Evaluating the Performance of K-Nearest Neighbors (KNN) compared to Multiple Linear Regression and Polynomial Regression in Forecasting Avocado Sales

Julius Calvin Saputra
Department of Information System, Multimedia Nusantara University, Indonesia
julius.calvin@student.umn.ac.id

Abstract – In order to predict avocado sales, this analysis journal looks into and contrasts the forecasting accuracy of K-Nearest Neighbors (KNN) with Multiple Linear Regression and Polynomial Regression. The study uses a dataset with average price, type, and year as well as other avocado-related features to evaluate the predictive power of multiple linear regression, polynomial regression, and KNN models for total sales volume. By employing assessment metrics like mean absolute error, mean squared error, and R-squared, the study seeks to clarify the relative advantages and disadvantages of every regression method in identifying the intricate relationships present in the avocado sales data.

Keywords - K-Nearest Neighbors; Multiple Linear Regression; Polynomial Regression; Forecasting; Avocado Sales

I. INTRODUCTION

The avocado sector, which is known for its rapid expansion and volatility [1], requires sophisticated forecasting tools in order to handle the intricacies of sales projections. Our study compares and assesses the forecasting abilities of three different regression techniques: polynomial regression, multiple linear regression, and K-nearest neighbors (KNN) in response to this necessity. The study, which focuses on the expectation of avocado sales, makes use of an extensive dataset that has been enhanced with a variety of avocado-related characteristics, such as average price, type, and year. Our goal as we explore the complexities of these regression models is to offer a thorough grasp of their individual advantages and disadvantages in terms of accurately representing the complex relationships found in the avocado sales data.

Our research seeks to provide a nuanced perspective on the relative performance of these regression techniques by going beyond traditional analysis and applying rigorous evaluation metrics like mean absolute error, mean squared error, and R-squared. This study not only meets the immediate needs of avocado industry stakeholders by illuminating their applicability and accuracy in forecasting total sales volume, but it also adds to the growing conversation on predictive modeling in the agricultural domain. As we begin this investigation, our results aim to provide researchers and industry practitioners with knowledge that goes beyond the confines of conventional forecasting techniques, encouraging a more profound comprehension of the ever-changing avocado market.

II. LITERATURE

A. Dataset

A dataset is an arranged and structured collection of data with a common theme or objective. Typically, these datasets consist of rows and columns, where each row denotes a single observation or record and each column a variable or attribute related to those observations. In almost every industry, including business, government, and machine learning, datasets are used for a range of purposes in order to train algorithms, gain insights, and make decisions that can be defended.

The "Avocado Prices (2020)" dataset, which includes historical data on avocado prices and sales volume in various US cities, states, and regions, is the dataset used in this journal.[2]

B. Data Processing

The process of converting data into information that can be used is called data processing. Validation, sorting, summarizing, aggregating, analyzing, reporting, and classification are some of the processes that are involved. Gathering data from accessible sources, such as data lakes and data warehouses, is the initial stage of data processing. After data collection, it moves on to the data preparation phase, where unprocessed data is cleaned and arranged in preparation for the next data processing step. In the processing phase, the data is processed for interpretation using machine learning algorithms.

Data processing is defined by the General Data Protection Regulation (GDPR) as a broad range of operations carried out on personal data, including gathering, logging, organizing, storing, retrieving, consulting, using, disclosing, and destroying personal data, whether through manual or automated means.[3]

C. *Data Visualization*

Data visualization are tools, which use visual elements such as charts, graphs, and maps, make it easy to see and understand trends, outliers, and patterns in data. It also gives workers and business owners a great way to present data to non-technical audiences without confusing them. Effective and good data visualization will give good insights [4].

Numerous data visualization techniques are available, such as heat maps, tree maps, bar charts, line charts, scatter plots, and more. Every type of visualization can be used to highlight different aspects of the data and is appropriate for a variety of data types.

D. *Machine Learning*

Algorithms are used in machine learning, a subfield of artificial intelligence, to help computers learn from data and make predictions or decisions without explicit programming. The creation of computer programs that can access data and use it to learn for themselves is the focus of this subset of artificial intelligence. Machine learning algorithms can be trained to identify patterns in data, classify data, and generate predictions based on data[5].

E. *Machine Learning Regression*

Regression is a kind of machine learning algorithm that forecasts a continuous result by evaluating the values of one or more predictor variables [6]. Being a subset of supervised learning, it entails that the algorithm is trained on data that has already been assigned a label or classification. Regression machine learning is the most appropriate technique for study because it can help predict the average price of avocados based on the other features in the dataset. This is a regression problem because the target variable (average price) is continuous.

F. *K-Nearest Neighbors*

The K-Nearest Neighbor (K-NN) algorithm can resolve a variety of classification issues. This method determines the K number of points that are closest to the test data after calculating the distance between all of the training points and the test data. Based on the chosen K points, the algorithm then attempts to predict the correct class for the test data [7].

G. *Polynomial Regression*

Regression analysis uses the supervised learning algorithm polynomial regression. It uses an degree polynomial to represent the relationship between the independent variable (x) and the dependent variable (y) [8]. The algorithm looks for the line that best fits the data points and provides the best possible representation. Usually, cross-validation methods or trial and error are used to determine the polynomial's degree. Numerous disciplines, including physics, engineering, and economics, frequently use polynomial regression [9].

H. *Multiple Linear Regression*

Multiple linear regression expands on simple linear regression. Since the response variable is directly correlated with a linear combination of the explanatory variables, the term "linear" is continued being used in both scenarios.By adding more than one explanatory variable, multiple linear regression expands on simple linear regression. The response variable is directly correlated with a linear combination of the explanatory variables, we continue to use the term "linear"[10].

III. METHODOLOGY

A. *Research Objectives*

The main goal of this journal is to methodically assess and contrast the forecasting abilities of three different regression techniques: polynomial regression, multiple linear regression, and K-nearest neighbors (KNN). This study uses a large dataset that includes a variety of avocado-related features, including average price, type, and year, to evaluate how well these regression models predict overall sales volume. By utilizing critical assessment metrics such as mean absolute error, mean squared error, and R-squared, our study aims to clarify the relative merits and demerits of every regression method in describing the complex relationships found in the avocado sales data.

B. *Dataset*

This journal analysis uses a dataset named “Avocado Prices (2020)”, which is This is an updated version of the avocado dataset originally compiled from the Hass Avocado Board (or HAB, for short) data and published on Kaggle by Justin Kiggins in 2018.
Kaggle Link: https://www.kaggle.com/datasets/timmate/avocado-prices-2020/data

C. *Handling Missing Values*

Missing Values can be tricky and will result in a decrease or loss of efficiency[11]. Because the avocado sales 2022 dataset does not contain any null or missing values, there won’t be a need for Handling Missing Values. However, if there were Missing Values in the dataset, then the dropna() function would be a good choice to use.

D. *Handling Outliers*

In most datasets, there is what is called an anomaly that is needed to be erased so that processing the data will run smoothly[12]. The way to remove these anomalies are with “Handling Outliers”. This study will use the Tukey method, also known as the Interquartile Range (IQR) method. This technique involves locating and eliminating the dataset's outliers using the interquartile range[13].

E. *Formatting*

Some columns in the dataset may not be compatible with the data processing we want to use, so we use Formatting to achieve Accuracy, consistency, and compatibility with analysis tools[14]. The method of formatting that will be used in this study is pd or pandas formatting (for example: pd.tp_datetime).

F. *Normalization*

Increasing the quality of data has emerged as one of the main issues facing machine learning, the use of normalization is a preprocessing method that turns features into a common scale for use in machine learning and data analysis.[15]. Normalization can be the solution

to handle and enhance the quality of data. This study uses Min-Max scaling technique to normalize the data,

G. Binning

Numerical values can be converted to categorical using binning. Depending on the type of data and the analysis's objectives, this procedure may be helpful in a variety of analysis scenarios[16]. The method of binning used in this study is called custom binning or manual binning, where price levels are determined by dividing into 5 categories; 'Very Low', 'Low', 'Average', 'High', 'Very High'.

H. Encoding

The process of transforming categorical variables with a set of possible values into a numerical form suitable for machine learning algorithms[17]. The method used in this study is called Label Encoding.

I. Grouping

A basic step in data analysis is grouping data, which is combining and arranging data according to predetermined standards. Grouping makes it possible to compile and evaluate data, which then can be used to compare data[18]. The method of grouping used in the study is called “grouping by multiple columns”.

J. Mean Absolute Error

The average of the differences between the expected and actual values is known as the mean absolute error, or MAE. The direction of the errors calculates the average magnitude of the errors in a set of predictions [19]. MAE is useful because it provides a simple measure of the models performance.

K. Mean Squared Error

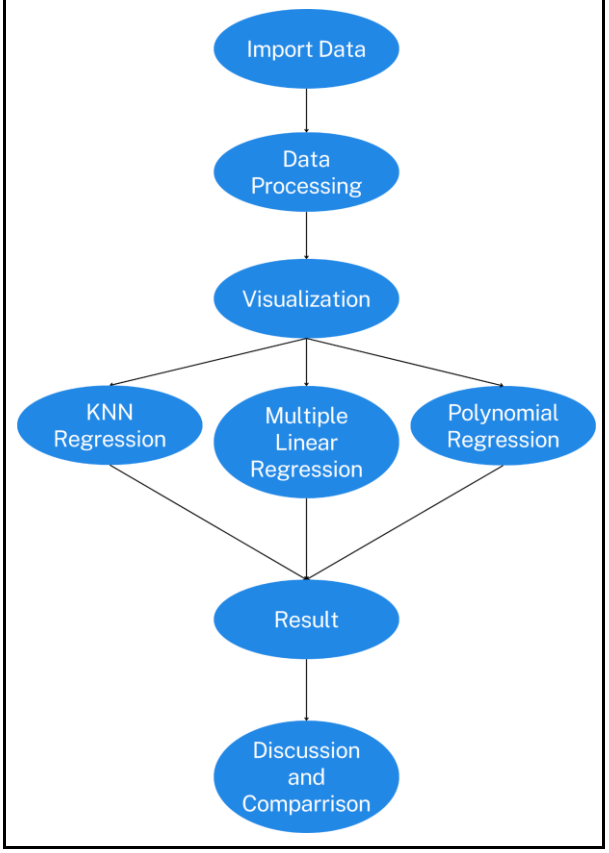
Mean Squared Error is the squared difference between the expected and actual values, averaged. Larger errors are given more weight because it calculates the average squared difference of the errors in a set of predictions[20]. Large errors are penalized more severely than small errors using MSE, which can be significant in some applications.

L. R-squared

R square is a statistical measure that illustrates the percentage of the independent variables' variance[21]. R squared gives a general indication of the goodness-of-fit of the model.

M. Research Flow

Fig.1 Research Flowchart

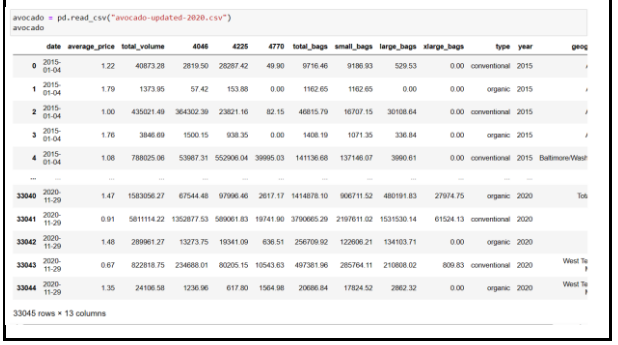


Based on the Flowchart above, the flow of this research is to import the data, next do Data Processing (Correcting the data), visualization(Pie charts, graphs, barcharts, and many more), next we do KNN, MUltiple Linear and Polynomial Regression, which we get the result and lastly we discuss and compare the results.

III. RESULT

A. Data Understanding

Fig 2. Data Import/Reading Data



```
avocado = pd.read_csv("avocado-updated-2020.csv")
avocado
```

	date	average_price	total_volume	4046	4225	4770	total_bags	small_bags	large_bags	xlarge_bags	type	year	geog
0	2015-01-04	1.22	40873.28	2819.50	28287.42	49.90	9716.46	9186.93	529.53	0.00	conventional	2015	
1	2015-01-04	1.79	1373.95	57.42	153.88	0.00	1162.65	1162.65	0.00	0.00	organic	2015	
2	2015-01-04	1.00	435021.49	364302.39	23821.16	82.15	46815.79	16707.15	30108.64	0.00	conventional	2015	
3	2015-01-04	1.76	3846.69	1500.15	938.35	0.00	1408.19	1071.35	336.84	0.00	organic	2015	
4	2015-01-04	1.08	788025.06	53987.31	552906.04	39995.03	141136.68	137146.07	3990.61	0.00	conventional	2015	BaltimoreWash
...	...	...	...	...	...	...	...	...	...	...	...	...	...
33040	2020-11-29	1.47	1583056.27	67544.48	97996.46	2617.17	1414878.10	906711.52	480191.83	27974.75	organic	2020	To
33041	2020-11-29	0.91	5811114.22	1352877.53	589061.83	19741.90	3790665.29	2197611.02	1531530.14	61524.13	conventional	2020	
33042	2020-11-29	1.48	289961.27	13273.75	19341.09	636.51	256709.92	122606.21	134103.71	0.00	organic	2020	
33043	2020-11-29	0.67	822818.75	234688.01	80205.15	10543.63	497381.96	285764.11	210808.02	809.83	conventional	2020	West Te
33044	2020-11-29	1.35	24106.58	1236.96	617.80	1564.98	20686.84	17824.52	2862.32	0.00	organic	2020	West Te

33045 rows × 13 columns

Figure 2 shows that after importing or reading the data (pd.read), “avocado” code is used to call and show the data as a whole from the csv that was imported. There are 33045 rows and 13 columns. The dependent columns will be 'AveragePrice', '4046', '4225', '4770', 'type', 'year', and the independent columns will be 'Total Volume'.

Fig 3. Avocado Info

```
avocado.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 33045 entries, 0 to 33044
Data columns (total 13 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   date            33045 non-null  object
 1   average_price   33045 non-null  float64
 2   total_volume    33045 non-null  float64
 3   4046            33045 non-null  float64
 4   4225            33045 non-null  float64
 5   4770            33045 non-null  float64
 6   total_bags      33045 non-null  float64
 7   small_bags      33045 non-null  float64
 8   large_bags      33045 non-null  float64
 9   xlarge_bags     33045 non-null  float64
 10  type            33045 non-null  object
 11  year            33045 non-null  int64
 12  geography       33045 non-null  object
dtypes: float64(9), int64(1), object(3)
memory usage: 3.3+ MB
```

Figure 3 shows that every column does not have a null count, the column date is object type, average_price, total_volume, 4046, 4225, 4770, total_bags, small_bags,large_bags,xlarge_bags is float 64 type. Type is object, year is int 64 and gepgra[hy is object type.

Fig. 4 Column Change

```
avocado.columns = ['Date', 'AveragePrice', 'Total Volume', '4046', '4225', '4770',
                   'Total Bags', 'Small Bags', 'Large Bags', 'XLarge Bags',
                   'type', 'year', 'region']
avocado.head()
avocado.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 33045 entries, 0 to 33044
Data columns (total 13 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Date          33045 non-null  object
 1   AveragePrice  33045 non-null  float64
 2   Total Volume  33045 non-null  float64
 3   4046          33045 non-null  float64
 4   4225          33045 non-null  float64
 5   4770          33045 non-null  float64
 6   Total Bags    33045 non-null  float64
 7   Small Bags    33045 non-null  float64
 8   Large Bags    33045 non-null  float64
 9   XLarge Bags   33045 non-null  float64
 10  type          33045 non-null  object
 11  year          33045 non-null  int64
 12  region        33045 non-null  object
dtypes: float64(9), int64(1), object(3)
memory usage: 3.3+ MB
```

Figure 4 shows that the columns has been renamed to'Date', 'AveragePrice', 'Total Volume', '4046', '4225', '4770', 'Total Bags', 'Small Bags', 'Large Bags', 'XLarge Bags', 'type', 'year', and 'region'. 4046,4225 and 4770 means PLU Code used on the avocado.

B. Preprocessing Data

Fig 5. Drop Null Data

```
avocado.dropna()
```

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	type	year	region
0	2015-01-04	1.22	40873.28	2819.50	28287.42	49.90	9716.46	9186.93	529.53	0.00	conventional	2015	Albany
1	2015-01-04	1.79	1373.95	57.42	153.88	0.00	1162.65	1162.65	0.00	0.00	organic	2015	Albany
2	2015-01-04	1.00	435021.49	364302.39	23821.16	82.15	46815.79	16707.15	30108.64	0.00	conventional	2015	Atlanta
3	2015-01-04	1.76	3846.69	1500.15	938.35	0.00	1408.19	1071.35	336.84	0.00	organic	2015	Atlanta
4	2015-01-04	1.08	788025.06	53987.31	552906.04	39995.03	141136.68	137146.07	3990.61	0.00	conventional	2015	Baltimore/Washington
...	...	...	...	...	...	...	...	...	...	...	...	...	...
33040	2020-11-29	1.47	1583056.27	67544.48	97996.46	2617.17	1414878.10	906711.52	480191.83	27974.75	organic	2020	Total U.S.
33041	2020-11-29	0.91	5811114.22	1352877.53	589061.83	19741.90	3790665.29	2197611.02	1531530.14	61524.13	conventional	2020	West
33042	2020-11-29	1.46	289961.27	13273.75	19341.09	636.51	256709.92	122606.21	134103.71	0.00	organic	2020	West
33043	2020-11-29	0.67	822818.75	234688.01	80205.15	10543.63	497381.96	285764.11	210808.02	809.83	conventional	2020	West Tex/New Mexico
33044	2020-11-29	1.35	24106.58	1236.96	617.80	1564.98	20686.84	17824.52	2862.32	0.00	organic	2020	West Tex/New Mexico

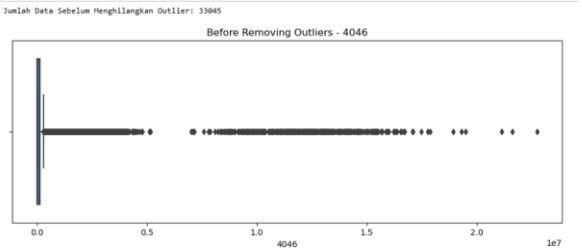
33045 rows × 13 columns

In Figure 5, avocado sales 2022 dataset does not contain any null or missing values, there won't be a need for Handling Missing Values. But, the dropna() function would still be used.

Fig 6. Removing '4046' Outliers

```
import matplotlib.pyplot as plt
import seaborn as sns
print("Jumlah Data Sebelum Menghilangkan Outlier:", len(avocado))
plt.figure(figsize=(12, 4))
sns.boxplot(x=avocado['4046'])
plt.title('Before Removing Outliers - 4046')
plt.show()
Q1 = avocado['4046'].quantile(0.25)
Q3 = avocado['4046'].quantile(0.75)
IQR = Q3 - Q1
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
avocado = avocado[(avocado['4046'] >= lower_bound) & (avocado['4046'] <= upper_bound)]
print("\nJumlah Data Setelah Menghilangkan Outlier:", len(avocado))
plt.figure(figsize=(12, 4))
sns.boxplot(x=avocado['4046'])
plt.title('After Removing Outliers - 4046')
plt.show()
```

```
Jumlah Data Sebelum Menghilangkan Outlier: 33045
```




```
Jumlah Data Setelah Menghilangkan Outlier: 28405
```

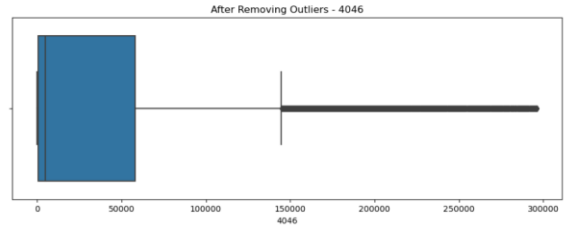


Figure 6 shows the usage of IQR to remove outliers in the 4046 column. The data before removing the outlier is 33045, after removing the outlier 28405 data remains.

Fig 7. Removing '4225' Outliers

```
print("Jumlah Data Sebelum Menghilangkan Outlier:", len(avocado))

plt.figure(figsize=(12, 4))
sns.boxplot(x=avocado['4225'])
plt.title('Before Removing Outliers - 4225')
plt.show()

Q1 = avocado['4225'].quantile(0.25)
Q3 = avocado['4225'].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

avocado = avocado[(avocado['4225'] >= lower_bound) & (avocado['4225'] <= upper_bound)]

print("\nJumlah Data Setelah Menghilangkan Outlier:", len(avocado))

plt.figure(figsize=(12, 4))
sns.boxplot(x=avocado['4225'])
plt.title('After Removing Outliers - 4225')
plt.show()
```

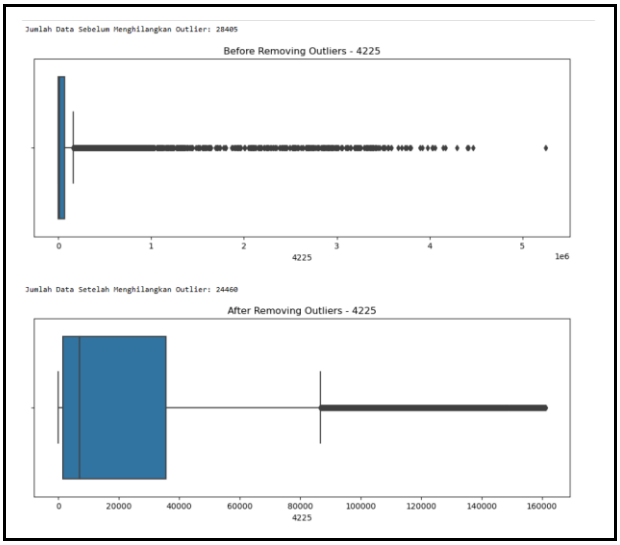


Figure 7 shows the usage of IQR to remove outliers in the 4225 column. The data before removing the outlier is 28405, after removing the outlier 24460 data remains.

Fig 8. Removing '4770' Outliers

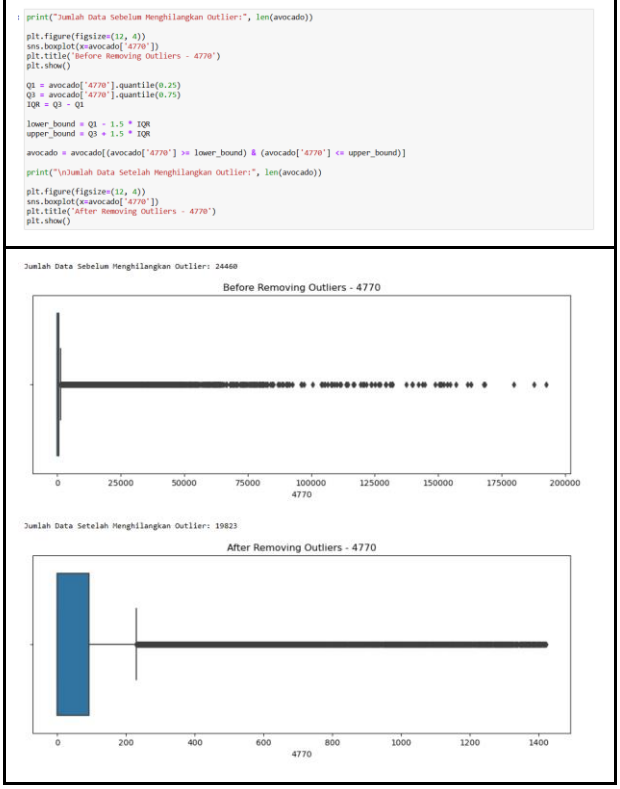


Figure 8 shows the usage of IQR to remove outliers in the 4770 column. The data before removing the outlier is 24460, after removing the outlier 19823 data remains.

Fig 9. Removing 'Total Volume' Outliers

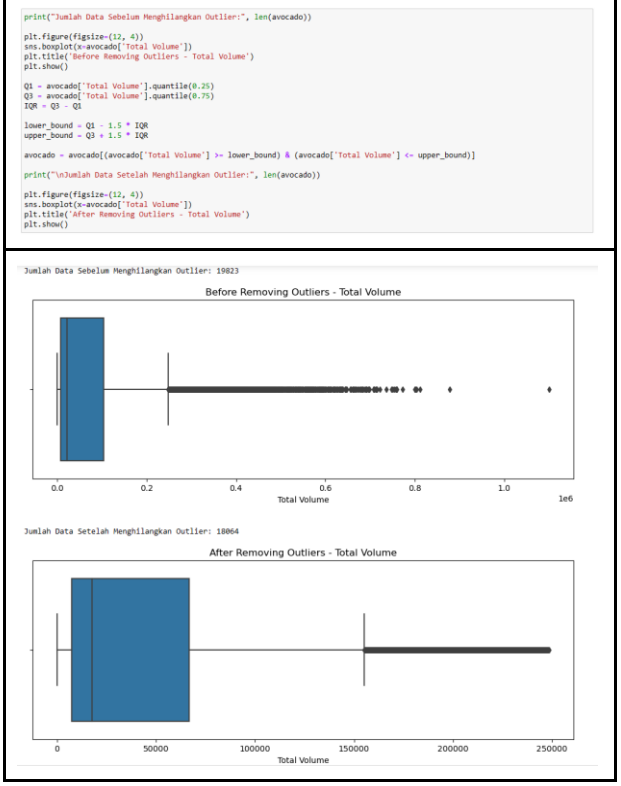


Figure 9 shows the removal of outliers in the Total Volume column, the study uses a boxplot to display the distribution of "Total Volume" and the IQR method to remove outliers. The data before removing the outlier is 19823, after removing the outlier 18064 remains.

Fig 10. Removing 'Average Price' Outliers

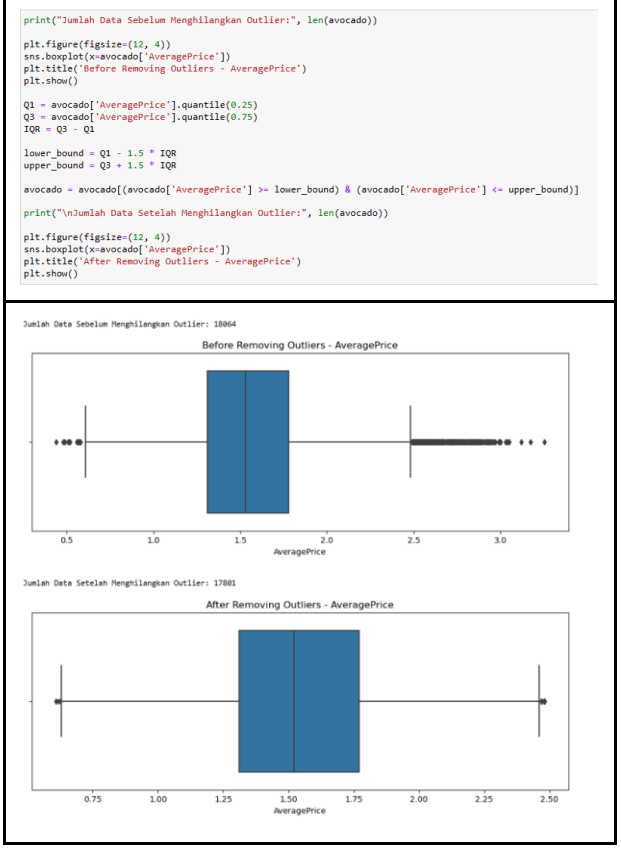


Figure 10 shows the usage of IQR to remove outliers in the Average Price column. The data before removing the outlier is 18064, after removing the outlier 17801 data remains. So now only 17801 data remains to be processed and used in this study's analysis.

Fig 11. Formatting Date

```
avocado['Date'] = pd.to_datetime(avocado['Date'])
datavis = avocado[['AveragePrice', 'Date']]
datavis['Date']= pd.to_datetime(datavis['Date'], infer_datetime_format=True)
print(datavis.dtypes)
```

```
AveragePrice           float64
Date            datetime64[ns]
dtype: object
```

```
C:\Users\calvi\AppData\Local\Temp\ipykernel_5632\2872335283.py:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
```

Figure 11 shows that the code formats Date column object into a datetime64 data type and create a new DataFrame object named datavis that contains only the AveragePrice and Date columns.

Fig 12. Normalization Average Price

```
from sklearn.preprocessing import MinMaxScaler
import pandas as pd
scaler = MinMaxScaler()
avocado['AveragePrice'] = scaler.fit_transform(avocado['AveragePrice'].values.reshape(-1, 1))
avocado.head()
```

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	type	year	region
0	2015-01-04	0.326203	40873.28	2819.50	28287.42	49.90	9716.46	9186.93	529.53	0.0	conventional	2015	Albany
1	2015-01-04	0.631016	1373.95	57.42	153.88	0.00	1162.65	1162.65	0.00	0.0	organic	2015	Albany
3	2015-01-04	0.614973	3846.69	1500.15	938.35	0.00	1408.19	1071.35	336.84	0.0	organic	2015	Atlanta
5	2015-01-04	0.363636	19137.28	8040.64	6557.47	657.48	3881.69	3881.69	0.00	0.0	organic	2015	Baltimore/Washington
7	2015-01-04	0.550802	1505.12	1.27	1129.50	0.00	374.35	186.67	187.68	0.0	organic	2015	Boise

Figure 12 shows a code that is used to scale the AveragePrice column using the MinMaxScaler function and display the first five rows of the DataFrame object after scaling it.

Fig 13. Binning Price Level

```
bin_edges = [0, 0.2, 0.4, 0.6, 0.8, 1.0]
bin_labels = ['Very Low', 'Low', 'Average', 'High', 'Very High']

avocado['Price Level'] = pd.cut(avocado['AveragePrice'], bins=bin_edges, labels=bin_labels, include_lowest=True)

avocado.head()
```

	Date	AveragePrice	Total Volume	4046	4225	4770	Total Bags	Small Bags	Large Bags	XLarge Bags	type	year	region	Price Level
0	2015-01-04	0.326203	40873.28	2819.50	28287.42	49.90	9716.46	9186.93	529.53	0.0	1	2015	Albany	Low
1	2015-01-04	0.631016	1373.95	57.42	153.88	0.00	1162.65	1162.65	0.00	0.0	2	2015	Albany	High
3	2015-01-04	0.614973	3846.69	1500.15	938.35	0.00	1408.19	1071.35	336.84	0.0	2	2015	Atlanta	High
5	2015-01-04	0.363636	19137.28	8040.64	6557.47	657.48	3881.69	3881.69	0.00	0.0	2	2015	Baltimore/Washington	Low
7	2015-01-04	0.550802	1505.12	1.27	1129.50	0.00	374.35	186.67	187.68	0.0	2	2015	Boise	Average

Figure 13 shows a code and result of binning The AveragePrice column into intervals.. A new column called "Price Level" is then created, containing the labels that correspond to each interval. The new column will consist of; 'Very Low', 'Low', 'Average', 'High', and 'Very High'.

Fig 14. Encoding Type Column

```
avocado['type'] = avocado['type'].replace({'conventional': 1, 'organic': 2})
print(avocado)
```

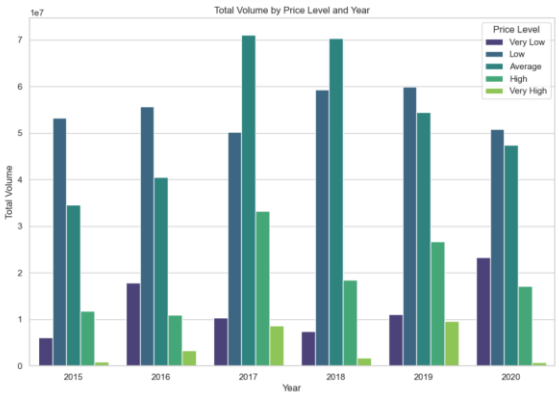
```
            Date  AveragePrice  Total Volume     4046     4225    4770  \
0     2015-01-04      0.326203      40873.28  2819.50 28287.42   49.90
1     2015-01-04      0.631016       1373.95    57.42   153.88    0.00
3     2015-01-04      0.614973       3846.69  1500.15   938.35    0.00
5     2015-01-04      0.363636      19137.28  8040.64  6557.47  657.48
7     2015-01-04      0.550802       1505.12     1.27  1129.50    0.00
...          ...           ...           ...      ...      ...     ...
33033 2020-11-29      0.117647     189187.58 78597.67  9497.22   65.16
33034 2020-11-29      0.657754       5898.33   677.71   912.70    0.00
33035 2020-11-29      0.181818      72128.91  6789.51 31201.09  627.87
33036 2020-11-29      0.454545       3191.59   166.36    89.78    0.00
33038 2020-11-29      0.181818      11883.88   101.71     0.00    0.00

       Total Bags  Small Bags  Large Bags  XLarge Bags  type  year  \
0         9716.46     9186.93      529.53         0.00     1  2015
1         1162.65     1162.65        0.00         0.00     2  2015
3         1408.19     1071.35      336.84         0.00     2  2015
5         3881.69     3881.69        0.00         0.00     2  2015
7          374.35      186.67      187.68         0.00     2  2015
...           ...         ...         ...          ...   ...   ...
33033   101027.53    93625.26     7402.27         0.00     1  2020
33034     4307.92     4301.25        6.67         0.00     2  2020
33035    33510.44    20587.54    11866.23      1056.67     1  2020
33036     2935.45     2618.57      316.88         0.00     2  2020
33038    11782.17    11782.17        0.00         0.00     2  2020

                region
0               Albany
1               Albany
3              Atlanta
5      Baltimore/Washington
7                Boise
...               ...
33033         St. Louis
33034         St. Louis
33035          Syracuse
33036          Syracuse
33038             Tampa
```

Figure 14 shows a code and result when using the replace() method, the values in theType column are changed to 1 for "conventional" and 2 for "organic". The code produces a dataframe with the same columns as the original dataframe, but with 1s and 2s in place of the values in the type column.

C. Data Visualization

Fig 15. Grouping Total Volume by Price

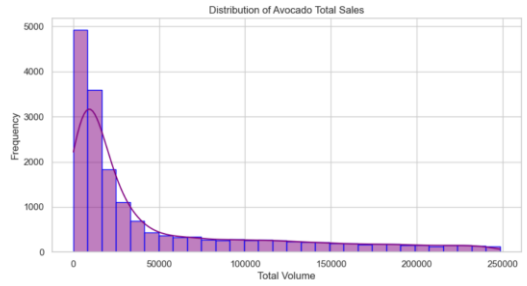
```
total_volume_by_price_year = avocado.groupby(['Price Level', 'year'])['Total Volume'].sum().reset_index()
total_volume_by_price_year.head()

sns.set(style="whitegrid")

plt.figure(figsize=(12, 8))
sns.barplot(x='year', y='Total Volume', hue='Price Level', data=total_volume_by_price_year, palette='viridis')

plt.xlabel('Year')
plt.ylabel('Total Volume')
plt.title('Total Volume by Price Level and Year')

plt.show()
```



Based on Figure 15, the barplot shows the Total Volume by Price level (This is the binning done previously) for each year. Where Avocado total volume peaked in the year 2017 and slowly decreasing by the next years to come.

Fig 16. Histogram Total Sales

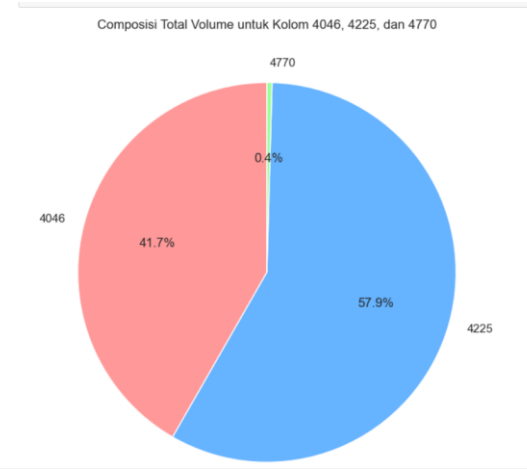
```
plt.figure(figsize=(10, 5))
sns.histplot(avocado['Total Volume'], color='purple', bins = 30, kde= True, edgecolor='blue')
plt.xlabel('Total Volume')
plt.ylabel('Frequency')
plt.title('Distribution of Avocado Total Sales')
plt.show()
```



In figure 16, a histogram is created to show the frequency of the total volume ranging from 0 to 250,000.

Fig 17. Pie Chart Total Volume by 4046, 4225 4770 Column

```
total_4046 = avocado['4046'].sum()
total_4225 = avocado['4225'].sum()
total_4770 = avocado['4770'].sum()

labels = ['4046', '4225', '4770']
sizes = [total_4046, total_4225, total_4770]
colors = ['#ff9999','#66b3ff','#99ff99']

plt.figure(figsize=(8, 8))
plt.pie(sizes, labels=labels, colors=colors, autopct='%1.1f%%', startangle=90)
plt.title('Composisi Total Volume untuk Kolom 4046, 4225, dan 4770')
plt.show()
```



Based on figure 17, a pie chart is used to represent the PLU 4046, 4225, and 4770 from the total volume. PLU 4046 is 41.7%, PLU 4225 is 57.9% and PLU 4770 is 0.4%.

Fig 18. Scatterplot Total Volume and Average Price

```
plt.figure(figsize=(10, 6))
sns.scatterplot(x='Total Volume', y='AveragePrice', data=avocado)
plt.title('Scatterplot: Total Volume vs AveragePrice')
plt.xlabel('Total Volume')
plt.ylabel('Average Price')
plt.show()
```

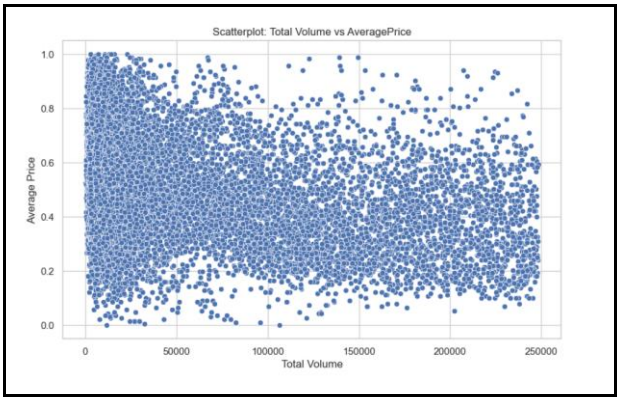
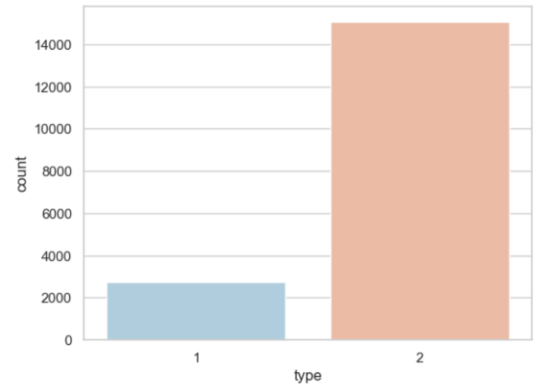


Figure 18 shows the corelation of Total Volume column and Average Price Column, There is no clear correlation between total volume and average price as indicated by the random distribution of data

Fig 19.Countplot Type Column

```
sns.set_style('whitegrid')
sns.countplot(x='type',data=avocado,palette='RdBu_r')
```

```
<Axes: xlabel='type', ylabel='count'>
```



From Figure 19, it shows that type 1 or type conventional has almost 3000 count, while type 2 or type organic on the other hand has over 14000 count..

Fig 20. Heatmap Correlation

```
numerical_columns = avocado.drop(['Date', 'region'], axis=1)
correlation_matrix = numerical_columns.corr()
plt.figure(figsize=(12, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
plt.title('Correlation Heatmap')
plt.show()
```

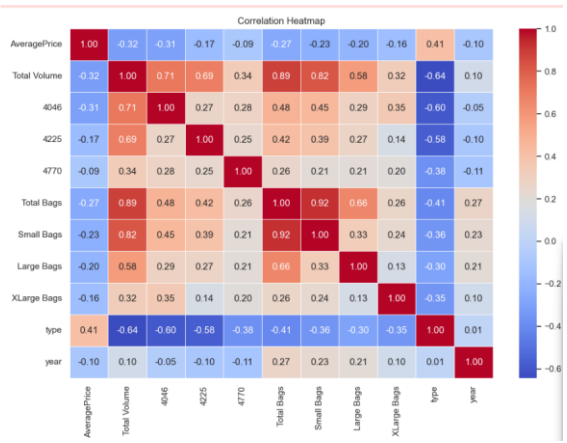


Figure 20 shows a corelation heatmap showing the relationships between avocado dataset columns including Average Price, Total Volume, Different Bag Sizes, Type, and Year. The colors can range from dark blue (-0.64), which means strong negative correlation, to dark red (correlation of 1), which means a perfect positive correlation. For Example, there is a significant positive correlation between the total number of bags and the number of small bags (0.92), indicating that the number of small bags tends to increase along with the total number of bags.

D. Multiple Linear Regression

Fig. 21 MLR Data Preparation

```
X = avocado[['AveragePrice', '4046', '4225', '4770', 'type', 'year']]
y = avocado['Total Volume']
y = y.values.reshape(-1, 1)

scaler = StandardScaler()

X_scaled = scaler.fit_transform(X)
y_scaled = scaler.fit_transform(y)

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y_scaled, test_size=0.2, random_state=42)
print(X_train.shape, X_test.shape)
print(y_train.shape, y_test.shape)
```

```
(14240, 6) (3561, 6)
(14240, 1) (3561, 1)
```

Figure 21 shows a line of code, the StandardScaler function is used to standardize the features and target variable. The standardized data is divided into training and testing sets using the train test split function. The training set contains 14240 rows and 6 columns, while the testing set contains 3561 rows and 6 columns.

Fig. 22. Training Data

```
model = linear_model.LinearRegression()
model.fit(X_train, y_train)
print('Coefficients: ',model.coef_)
print('Intercept: ',model.intercept_)
```

```
Coefficients:  [[-0.0498538   0.57374552  0.56741675  0.07202317  0.0703294   0.19089092]]
Intercept:  [-0.00224199]
```

Figure 22 shows a code that acts to train a linear regression model on the training set. The linear_model.LinearRegression() function is used to create an instance of the linear regression model. THe result will show the coefficient of each columns in the dataset,

Fig. 23. MLR Prediction

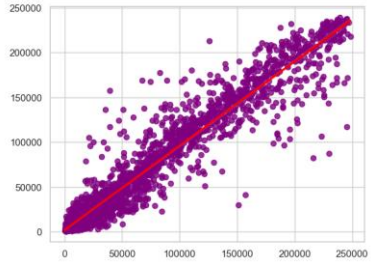
```
y_pred = model.predict(X_test)
print('Predicted: ', y_pred)
```

```
Predicted:  [[-0.25308275]
 [ 1.75275325]
 [-0.13618126]
 ...
 [-0.18343706]
 [ 0.90383144]
 [-0.6384793 ]]
```

Figure 23 shows a prediction, the predict() method is used to generate predicted values of a certain variable based on certain features.

Fig. 24. MLR Scatterplot

```
y_test_inverse = scaler.inverse_transform(y_test.reshape(-1,1))
y_pred_inverse = scaler.inverse_transform(y_pred.reshape(-1,1))
line_color = 'red'
ax = sns.regplot(x=y_test_inverse, y=y_pred_inverse, scatter_kws={'color': 'purple'}, line_kws={'color': line_color})
plt.show()
```



From figure 24, a scatter plot is used to visualize the relationship between the real values and the prediction values from a regression model. the variables show a good correlation as both the lines and scattered data are closely going towards the same way and the data is not that scattered.

E. Polynomial Regression

Fig. 25. polynomial Features

```
from sklearn.preprocessing import PolynomialFeatures
polynomial = PolynomialFeatures(degree = 5)
polynomial_train_x = polynomial.fit_transform(X_train)
polynomial_train_x
```

```
array([[ 1.00000000e+00, -2.20037988e-01, -4.08157893e-01, ...,
         7.37971516e-03,  5.94909850e-03,  4.79581829e-03],
       [ 1.00000000e+00, -3.39822579e-01, -3.66322668e-01, ...,
        -8.54353852e-02, -9.08765886e-03, -9.66643310e-04],
       [ 1.00000000e+00,  7.38238737e-01, -3.69670424e-01, ...,
        -5.38093150e-01,  1.81220912e+00, -6.10322190e+00],
       ...,
       [ 1.00000000e+00, -1.89702226e+00, -3.94808610e-01, ...,
        -1.08768131e-01,  2.14981249e-01, -4.24912491e-01],
       [ 1.00000000e+00, -5.49445612e-01,  3.23026388e+00, ...,
        -1.62859662e+01, -9.96979766e+00, -6.10322190e+00],
       [ 1.00000000e+00,  1.51683858e+00, -4.16894514e-01, ...,
         6.51103994e-01,  2.33667784e+00,  8.38585445e+00]])
```

Figure 25 shows a code that all polynomial combinations of features with a degree less than or equal to the specified degree are produced as a new feature matrix.

Fig. 26. Polynomial Scatterplot

```
y_test_poly_inverse = scaler.inverse_transform(y_test.reshape(-1,1))
y_pred_poly_inverse = scaler.inverse_transform(y_pred_poly.reshape(-1,1))
ax = sns.regplot(x=y_test_poly_inverse, y=y_pred_poly_inverse,
                 order = 2, scatter_kws = {"color" : "pink"}, line_kws = {"color" : "red"})
plt.show()
```

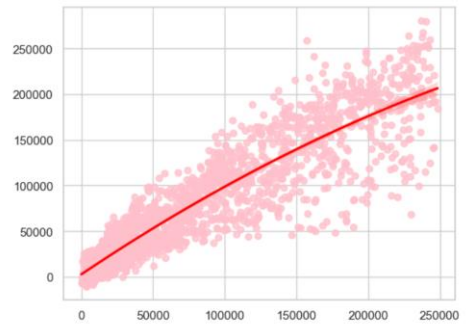
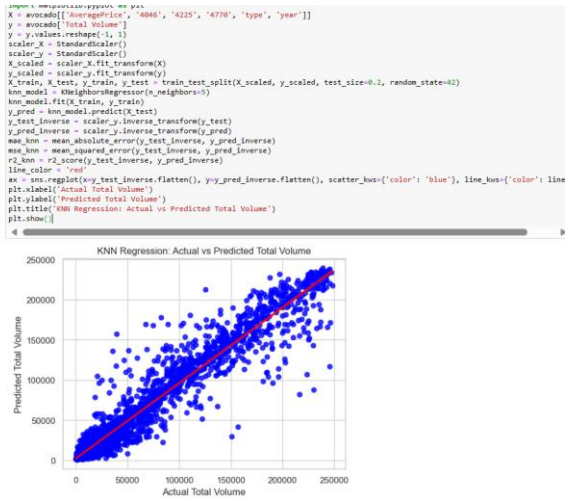


Figure 26 shows a scatterplot based on the Polynomial Regression method, it shows a small positive correlation since the data scatters more than the MLR Scatterplot done previously.

F. K-Nearest Neighbors

Fig 27. KNN Scatterplot



Based on Figure 27 shows a scatterplot with a much stronger correlation than the one shown using MLR or Polynomial Regression, the one using KNN is correlating more positively even though it has several scattered data, the main correlating data is not scattered.

IV. DISCUSSION

On the previous group task, the Regression methods used are MLR and Polynomial, in this study we will compare the two regression methods with KNN regression methods, and see which one is better.

To compare the method, we see the Mean Absolute Error (MAE), Mean Squared Error (MSE), and R-Squared from each regression method. In MAE and MSE, the smaller the number the better the prediction. For R-squared values range between 0 and 1, where a value of 1 indicates a perfect model. The higher the R-squared value, the better the model is at explaining variations in the data.

Fig 28. MAE,MSE, RSquared of Multi Linear Regression

```
print("Mean Absolute Error: %.2f" % mae_multi)
print("Mean Squared Error: %.2f" % mse_multi)
print("R-squared:", r2_multi)
```

```
Mean Absolute Error: 0.26
Mean Squared Error: 0.18
R-squared: 0.8143322822814907
```

Figure 28 discusses the MAE,MSE, RSquared of Multi Linear Regression. The Mean Absolute Error has a value of 0.26 which indicates that the average absolute difference between the observed total volume and the total volume predicted by the model is about 0.26.

The Mean Squared Error has a value of 0.18 which indicates that the average squared difference between the observed total volume and the total volume predicted by the model is about 0.18.

The RSquared has a value of 0.814(Rounded up) which indicates 81.43% of the variation in total volume can be explained by the independent variable (Total Volume)

Fig 29. MAE,MSE, RSquared of Polynomial Regression

```
print("Mean Absolute Error: %.2f" % mae_poly)
print("Mean Squared Error: %.2f" % mse_poly)
print("R-squared:", r2_poly)
```

```
Mean Absolute Error: 0.22
Mean Squared Error: 0.13
R-squared: 0.8730429776166446
```

Figure 29 shows the MAE, MSE and R squared of Polynomial Regression. The MAE has a score of 0.22, the MSE has a score of 0.13 and the R-Squared has a score of 0.873 or 87.3%. The MAE, MSE and R squared percentage of the polynomial regression method has a higher number compared to using Multi Linear Regression, now let us compare it to the KNN regression method.

Fig 30. MAE,MSE, RSquared of K-Nearest Neighbors

```
print("Mean Absolute Error: %.2f" % mae_knn)
print("Mean Squared Error: %.2f" % mse_knn)
print("R-squared:", r2_knn)
```

```
Mean Absolute Error: 8345.46
Mean Squared Error: 242068419.00
R-squared: 0.9372185099931346
```

Figure 30 shows that the Mean Absolute Error is 8345.46, MSE is 242068419.00. However the R squared has the highest percentage out of all three of them, at 0.937 or 93.7%.

V. CONCLUSION

In Conclusion, a comparison between Multiple Linear Regression and Polynomial Regression and K-Nearest Neighbors (KNN) for avocado sales forecasting shows that KNN obtained an R-squared value of 93.7%, a Mean Absolute Error (MAE) of 8345.46, and a Mean Squared Error (MSE) of 242068419.00 when comparing the performance metrics among the three models. even though the polynomial regression model's MAE and MSE are significantly lower than those of the KNN, the R-squared percentage is significantly higher, meaning that KNN explains 93.7% of the variation in total volume, outperforming both Multiple Linear Regression at 81.43% and Polynomial Regression at 87.3%.

ACKNOWLEDGMENT

I would like to acknowledge my lecturer Bu Irmawati for guiding me through making this journal and study. My deepest gratitude goes to Bu Irmawati as she has taught me many important things and information regarding data analytics

REFERENCES

[1] Rincon, J. (2018). Estimating Avocado Sales Using Machine Learning Algorithms and Weather Data.

[2] KORNEV, T. (2020) Avocado Prices (2020). Available at: https://www.kaggle.com/datasets/timmate/avocado-prices-2020/data.

[3] Data Protection Regulation, G. (n.d.). https://gdpr-info.eu

[4] Waskom, M.L. (2021) seaborn: statistical data visualization, https://joss.theoj.org/papers/10.21105/joss.03021.

[5] Jordan, M. (no date) Foundations and Trends® in Machine Learning. Available at: https://www.nowpublishers.com/MAL.

[6] Saleh, H. and Layous, J.A. (2022) Machine Learning -Regression. Available at: https://www.researchgate.net/publication/357992043_Machine_Learning_-Regression.

[7] Hidayati, N. and Hermawan, A. (2021) K-Nearest Neighbor (K-NN) algorithm with Euclidean and Manhattan in classification of student graduation. Available at: https://journal.uny.ac.id/index.php/jeatech/article/view/42777.

[8] Li, Y. (2023) Evolutionary polynomial regression improved by regularization methods. Available at: https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0282029.

[9] Ostertagová, E. (2012)Modelling using Polynomial Regression Available at: https://www.sciencedirect.com/science/article/pii/S1877705812046085.

[10] Tranmer, M. and Murphy, J. (2020) Multiple Linear Regression. Available at: https://hummedia.manchester.ac.uk/institutes/cmist/archive-publications/working-papers/2020/multiple-linear-regression.pdf.

[11] Kaiser, J. (2014) Dealing with Missing Values in Data. Available at: https://www.researchgate.net/publication/304500093_Dealing_with_Missing_Values_in_Data.

[12] Nasserddine, G. and Younis, J. (2023) Detecting Data Outliers with Machine Learning. Available at: https://www.researchgate.net/publication/370816878_Detecting_Data_Outliers_with_Machine_Learning.

[13] King, A.P. and Eckersley, R.J. (2019) Descriptive Statistics II: Bivariate and Multivariate Statistics. Available at: https://www.sciencedirect.com/topics/mathematics/iqr.

[14] Chaudekar, S. (2022) An Overview of Python for Data Analytics. Available at: https://www.academia.edu/93531071/An_Overview_of_Python_for_Data_Analytics.

[15] Singh, D. and Singh, B. (2021) Feature wise normalization: An effective way of normalizing data. Available at: https://www.sciencedirect.com/science/article/abs/pii/S0031320321004878.

[16] Duca, A.L. (2020) Data Preprocessing with Python Pandas. Available at: https://towardsdatascience.com/data-preprocessing-with-python-pandas-part-5-binning-c5bd5fd1b950#:~:text=An%20overview%20of%20Techniques%20for%20Binning%20in%20Python.&text=Data%20binning%20is%20a%20type,sample%20(quantise)%20numeric%20values.

[17] Potdar, K., Pai, C. and Pardawala, T. (2017) A Comparative Study of Categorical Variable Encoding Techniques for Neural Network Classifiers. Available at: https://www.researchgate.net/publication/320465713_A_Comparative_Study_of_Categorical_Variable_Encoding_Techniques_for_Neural_Network_Classifiers.

[18] Zhou, J. et al. (2023) Using the grouping function of machine learning algorithm to reduce the influence of information avoidance tendency during reading behavior. Available at: https://www.researchgate.net/publication/375961134_Using_the_grouping_function_of_machine_learning_algorithm_to_reduce_the_influence_of_information_avoidance_tendency_during_reading_behavior.

[19] Suryanto, A.A. and Muqtadir, A. (2019) PENERAPAN METODE MEAN ABSOLUTE ERROR (MEA) DALAM ALGORITMA REGRESI LINEAR UNTUK PREDIKSI PRODUKSI PADI .

[20] Hodson, T.O. and Foks, S.S. (2021) Mean Squared Error, Deconstructed. Available at: https://agupubs.onlinelibrary.wiley.com/doi/full/10.1029/2021MS002681.

[21] Chicco, D., Jurman, G. and Warrens, M.J. (2021) The coefficient of determination R-squared is more informative than SMAPE, MAE, MAPE, MSE and RMSE in regression analysis evaluation. Available at: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8279135/.