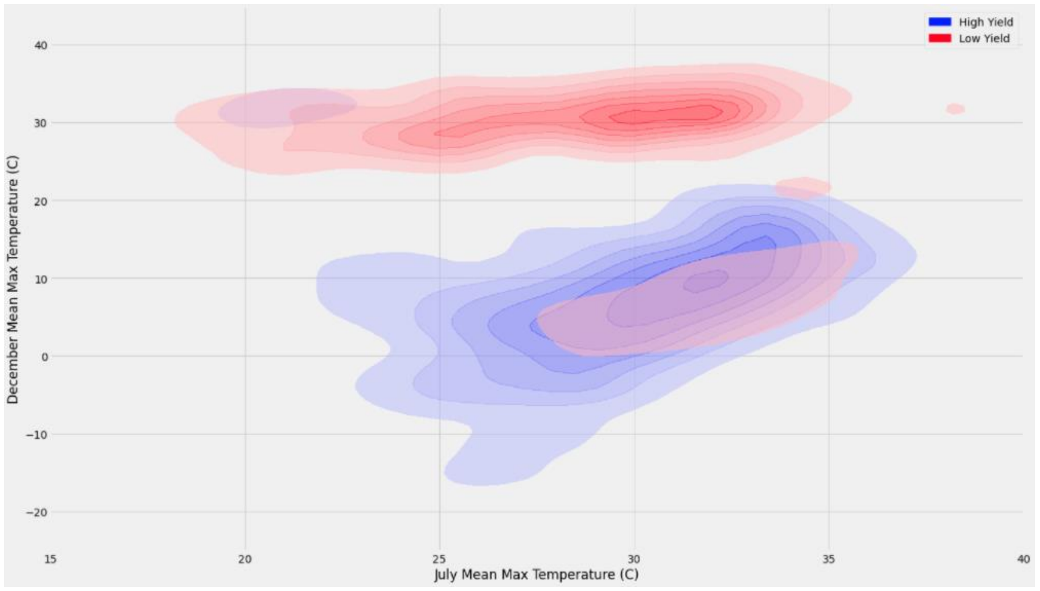

Figure 2. Density of high yield and low yield for temperature in December and July.

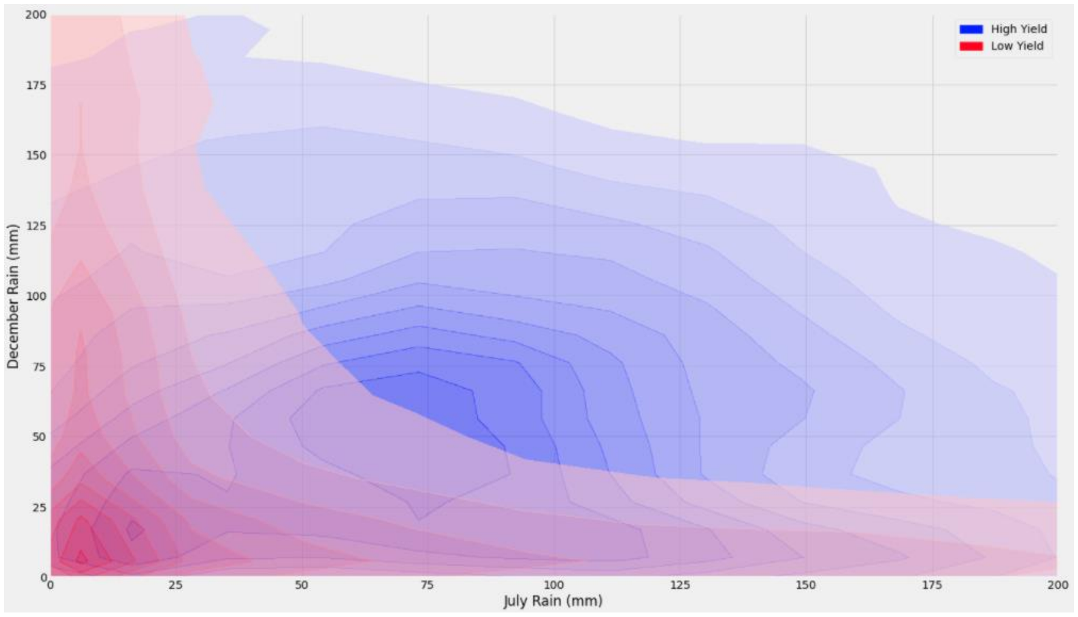

Figure 3. Density of high yield and low yield for rainfall in December and July.

Next a correlation matrix was used to visualize the impact that the features may have on the target variable, as well as the correlation and similarity between the features. This correlation matrix is shown in Figure 4. One can see from Figures 2 and 3 that the target variable is separable between high and low crop yield for December and July.

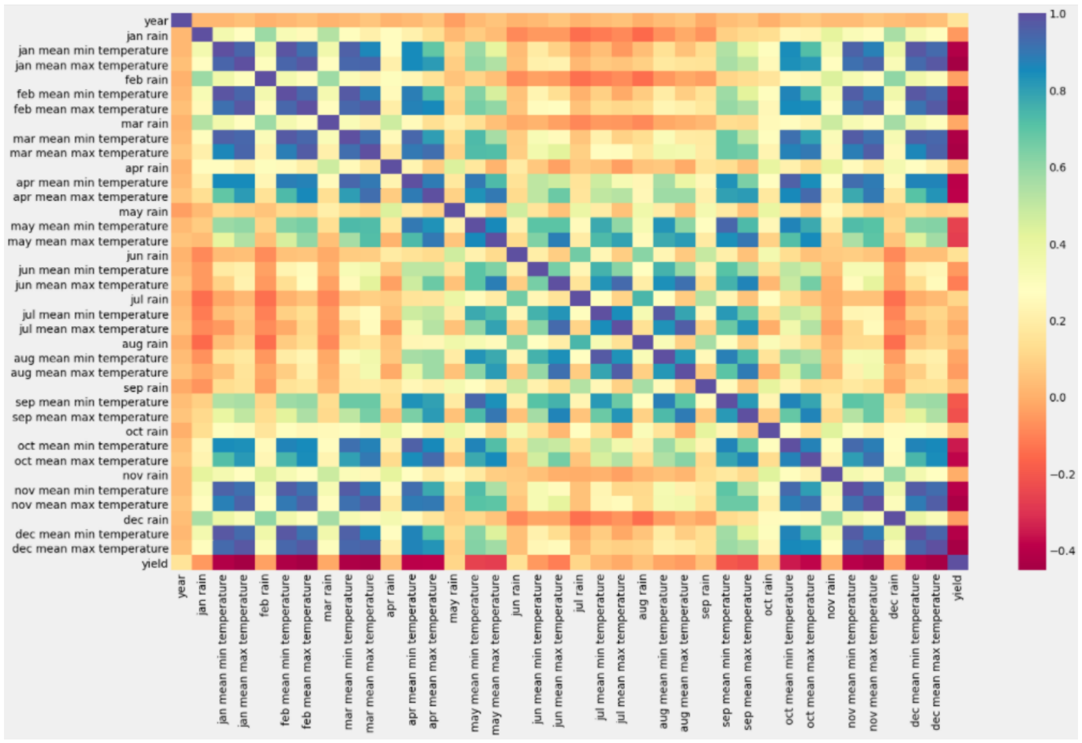

Figure 4. Correlation matrix for all features and target variable.

The average yield for each year was also plotted and linear regression was performed on this data, shown in Figure 5. The consistent growth in average crop yield is likely due to advances in farming techniques. This is good reason to either use the year as a feature for learning and predicting the target variable or normalizing the target variable according to this growth so to isolate the effect of the climate on the crop yield. I chose the former, though I did some initial testing for yield normalization as well.

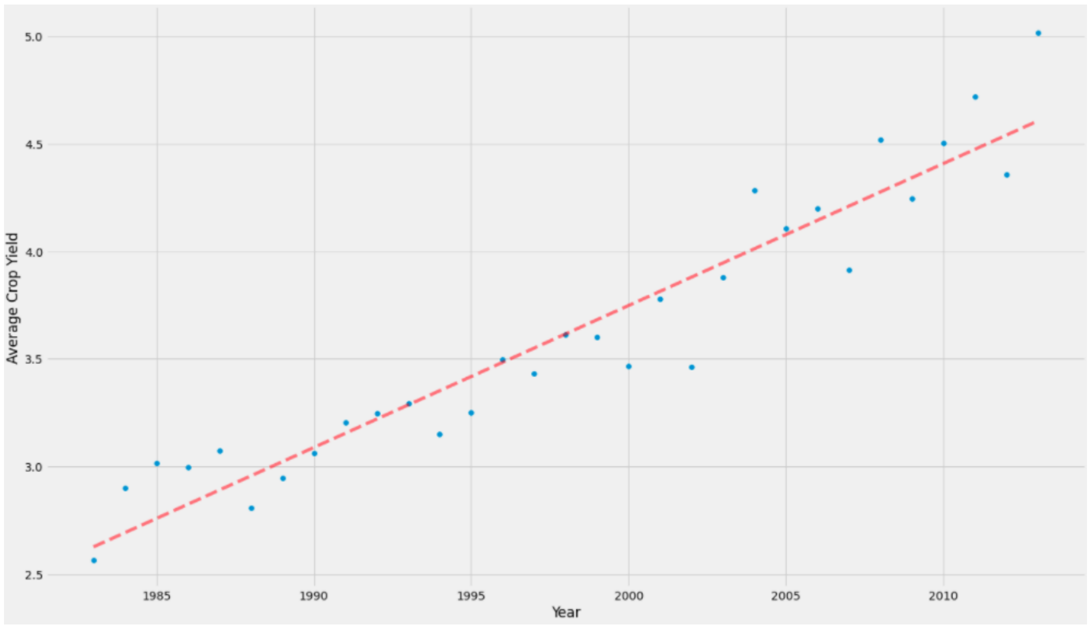

Figure 5. Linear regression on the average crop yield over time.

2. How was the dataset prepared?

The data was prepared consistently between models before an algorithm was applied. The rows of the dataset were shuffled so as to treat each row as one exemplar. The dataset was split by a custom function into train, test, and validation sets with ratios of 60%, 20%, and 20%, respectively. These train, test, and validation sets were then normalized so that the mean data had a mean of 0 and standard deviation of 1. The target variable was then split from each dataset so that there was an input (x) and output (y) set. Each model was then trained on the training set, hyperparameter optimization was performed on the test set, and final scoring on the validation set. The predicted values of the algorithms had to be unnormalized to match the ranges of the original data for comparison.

3. How does a regression forest work?

3.1 Decision Tree

A decision tree works by iterating over the features of the data to construct a tree of decisions ending with leaf nodes that make classification or regression predictions. The 'impurity' of each feature is measured to assess its importance in predicting the target variable. Each split is constructed my choosing the feature that minimizes the impurity. The impurity of each split is measured according to many metrics, the most common of which include the Gini impurity, entropy, information gain, variance, and others. The decision tree will stop splitting according to features when the impurity from a split is zero or is greater than the impurity of the previous split in the branch. A common hyperparameter used for decision trees is the maximum depth of a tree, which is the last condition for creating a leaf node and ending the feature iteration process.

3.2 Random Forest

A random forest merges the decision tree algorithm with ensemble machine learning which often creates more reliable and accurate predictions. An ensemble method involves viewing many predictions, effectively combining many estimators into one cohesive value or final prediction value. Essentially, a random forest creates multiple decisions trees, the number of which is specified according to the 'number of estimators' hyperparameter. The final leaf nodes of each tree are combined, usually by taking the average, to create the final regression predictions of the model.

4. How does a Gaussian process work?

A gaussian process works by creating a normal probability distribution over the possible functions that fit the set of points in the training data. A covariance matrix, which is calculated from the training data, is used along with a mean function as the parameters for a normal distribution. The expected value of this normal (gaussian) distribution of possible functions is a gaussian process. First an arbitrary prior distribution is defined by a mean function, this mean function can usually be set to zero. A covariance matrix is derived from the kernel function applied to the training data. This is used to measure the similarity of data points. The kernel function is typically a proxy for the similarity by calculating distance between data points. The prior distribution is ultimately converted to a posterior distribution after considering the data. This is the new information or evidence which updates the prior to become the posterior.

5. Which additional algorithm did you choose and why?

K Nearest Neighbor (KNN) was the additional algorithm chosen to predict the crop yield. It was chosen because this problem has relatively little features (approx. 37) compared to the thousands of features often present in many other machine learning problems. KNN is typically very slow when used on high dimension problems and loses accuracy quickly with increases in dimensionality (Pestov 2013). In addition, KNN is very simple to implement and easy to understand how the model is working and arrives at its conclusions. The KNN algorithm also has only one hyperparameter and is therefore less prone to overfitting than other ML models.

6. What are the pros and cons of the algorithms?

A brief pros and cons list for each algorithm is provided below, the pros and cons of each are in relative terms to the other algorithms used.

Linear Regression

Pros

- Simple, intuitive, and easily explainable
- Few hyperparameters
- Resistant to overfitting

Cons

- Limited to fit linear data
- Typically, low in precision, prone to underfitting

Random Forest

Pros

- Resistant to the curse of dimensionality i.e. can handle lots of features
- Ensemble method typically has higher accuracy with many estimations.
- Highest accuracy after KNN for this task

Cons

- Operates as ‘black-box’, difficult to understand
- Has the longest training time and complexity of the algorithms.

Gaussian Process

Pros

- Usually, high accuracy
- Good at fitting non-linear data
- Incorporates uncertainty in results

Cons

- Difficult to understand model process
- Difficult to explain conclusions

K Nearest Neighbor

Pros

- Simple to implement
- Only one hyperparameter
- Easily explainable and intuitive
- Highest accuracy for this task

Cons

- More affected by the curse of dimensionality
- Requires similarly scaled features i.e. normalization
- Sensitive to outlier data points because it is simply using distance between points

7. Describe the toy problem used to validate the algorithms, and explain its design?

The toy problem designed is a one-dimensional polynomial function with randomized coefficients and added random noise. This allows for easy visualization of the model learning the function as well as noise to simulate random variables that are present in most real-world problems. This also allows to visualize how prone a model is to overfitting by observing the ML algorithms tendency to model the noise against the underlying function. Figure 6 shows an example prediction on the toy problem by the K-nearest neighbor (KNN) algorithm. The solid blue line shows the underlying function that created the input data, the scattered data is created using this function plus randomized noise to simulate a real-world problem. The closer the predicted values from the KNN model are to the solid blue line, the less overfit the model results as the model is more closely representing the underlying function and less likely to model the noise in the training data.

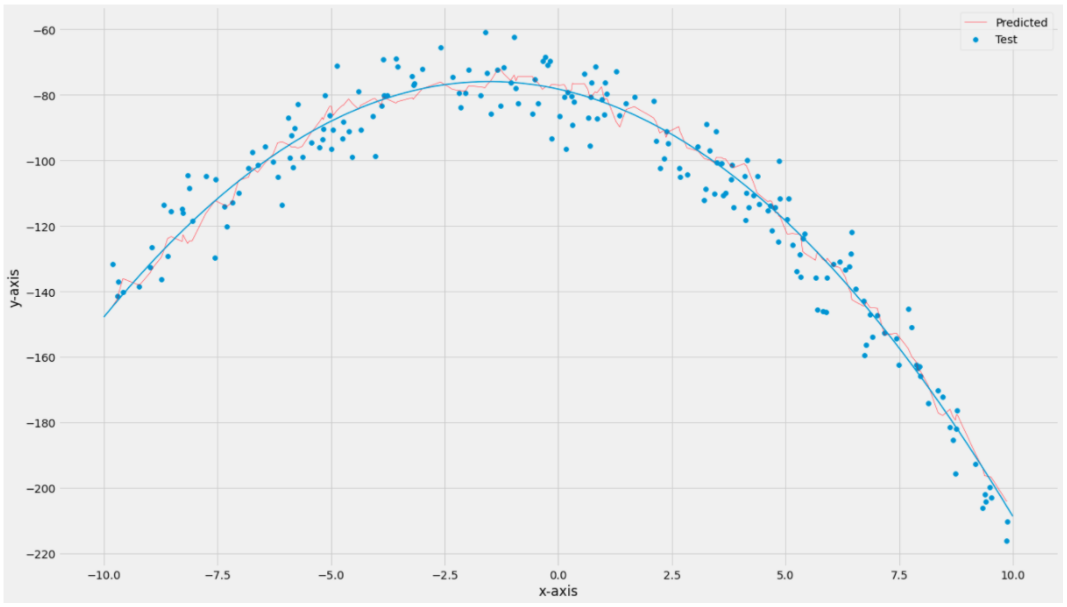

Figure 6. K Nearest Neighbor algorithm run on the toy problem, showing the effect of noise on the model output.

8. What evidence of correct, or incorrect, implementation did the toy problem provide?

Utilizing the toy problem was helpful to test the implementation of each ML algorithm. Plotting the predicted values from the model with the noisy input data and the underlying function showed that each model was versatile and that the model was able to fit a variety of randomly generated functions. Another toy problem example can be seen below for the linear regression model in Figure 7. Despite this model tending to underfit non-linear data, the toy problem helped identify incorrect implementations during testing of the algorithm. I was able to use the testing on the toy problem for debugging of the gradient descent function for linear regression. It was helpful because I could easily visualize that the predicted values were failing to match a negatively sloped function, while the model seemed to match positively sloped functions.

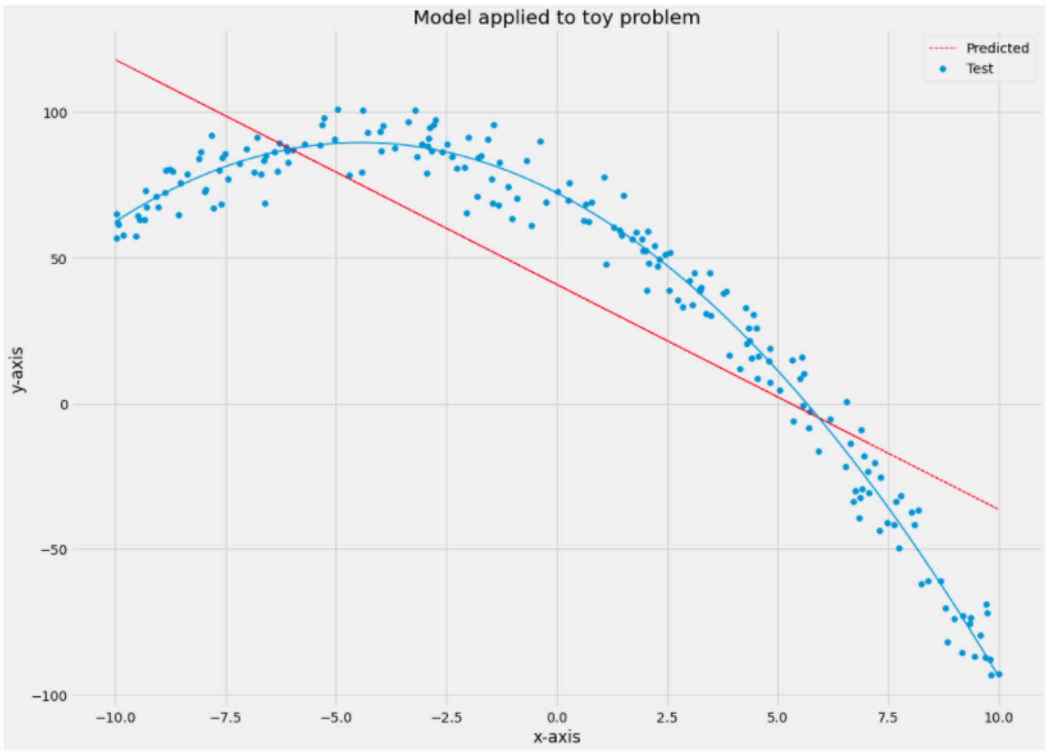

Figure 7. Linear regression model applied to the arbitrary toy problem function with noisy data.

9. How were the hyperparameters optimised?

A grid search method was used to find the optimal hyperparameters for each ML algorithm. Grid search works by testing a specified discrete space for each parameter, every possible combination was tested, and the results were recorded in a dictionary. This method was chosen because it's simple to implement, collects consistent data for diagnostics and exploration, the range of likely optimal parameters is known and limited, and the algorithms deployed only had one or two hyperparameters each.

A portion of the results for the grid search for linear regression and the gaussian process can be seen below in Figures 8 and 9, respectively. More results depicting the optimization space can be found in the appendix.

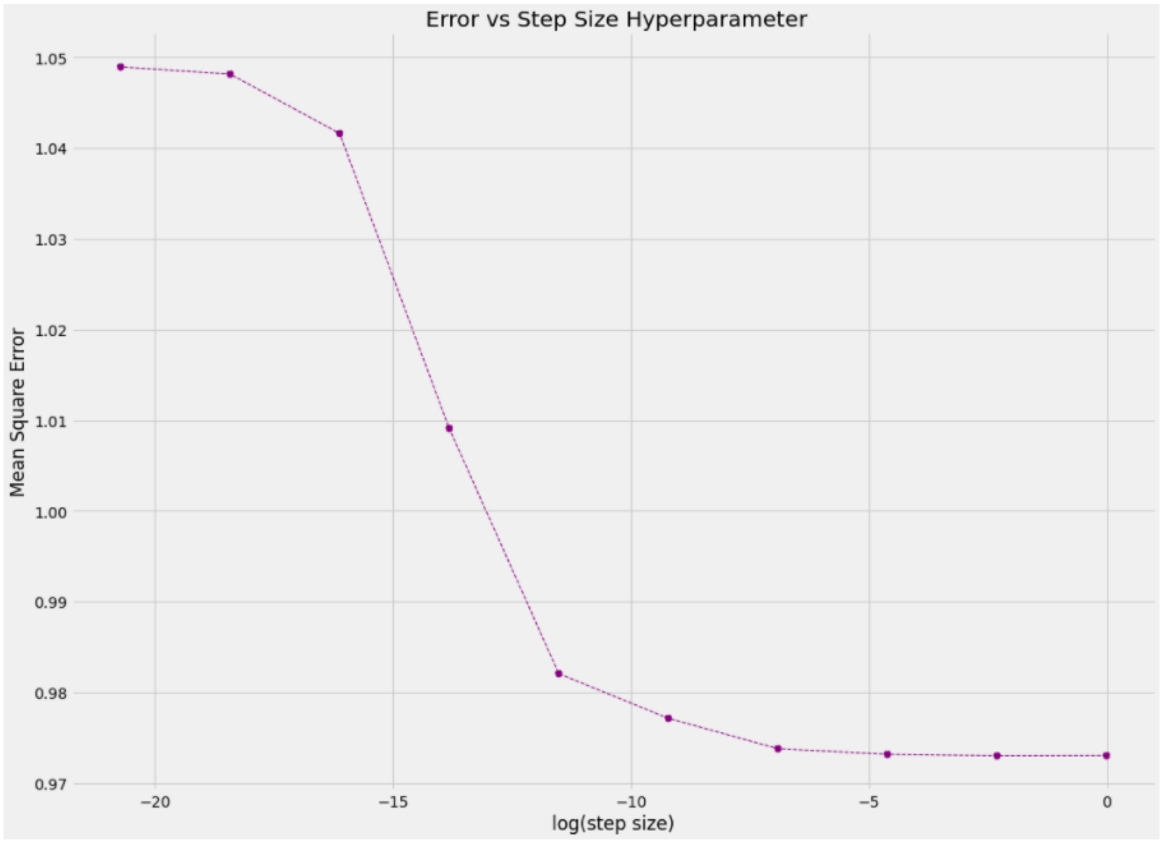

Figure 8. Error vs log of the step size hyperparameter for linear regression.

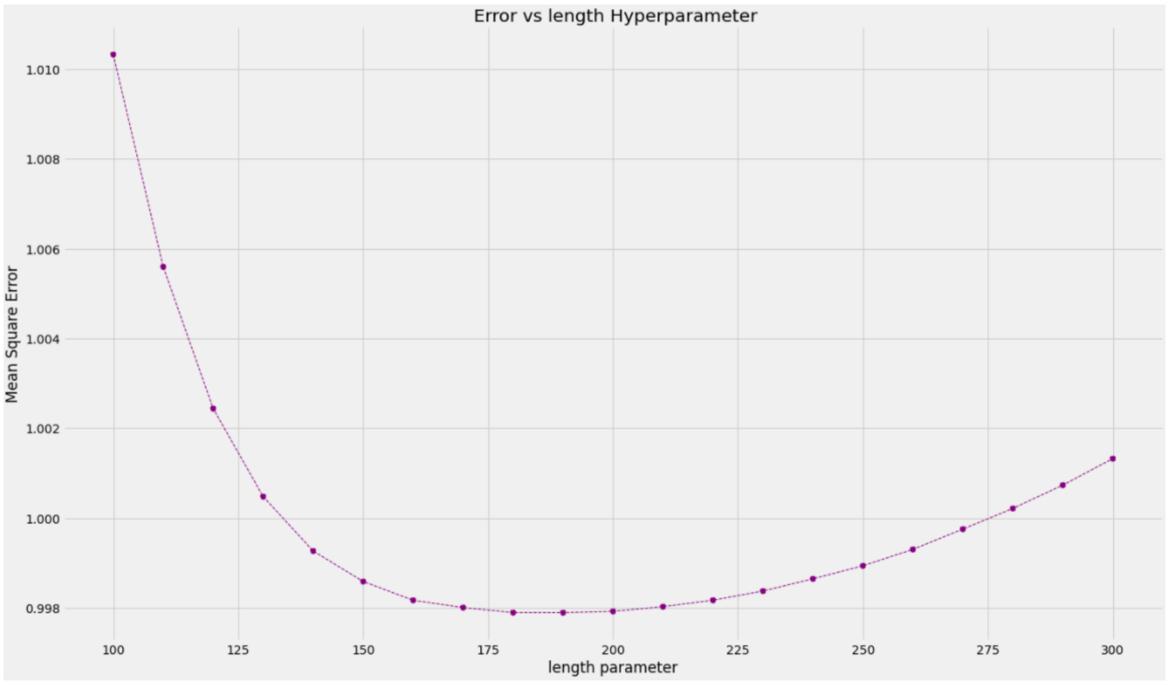

Figure 9. Mean square error vs length parameter for gaussian process regression.

To avoid overfitting, the hyperparameters were optimized using each model on the test set and the results were deployed using the validation dataset, this prevented overtraining on one dataset. The coefficient of determination for the final optimized model can then be compared when run on the training set and the validation set. A value significantly higher for the training set than the validation set would indicate the model is likely overfit to the training data.

10. What results are obtained by the algorithms?

The results for each algorithm are shown below in Table 1. Each metric shown in the table is derived from the best test results obtained for an optimized model on the validation dataset. As one can see from the resulting metrics, K Nearest Neighbor had the highest regression performance on predicting the crop yield based on the input features.

*Table 1. Metrics indicating model results for each algorithm. * 'correct' in this context is defined as a prediction within 2 of the true value.*

Algorithm	**Coefficient of determination**	**Mean Square Error**	**Mean Absolute Error**	**Probability of 'correct'* prediction**
Linear Regression	0.56	4.79	2.20	0.60
Gaussian Process	0.72	3.47	1.78	0.70
Random Forest	0.78	2.35	1.51	0.74
K Nearest Neighbor	0.82	2.16	1.34	0.79

11. How fast do the algorithms run and how fast could they run?

The run time for each optimized model along with the O-complexity is shown below in table 2. In addition, the change in run time with changes in hyperparameters for each of the algorithms was also explored. This exploration can be seen in Figures 10 – 16 below.

Table 2. Run time and big O complexity of each ML algorithm developed.

Algorithm	**Optimal Run time (seconds)**	**O-Complexity**
Linear Regression	3.50	$O(m * n)$
Gaussian Process	184.30	$O(n^3)$
Random Forest	1742.4	$O(k * n * \log(n))$
K Nearest Neighbor	7.45	$O(k * n)$

K Nearest Neighbor Optimization

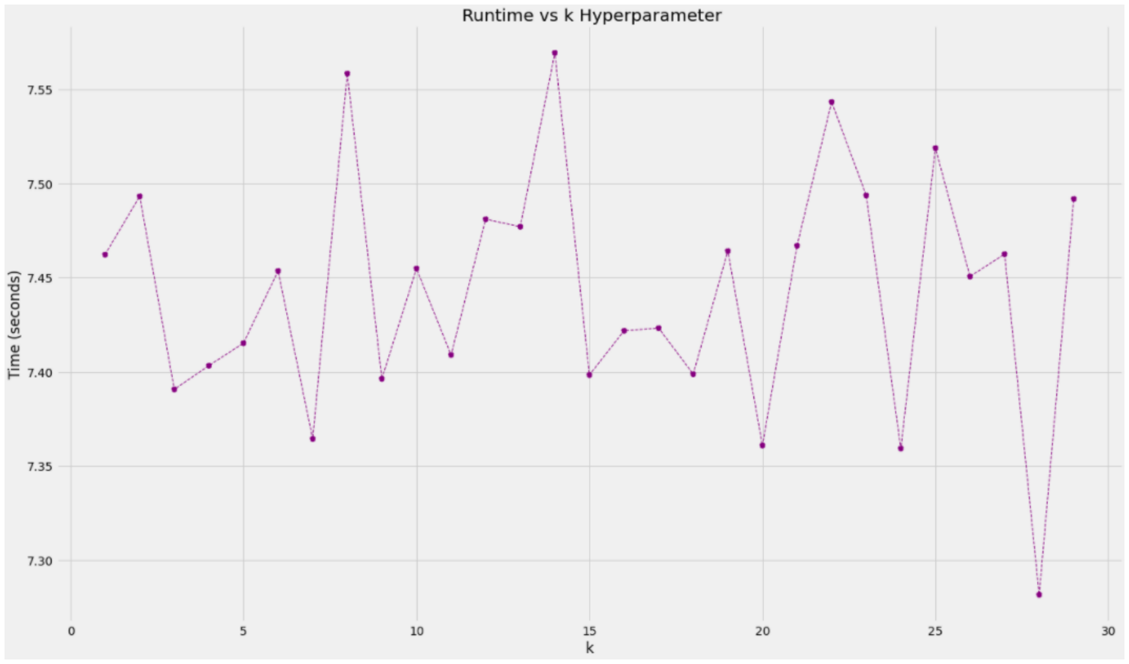

Figure 10. Run time relationship between k hyperparameter for KNN.

Linear Regression Optimization

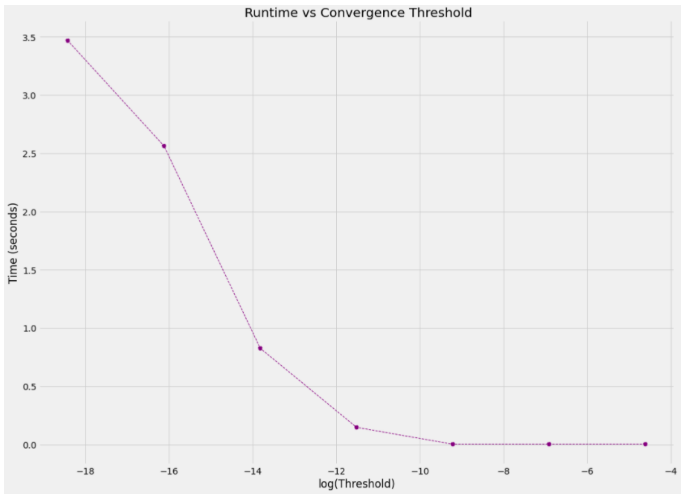

Figure 11. Run time for convergence threshold for linear regression.

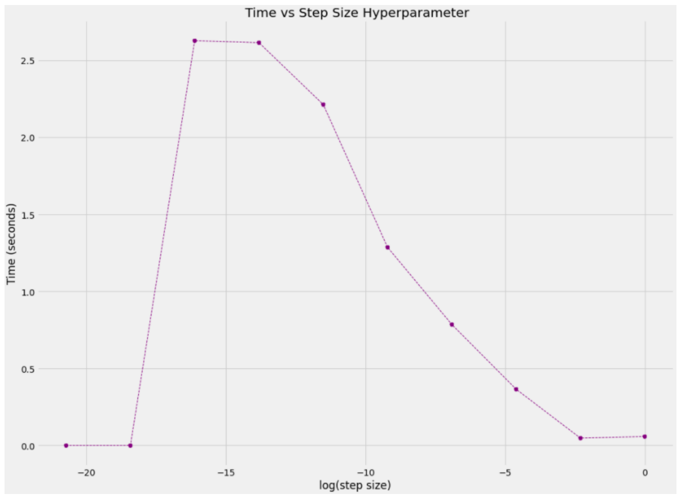

Figure 12. Run time of relationship between convergence threshold and log of the step size for linear regression.

Gaussian Process Optimization

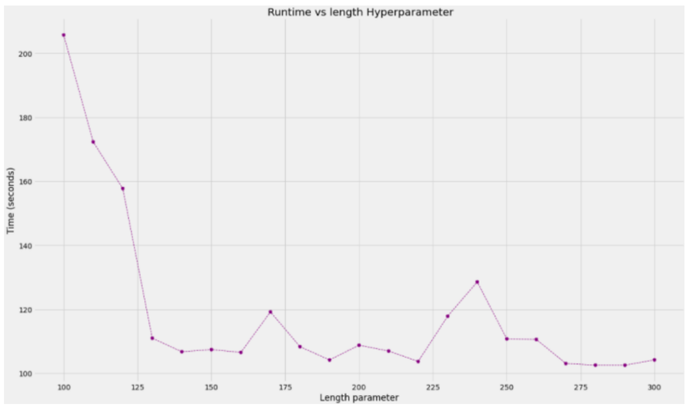

Figure 13. Change in run time with change in length hyperparameter for gaussian process.

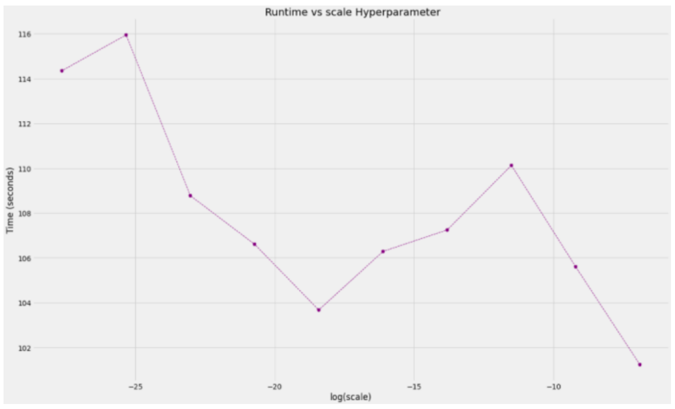

Figure 14. Run time for gaussian process with changes in the scale hyperparameter.

Random Forest Optimization

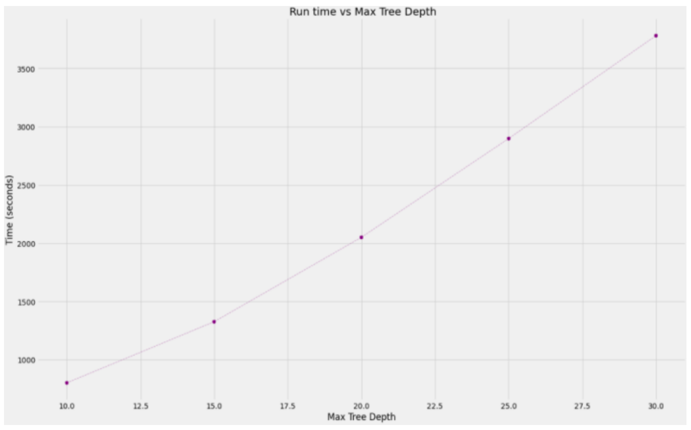

Figure 15. Run time of random forest algorithm with variations in tree depth.

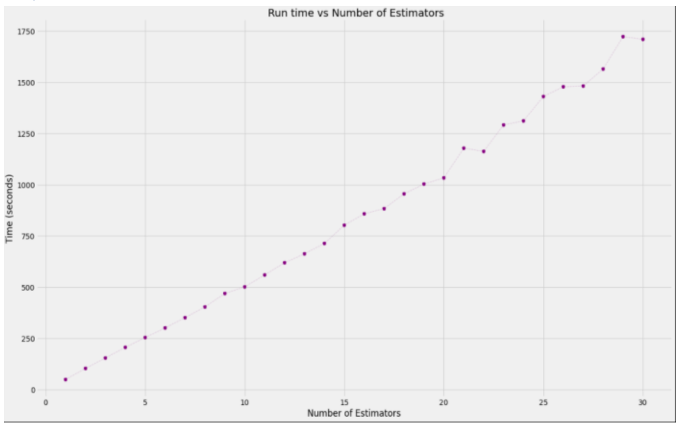

Figure 16. Increase in run time of random forest with increase in number of estimators used.

12. Which algorithm would you deploy and why?

Choosing between the algorithms developed and tested on the dataset I would deploy K Nearest Neighbors to predict the crop yield from the climate data. This algorithm had the highest precision when comparing its' predictions to the actual yield values. This model also had the shortest training time and is computationally inexpensive to run. This would mean it could be run often if necessary. In addition, KNN is relatively easy to implement, simple to understand, and only has one hyperparameter making it easier to optimize for industrial use. Lastly, on the dataset provided, KNN tended to overfit less when compared to random forest and gaussian process algorithms. This was measured by comparing the optimized results on the validation set with optimized results on the training set. A higher performance on the training set when compared to the validation set is a result of overfitting to the data.

13. How could the best algorithm be improved further?

The KNN algorithm could be converted to a weighted KNN using a kernel function. The idea is simple, a kernel function calculates weights for each point based on the distance calculated (Weighted KNN, 2020). This would decrease the effect of outliers on the final model and it would fit the model more closely to the underlying function present in the data instead of fitting to the noise of the more distant points. In addition, feature selection, gathering new data, and model stacking are all possible ideas to improve the accuracy. Feature selection would entail removing features with low correlation or that may have little impact on the target variable. One could also add new features that may have influence on the crop yield such as soil characteristics. Lastly, it may increase accuracy to stack KNN with another ML model and use the output of the first model as input into the KNN algorithm.

14. If you were to try another algorithm then which one and why?

I would suggest an implementation of a kernel-based Support Vector Machine (SVM) regression to predict the crop yield. The number of exemplars compared to the number of features indicates that the curse of dimensionality would not limit the precision of the algorithm. SVM is also highly flexible in the shapes of functions it will map to, allowing for better performance on non-linear datasets. An SVM algorithm would also likely take longer to train than a KNN, however, depending on the context, training time would likely not be of significance in a scenario to predict the crop yield. Lastly, SMV is known to work well when there is evidence that the data is separable in some dimension. The density plots shown previously in Figures 2 and 3 indicate that the crop yield is likely separable in a higher dimension with regards to the weather and rainfall features.

15. Are the results good enough for real world use?

Real world use to predict the crop yield depends on context of usage and the required precision or confidence one would need to use the resulting predictions. Figure 17 below shows the relationship between the accuracy and the margin of error considered acceptable in the predictions. I will use these precision estimates to evaluate two potential uses for this model.

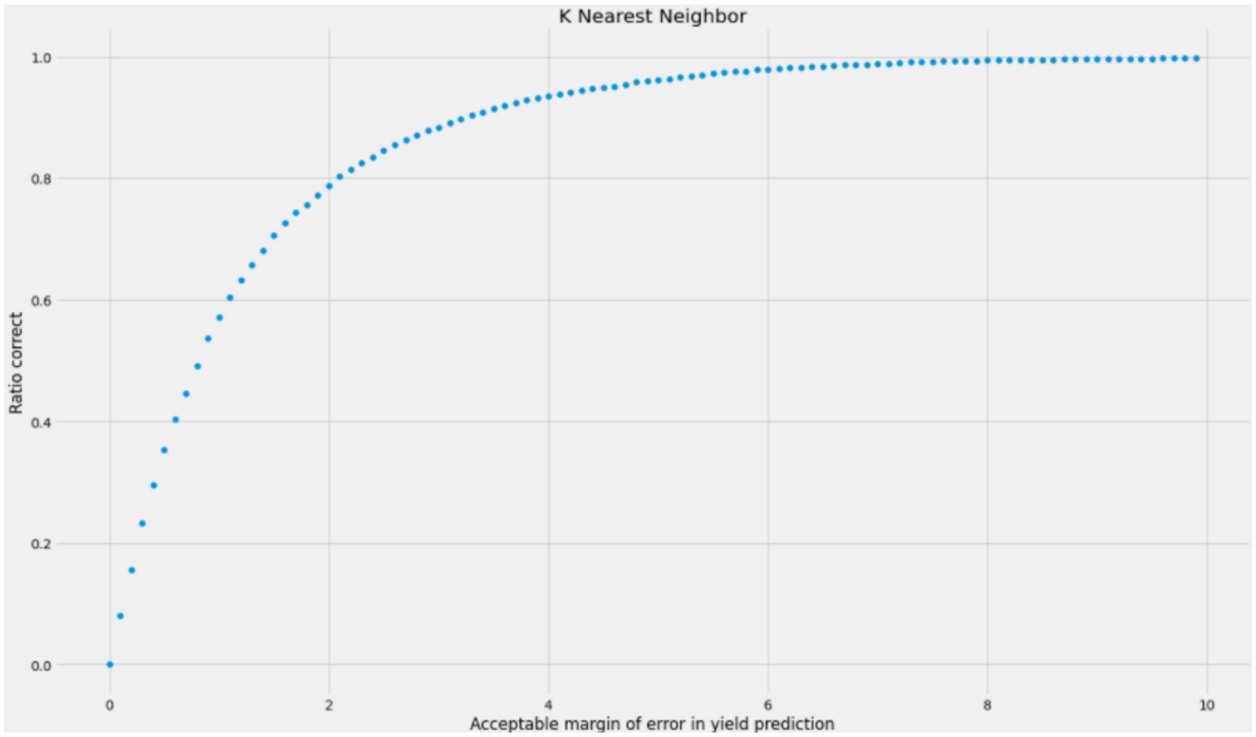

Figure 17. K Nearest Neighbor relationship between accuracy and acceptable margin of error.

1st Scenario

In this scenario the model would be used to predict crop supply in order to generate a profit in commodity futures trading. Knowing when the supply might be low may indicate a future surge in prices, a model predicting the crop yield for a specific commodity would give a trader an edge and help them generate a profit. Precision for a task such as this would likely need to be relatively low. This is because simply understanding if the yield is likely to increase or decrease would give a trader a large edge in the market. Figure 17 shows that one can be about 80% confident that the yield will be within 2 of its yield predictions. This would likely be more than enough confidence to secure an edge in trading.

2nd Scenario

The model would be used to estimate storage needs for the yield supply. This scenario would likely have a low precision requirement. Though, the consequences would be high if the prediction is off by a significant amount. This is because excess supply from the crop yield would probably need to be thrown out. I would say that the model would be fit for use in this scenario, however one would want to be very confident, planning for an error of 4 to 5 to achieve over 95% confidence crop yield would not be wasted.

16. How could this solution fail?

Any of the ML models developed could fail for many reasons. The primary potential reasons for failure would involve either a drastic change in relationship between the input data and the crop yield, or a change in long term crop yield trends. For example, it's possible that technological advances and other agricultural innovations drastically increases the efficiency of crop yield. This could be so drastic that even drastic changes in weather and climate could render the effects on yield to be negligible. Another situation may involve an external impact on the long-term trends of the crop yield that may not be incorporated in the model predictions. This could include the effects of climate change on crop yield or even a drastic slowdown in innovation. One can see from Figure 18 there has been a steady increase in the average crop yield since the starting year of the dataset in 1983. This average yearly increase can likely be attributed to technological advances and improvements in agricultural techniques.

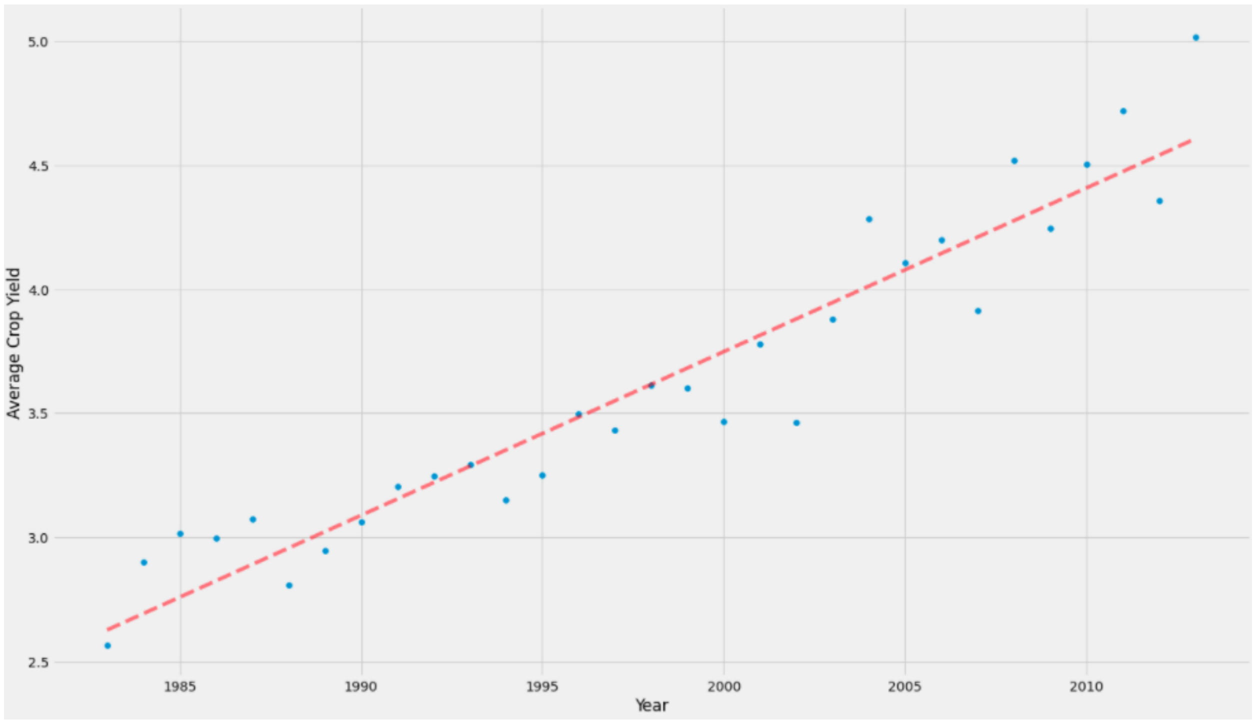

Figure 18. Average yearly increase in crop yield over time.

17. What improvements could be made to the data set?

The addition of new features related to atmospheric conditions, soil characteristics, and even satellite imagery could drastically improve the predictive ability of many ML models. Some improving features may include, humidity, soil pH, organic content, moisture content, nutrient content, and satellite imagery. Aside from satellite imagery, the survival rate and health of many crops are directly influenced by these features. For example, many crops can absorb moisture through the moisture content of the air (humidity) or the soil (USGS, 2020). Many crops also need a relatively neutral pH soil (How Soil pH affects Wheat and Corn Yields) with specific nutrient contents to maintain health (Bould, 1965). In addition, if processed correctly, the pixel color of satellite images may correlate with the future crop yield and have been used for similar problems in the past. This correlation may allow for improvement in predictive ability for many models.

Appendix

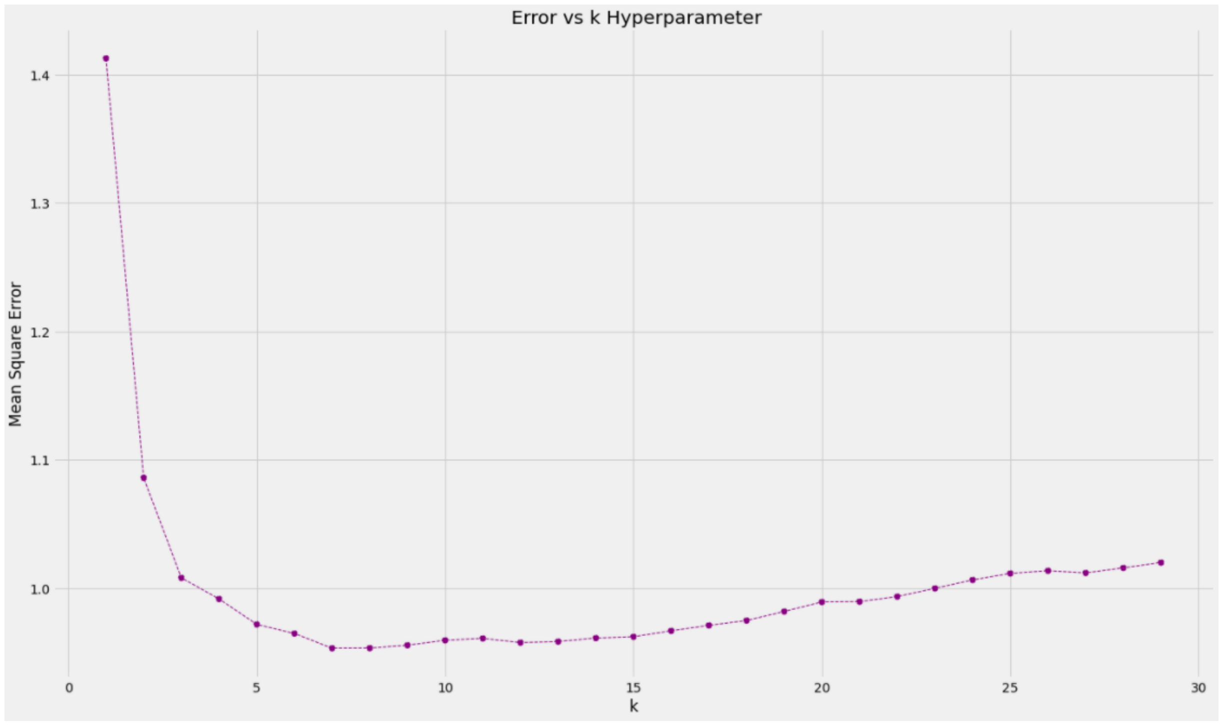

Figure A.19. Error of KNN algorithm with changes to the k hyperparameter.

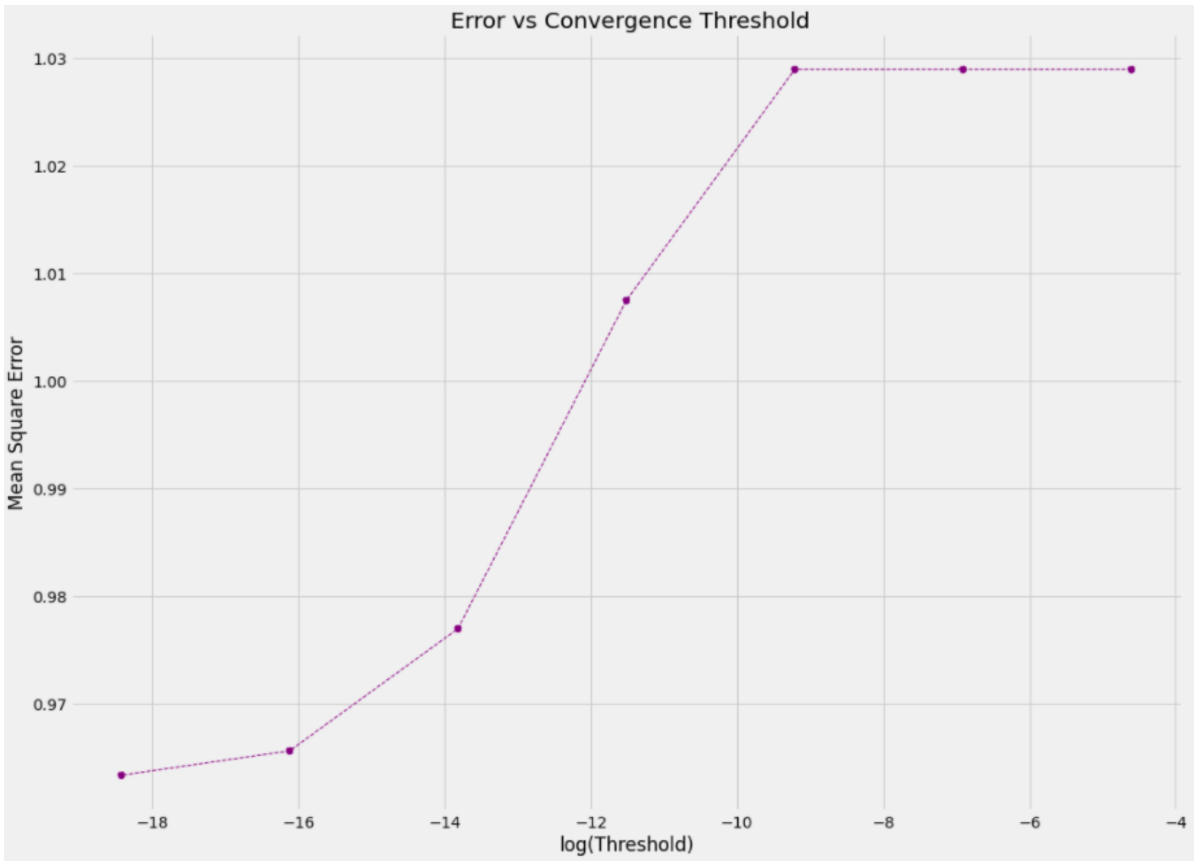

Figure A.20. Error of linear regression with changes in the convergence threshold.

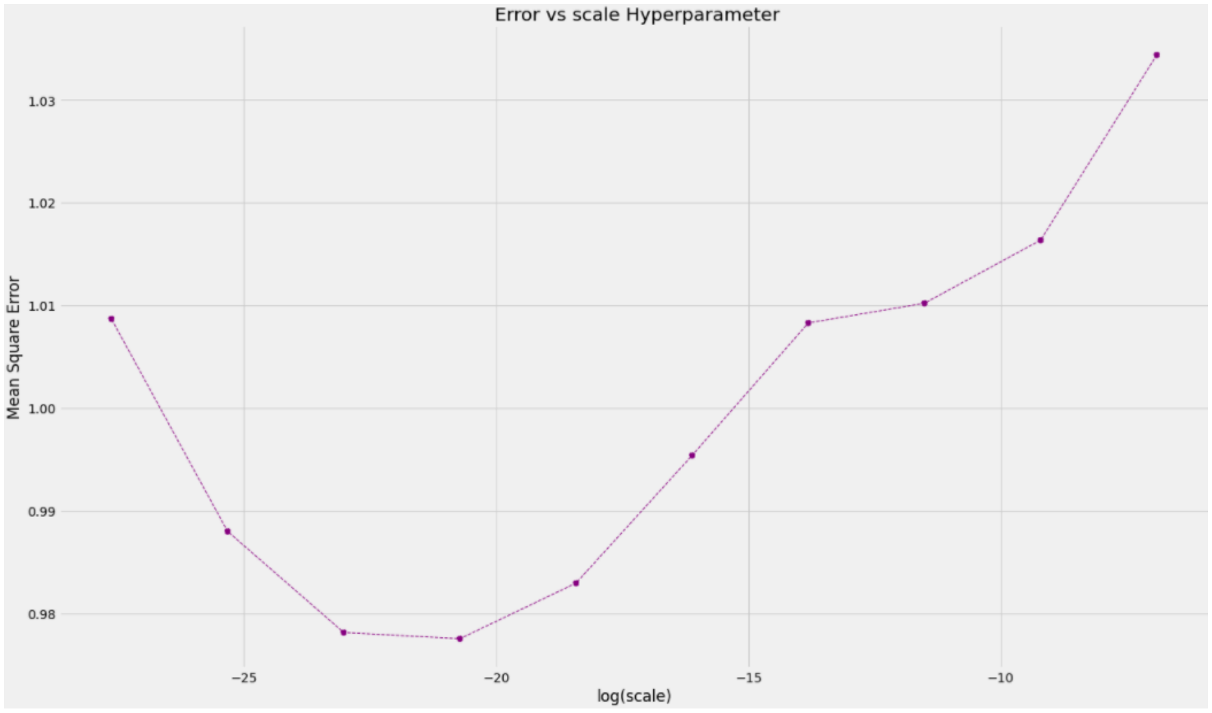

Figure A.21. Error in gaussian process regression with changes in scale parameter.

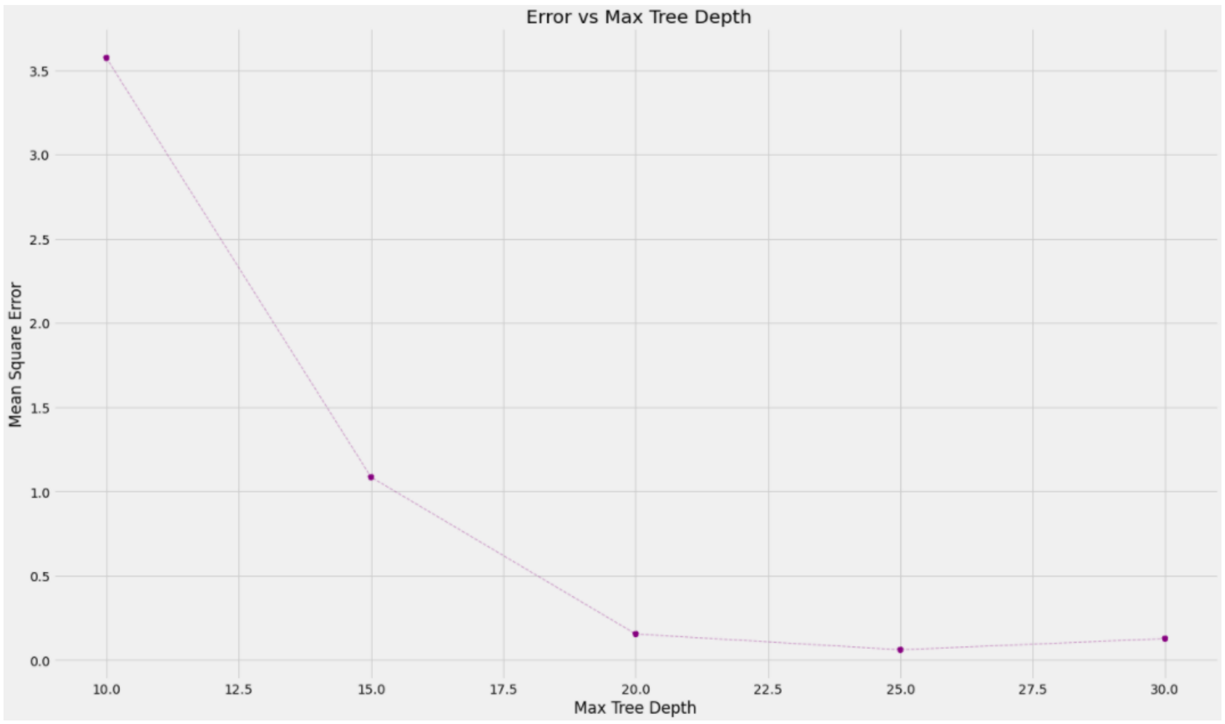

Figure A.22. Error of random forest with variation in maximum tree depth.

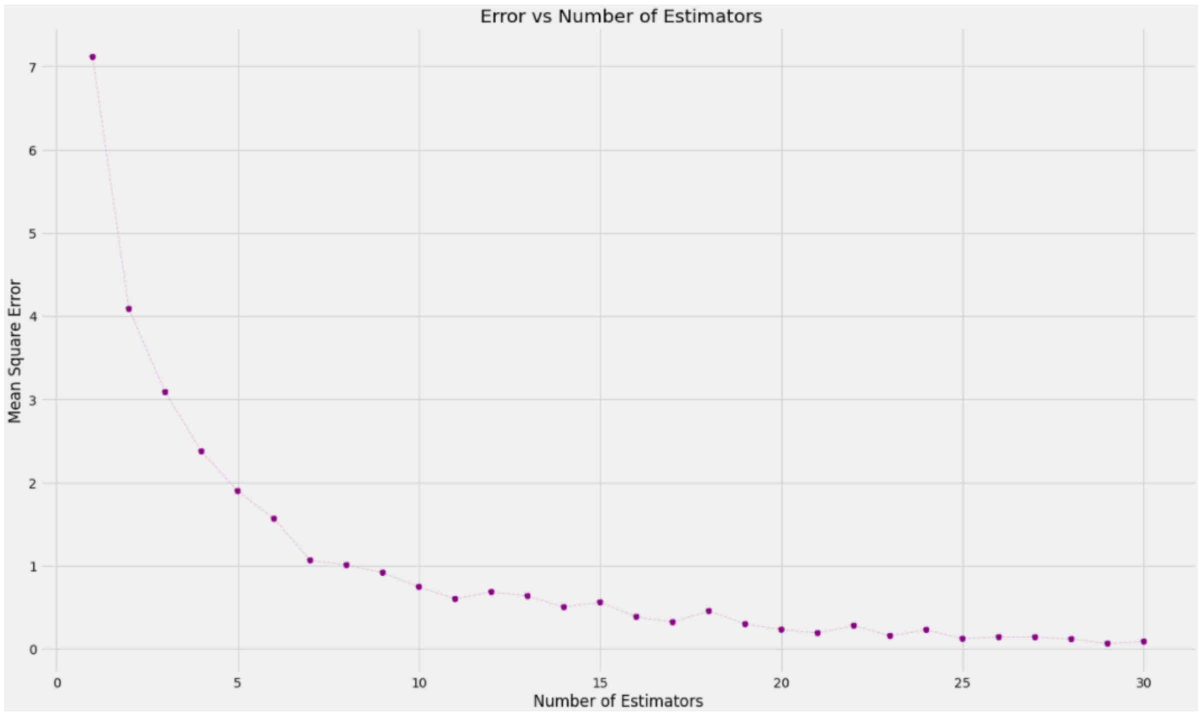

Figure A.23. Error of random forest algorithm with variation in the number of estimators used.

References

Bould, C. (1965). Soil and plant nutrient content in relation to crop yield. *Proceedings of the Nutrition Society, 24*(1), 21-29. doi:10.1079/PNS19650006

How Soil pH Affects Wheat and Corn Yields. (2020). Retrieved January 15, 2021, from https://corn.ces.ncsu.edu/corn-production-information/how-soil-ph-affects-wheat-and-corn-yields/

USGS (2020). Evapotranspiration and the Water Cycle. Retrieved January 15, 2021, from https://www.usgs.gov/special-topic/water-science-school/science/evapotranspiration-and-water-cycle?qt-science_center_objects=0#qt-science_center_objects

Vladimir Pestov,Is the k-NN classifier in high dimensions affected by the curse of dimensionality?,Computers & Mathematics with Applications,Volume 65, Issue 10,2013,Pages 1427-1437,ISSN 0898-1221,https://doi.org/10.1016/j.camwa.2012.09.011.

Weighted K-NN. (2020, April 07). Retrieved January 15, 2021, from https://www.geeksforgeeks.org/weighted-k-nn/#:~:text=In%20weighted%20kNN%2C%20the%20nearest,points%20which%20are%20farther%20away.&text=The%20simple%20function%20which%20is%20used%20is%20the%20inverse%20distance%20function.