La Pyramide

Morgan Stanley Quant challenge

Bóta Attila, Ernyei Zoltán, Freund László

1. Introduction

The purpose of this document is to provide a detailed explanation of the methodology, tests and results used to build a model that predicts the potential loss of productivity on plots of land in Minnesota due to the impact of climate change on agriculture. This document helps to gain a deeper understanding of the approach taken to develop the prediction model and to evaluate its performance. We intend to provide clear and concise information on the data we used, the methodology, the tests conducted to validate the model and the results achieved in order to help the bank assess the climate risk of its investments in agriculture and make informed decisions.

We used Pandas, Tensorflow and Sklearn for data manipulation and model building, Matplotlib for data visualization, and with the help of Google Colab training was accessible on high end GPUs. For model architecture, our first decision was an LSTM model, because the data given was sequential in the form of weather information, but later in the development process we tried an XGBoost model and it showed more promising results. There is a clear explanation of the LSTM and XGBoost model in the Methodology and model building (3) section of the document.

2. Data cleaning and Feature selection

Data Preprocessing: We began by cleaning and preparing the data for analysis. This involved removing any missing or inconsistent data, filling out missing values, transforming variables, and aggregating data at the appropriate level.

We decided to use just the data provided on the competition site, because the model makes predictions based on weather data only:

agri:

- minnesota_county_yearly_agricultural_production
- minnesota_county_location

weather:

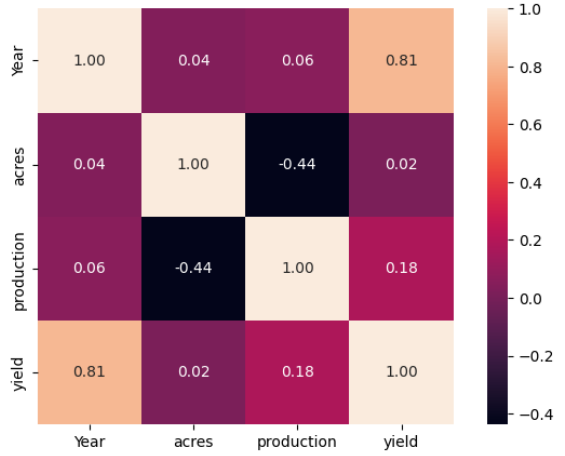
- prediction_targets_daily (for prediction)
- minnesota_daily
- minnesota_station_location

To use this data, first we had to complete the missing minimum and maximum temperature values in the weather dataframe. Assuming that the climate change is negligible in a few year timeframe, we searched for the closest existing value on that same day and averaged it with the values of the closest days, where minimum and maximum temperatures were given to get a close estimate of min and max values.

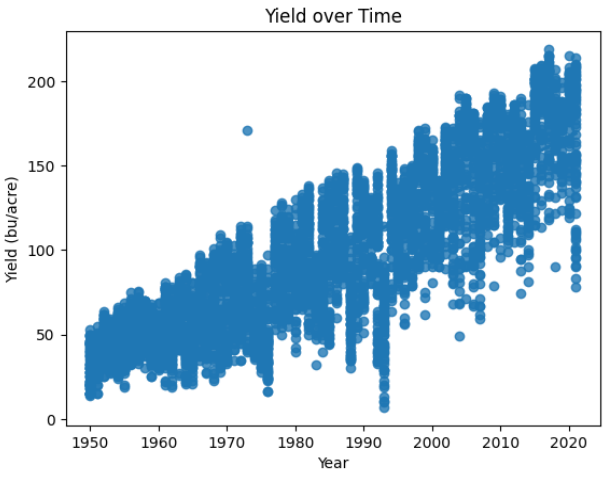
With the daily average temperature we had the same issue, as we had missing values, but implemented a different approach. Using the other weather data and the location, an XGBRegressor was our choice for completing the average data. We split the average temperatures to a training set and a validation set, and managed to fill out the missing values with around 0.79 RMSE, which is an accurate enough estimation.

The agricultural yield data frames were simplified to only retain information on yield, latitude, and longitude. Matching the counties with the yield data to the most relevant weather station was the next step. We calculated the distance from the counties and weather stations and determined the closest. Specifying a threshold was necessary to have good data for training, so we did not use the ones where there were no weather stations in a 120-km circle from the farm.

Exploratory Data Analysis: We conducted exploratory data analysis to identify patterns and relationships between variables in the data. This helped us gain a deeper understanding of how different factors influence crop yields. From the crop yield data we selected the relevant crops. We made a correlation study on the yield and other features. It shows that the yield has a high correlation with the year, but this was expected.



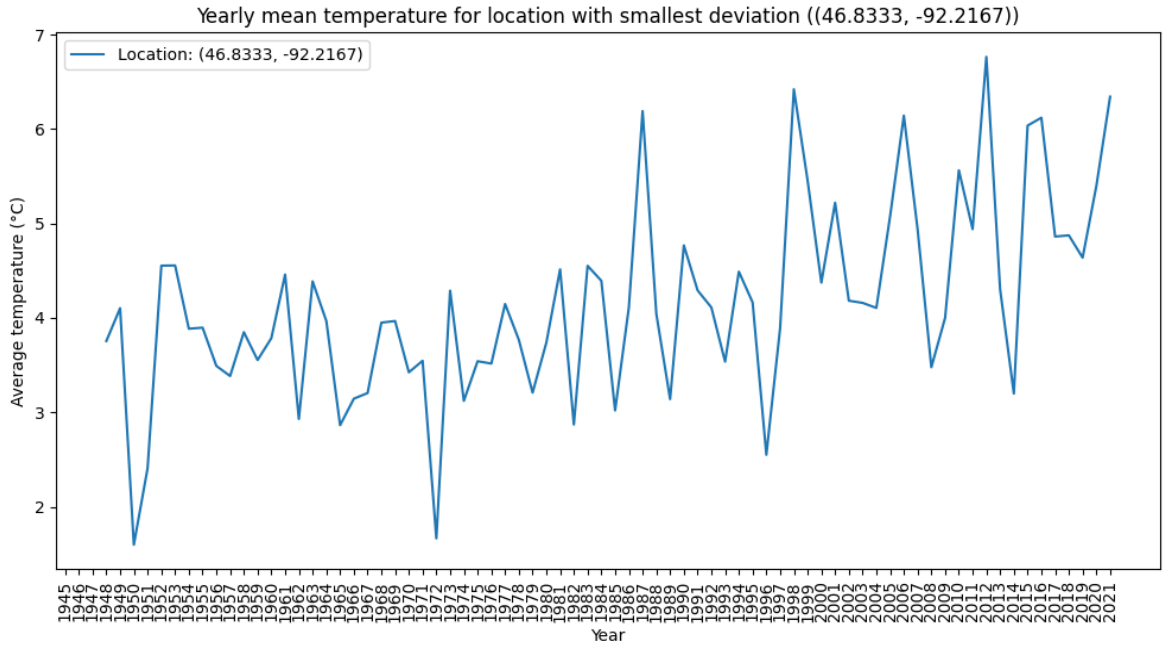
Correlation heatmap of features



Yield increasing over time

Feature Selection: To get more relevant data we created weekly averages. For each week, we examined the normal distribution of the average, minimum, maximum, and rain values and selected their means and standard deviations as features. These 8 features, along with the year, were used to define a training example. For training we only kept the years where at least 50 weeks had complete data, but filled out the missing weeks with the mean of that week for different years.

We normalized our data by using the z-score normalization technique, which involves subtracting the mean of each feature and dividing by its standard deviation to standardize the scale of the dataset. This helped us build a more accurate model.



Weather change over time at one location

The data was split into 90% training data and 10% validation data, and at last we adjusted the shapes of the data for training and validation. This point we had sequential data and were ready to train the LSTM model. For the XGBoost model other changes were needed in the data structure.

3. Methodology and model building

Our goal was to create a model for predicting crop yields using selected features, so we tried out various machine learning algorithms. We trained the model on historical data and then tested its performance on a separate validation dataset.

LSTM (Long Short-Term Memory) is a type of recurrent neural network that addresses the problem of vanishing gradients in traditional RNNs. It uses a memory cell that can selectively forget or add information to its state based on a set of controlling gates. LSTMs can capture long-term dependencies in data due to their memory cell. We thought this model would be a good solution because it can capture the temporal dependencies of weather and other environmental factors on crop growth and yield.

```
Model: "model_1"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
==================================================================================================
 input_3 (InputLayer)           [(None, 52, 8)]      0           []

 lstm_2 (LSTM)                  (None, 52, 128)      70144       ['input_3[0][0]']

 batch_normalization (BatchNorm  (None, 52, 128)     512         ['lstm_2[0][0]']
 alization)

 dropout_2 (Dropout)            (None, 52, 128)      0           ['batch_normalization[0][0]']

 lstm_3 (LSTM)                  (None, 64)           49408       ['dropout_2[0][0]']

 batch_normalization_1 (BatchNo  (None, 64)          256         ['lstm_3[0][0]']
 rmalization)

 dropout_3 (Dropout)            (None, 64)           0           ['batch_normalization_1[0][0]']

 input_4 (InputLayer)           [(None, 1)]          0           []

 concatenate_1 (Concatenate)    (None, 65)           0           ['dropout_3[0][0]',
                                                                  'input_4[0][0]']

 dense_2 (Dense)                (None, 32)           2112        ['concatenate_1[0][0]']

 dense_3 (Dense)                (None, 1)            33          ['dense_2[0][0]']

==================================================================================================
Total params: 122,465
Trainable params: 122,081
Non-trainable params: 384
__________________________________________________________________________________________________
```

Our LSTM model layer architecture

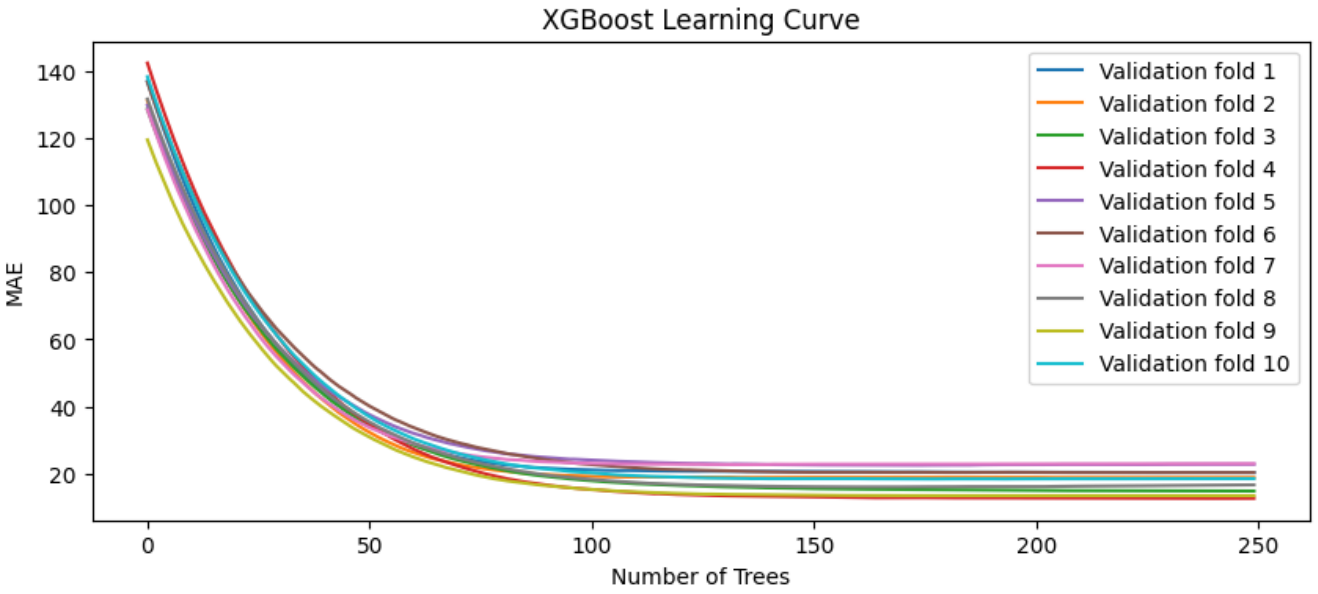
After training our LSTM model to predict the yield of corn crops, we were disappointed to find that it had only achieved a mean absolute error of 15 bushels per acre. Despite our best efforts to tune the hyperparameters and optimize the architecture, we simply couldn't get the performance we were hoping for. However, we were determined to improve our results, so we decided to explore other machine learning algorithms that might be better suited for the task. After researching and trying a few options, we settled on XGBoost as our next approach.

XGBoost (Extreme Gradient Boosting) is an algorithm that utilizes decision trees to improve prediction accuracy. It works by iteratively adding decision trees to the model, where each subsequent tree corrects the errors of the previous tree. The algorithm uses gradient boosting to optimize a user-specified objective function by calculating the gradient of the loss function with respect to the predicted values. Additionally, XGBoost implements a number of advanced features such as regularization to prevent overfitting and handling missing values.

```
params = {'n_estimators': 1000,
'min_child_weight': 7,
'max_depth': 7,
'learning_rate': 0.03,
'gamma': 0.3,
'colsample_bytree': 0.3}
```

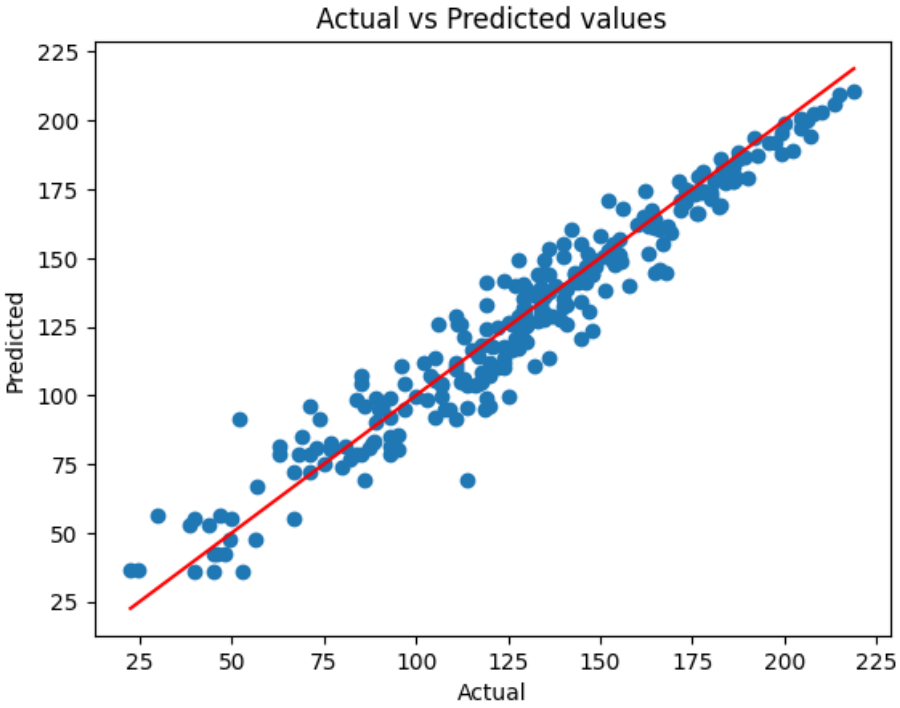
Parameters for the XGBoost model

We flattened all the features to train the XGBoost model, and to determine the parameters for the model training we used hyperparameter tuning, which is the process of selecting the optimal combination of hyperparameters for a machine learning model. In the case of the XGBoost model, hyperparameters such as the learning rate, the number of estimators, and the maximum depth, minimum child weight, gamma, column sample of the trees can be tuned to improve its performance. We used random search for hyperparameter tuning. The optimal combination of hyperparameters is selected based on the evaluation metric, such as accuracy or mean squared error on a validation set.



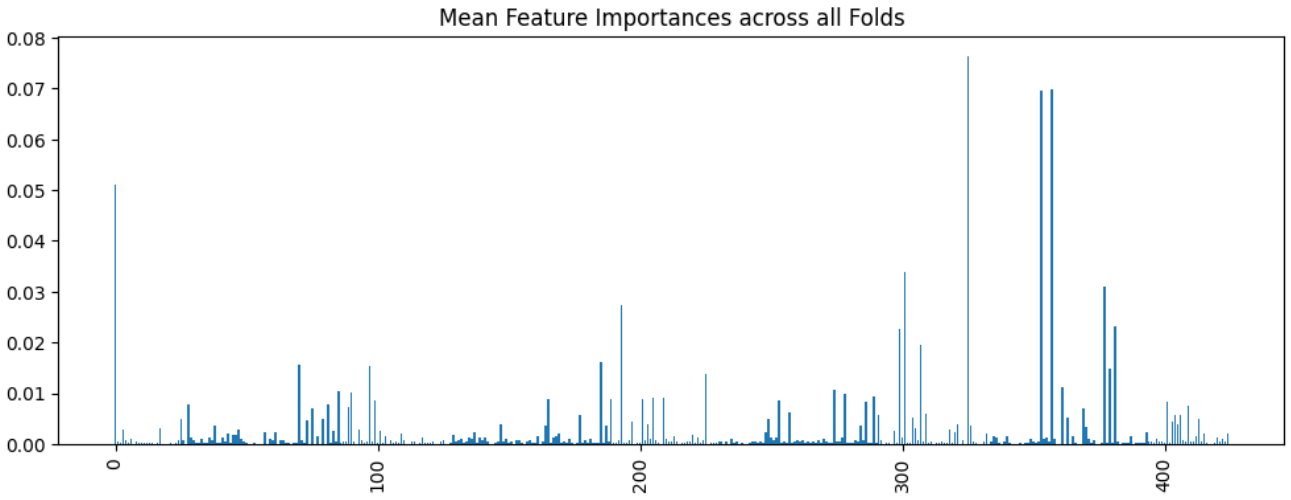
The XGBoost learning curve shows convergence

To train the XGBoost model, we used K Fold to split the training set into 30 subsets, and trained a model on each subset, while withholding one for validation (we only plotted 10 folds on the graph for clarity). This process yielded 30 models, each with its own validation score. To build the final model, we selected the models with a mean absolute error below 11, and with the ensemble method we combined them to a refined final model.



Accuracy of the XGBoost model on a cross validation set on corn

To our delight, the XGBoost model performed much better than the LSTM. In fact, it was able to minimize the mean absolute error to around 10 bushels per acre, a significant improvement over our previous results.



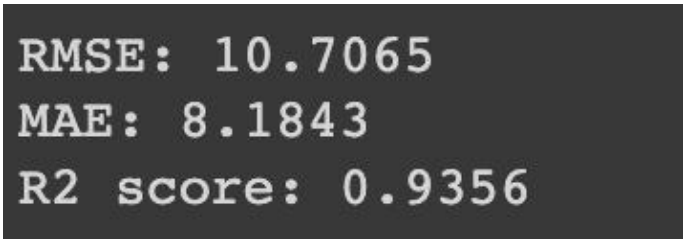
Feature importances across one year

The first data point represents the year under consideration. Subsequently, the consecutive sequence of weekly eight features follows, symbolizing the weekly cycle. On the graph, the grouping of data points in sets of eight represents a single week. As shown on the feature importance graph the most important weeks are when the harvest happens, which is correct according to our research on Minnesota crops.

Based on the validation results, we fine-tuned the model by adjusting its parameters and re-testing its accuracy. We repeated this process until we were confident that the model was performing well.

4. Conclusion

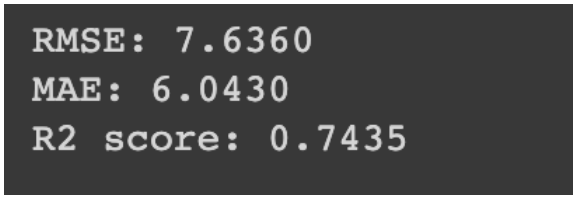
Our final model achieved a mean absolute error of 8.2 bushels per acre for corn crop on the training dataset. This indicates that our model has a good fit and is able to predict crop yields with reasonable accuracy.



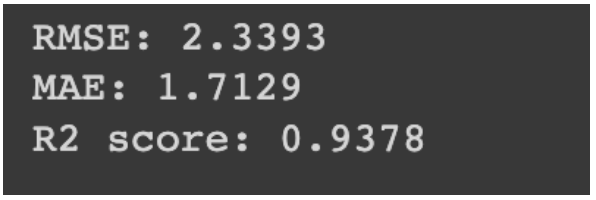
Validation scores for the final corn model

After applying the same data preprocessing on the provided climate trajectories with the exception of padding every year to 52 weeks. We used them as inputs to our model and generated forecasts in the same format as provided, we only predicted where enough data were provided.

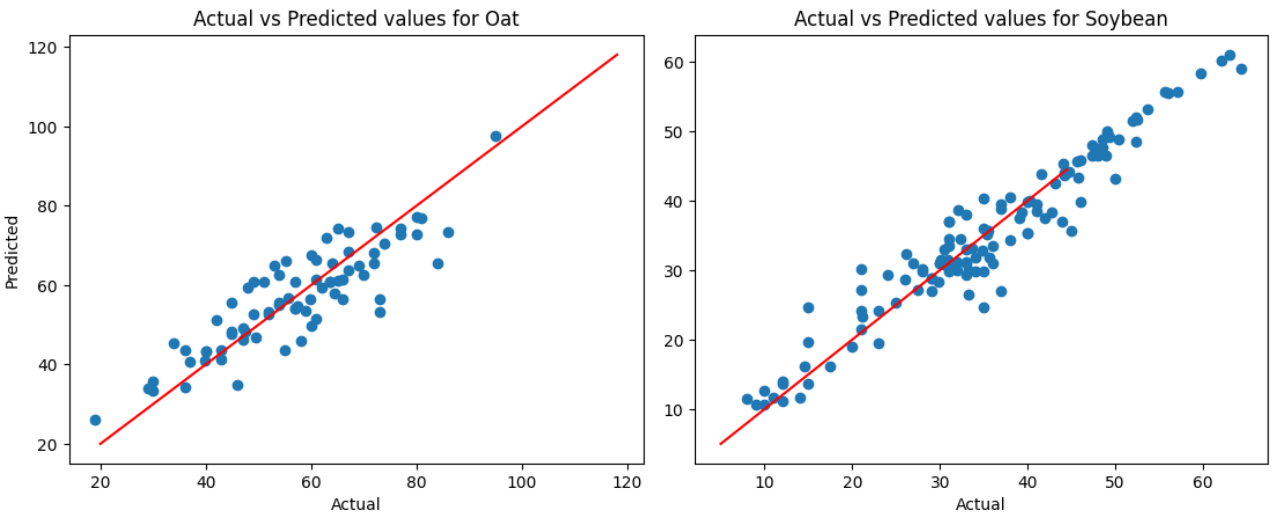
We have developed separate models for both oats and soybeans. These models operate in the same manner as the model for corn. Encouraging validation results have been obtained for both crops, however, the weaker r2 score for oats suggests that yield is not as strongly correlated with weather data.



Validation scores for the final oat model



Validation scores for the final soybean model



Predicted vs actual values on validation sets for different crops

In conclusion, we developed models that can predict potential productivity on plots of land in Minnesota based on historical crop yields, weather data, and future climate scenarios. We used a rigorous methodology to prepare the data, select features, build and validate the models, and generate forecasts. Our models achieved good performance metrics on the provided data and provided accurate forecasts for different climate scenarios. These models can be used by the bank to assess the climate risk of its investments in agriculture and make informed decisions.