

**Principles of Data Science and Computing Systems**

**10204280**

**Section (3)**

**Application of Data Science Life Cycle**

**Submitted to**

Dr. Murad A. Yaghi

**Submitted on**

January 30th, 2023

**Submitted by**

Marwan Tareq Shafiq Al Farah

**Student ID**

21110011

**Fall 2022 - 2023**

**Table of Content**

[***Report*** 3](#_Toc125880250)

[**Introduction** 3](#_Toc125880251)

[**Type of Machine Learning** 3](#_Toc125880252)

[**Supervised Machine Learning** 4](#_Toc125880253)

[**Computing Systems in Cloud Computing Services** 4](#_Toc125880254)

[**The Data Science Life Cycle of the Developed Prediction Model** 5](#_Toc125880255)

[**Data Preprocessing Steps and Justifications** 9](#_Toc125880256)

[**Linear Regression Model** 11](#_Toc125880257)

[**Classification Model** 12](#_Toc125880258)

[**Comparison Between the Different Models** 14](#_Toc125880259)

[**Linear Regression Model with Different Learning Rates and Different Normalizations** 14](#_Toc125880260)

[**KNN Model with Different k Values and Different Normalizations** 15](#_Toc125880261)

[**Analysis of the Results** 16](#_Toc125880262)

[**The Effectiveness of Different Models** 17](#_Toc125880263)

[**Conclusion** 18](#_Toc125880264)

[***References*** 19](#_Toc125880265)

# ***Report***

**Introduction:**

This report describes the data analysis and prediction models created for a food trading company. The company’s data describes various items that were sold last month at various retailers throughout the country. By gathering, cleaning, manipulating, preprocessing, and analyzing diverse datasets using cutting-edge technologies, platforms, and techniques, the purpose of this analysis is to get deeper insights that lead to innovative solutions. Linear Regression was used to predict total product sales, while KNN (K-Nearest Neighbors) was used to predict the store from which the product was purchased from. The prediction models’ performance was expressed using appropriate evaluation metrics and visual representations. Different optimization strategies were also applied to increase the models’ performance. This report will describe the main differences between supervised and unsupervised learning, regression, and classification techniques, as well as discuss the prediction models and their results.

**Type of Machine Learning:** (Deula, 2015; Seldon, 2021; Alteryx, 2022)

|  |  |  |
| --- | --- | --- |
|  | Supervised Learning | Unsupervised Learning |
| Definition | A type of machine learning in which a model is trained on a labeled dataset to make predictions about new, unlabeled data. When given input and output pairs, the model learns a mapping from input to output | A type of machine learning in which a model is trained on an unlabeled dataset to discover patterns or structure in the data. The model is not given a specific output, and the purpose is to find interesting features or relationships in the input data |
| Application (use) | Used for predictions, such as forecasting the price of a house based on particular attributes | Used to uncover patterns or structure in data, such as grouping similar data points together or discovering correlations between distinct variables |
| Strengths | Model evaluation is simple since it may be done by comparing predictions to true values | Can uncover hidden patterns or links in the data that might otherwise go undetected |
| Limitations | Relies on labeled data and may not be able to generalize well to new, unseen data | Model evaluation might be difficult because the actual outcome is unknown |
| Common Algorithms | Classification and Regression | Clustering, Dimensionality Reduction, Association Rule Learning |

**Supervised Machine Learning:** (Sakshi, 2021; Terra, 2022a, 2022b)

|  |  |  |
| --- | --- | --- |
|  | Classification | Regression |
| Type of Learning | Supervised learning, used to predict categorical data | Supervised learning, used to predict continuous data |
| Application (use) | Predicting categorical output variables, such as whether or not an email is spam | Predicting continuous output variable, such as the price of a property based on particular characteristics |
| Strengths | Can handle categorical variables and performs better with discrete values | Can handle continuous variables and performs better with numeric values |
| Limitations | Sensitive to uneven class composition | Sensitive to outliers |
| Common Algorithms | Decision Trees, Random Forests, Naïve Bayes, Logistic Regression | Linear Regression, Polynomial Regression, Support Vector Regression, Random Forest Regression |

**Computing Systems in Cloud Computing Services:** (CLOUD ACADEMY TEAM, 2018; NetApp, 2020; ACG Technical Editors Team, 2022; Gil, 2022; Jones, 2022; Microsoft, 2022)

Two of the most popular cloud computing services are Amazon Web Services (AWS) and Microsoft Azure. Both of these services provide a diverse set of computer systems and resources, such as virtual machines, storage, and databases. In this comparison, we will concentrate on the structure, GPU, CPU, storage devices used, and other hardware features of these two services.

1. **Structure:** A distributed architecture is used by both AWS and Azure to enable scalability and fault tolerance. AWS has a global infrastructure organized into regions and availability zones, whereas Azure employs a similar structure known as regions and availability sets.
2. **GPU:** AWS provides GPU instances for a wide range of tasks, including machine learning, gaming, and scientific computing. These instances are powered by NVIDIA GPUs and come in the following configurations: P2, P3, G3, and G4. Azure also provides GPU instances, which are powered by NVIDIA GPUs and come in the following configurations: NC, NCv2, NCv3, and ND.
3. **CPU:** AWS provides a variety of CPU options for its instances, including Intel Xeon and AMD EPYC processors. Azure also provides a number of CPU alternatives, including Intel Xeon and AMD EPYC processors.
4. **Storage Devices:** Elastic Block Store (EBS) for block storage, Simple Storage Service (S3) for object storage, and Elastic File System (EFS) for file storage are all available through AWS. Azure has storage choices that are similar, such as Azure Disk Storage for block storage, Azure Blob Storage for object storage, and Azure Files for file storage.

|  |  |  |
| --- | --- | --- |
| Feature | AWS | Azure |
| Structure | Physical servers, virtual machines, and containers are used to create a virtualized infrastructure | Physical servers, virtual machines, and containers are used to create a virtualized infrastructure |
| GPU | P3, G3, and G4 EC2 instances that support GPUs are available. These instances are powered by NVIDIA GPUs | NVIDIA GPUs power the N-series virtual machines. These virtual machines are intended for computationally intensive applications such as machine learning, scientific simulation, and video transcoding |
| CPU | EC2 instances can be ordered with a range of CPUs, including Intel and AMD-based processors | Azure virtual machines can be configured with a range of CPUs, including Intel and AMD-based processors |
| Storage Devices | AWS provides a number of storage solutions, including EBS (Elastic Block Store) for long-term data storage, S3 (Simple Storage Service) for object storage, and Glacier for long-term archive storage | Azure provides a number of storage choices, such as managed drives for long-term data storage, Azure Blob Storage for object storage, and Azure Archive Storage for long-term archival storage |
| Other Hardware Aspects | Elastic Load Balancing, Elastic Block Store, and Elastic File System are further hardware-related services offered by AWS | Azure also offers hardware-related services including Azure Load Balancer, Azure Disk Storage, and Azure File Storage |

**The Data Science Life Cycle of the Developed Prediction Model:** (Deepali, 2022; Hotz, 2022; Online Manipal Editorial Team, 2022)

The data science life cycle is a process that involves several stages to successfully develop and implement a prediction model. The developed prediction model in this case is KNN (K-Nearest Neighbors), and the data science life cycle for this model would involve the following stages:

1. **Understanding the Problem:** This is the first step in the data science life cycle. It involves identifying the problem that needs to be solved and understanding the goals and objectives of the project. In this case, the problem is to predict the store from which the item was bought from.
2. **Gathering Data:** Once the problem is understood, the next step is to gather the necessary data for the project from the data sources available. The data is usually in text, Tab Separated Values (TSV), or Comma Separated Value (CSV) format. In the case of the developed model, the food trading company provided us with a CSV file containing the dataset that was stored into a pandas’ DataFrame.
3. **Cleaning Data:** After the data is gathered, it's important to clean and preprocess it to ensure that it is in the appropriate format for analysis. This step involves checking for missing or duplicate data, removing outliers, replacing values, and handling any other issues that may affect the quality of the data. This phase is time-consuming yet crucial because the data utilized usually determines the model's dependability. In the case of the developed model, each preprocessing step is explained in the next section.
4. **Exploring Data:** Once the data is cleaned, the next step is to explore and analyze it. This step involves using various techniques such as visualization and statistical analysis to understand the underlying patterns and relationships in the data. Below are some of the charts that were used to better help explore the data understanding the content of the dataset:

**Numeric Data Visualizations:**

Timeline

Description automatically generated

Figure 1 – I\_MRP\_US

A picture containing text

Description automatically generated

Figure 2 – I\_O\_Sales

A picture containing table

Description automatically generated

Figure 3 – I\_Vis

A picture containing timeline

Description automatically generated

Figure 4 – I\_W

Timeline

Description automatically generated

Figure 5 – O\_Establ\_Y

**Categorical Data Visualizations:**

**Chart, bar chart

Description automatically generated**Chart, bar chart

Description automatically generated

Figure 6 – I\_category Figure 7 – I\_Fat\_C

Chart, bar chart

Description automatically generatedChart, bar chart

Description automatically generated

Figure 8 – I\_Recalled Figure 9 – O\_Id

Chart, bar chart

Description automatically generatedChart, bar chart

Description automatically generated

Figure 10 – O\_Id Figure 11 – O\_Loc\_T

Chart, bar chart

Description automatically generatedChart, bar chart

Description automatically generated

Figure 12 – O\_Size Figure 13 – O\_T

1. **Feature Engineering:** Following the exploration of the data, the following stage is to choose the relevant features that will be employed in the model. At this point, the Data Scientist must select the critical features that will directly aid the model's prediction because, in some cases, reducing the dimension of the data set is necessary because not every value and feature is required for the prediction of the results.
2. **Modeling Data:** With the relevant features selected, the next step is to build the model. In this case, a K-Nearest Neighbors Classifier is used, which is initialized with k = 3, 7, and then 13. The model is then trained on the selected features and also split the data into training and testing sets for validation of our model. We used the KNN algorithm as it is suited for the classification problems at hand.
3. **Interpreting Data:** The final step is to interpret and evaluate the performance of the model. This step involves using various metrics such as accuracy, precision, recall, and F1-score to evaluate the model's performance. This step helps in understanding how well the model is performing and what improvements can be made to it. The aim of this stage is to make the model’s outcomes comprehensible to non-technical stakeholder, as the data scientist could utilize various visualization methods (such as bar charts, box plots, radars, etc…) to clarify the result and make it easier for them to understand it. The ultimate aim of this phase is to provide accurate dependable, and easily understandable results that address the food trading company’s objectives.

**Data Preprocessing Steps and Justifications:**

The data preprocessing techniques utilized for building prediction models in the given code include several steps that help to clean, handle and transform the data so that it is suitable to be used as input for a machine learning model. These steps are important to improve the performance and accuracy of the model.

1. **Column Dropping:** The code uses the .drop() method to drop the ‘I\_MRP\_JD’ column from the original DataFrame. This technique is used to remove unnecessary or irrelevant columns, in this case redundant data with the ‘I\_MRP\_US’, from the data that do not add any value to the prediction model, because removing irrelevant columns can improve the performance of the model by reducing the complexity of the data and removing any noise that might affect the model’s accuracy.
2. **Data Cleaning:** data cleaning is performed on the ‘I\_Fat\_C’ column of the DataFrame ‘df1’ using the .replace() method. Specifically, the code replaces the values ‘LF’ and ‘reg’ with ‘Low Fat’ and ‘Regular’ respectively. The .replace() method is used to replace values in a DataFrame based on the dictionary {‘LF’: ‘Low Fat’, ‘reg’: ‘Regular’} when it is passed as an argument to the method, where the keys ‘LF’ and ‘reg’ represent the values to be replaced and the values ‘Low Fat’ and ‘Regular’ represent the replacement values. This data cleaning step is important because it ensures that the data is consistent and accurate. In this case, it ensures that the values ‘LF’ and ‘reg’ are replaced with ‘Low Fat’ and ‘Regular’ respectively to standardize the data and make it more consistent, thus improving the accuracy of the model’s predictions. This specific cleaning step is particularly useful especially if the data was entered manually or by different people, as it’s common for people to make typos or abbreviate words.
3. **Handling Missing Data:**
   1. Handling missing values is performed on the ‘I\_W’ column of the DataFrame ‘df1’ using the .loc() method and the .fillna() method. The first step in handling missing values is to identify any missing values. The code uses the .notnull() method to select all the rows where the ‘I\_W’ column is not null. Then it uses the .loc() method to update the ‘I\_W’ column of those rows with the first value of the ‘I\_W’ column for each ‘I\_Id’ group. This step ensures that the ‘I\_W’ column is filled with the first value for each ‘I\_Id’ group, which can help to preserve the integrity of the dataset. The second step is to fill in any remaining missing values. The code uses the .fillna() method to fill in any remaining missing values in the ‘I\_W’ column with the mean of the column. This step ensures that all the missing values in the column are filled in with a suitable value, which can help to improve the performance of the model. This specific handling missing values step is useful when the data is large and the missing values are not so many, as it’s common for machine learning models to not work well with missing data. This step can help to preserve the integrity of the dataset and make it more complete, which can improve the performance of the model.
   2. Handling missing values in the ‘O\_Size’ column is performed using a combination of techniques, including data splitting, machine learning and data concatenation. The first step is data splitting, where the data is split into training and testing sets using the train\_test\_split() function. The code uses the .notnull() method to select all the rows where the ‘O\_Size’ column is not null. Then it uses the .drop() method to remove the ‘O\_Size’ column from the selected rows and store it in the variable ‘X’. Also, it uses the same method to select all the values of ‘O\_Size’ where it is not null and store it in the variable ‘y’. The second step is using machine learning, where the code uses the Random Forest Classifier algorithm to train the classifier using the training data, and then uses the classifier to predict the missing values in the ‘O\_Size’ column of the DataFrame ‘df2’ using the .predict() method. The third step is data concatenation, where the code concatenates the ‘O Size’ column of df2 with the values of df1 where ‘O\_Size’ is not null using the pd.concat() method. Then it updates the ‘O\_Size’ column of df3 with the concatenated values using the assignment operator. This step ensures that the missing values in the ‘O\_Size’ column are filled in with the predicted values from the machine learning model, which can help to preserve the integrity of the dataset and improve the performance of the model.
4. **Data Splitting:** The code uses the train\_test\_split() function to split the data into training and testing sets with a test size of 20%. This technique is used to divide the data into separate sets for training and testing the model, which ensures that the model is trained and evaluated on different data. Data splitting is important to ensure that the model is not overfitting, which can lead to poor generalization performance on unseen data.
5. **Factorization:** The code uses the .apply() method and the pd.factorize() function to factorize all the columns in ‘cat\_columns’ and store the result in the same columns. This technique is used to convert categorical variables into numerical variables, which allows the use of machine learning algorithms that require numerical inputs. Factorization is important as many machine learning algorithms can only work with numerical data, and converting categorical variables into numerical variables can help to improve the performance of the model.
6. **Data Normalization:** The code uses the MinMaxScaler and StandardScaler to normalize the data. The MinMaxScaler scales the data between 0 and 1, while the StandardScaler standardizes the data by subtracting the mean and dividing by the standard deviation. Normalization is a common step in machine learning to ensure that all input features are on a similar scale and prevent large scale features from dominating the model.

In conclusion, data preprocessing is a crucial step in building prediction models as it helps to clean, handle, and transform the data so that it is suitable to be used as input for a machine learning model. The use of these preprocessing techniques can lead to improved performance by removing irrelevant data, handling missing values, and converting categorical variables into numerical variables. A performance evaluation and comparison should be done to evaluate the impact of these preprocessing techniques on the specific dataset and model being used.

**Linear Regression Model:** (Yale, 2011; Statistics Solutions, 2013; HAYES, 2022; IBM, 2022)

Linear regression is a mathematical approach for analyzing the relationship between a dependent variable and one or more independent variables. It is a form of supervised learning algorithm used for regression problems. The purpose of linear regression is to determine the best-fitting line through a set of data points. A mathematical equation of the form:

y = theta0 + theta1\*x1 + theta2\*x2 + ... + thetan\*xn

Where y is the dependent variable, theta0 is the constant or the y-intercept, and theta1, theta2, ..., thetan are the coefficients of the independent variables x1, x2, ..., xn. The goal is to find the values of theta0, theta1, theta2, ..., thetan that minimize the difference between the predicted values and the actual values.

Single linear regression is a type of linear regression that involves only one independent variable. In this case, the goal is to find the best-fitting line that describes the relationship between the dependent variable and the single independent variable. The equation for single linear regression is:

y = theta0+theta1\*x

Where y is the dependent variable, theta0 is the constant, theta1 is the coefficient of the independent variable x, and x is the independent variable.

In contrast, multiple linear regression involves more than one independent variable. The goal in this example is to select the line that best reflects the relationship between the dependent variable and all of the independent variables. The multiple linear regression equation is:

y = theta0 + theta1\*x1 + theta2\*x2 + ... + thetan\*xn

Where y is the dependent variable, theta0 is the constant, theta1, theta2, ..., thetan are the coefficients of the independent variables x1, x2, ..., xn.

One way to optimize the linear regression is by using the gradient descent algorithm. Gradient descent is an optimization algorithm that helps to minimize the cost function of the linear regression model. The cost function is a measure of how well the model is fitting the data. In the case of linear regression, the cost function is the sum of the squared differences between the predicted values and the actual values, also known as mean squared error (MSE). The gradient descent algorithm updates the parameters of the model (in this case, theta) to minimize the cost function.

The code provided is an implementation of the gradient descent algorithm for linear regression. The class SGDRegressor has three main methods: fit, predict, and evaluation. The fit method updates the theta values to minimize the cost function using the gradient descent algorithm. The predict method predicts the output value for the given input. The evaluation method prints the accuracy of the model, mean squared error, root mean squared error, and mean absolute error. Alpha is a parameter in the gradient descent algorithm that determines the step size of the updates to theta values and it is also known as the learning rate. A smaller value of alpha will result in smaller updates to theta values, which can lead to a slower convergence of the algorithm, but it can also help to avoid overshooting the minimum of the cost function while a larger value of alpha will result in larger updates to theta values, which can lead to a faster convergence of the algorithm, but it can also lead to overshooting the minimum of the cost function, in this way the optimization of the linear regression model can be changed by adjusting the value of alpha.

The fit method takes the X\_train, y\_train, iters, alpha, and theta and it starts by initializing the theta values and creating an array to store the cost values. It then iterates a given number of times (iters) and in each iteration, it calculates the error between the predicted values and the actual values. Then it calculates the cost by dividing the error by the total number of data points. The theta values are then updated using the gradient descent algorithm. Theta values are updated in the direction of the negative gradient of the cost function. The new theta values are then used to calculate the cost again and the cost is stored in the cost\_array. After each iteration, the cost\_array is checked to see if the current cost is less than the minimum cost so far. If it is, the current theta values are stored as the minimum theta values. After all the iterations are complete, the theta values are set to the minimum theta values, and the cost\_array and the number of iterations are returned.

The predict method takes in a test dataset (X\_test) and calculates the predicted values by multiplying the test dataset with the theta values. The method then returns the predicted values.

The evaluation method takes in the normalization type, the actual values of the test dataset (y\_test) and the predicted values (y\_pred). It then prints out the accuracy of the model using the r2\_score function, the mean squared error using the mean\_squared\_error function, the root mean squared error using the square root of the mean squared error, and the mean absolute error using the mean\_absolute\_error function.

In summary, the code provided is a simple implementation of the linear regression algorithm using the gradient descent optimization method. It can be used for both single and multiple linear regression problems. The evaluation method provides a way to evaluate the performance of the model, and the predict method allows for making predictions on new data.

**Classification Model:** (Javatpoint, 2019, 2021; Emeritus, 2021; Joby, 2021)

Classification is a method for forecasting a category output variable from one or more input variables. The goal of classification is to learn a decision boundary that separates the different classes in the data. Classification can be used for problems such as image classification, spam detection, or credit fraud detection. Some of the main characteristics of classification include:

* Classification is used to predict categorical output variables, meaning that the output variable can take only a limited number of discrete values. For example, a classification model can predict whether an email is spam or not, or whether a customer will default on a loan or not.
* In classification, the goal is to learn a decision boundary that separates the different classes in the data. This can be done using various types of classification algorithms such as logistic regression, decision trees, random forests, Naive Bayes, etc.
* Classification models strive to maximize prediction accuracy. In classification problems, the most popular evaluation metrics are accuracy, precision, recall, and fl- score. The percentage of right predictions is measured by accuracy, the percentage of true positives within predicted positives is measured by precision, the percentage of true positives within actual positives is measured by recall, and the fl-score is the harmonic mean of precision and recall.
* Classification models can be used for a wide range of problems such as image classification, natural language processing, speech recognition, and many more.
* Some classification problems may have imbalanced classes, meaning that the number of examples in one class is much larger than the other. This can cause the model to be biased towards the majority class, thus metrics like precision, recall, fl-score, etc are important to evaluate the model performance.

The K-nearest neighbors (KNN) technique is a supervised machine learning algorithm that is often used both for classification and regression. In this algorithm, the main idea is to find the K-nearest data points in the feature space for a given query point and then make a prediction based on the majority class or average value of the K-nearest data points.

KNN is used in classification to predict a categorical class label for a new data point. A KNN classifier, for example, may predict the class label (i.e. type of flower) for a new data point based on the class labels of its K-nearest neighbors given a set of labeled data points representing different types of flowers. The class label is predicted using the majority vote of the K-nearest neighbors. As the prediction, the class label with the most votes is chosen.

In the code, the KNeighbors Classifier class is defined and initialized with k = 3, 7, then 13. The ‘fit’ method is used to store the training data and the ‘predict’ method is used to make predictions on new data points. The ‘predict’ method finds the k nearest neighbors by computing the distances using numpy’s np.linalg.norm() function.

For each row in the test set, the code finds the k nearest neighbors by computing the distances using numpy’s np.linalg.norm() function. Then it finds the nearest\_neighbor\_ids by sorting the distances and taking the first k elements. With the nearest\_neighbor\_ids, it finds the corresponding class labels in y\_train. The mode of the nearest neighbors is then taken as the prediction for that row.

In the end, the code shows how to evaluate the model. It uses several evaluation metric like accuracy, precision, recall and F1 score to evaluate the model performance.

Finally, KNN is a simple yet effective method that can be applied to both classification and regression issues. The results are simple to implement and analyze, but the training data must be stored in a huge quantity of memory. The algorithm's performance can also be affected by the value of k and the distance measure used. As a result, it is critical to determine the best k value and distance measure for a given problem.

**Comparison Between the Different Models:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Type | Training Method | Prediction Method | Evaluation Metrics |
| KNeighborsClassifier | Classification | Storing training data, and computing distances between testing and training data | Using the k nearest neighbors (determined by Euclidean distance) to make predictions for each row in the test set | Accuracy, Precision, Recall, F1-score |
| SGDRegressor | Regression | Updating the parameters (theta) using the Gradient Descent to minimizing the cost function | Using these learned parameters (theta) to make predictions for each row in the test set | R2, MSE, RMSE, MAE |

**Linear Regression Model with Different Learning Rates and Different Normalizations:**

The number of iterations will be constant, and it will be 10,000 iteration.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Normalization | Alpha (LR) | R2 | MSE | RMSE | MAE |
| Without Normalization | 0.001 | -4.0956\*107 | 1.1987\*1014 | 1.0949\*107 | 1.0921\*107 |
| 0.01 | -4.0971\*109 | 1.1992\*1016 | 1.0951\*108 | 1.0923\*108 |
| 0.125 | -6.4021\*1011 | 1.8738\*1018 | 1.3689\*109 | 1.3654\*109 |
| Min-Max Normalization | 0.001 | 0.2698 | 0.0125 | 0.1119 | 0.0867 |
| 0.01 | 0.3793 | 0.0107 | 0.1033 | 0.0779 |
| 0.125 | 0.3926 | 0.0104 | 0.1021 | 0.0772 |
| Z-Score | 0.001 | 0.3759 | 0.6273 | 0.7920 | 0.5978 |
| 0.01 | 0.3925 | 0.6106 | 0.7814 | 0.5901 |
| 0.125 | 0.3926 | 0.6105 | 0.7813 | 0.5903 |

The table above shows the results of a linear regression model trained and evaluated with different learning rates (alpha) and different types of normalization applied to the input data. The evaluation metrics shown are R-squared, mean squared error (MSE), root mean squared error (RMSE), and mean absolute error (MAE). The first column of the table shows the type of normalization applied to the input data, which can be no normalization, Min-Max normalization, or Z-score normalization. The second column shows the learning rate (alpha) used in the training process, and the remaining columns show the evaluation metrics calculated on the test set. The model’s performance varies depending on the learning rate and the normalization applied.

Chart, scatter chart

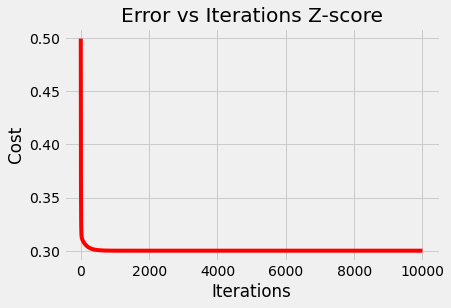
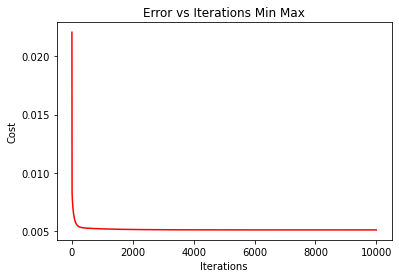
Description automatically generated

Figure 14 – GD & WN Figure 15 – GD & MM Figure 16 - GD & ZS

These charts represent the relation between the cost of the gradient descent algorithm with the number of iterations with 3 normalizations (Without Normalization, Min-Max Normalization, and Z-score Normalization) when alpha is equal to 0.125 and the number of iterations is equal to 10,000. The analysis of the results of both the table and the charts will be provided in the analysis of the results section.

**KNN Model with Different k Values and Different Normalizations:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Normalization | K value | Accuracy | Precision | Recall | F1-Score |
| Without Normalization | 3 | 0.7208 | 0.5584 | 0.5030 | 0.5069 |
| 7 | 0.7355 | 0.5671 | 0.4824 | 0.4667 |
| 13 | 0.7390 | 0.5229 | 0.4649 | 0.4389 |
| Min-Max Normalization | 3 | 1.0 | 1.0 | 1.0 | 1.0 |
| 7 | 1.0 | 1.0 | 1.0 | 1.0 |
| 13 | 1.0 | 1.0 | 1.0 | 1.0 |
| Z-Score | 3 | 0.9965 | 0.9944 | 0.9965 | 0.9954 |
| 7 | 0.9941 | 0.9894 | 0.9956 | 0.9925 |
| 13 | 0.9935 | 0.9883 | 0.9954 | 0.9918 |

The table above shows the results of a KNN (K-Nearest Neighbours) model trained and evaluated with different values of K (number of nearest neighbours considered) and different types of normalization applied to the input data. The evaluation metrics shown are accuracy, precision, recall, and F1-score. The first column of the table shows the type of normalization applied to the input data, which can be no normalization, Min-Max normalization, or Z-score normalization. The second column shows the K value used in the model, and the remaining columns show the evaluation metrics calculated on the test set.

**Chart, bar chart

Description automatically generatedChart, radar chart

Description automatically generated**

Figure 17 – KNN Bar Figure 18 – KNN Radar

These charts represent the evaluation measures (accuracy, precision, recall, and f1-score) of the KNN model with 3 normalizations (Without Normalization, Min-Max Normalization, and Z- score Normalization) when the number of neighbours is equal to 13. The analysis of the results of both the table and the charts will be provided in the following section.

**Analysis of the Results:**

From the first table, it can be seen that the model’s performance varies depending on the learning rate and the normalization applied. Without normalization, the model performed poorly with R-squared values between -4.0956\*107 and -6.4021\*10¹¹ and large MSE, RMSE and MAE values. R-squared is a measure of how well the model fits the data, where a value of 1 represents a perfect fit and a value less than 0 indicates that the model is a poor fit. In this case, the negative R-squared values indicate that the model is a very poor fit for the data when no normalization is applied.

The MSE, RMSE, and MAE values are also extremely high, which indicates that the model is making large errors when predicting the target variable. The high values of these errors are likely due to the large scale differences between the input features. Without normalization, the different input features could be on different scales, which can cause the model to be sensitive to certain features that have large values.

When Min-Max normalization is applied, the performance of the model improves significantly. The R-squared values range from 0.2698 to 0.3926, indicating a better fit to the data. The MSE, RMSE, and MAE values are also significantly lower, which indicates that the model is making smaller errors when predicting the target variable with each iteration. Min-Max normalization scales the input features to a specific range, usually between 0 and 1, which can help to mitigate the effect of features with large values and improve the model’s performance. Finally, it was also noticed that when the alpha increased the accuracy of the model increased as well, while the MSE, RMSE, MAE decreased.

Z-Score normalization also improves the model’s performance similarly to Min-Max normalization but with slightly lower MSE, RMSE and MAE values. Z-Score normalization standardizes the input features by subtracting the mean and dividing by the standard deviation. This can help to centre the input features around 0 and make them have similar scales, which can also improve the model’s performance. Finally, it was also noticed that when the alpha increased the accuracy of the model increased as well, while the MSE, RMSE, MAE decreased.

As for the charts (Figure 14-16), we can see that when the data was not normalized, the cost of that algorithm increased exponentially, in contrary to the two other charts, and that the number of iterations was only about 25 iterations as it was unable to calculate the cost after the 25 iterations although it was given that its number of iterations is 10,000 due to the extremely high values of the cost and of theta. As for the charts when the data was normalized using Min-Max and Z-score Normalizations, we can see that with each iteration the cost kept decreasing until almost remaining stable, which means that the cost reached a very low value.

As for the second table and the charts (Figure 17-18), it shows the results of a k-nearest neighbours (KNN) model trained and evaluated with different values of k and different types of normalization applied to the input data. The evaluation metrics shown are accuracy, precision, recall, and F1-score.

The first column shows the type of normalization applied to the input data, which can be no normalization, Min-Max normalization, or Z-score normalization. The second column shows the value of k used in the KNN model. The remaining columns show the evaluation metrics calculated on the test set.

It can be seen that the model’s performance varies depending on the k value and the normalization applied. Without normalization, the model performed poorly with accuracy values between 0.7208 and 0.7390 and precision, recall, and F1-score values between 0.5069 and 0.4389. Finally, it was also noticed that when the number of neighbours increased the accuracy, precision, recall, and f1-score of the model increased as well.

When Min-Max normalization is applied, the performance of the model improves significantly. The accuracy, precision, recall, and F1-score values are all equal to 1.0, indicating a perfect fit to the data. Finally, it was also noticed that when the number of neighbours was changed it did not affect the values of the evaluation measures.

When Z-Score normalization is applied, the performance of the model improves significantly when comparing it to the one without normalization. The accuracy values are between 0.9935 and 0.9965, indicating that the model performs extremely well but not perfect. Finally, it was also noticed that when the number of neighbours increased the accuracy, precision, recall, and f1-score of the model decreased.

In conclusion, normalization is an important step in pre-processing the data before training a machine learning model. Without normalization, the model may be sensitive to certain features with large values and have poor performance which was verified by the charts along with the tables. By normalizing the data, the input features can be transformed to have similar scales, which can help to improve the model’s performance.

**The Effectiveness of Different Models:**

The effectiveness of a model is determined by its ability to accurately predict unseen data. The K-Nearest Neighbors (KNN) model and the Linear Regression model with Gradient Descent are two different models that have been trained and evaluated on a given dataset.

The KNN model is a classification model that uses the concept of distance to find the k nearest neighbors to a test data point. The majority class among these k nearest neighbors is then taken as the prediction for that test data point. The effectiveness of the KNN model can be determined by looking at the accuracy, precision, recall, and F1-score of the model on the test set. In the case of the provided implementation, the KNN model performed extremely well when using the Min-Max technique with an accuracy and F1-score of 1.0, and when using the Z-Score normalization technique with an accuracy of 0.9965 and F1-score of 0.9954. However, when no normalization was applied, the model performed poorly with an accuracy of 0.7390 and F1- score of 0.4389. This is likely due to the fact that without normalization, the data may be skewed and result in poor distance calculations, leading to poor predictions.

The Linear Regression model with Gradient Descent is a regression model that finds the best- fitting line through a set of data points. The effectiveness of this model can be determined by looking at the R-squared score, mean squared error, root mean squared error, and mean absolute error of the model on the test set. In the case of the provided implementation, the Linear Regression model with Gradient Descent performed well when using Min-Max and Z-Score normalization techniques, with R-squared values of 0.2698 and 0.3759 respectively. However, when no normalization was applied, the model performed poorly with R-squared values between -4.0956\*10 and -6.4021\*10¹¹, and large MSE, RMSE and MAE values. This is likely due to the fact that without normalization, the data may be skewed and result in poor predictions.

Overall, the KNN model and Linear Regression model with Gradient Descent are two different models that have different strengths and weaknesses. The KNN model performed well when using Min-Max and Z-Score normalization techniques, but performed poorly when no normalization was applied. The Linear Regression model with Gradient Descent performed well when using Min-Max and Z-Score normalization techniques, but performed poorly when no normalization was applied.

It’s important to note that the provided implementation is a simplified version of the models, and the results may vary when applied to other datasets or with different parameter settings. To choose between these models, you have to consider the characteristics of the problem and the data you have. If your problem is a classification problem, you might consider using the KNN model with appropriate normalization. If your problem is a regression problem, you might consider using the Linear Regression model with Gradient Descent with appropriate normalization.

**Conclusion:**

In conclusion, this project aimed to analyze data related to a food trading company and develop two prediction models using Linear Regression and KNN to predict total product sales and store location, respectively. Through the implementation of various data preprocessing techniques, the performance of the prediction models was optimized and evaluated using appropriate measures. The project also highlighted the importance of understanding the data science life cycle, the differences between supervised and unsupervised learning, and the role of computing systems in data science projects. Overall, the project reinforced the importance of thorough data analysis and preprocessing in the development of effective prediction models. Furthermore, it also highlighted the significance of appropriate performance evaluation and communication of results to non-technical stakeholders.

# ***References***

ACG Technical Editors Team (2022) *Compute compared: AWS vs Azure vs GCP | A Cloud Guru, A Cloud Guru.* Available at: <https://acloudguru.com/blog/engineering/compute-compared-aws-vs-azure-vs-gcp>

Alteryx (2022) *Supervised vs. Unsupervised Learning; Which Is Best?, Alteryx.* Available at: [https://www.alteryx.com/glossary/supervised-vs-unsupervised-learning](https://www.alteryx.com/glossary/supervised-vs-unsupervised-learning%20)

CLOUD ACADEMY TEAM (2018) *Analyze CPU vs. GPU Performance for AWS Machine Learning, Cloud Academy.* Available at: <https://cloudacademy.com/blog/analyze-cpu-vs-gpu-performance-for-aws-machine-learning/>

Deepali, V. (2022) *What Is Data Science Life Cycle? Steps Explained, Knowledge Hut.* Available at: [https://www.knowledgehut.com/blog/data-science/what-is-data-science-life-cycle](https://www.knowledgehut.com/blog/data-science/what-is-data-science-life-cycle%20)

Deula, J. (2015) *Supervised vs. Unsupervised Learning: What’s the Difference?, IBM.* Available at: <https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning>

Emeritus (2021) *What is Classification in Machine Learning and Why is it Important?, Emeritus.* Available at: <https://emeritus.org/blog/artificial-intelligence-and-machine-learning-classification-in-machine-learning/>

Gil, L. (2022) *Cloud Pricing Comparison for 2022: AWS vs. Azure vs. Google Cloud Platform, Cast AI.* Available at: [https://cast.ai/blog/cloud-pricing-comparison-aws-vs-azure-vs-google-cloud-platform/](https://cast.ai/blog/cloud-pricing-comparison-aws-vs-azure-vs-google-cloud-platform/%20)

HAYES, A. (2022) *Multiple Linear Regression (MLR) Definition, Formula, and Example, Investopedia.* Available at: <https://www.investopedia.com/terms/m/mlr.asp>

Hotz, N. (2022) *What is a Data Science Life Cycle?, Data Science Process Alliance.* Available at: <https://www.datascience-pm.com/data-science-life-cycle/>

IBM (2022) *About Linear Regression, IBM.* Available at: <https://www.ibm.com/uk-en/topics/linear-regression>

Javatpoint (2019) *Classification Algorithm in Machine Learning - Javatpoint, Javatpoint.* Available at: [https://www.javatpoint.com/classification-algorithm-in-machine-learning](https://www.javatpoint.com/classification-algorithm-in-machine-learning%20)

Javatpoint (2021) *K-Nearest Neighbor(KNN) Algorithm for Machine Learning - Javatpoint, Javatpoint.* Available at: <https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning>

Joby, A. (2021) *What Is K-Nearest Neighbor? An ML Algorithm to Classify Data, Learn Hub.* Available at: <https://learn.g2.com/k-nearest-neighbor>

Jones, E. (2022) *AWS vs Azure: Comparing the Cloud Computing Giants*, *Kinsta*. Available at: [https://kinsta.com/blog/aws-vs-azure/](https://kinsta.com/blog/aws-vs-azure/%20)

Microsoft (2022*) Comparing AWS and Azure storage services - Azure Architecture Center | Microsoft Docs, Microsoft.* Available at: [https://docs.microsoft.com/en-us/azure/architecture/aws-professional/storage](https://docs.microsoft.com/en-us/azure/architecture/aws-professional/storage%20)

NetApp (2020) *Azure vs AWS Pricing: Comparing Apples to Apples.* Available at: [https://cloud.netapp.com/blog/azure-vs-aws-pricing-comparing-apples-to-apples-azure-aws-cvo-blg](https://cloud.netapp.com/blog/azure-vs-aws-pricing-comparing-apples-to-apples-azure-aws-cvo-blg%20)

Online Manipal Editorial Team (2022) *What is the data science lifecycle?, Online Manipal.* Available at: <https://www.onlinemanipal.com/blogs/data-science-lifecycle-explained>

Sakshi, G. (2021) *Regression vs. Classification in Machine Learning: What’s the Difference?, Springboard.* Available at: <https://www.springboard.com/blog/data-science/regression-vs-classification/?msclkid=a86e7be3a93f11ec979c3c0d709a2098>

Seldon (2021) *Supervised vs unsupervised learning explained , Seldon Technologies.* Available at: <https://www.seldon.io/supervised-vs-unsupervised-learning-explained>

Statistics Solutions (2013) *What is Linear Regression? - Statistics Solutions, Statistics Solutions.* Available at: <https://www.statisticssolutions.com/what-is-linear-regression/>

Terra, J. (2022a) *Regression vs. Classification in Machine Learning for Beginners, Simplilearn.* Available at: <https://www.simplilearn.com/regression-vs-classification-in-machine-learning-article>

Terra, J. (2022b) *Regression vs. Classification in Machine Learning for beginners, Javatpoint.* Available at: <https://www.simplilearn.com/regression-vs-classification-in-machine-learning-article>

Yale (2011) *Linear regression, Munro’s Statistical Methods for Health Care Research: Sixth Edition.* Available at: <https://doi.org/10.4324/9780203499894-15.>