Regression Analysis of War and Agricultural Production

Regression Analysis of War and Agricultural Production

A report on the impact of disorder events on agricultural markets

# Desiree Teter

# MSc. Data Analytics, Student ID 000809943

Regression Analysis of War and Agricultural Production

A report on the impact of disorder events on agricultural markets

1. Research Question

What is the relationship between violent disorder or war and agricultural production value?

When working for large multinational agricultural corporations, this question comes up recurrently. Violent events within a country seem to shake up agricultural markets, or at least relevant stocks, but there is little research attempting to directly investigate the relationship between agriculture and war. This project is a step in the direction of filling that perceived gap in understanding.

The project will evaluate the relationship between "disorder events" occurring within a state (battles, violent demonstrations, etc.) and fluctuations in its annual gross agricultural production value on various markets.

It will do so by testing the following hypothesis:

H0: # of disorder events in a country has no impact on the growth % of its agricultural production value in most subindustries.

H1: # of disorder events in a country has a negative impact on the growth % of its agricultural production value in most subindustries.  
 It may also prove beneficial, for international agriculture companies as well as for humanitarian organizations, to be able to predict the way military operations and violent occurrences impact local crop and livestock production. To that end a preliminary predictive model will be developed and tested on the dataset used to test the stated hypothesis.

# of disorder events will be represented by a % change from the previous year’s number of disorder events, as using year over year change vs absolute number proved to have a few advantages once analysis was underway.

1. Data Collection

Data for this project was taken from 2 different sources, both public and found online. The datasets were carefully cleaned and combined.

The first data source was produced by the Food and Agricultural Organization of the United Nations. It is a dataset of 11 columns (Area Code, Area, Item Code, Item, Element Code, Element, Year Code, Year, Unit, Value, Flag) and 3085376 observations, and contains data from over 200 different countries. It was collected from records kept as part of the FAO’s initiative to improve global food security. (Statistics, 2020) (Global Food & Agriculture Statistics, 2020)

The UNData terms of use state that all data on the website are available free of charge and may be used as long as the UN is cited. (UNdata | conditions of use, 2020)

The second source of data is an aggregated count of all “disorder events” by country and year, downloaded in excel format. The data set contains 1596 rows and 3 columns (Country, Year, Events). It was collected by the Armed Conflict Location & Event Data Project.

The Armed Conflict Location & Event Data Project is a largely volunteer-based data project designed to enable crisis mapping and conflict analysis. Data is gathered by volunteers in multiple countries who report on and collate historical and contemporary media accounts of “disorder” including battles and violent protests, an adequate way to track upheaval provided there is ample media coverage within the country. Licensing for this dataset is public provided credit is given per their specifications on download. (Curated Data | ACLED, 2020)

There were relatively few challenges associated with the project’s intermediary data collection methodology. Finding sources for data relevant to the research question that were publicly available and “compatible” enough for joining was simply a matter of patient sifting and knowing where to look. The search was started on Kaggle, then taken Google more broadly. Once possible sources were located, data was downloaded directly from the source sites and vetted for appropriateness.

Initially the project intended to include natural disaster event indicators in its dataset. There are many factors that potentially contribute to production changes in ag markets that should ideally be controlled for, the weather foremost among them. However the process of cleaning the disaster event datasets that were available would have involved manual collation into excel sheets and correlating geocoordinates with country names and/or would have required seeking written permission from the data publishers for its use. In the end it was decided that, due to lack of easy availability of pre-cleaned datasets, natural disaster data should not be included in the scope of the current work.

C.  Data Extraction and Preparation

All data extraction and preparation for this project was done in Python. Python is typically ideal for handling large amounts of data such as this project required, was familiar to the analyst, and has packages more than sufficient for both cleaning and analyzing data.

The externally sourced data was downloaded in .csv and .xlsx format from their respective websites, with file type being based on availability. They were loaded into Python using the pandas read\_csv and read\_excel functions. Pandas as a package was made heavy use of throughout cleaning and analysis due to its impressive inbuilt data manipulation functions.  


Figure 1 Python code used to read data files

Countries outside Africa were excluded, and the dataset was subsetted to include only years between 1997-2017 . This is because data from the second, disorder events dataset was most complete for Africa between the years 1997-2017.



Figure 2 Code used to subset ag data set by year

The agricultural dataset was further subsetted to ensure that only rows with certain Element and Unit (both currency indicators) values were included. This is because some country and year combinations had rows listing gross production value in several different currency formats. It functionally resulted in duplicate data. 1 currency unit per country/year/market was chosen for use in the data set. The project defaults to using rows where value was recorded in USD where possible, but it was not possible in all cases.

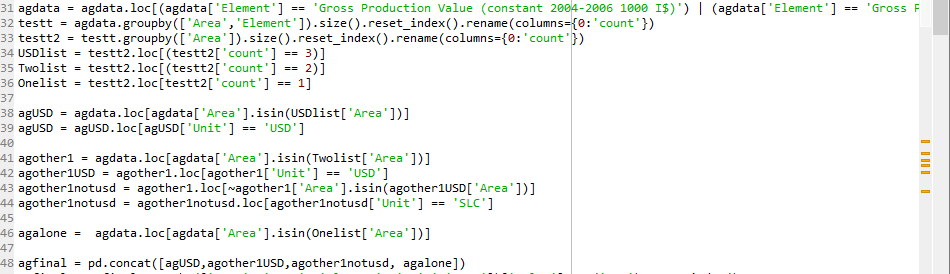


Figure 3 Code used to subset ag data by currency

Gross production value was summed by Area, Year and Item (the type of produce). Areas that were excessively broad (not country-specific) were excluded.

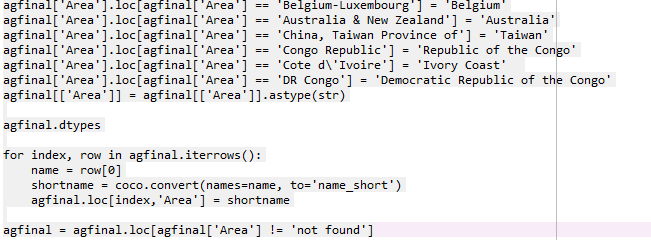
Cleaning the datasets so that they could be used in conjunction proved one of the most challenging aspects of the project. The two must be joined on a country name (along with another key, year) using a series of embedded for loops. A Python library geared toward standardizing country names was used to sync and standardize the country naming conventions for each data set, so that attempted matching on country name (called ‘Area’ in the ag dataset) would be successful. The Python library that was used, country\_converter, was chosen based on the clarity of its documentation.  


Figure 4 Code used to convert ag dataset country names to standardized format

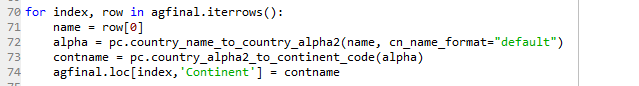
The dataset was also subsetted to only include data from African countries, as conflict data from other continents contained known missing data. The pycountry\_convert library was used to assign a continent value to rows based on country name, which was then used to limit the data frame to include only rows from African countries.  


Figure 5 Code used to assign continent based on country name for Ag data set

In some cases the regex used by these libraries was insufficient for matching. Cleaning them had to be done using manually written lines of Python code. Without this manual cleaning the library functions in packages like country\_converter (coco) errored out. Instances of line-by-line cleaning can be seen in several places in the country\_converter code above.

The library functions were extremely long-running (in the 10s of hours). Data was frequently saved into intermediary CSVs to be reloaded later.

The ag and war datasets were eventually joined successfully on country name and year – information on the number of disorder events per country per year, and the binary occurrence of disorder events per country per year, were added to the ag data set using the same embedded for loop series.

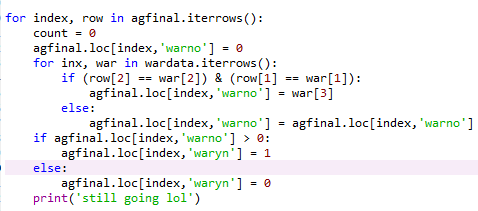


Figure 6 Code used to join relevant columns from second data set to the first data set

The dependent variable used in the analysis was then computed by taking the % difference change in gross production value, from the previous year, for ag markets. This % was stored as a decimal, with positive and negative signs indicating the direction of the change.

It was decided that the project should attempt to predict % difference change year over year rather than actual gross production value (GPV) because attempting to standardize values for countries where GPV was only recorded in local currency and not in USD would have been a near impossible task. Using % change rendered the exact currency the value was recorded in irrelevant while remaining loyal to the intent of the analysis.

The dataset was subsetted yet again to only include independent, dependent and control variables. Rows where the % GPV change was NA were removed.

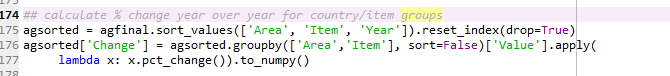
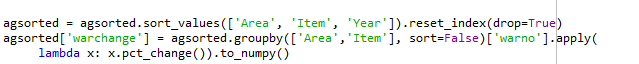


Figure 7 Code to calculat target variabe

At length it was also decided to calculate a % year over year change for # of disorder events in a year, use it as the primary predictor variable in lieu of # of events, and include the absolute # of disorder events as a control variable. This new variable was titled “warchange.” Using changes in levels of political violence within a country (as represented by changes in the number of disorder events) rather than the absolute number of disorder events as a predictor better allows the project to account for the human tendency to adapt to circumstance.   
  
 Rows where warchange was NA or INF were converted to 0 or removed. Finally, categorical variables like Item were encoded to allow for their inclusion in my regression model(s), resulting in two new variables, Area\_cat and Item\_cat.

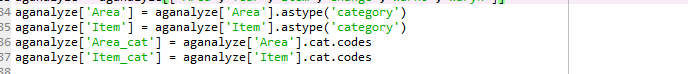


Figure 8 Code used to encode categorical variables

This left me with the following, penultimate dataset:

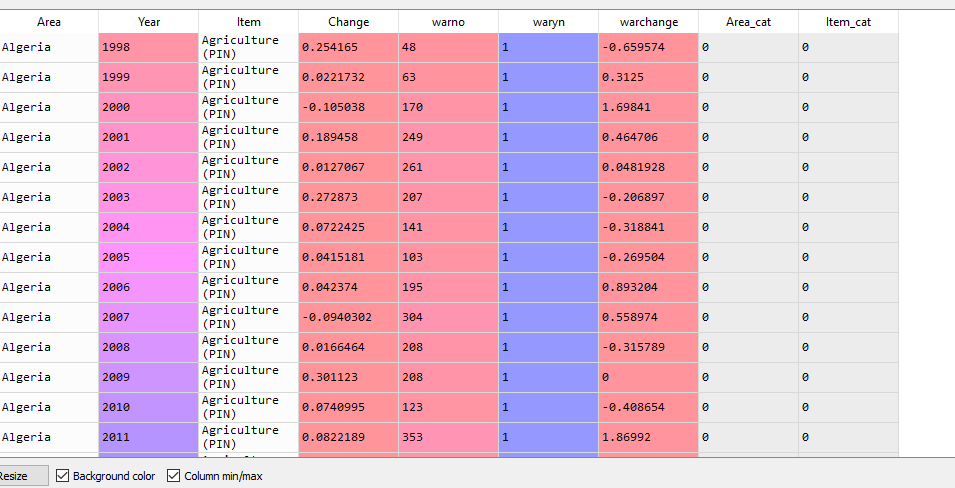


Figure 9 Data set including all analyzed variables

Later, once the need for regularization became apparent (see report section D) and the project began including a Lasso model in its testing, Change values 2 or greater standard deviations from the mean were eliminated to meet the higher demands of that model with regard to reducing the number of outliers:   
  


The number of observations in the final dataset is 41377. This is significantly larger than the number reported in the project proposal. The Python for loop being used to join the two source datasets prior to submitting the proposal encountered an error, and an incomplete dataset with only around 8000 observations was mistakenly generated. As the 41377 observations of the final dataset is well above the minimum # of observations required for the project, and as the cleaning methodology is unchanged – but better executed – this was not an impediment to analysis.

1. Analysis

The language used for analysis of the data in this project was, again, Python. This is true for the same reasons listed concerning its choice as a data cleaning tool.

Python packages used in analysis of this data include:

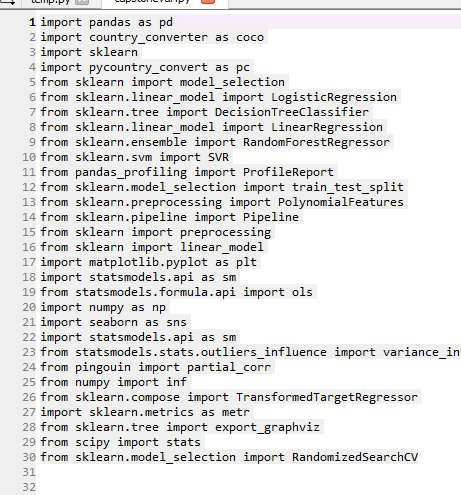


Figure 10 Code to load all packages used in cleaning and analysis

Packages are also cited in the Appendix where possible. (Conventions for citing packages in programming languages like Python are not yet standardized and only some packages have published journal articles available for easy citation.)  
 The primary package used for model development was scikit learn (sklearn). It provides a wide range of models and model-modifying tools (like normalization techniques and Polynomial Feature transforms) that can be used in conjunction with its selection of models. (Pedregosa, 2011)

The intention for this project was not just to test the hypothesis on the relationship between change in GPV for ag markets based on conflict and disorder events occurring within a region. It was also to predict changes in GPV based on occurring conflict events. Prediction allows for mitigation – it is a more actionable goal. To that end the project attempts to fit a regression model in addition to testing the hypothesis via use of Spearman correlation.

The first practical step in analyzing the data was to profile the data using exploratory methods. A main concern was identifying possible instances of multicollinearity and eliminating them. Multicollinearity is the tendency of 1 predictor variable to be predictable based on other predictor variables. When it occurs, it increases the standard error, and results that are statistically significant can appear statistically insignificant. (Minitab Editor 2013) Pandas\_profiling was chosen for the task of evaluating multicollinearity, as it eliminates collinear values automatically.

Profiling did turn up a few minor points of interest such as the high cardinality of Item and the skewness of the dependent variable, which are addressed later in the analysis process.

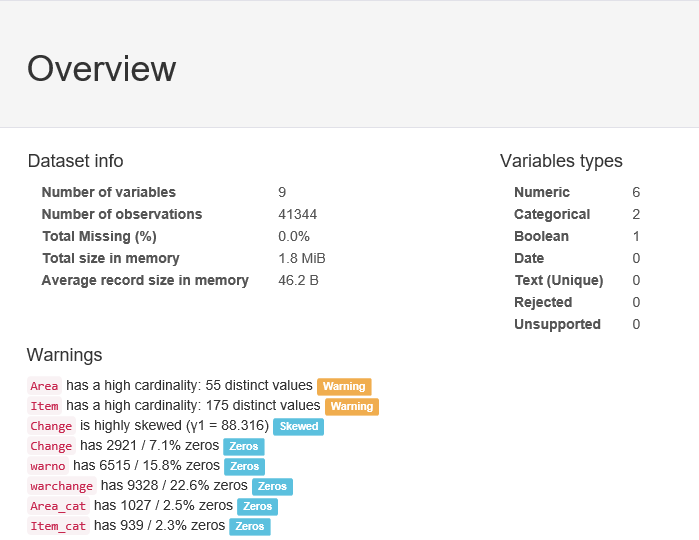


Figure 11 Pandas profiling overview showing no rejected variables and other features

None of the variables included in profiling were rejected, as seen in Figure 11. This suggested that muticollinearity was not an issue for this data set.

For the sake of stringency further testing was conducted. Variance Inflation Factor (VIF) measures multicollinearity in a controlled, direct way by comparing the overall model variance to the variance of models based on each independent variable alone. VIF scores were calculated for the proposed independent and control variables with a technique blogged about by Tavares. (Tavares, 2020)

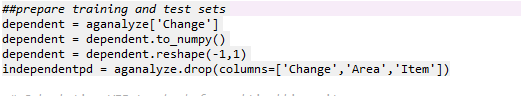


Figure 12 Code showing early preparation for test and training data

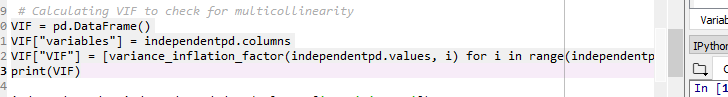


Figure 13 Code showing calcualtion of variance inflation factor

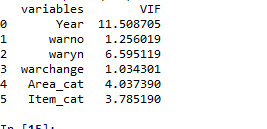


Figure 14 Output showing variance inflation for each variable

As shown in Figure 14, the scores for Year and waryn were somewhat high, at 11.5 and 6.5 respectively. This indicated that the two variables are predictable based on other variables in the dataset – thus are multicollinear with other variables. Based on these scores the two variables were eliminated from inclusion in the model.

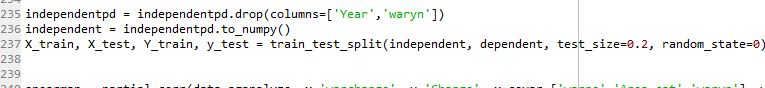


Figure 15 Code showing final preparation of test and training data sets

Before moving forward with testing and creating a regression model for prediction, a formal test of the hypothesis, that there is a relationship between occurrence of disorder events in a country within a year and changes in the gross production value of its agricultural markets, was conducted. Although the dependent variable was not ordinal, it is, as seen above, highly skewed, which lead to the choice of Spearman correlation over Pearson for hypothesis testing. I made use of the partial\_corr function from the pingouin package to control for covariates for both my chosen independent variable and my chosen dependent variable. X covariates were warno, Area\_cat and waryn. Y covariates were Item\_cat and Area\_cat.



Figure 16 Code for calculating partial spearman correlation

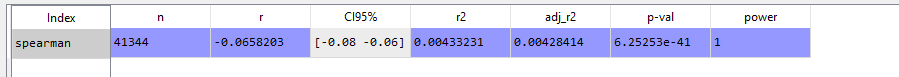


Figure 17 Code showing partial spearman correlation

With a correlation coefficient (see r in Figure 17) of -.0658, there does appear to be a slight negative correlation between changes in the # of disorder events in a country/for a year and changes to gross production value in its agricultural markets. When the # of disorder events increase, ag production value tends to decrease. Although the correlation is quite slight, the P value is very close to 0. This indicates that the correlation is statistically significant.

The null hypothesis, that there is no relationship between violent events in a country and production in its agricultural markets, can thus be rejected.

Based on the results of the hypothesis test, it would be reasonable to expect that any developed regression model will only achieve modest predictive accuracy. There are very likely other factors that contribute as much to shifts in agricultural production value as disorder event occurrence does, and those factors are not accounted for in my data.

For experimentation’s sake, several possible regression models were tested for basic suitability/accuracy before a direction for predictive modeling was decided upon. The first step was an examination of the possible impact of the skewness of the intended target variable. Having skewed actual values for Change (as detected in the pandas\_profiling report) does not guarantee skewness of the residuals of Change, but does flag residual skew as a possible problem. A quick, simple linear model was fitted, and the residuals were calculated and plotted to get a sense of their distribution.

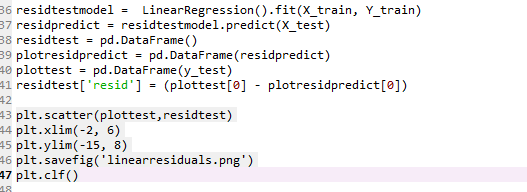


Figure 18 Code to fit simple model, calculate and plot residuals for Change

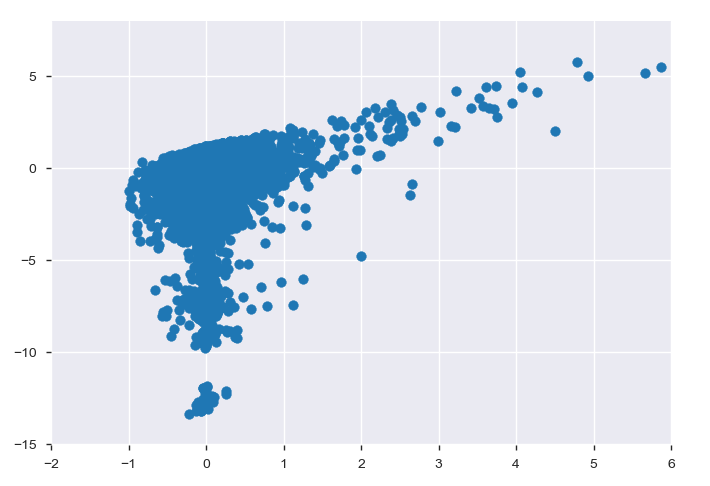


Figure 19 Graph showing distribution for residuals of dependent variable Change

Clearly the distribution of the residuals for the Change variable are quite skewed/non-normal. This alone makes it unlikely a simple Linear model would provide an accurate fit to the data as Linear Regression assumes normally distributed residuals for the dependent variable. Rather than rule out the possibility of using Linear Regression immediately, a log transform was applied to the Change variable to normalize it and make the data friendlier to a wider variety of models.

Because the target variable represents a % year over year change in gross production value, up or down, it can be a negative. As a result it was necessary to add a constant to the variable to make it positive, so that Log could be properly applied as a normalization technique. The constant would need to be subtracted from the model’s predictions in order to plot actual predicted change values.

To apply Log to the y variable, the project made use of scikit learn’s TransformedTargetRegressor, which allows users to define transformations and reverse transformations for a model. Defined transformations can then be applied to any regression model’s dependent variable. This was done for all compared models.

The models selected for testing were Random Forest, Linear Regression, Polynomial regression, and Lasso regression.

The initial proposal was simply to compare Random Forest, Linear Regression and Polynomial regression models. Linear regression is comparatively simple and simplicity is desirable where it is achievable, although it is also not the best fit for the dataset. Random Forest is suitable for use with categorical variables (though the remaining categorical variables were encoded), robust to issues with data normalcy, robust to outliers, and robust to noise. Polynomial data is robust to non-normally distributed data and useful for building out models with multiple predictor and/or control variables.

Initial testing consistently lead to negative cross-validation scores, which suggested a tendency toward fitting noise/over-fitting and a need for regularization. Hence the addition of Lasso as an alternative to a simple Linear model.

Lasso was judged more suitable than Ridge Regression as a regularized model as it reduces coefficients to zero where mathematically sensible and thus performs feature selection, avoiding the need for the addition of step-wise selection. (Bhattacharyya, 2018)

Random Forest is naturally somewhat resistant to the type of overfitting that was observed, because of the way it uses randomized samples. Fine-tuning of its parameters can make it even more resistant to overfitting, but for initial model selection the sklearn default parameters were kept.

The models were defined using scikit learn’s modeling functions. Polynomial features were

added to some models using sklearn’s Pipeline functionality.

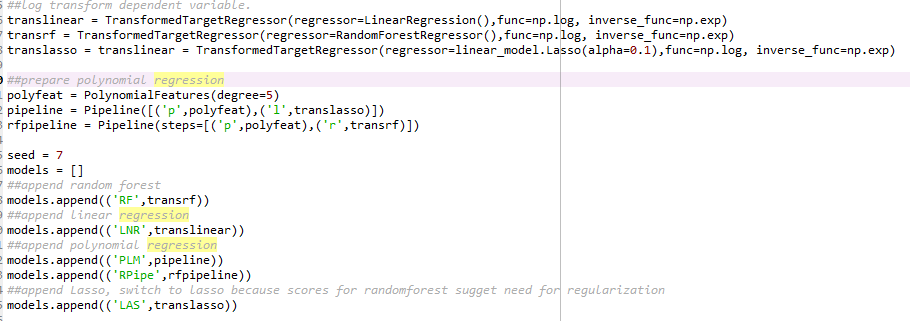


Figure 20 Code used to define each tested model

To address each model in detail:

The default Linear and Random Forest models used the Transformed Target Regressor to logarithmically transform the dependent variable, Change, but otherwise used scikit defaults.

The Lasso model also used a logarithmically transformed independent variable. Its alpha value of .1 – used for moderating the reduction of coefficients to 0 – was chosen for its comparable level of accuracy after a round of cross-validation.

The Polynomial model was produced by using scikit’s PolynomialFeatures to transform the data to a degree of 5 (1 less than the # of initial predictor/control variables) and used the Lasso regressor with log transformation on the target variable.

Finally, a version of Random Forest with Polynomial Features to a degree of 5 was introduced to improve Random Forest’s feature engineering.

Model accuracy was tested using KFold cross validation because, with the target variable being skewed at the outset, the use of multiple folds for validation helped to balance the outcomes and give a more consistent read on dataset performance. The value for k (# of folds) = # of models being evaluated (4).

The scoring technique used for the models was explained variance. This method was appropriate because of the dataset’s tendency toward skewed residuals: the formula for explained variance takes residual skew into account whereas the more common r2 score does not.

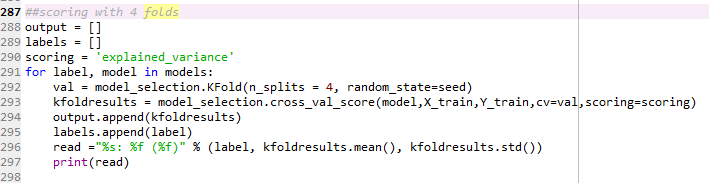


Figure 21 Code showing KFold evaluation of all models

Mean and standard deviation for model scores:



Figure 22 Mean explained variance for Random Forest and Linear models



Figure 23 Mean explained variance for Polynomial model



Figure 24 Mean explained variance for Random Forest with Polynomial features and Lasso models

Based on the explained variance score and standard deviation produced by KFold cross-validation, the Random Forest Regressor with added Polynomial Features was chosen to build out the final predictive model. Although many of the other models had much lower standard deviations in their scores across folds, their explained variance scores were so low as to suggest completely negligible prediction accuracy. The .36 accuracy score of the chosen model was modest but the highest of the values produced by the test

Because the explained variance score seemed so low for even the best performing model, another function of scikit’s model\_selection package was used to evaluate a number of different parameter sets. These new parameters were used in place the default parameters for the Random Forest Regressor that were untouched while using KFold to decide between vastly different models. This was accomplished by creating a random grid of potential parameters for all of the Random Forest Regressor’s parameters, then using the RandomizedSearchCV to evaluate their use in a Random Forest model trained on the project data. It was then possible to select optimized parameter values, define a model using those values, and test their accuracy against the accuracy of a model using scikit’s default parameter values.

The technique used to evaluate hyperparameters for Random Forest was effectively the one described by Koehrson (Koehrson, 2018).

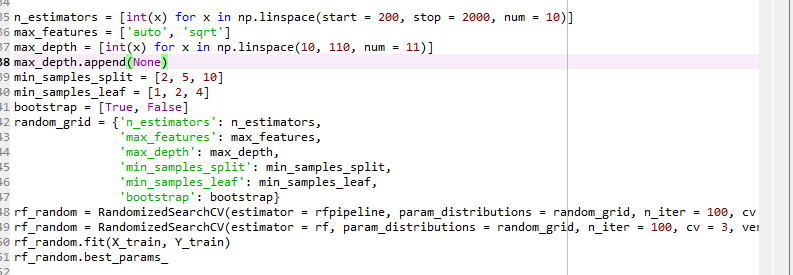


Figure 25 Code for evaluating Random Forest optimal parameters

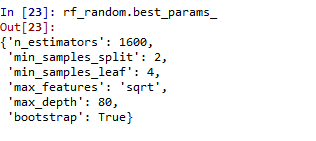


Figure 26 Random Forest optimal parameters

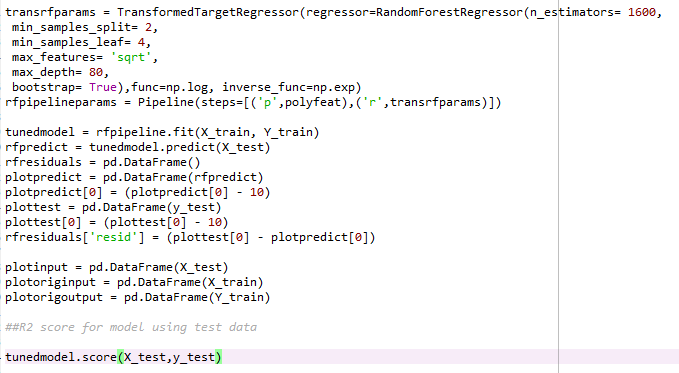


Figure 27 Code for defining the final model with its optimized parameters, training the model, and evaluating its accuracy



Figure 28 Output R2 (accuracy) score for final model, tested on test data set

The model with optimized parameters, tested on the test data set, has a predictive accuracy (using R2 score) of around .65. This is better than the accuracy of the original model with default parameters which was closer to .5. Unsurprisingly given the modesty of the correlation between my main predictor variable (% change # of disorder events per country per year) and my target variable (% change gross production value in ag markets), the accuracy of the model could still use significant improvement.

1. Data Summary and Implications

Using a combination Spearman correlation with covariates for x and y, the project established that there is a slim yet statistically significant correlation between % change in number of disorder events and % change in gross production value in ag markets, for African countries between the years of 1997 and 2017.

The dataset was then used to develop a model for predicting % change in a country’s agricultural production value for a given year based on changes in the number of violent events occurring within the country that year, using Random Forest with Polynomial Features. The model was carefully optimized, yet still apparently hovers around only 65% accuracy in its predictions, per statistical testing.

Although the null hypothesis was rejected, the weakness of the correlation and greater-than-chance-but-low accuracy of the model suggest that there may be confounding variables not being accounted for in the current data set.

An important area of possible future work for this project would include expansion of the dataset / identification of important controls. These would almost certainly include the natural disaster events that were initially slated for inclusion in the dataset, as well as factors like religion, poverty rates, demographic homogeneity, international aid received, etc. It would be helpful to obtain data on a wider range of countries for a larger span of time, and to massage the existing features by reducing the cardinality of the Item / agricultural market.

The second area of suggested future work is further tuning of possible model parameters, exploration of other forms of data regularization, and exploration of a greater variety of models.

Regardless, the research has proved fruitful in that it successfully establishes that # of and/or % change in the # of disorder events in a country for a given year is negatively correlated with the gross production value of its ag markets. Disorder and violent events in a country have an adverse effect on the agricultural economy, to some degree. This and similar factors can be used to help predict fluctuations in agricultural production, especially if the predictive model is improved further.

# Based on this initial research, it is recommended that large agricultural companies and their stakeholders do additional research on possible interventions to buffer against decreases in gross production value for agricultural markets, during periods where social and political disorder is on the rise. Such interventions might not only help to protect the profits of ag companies and farmers, but also result in greater food security for local populations.

1. Appendix (References)

Battacharyya, S. (2018). *Ridge and Lasso Regression: L1 and L2 Regularization*. Medium. Retrieved 22 October 2020, from https://towardsdatascience.com/ridge-and-lasso-regression-a-complete-guide-with-python-scikit-learn-e20e34bcbf0b.

*Curated Data | ACLED*. ACLED. (2020). Retrieved 22 October 2020, from <https://acleddata.com/curated-data-files/>.

*Global Food & Agriculture Statistics*. Kaggle.com. (2017). Retrieved 22 October 2020, from https://www.kaggle.com/unitednations/global-food-agriculture-statistics.

Harris, Charles R. (2020) **Array programming with NumPy**, *Nature*, **585**, 357–362 (2020), [DOI:10.1038/s41586-020-2649-2](https://doi.org/10.1038/s41586-020-2649-2)

Hunter, John D. (2007) **Matplotlib: A 2D Graphics Environment***, Computing in Science & Engineering*, **9,** 90-95, [DOI:10.1109/MCSE.2007.55](https://doi.org/10.1109/MCSE.2007.55)

Koehrson, W*.* (2018) *Hyperparameter Tuning the Random Forest in Python*. Medium. Retrieved 22 October 2020, from <https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74>.

McKinney, W. (2010) **Data Structures for Statistical Computing in Python*,*** *Proceedings of the 9th Python in Science Conference,* 51-56 ([publisher link](http://conference.scipy.org/proceedings/scipy2010/mckinney.html))

Minitab Editor. (2020). *Enough Is Enough! Handling Multicollinearity in Regression Analysis*. Blog.minitab.com. Retrieved 22 October 2020, from <https://blog.minitab.com/blog/understanding-statistics/handling-multicollinearity-in-regression-analysis>.

Pedregosa, F. (2011) **Scikit-learn: Machine Learning in Python**, *Journal of Machine Learning Research*, **12**, 2825-2830 ([publisher link](http://jmlr.org/papers/v12/pedregosa11a.html))

*Statistics*. Food and Agriculture Organization of the United Nations. (2020). Retrieved 22 October 2020, from <http://www.fao.org/statistics/en/>.

Tavares, E. (2017). *Variance Inflation Factor (VIF) Explained - Python*. Etav.github.io. Retrieved 22 October 2020, from <https://etav.github.io/python/vif_factor_python.html>.

*UNdata | conditions of use*. Data.un.org. (2020). Retrieved 22 October 2020, from https://data.un.org/Host.aspx?Content=UNdataUse.