Crop Yield Prediction

Data Mining Project Final Paper

Group 5

ISM 6136 | Professor Balaji Padmanabhan

Team Members:

Naga Chaitanya Chintalapudi

Pranith Reddy Saireddy

Sri Sharmili Sowmya Nanduri

Yogitha Ramaraju

Table of Contents

1.Introduction

1.1 Objective:

Our goal for this project is to forecast production or farm yield based on other variables such as location, season, crop, and land area. We want to know how crop yield is affected by the season and geography, as both impact Yield. We also want to find how variation of yield is by the area under cultivation.

1.2 Problems to be addressed:

- Investigate the data and create new features
- Estimate each farm crop's production.
- Create a sourcing plan for an ingredient based on expected demand for the next few months.

1.3 Approach:

For this analysis we have taken agricultural dataset and climate dataset from Kaggle and merged them based on the common fields to predict crop yield. We built 5 different models and compared the results to find the best model for our analysis.

2. Data Synthesis

2.1 Data Overview

Our data is around 2080580 bytes in size, with 17 properties. Farm data, train data, and weather data, all of which contain a combination of category and numeric variables, are contained in three datasets. The attribute yield is an independent variable, while the rest of the attributes are dependent. We have used the data from year 2016 as training data and used the 2017 data to as test data to do predictions.

Farm Yield Data

Farm id	Unique farm ids
Date	Dates per hour from 2016 in train and from 2017 in test
Ingredient_type	Type of ingredient in the farm : There are 4 types - w,x,y,z
Yield	Yield for each farm per hour

Farm Data

Farm id	Unique farm ids
operations_commencing_year	Year the farm has started
num_processing_plants	processing plants present in the location/ farm
farm_area	Area of the given farm
farming_company	The company that owns the farm
deidentified_location	Location of the farm

Weather data

Timestamp	Dates at which the weather was calculated at each hour
deidentified_location	Location of the farm
temp_obs	Temperature at that hour
Cloudiness	Clouds present in the sky at that hour
wind_direction	The direction of the wind at the hour
dew_temp	Dew temperature at the hour
pressure_sea_level	Pressure sea level at the hour
Precipitation	Rainfall at the hour
wind_speed	Wind speed at that hour

2.2 Data Processing:

1. Converted date, timestamp to datetime format
2. Checking and clearing duplicates
3. **135568** duplicate records have different yield values
   - Dropped the duplicates by copying to another data frame
   - Taken average and replaced the mean for duplicate values
   - Merge the average replaced records back with original data
   - **148920** records are dropped as whole rows are duplicated in the table
4. Identified the percentage of null values in each column
   - Dropped columns with missing value percentage > 40%
   - Imputation is performed for records with missing values < 40% using mean.
   - Imputing with mode i.e. as 0 is occurring majority of the times in precipitation
5. Merge operation on Data
   - Farm_data + test -> initial_merged_test (+ weather_test) -> final test
   - Farm_data + train -> initial_merged_train (+ weather_train) -> final train
6. Removed outliers with **z-score >** 1.96
7. Removing columns that are no longer needed like Weekday, Index, Date, and Timestamp
8. Label Encoding is done for categorical columns
9. Extracted Features from Time stamp such as:
   - Time Stamp: Extracting features from datetime as weekday, day_name, dayofyear, day, month, is_month_end, is_month_start
   - Weekend or Weekday: Appending weekend with 0 and weekday with 1
   - Morning Evening Night: Segregated time into categorical variables
   - Splitting the data for validation Data split is done based on time stamp. For train data we used 2016 data and for test data, 2017 data is used for all months and compared yields based on months.

3. MODEL BUIDLING

3.1 LINEAR REGRESSION MODEL

Dummification is performed on columns such as is_month_end, is_month_start, day_name,time_of_day

```
cols_to_dummi = ['is_month_end', 'is_month_start', 'day_name','time_of_day']
```

```
temptrain= pd.get_dummies(temptrain,columns=cols_to_dummi,drop_first=True)
```

```
tempval= pd.get_dummies(tempval,columns= cols_to_dummi,drop_first=True)
```

```
temptest2= pd.get_dummies(temptest2,columns= cols_to_dummi,drop_first=True)
```

```
temptest2.sample(10)
```

is_month_end_True	is_month_start_True	day_name_Monday	day_name_Saturday	day_name_Sunday	day_name_Thurs
0	0	0	0	0	0
0	0	0	0	0	1
0	0	0	0	1	0
0	1	0	0	0	0
0	0	0	0	0	1
0	0	0	1	0	0
0	0	0	0	0	0
0	0	0	1	0	0
0	0	1	0	0	0
0	1	0	0	0	0

After dummification, Linear Regression model is performed which resulted in **mean absolute error** of 573.68 and mean squared error is 5663965.13.

```
Slope: [-7.55916307e-04 -1.77745429e-01  2.00164376e-04 -2.00476621e-01
 -2.62082818e-02  2.55167938e-03  2.25684547e-03 -4.70129152e-03
  1.09904970e-02  1.80197434e-02 -5.42123621e-03 -1.55184040e-01
 -1.46545233e-02 -1.50951578e-01  4.34722441e-01  2.39625538e-02
 -3.91074748e+01  6.28139861e+00 -4.13708846e+00 -5.14572199e+00
 -5.05108086e+00 -4.58629546e+00 -3.54234291e+00 -3.07732076e+00
 -3.63852046e-01 -2.72793613e+01 -9.04176802e+00 -7.32505952e+01]
Intercept: 498.03901226366344
mean squared error:  5663965.132143057
root mean squared error train:  2379.908639453006
root mean squared error test:  2406.0053770609707
mean absolute error:  573.684734940103
```

3.2 Decision Tree

Decision Tree is performed without outliers and with hyperparameter and we get to compare the results.

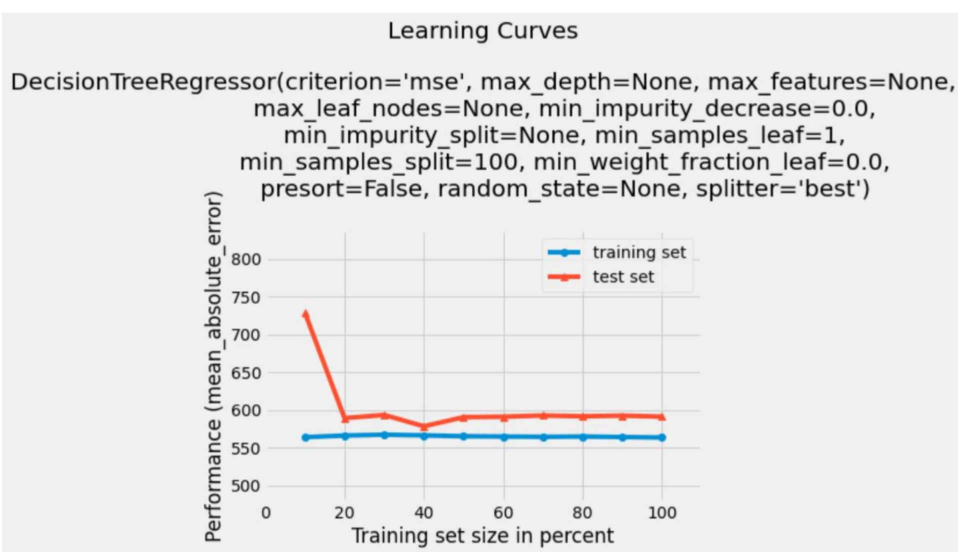
Without Outliers:

```
mean squared error for train:  1415713.4806902749
mean squared error for validation:  7902171.526440854
root mean squared error for train:  1189.8375858453435
root mean squared error validation:  2811.080135186625
mean absolute error for train:  143.86279821314398
mean absolute error for validation:  641.2978784826779
```

Hyperparameter:

```
mean squared error for train:  5348551.552265162
mean squared error for validation:  5918989.492931039
root mean squared error for train:  2312.693570766599
root mean squared error validation:  2432.8973453335507
mean absolute error for train:  563.3067030564814
mean absolute error for validation:  590.9931833981524
```

Learning Curve:



3.3 Random Forest

Before changes we got Mean absolute error of 587.83 and mean squared error was 5812902.98

```
mean squared error for train:  5314819.1795634935
mean squared error for validation:  5812902.98563557
root mean squared error for train:  2305.389160112343
root mean squared error validation:  2410.9962641272527
mean absolute error for train:  558.6791473841952
mean absolute error for validation:  587.8301505747415
```

3.4 XG Boost

Preprocessing: Label Encoding done on columns dayofyear, day and hour

MAE value for XG Boost is 575.912

```
mean squared error for train:  5663671.5
mean squared error for validation:  5788779.0
root mean squared error for train:  2379.847
root mean squared error validation:  2405.988
mean absolute error for train:  573.64453
mean absolute error for validation:  575.912
mean absolute percentage error for train:  inf
mean absolute percentage error for validation:  inf
```

4. Evaluations and Results

4.1 Evaluation Methods

1. Compared the errors of all the models performed and evaluated the results

Linear Regression	Decision Tree without outlier	Decision Tree with Hyperparameter	Random Forest	XG Boost
573.8	143.86	563.30	558.67	573

When compared to all other models, the **Decision Tree without outliers** produces the least error, hence it is the best model to utilize for agricultural yield production.

2. Compared Yield produced - Actual Versus Original Demand:

Predicted yield for every month for 2017 is presented below

EXPECTED YIELD FOR EACH MONTH IN 2017:

```
extra = extra2 -  org_demandlist[3]
print("Extra produce after meeting 4th month's demand : ",extra)
extra2 = extra + twtlist[3]
extra = extra2 -  org_demandlist[4]
print("Extra produce after meeting 5th month's demand : ",extra)
extra2 = extra + twtlist[4]
extra = extra2 -  org_demandlist[5]
print("Extra produce after meeting 6th month's demand : ",extra)
extra2 = extra + twtlist[5]
extra = extra2 -  org_demandlist[6]
print("Extra produce after meeting 7th month's demand : ",extra)
extra2 = extra + twtlist[6]
extra = extra2 -  org_demandlist[7]
print("Extra produce after meeting 8th month's demand : ",extra)
extra2 = extra + twtlist[7]
extra = extra2 -  org_demandlist[8]
print("Extra produce after meeting 9th month's demand : ",extra)
extra2 = extra + twtlist[8]
extra = extra2 -  org_demandlist[9]
print("Extra produce after meeting 10th month's demand : ",extra)
extra2 = extra + twtlist[9]
extra = extra2 -  org_demandlist[10]
print("Extra produce after meeting 11th month's demand : ",extra)
extra2 = extra + twtlist[10]
extra = extra2 -  org_demandlist[11]
print("Extra produce after meeting 12th month's demand : ",extra)
extra2 = extra + twtlist[11]
print("Extra produce remaing after year end : ",extra2)
```

```
Extra produce after meeting 2nd month's demand :   935387147.0
Extra produce after meeting 3rd month's demand :   1724586864.0
Extra produce after meeting 4th month's demand :   2571419840.0
Extra produce after meeting 5th month's demand :   3542250171.0
Extra produce after meeting 6th month's demand :   4149326571.0
Extra produce after meeting 7th month's demand :   4950657571.0
Extra produce after meeting 8th month's demand :   5709175940.0
Extra produce after meeting 9th month's demand :   6480964754.0
Extra produce after meeting 10th month's demand :   7142384354.0
Extra produce after meeting 11th month's demand :   7823551988.0
Extra produce after meeting 12th month's demand :   8525427878.0
Extra produce remaing after year end :   9320030078.0
```

4.2. Results and Findings

Extra Yield is calculated by comparing actual consumption to the original demand, making it easier to predict and develop crops to meet future needs.

5. Conclusions and Future Work

5.1. Conclusions

- Among all the models evaluated, the Decision Tree is the most appropriate.
- The monthly harvest is more than enough for consumption.
- Any excess product can be preserved for future use.

5.2. Limitations

- Crop forecast was based on 2 years of data; however, the number of years analyzed might be increased to obtain a more efficient conclusion based on annual demand increases.
- Future demand can be forecasted using previous data for the next number of years, rather than just one or two.

5.3. Potential Improvements or Future Work

- Density Based clustering Technique can be used on Crop yield Prediction.
- Factors affecting agricultural yield output can be identified and worked towards an efficient method of yield prediction using the density-based clustering technique.

6.References

- https://realpython.com/linear-regression-in-python/
- https://towardsdatascience.com/random-forest-in-python-24d0893d51c0
- https://arxiv.org/ftp/arxiv/papers/2105/2105.01282.pdf
- https://ai.plainenglish.io/hyperparameter-tuning-of-decision-tree-classifier-using-gridsearchcv-2a6ebcaffeda
- https://towardsdatascience.com/step-by-step-guide-building-a-prediction-model-in-python-ac441e8b9e8b