**CCT College Dublin**

**Assessment Cover Page**

|  |  |
| --- | --- |
| **Module Title:** | Data Preparation |
| **Assessment Title:** | CA1\_DataPrep\_HDip |
| **Lecturer Name:** | David McQuaid |
| **Student Full Name:** | Federico Ariton |
| **Student Number:** | sba22090 |
| **Assessment Due Date:** | 05/10/2023 |
| **Date of Submission:** | 03/11/2023 |

**Declaration**

|  |
| --- |
| By submitting this assessment, I confirm that I have read the CCT policy on Academic Misconduct and understand the implications of submitting work that is not my own or does not appropriately reference material taken from a third party or other source. I declare it to be my own work and that all material from third parties has been appropriately referenced. I further confirm that this work has not previously been submitted for assessment by myself or someone else in CCT College Dublin or any other higher education institution. |

Index

[Introduction 2](#_Toc149940044)

[Characterisation 2](#_Toc149940045)

[EDA and DATA CLEANING 3](#_Toc149940046)

[PCA 4](#_Toc149940047)

[The curse of dimensionality 4](#_Toc149940048)

[References: 5](#_Toc149940049)

Analysis of Scania Truck APS System Failures

# Introduction

The primary goal of this analysis is to understand the APS (Air Pressure System) failures in heavy Scania trucks, which can help the company in deciding their investment strategy for the upcoming year. We will be using Python for all data wrangling, analysis, and visualizations. The steps are designed to meet the minimum requirements set by the company's CTO, including data characterization, data preparation and evaluation, and Principal Component Analysis (PCA) for dimensionality reduction.  
Characterisation of the data

# Characterisation

First, we import the Pandas framework and assign it the alias 'pd' for ease of code management, allowing us to avoid lengthy references. We proceed to read the file by creating a variable with the same name as the file and utilize the 'pd.read\_csv("filename.csv")' method. The 'read\_csv' function is a component of the Pandas library designed for file reading, and we specify the file's name within the brackets.

In the following step, I ensure that the correct file has been successfully loaded. To achieve this, I use the code .head(). , there are various methods to visualize the table, such as displaying the column names.

To gain insight into the dimensions of our dataset, I implement the code .shape, which allows us to determine that our dataset is of size 60,000 x 171. This implies that we have 60,000 observations and 171 attributes in our dataset.

By utilizing the .describe() function to analyze the data, we observe that it displays information for only one column. This could indicate that this particular column lacks numeric values necessary for calculating statistical measures such as the mean and median. Consequently, the output provides information for just a single column.

## EDA and DATA CLEANING

Let's investigate the presence of missing values in our dataset. Currently, it appears that there are no missing values, which is somewhat unusual.

To further explore this, we can use the .dtypes method to analyze the data types of the columns. Indeed, it seems that the issue lies in the fact that some columns are of type 'object' rather than 'integer' or 'float'. This suggests that there might be "na" values present in these columns, causing them to be classified as 'object' data types.

Let's perform a more comprehensive description of the table, including the columns with 'object' values. What becomes apparent is that the value 'na' is repeatedly present in columns like 'ab\_000' and 'ad\_000,' with 'na' occurring frequently. This situation adds complexity to our analysis, and we need to address it.

Upon changing the values in the dataset and reevaluating for missing values, we discover that there are indeed many missing values present. When we describe the data again, all columns are now included,e notice that there are numerous '0' values, which may not be ideal for our calculations

Following the conversion of 'na' values to 'NaN,' the system can now properly recognize empty values as missing data, as opposed to interpreting them as strings.

Subsequently, I calculated the percentage of missing data in each column, revealing that certain columns have missing data exceeding 70% or 80%. To streamline our dataset and facilitate better data management, I've decided to remove columns with more than 60% missing data. This helps in maintaining a dataset with fewer missing values, enhancing our data handling and analysis.

After converting the 'NaN' values, the next step involves deciding how to handle these missing values. One common approach is to replace them with the mean (average) value of the respective column. This is done to ensure that missing data does not significantly skew our analysis, and it provides a reasonable approximation of the missing information. This method is particularly useful when the missing data is assumed to be missing at random and doesn't introduce bias into our analysis.

the next step is to consider removing columns where more than 60% of the values are equal to 0.0. This decision is made because columns with a high percentage of zero values may not contribute much meaningful information to our analysis, and they could potentially introduce noise or bias into our models.

Before conducting exploratory data analysis (EDA), it's essential to address the data cleaning process. Initially, during EDA, I encountered limitations since I could only visualize the first column due to the others being of 'object' data type. These columns should ideally be converted to numerical data types, such as float or integer.

After completing the data cleaning, we are now ready to perform EDA effectively. One of the insightful visualizations we can create is a box plot representing each class and a histogram displaying the distribution of positive and negative values within the dataset. These visualizations provide valuable insights into the data and help us better understand the characteristics of our variables.

after completing the data cleaning and preprocessing steps, the next stage is to perform Principal Component Analysis (PCA) to reduce the dimensionality of our data.

# PCA

Before applying PCA, it's crucial to standardize our data. Standardization is necessary because it scales the features, ensuring that no single feature has a disproportionately high influence on the analysis. As Jaadi (2019) pointed out, it equalizes the distances between data points, making the analysis more meaningful.

To determine the number of principal components to retain, we initially initialize a PCA object without specifying the number of components explicitly. This default setting will consider as many components as there are features. We then calculate the cumulative sum of variance explained by each component and find the number of principal components required to retain 99.5% of the variance in the data.

After identifying the optimal number of components, we initialize a new PCA object to reduce the data to this dimensionality. Finally, we print the shape of the reduced data to confirm the successful dimensionality reduction process.

On the other hand, there is another aspect of PCA, which aims to identify the number of components that explain a substantial portion of the variance in the dataset. This is typically done by creating a plot that shows the cumulative explained variance against the number of components. This visual representation helps us determine how many components are needed to capture the majority of the data's variance effectively.

Once the optimal number of components is established, we proceed to transform the data into a lower-dimensional space, typically a 4-dimensional space in this case, using PCA. This reduction in complexity retains the essential information from the original dataset while simplifying its representation.

After applying PCA transformation and labeling the data, the next step involves splitting it into training and validation sets. Subsequently, we train a model on the training data, and when evaluating its performance on the validation set, we achieve an accuracy score of approximately 79.88%. This score demonstrates the model's ability to make accurate predictions based on the reduced-dimensional representation of the data.

# The curse of dimensionality

Understandably, navigating the complexities of high-dimensional data spaces is much like the challenge of finding a specific landmark in a vast and varied landscape. Each characteristic or feature of the data introduces a new dimension, just as additional geographic features such as elevation, temperature, or rainfall would add layers of complexity to a map.

As the number of features (dimensions) in a dataset increases, the difficulty in finding patterns, or in the case of our analogy, landmarks (data points), escalates. This phenomenon is commonly referred to as the "Curse of Dimensionality" (Vishwesh, 2020). Essentially, in a high-dimensional space, the data becomes so spread out that any new data point and can appear to be far from any existing points. This vastness makes it incredibly challenging to perform analyses or predictions because traditional statistical techniques lose their effectiveness as the space expands (Jordan, Kleinberg and Schölkopf, n.d.).

It's crucial to strike a balance. While dimensionality reduction techniques such as Principal Component Analysis (PCA) can help by removing redundant or irrelevant features, thereby simplifying the landscape and making it easier to find our landmarks, there is a caveat. If we simplify too much, akin to removing too many details from our map, we risk "underfitting" our model. Underfitting occurs when our model is overly simplified and lacks the complexity needed to capture the underlying patterns and structures in the data, much like a map that lacks the necessary details to navigate the terrain effectively (James et al., 2013).

In my analysis can affect the curse of the dimensionality the presence of numerous variables or features, leads to increased computational complexity, data sparsity, and overfitting risk making it challenging to derive meaningful insights from the data (Built In, n.d.).

# References:

Jaadi, Z. (2019). A Step by Step Explanation of Principal Component Analysis. [online] Built In. Available at: https://builtin.com/data-science/step-step-explanation-principal-component-analysis.

Vishwesh, K. (2020). Curse of Dimensionality: An intuitive and practical explanation with examples. [online] Flutter Community. Available at: https://medium.com/flutter-community/curse-of-dimensionality-an-intuitive-and-practical-explanation-with-examples-399af3e38e70.

Jordan, M., Kleinberg, J. and Schölkopf, B. (n.d.). Information Science and Statistics. [online] Available at: http://users.isr.ist.utl.pt/~wurmd/Livros/school/Bishop%20-%20Pattern%20Recognition%20And%20Machine%20Learning%20-%20Springer%20%202006.pdf.James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning. Springer.

Built In. (n.d.). Curse of Dimensionality. [online] Available at: https://builtin.com/data-science/curse-dimensionality.