# Manufacture Defect Detection Report

Applied Data Intelligence

Project Deliverable – 1

By

Ajala Reddy Akiti

Dhatri Mulpuri

**Table of Contents**

* Introduction
* Project Description
* Data Visualization
* Data Pre-processing and Data Splitting
* Model Description
  + Logistic Regression
  + Decision Tree Classifier
* Model Selection
  + Resampling Method
  + Performance Metric
* Results
  + Confusion Matrix
  + ROC Curve and AUC
  + Accuracy
* Conclusion

# Introduction

In this project, we apply machine learning strategies to identify defects in a dataset tailored for Manufacturing Defect Detection. Deliverable 1 starts our attempt, focusing on logistic regression and decision tree algorithms. These initial techniques are selected due to their proven track record in efficiently solving classification problems. Through applying these algorithms, our objective is to lay the foundational knowledge on how machine learning methodologies can be utilized to spot flaws and irregularities in manufacturing outputs.

# Project Description

This project focuses on leveraging advanced data science techniques to explore the state of quality control in manufacturing by analyzing detailed manufacturing data. The project aims to pinpoint and predict defects that may arise during production by utilizing a comprehensive dataset containing manufacturing parameters and outcomes. The project strives to predict defect occurrences accurately through meticulous data cleansing and applying advanced machine learning algorithms like Logistic Regression and Decision Trees. Its primary objectives include identifying underlying defect patterns and anomalies, refining defect detection mechanisms, and enhancing product quality standards. Through this data-driven approach, the project seeks to showcase the transformative potential of data science in revolutionizing quality control processes within the manufacturing industry.

# Data Visualization

1. **Boxplots of Features by Response**: The first image presents a series of boxplots for different features (X1 to X30) categorized by a binary 'Response' variable. Boxplots are excellent for visualizing data distribution and highlighting the median, quartiles, and potential outliers. If the medians of the boxplots for the two responses are significantly different, this might indicate a feature's potential for distinguishing between the two response classes.

A chart of different colored bars

Description automatically generated with medium confidence

1. **Skewness of Each Feature**: The second image shows a bar plot of the skewness of each feature. Skewness measures the asymmetry of the probability distribution of a real-valued random variable. A zero value indicates a symmetric distribution, while negative or positive values indicate skew. Features with high skewness might require transformation to make the data more normally distributed, which can improve the performance of many machine learning models.

A graph with different colored squares

Description automatically generated **A graph of a number of bars

Description automatically generated**

1. **Feature Distribution Histograms**: The final image is a histogram plot showing the distribution of values for a subset of features (X1 to X5). Histograms help understand the data's distribution's central tendency, dispersion, and shape.
2. **Correlation Heatmap**: The third image is a correlation heatmap that provides insight into the linear relationships between the features. Values close to 1 or -1 indicate a strong positive or negative correlation. Features with high correlation may contain redundant information, and consideration might be given to dimensionality reduction techniques like Principal Component Analysis (PCA).

**A screen shot of a graph

Description automatically generated**

# Data Pre-processing and Data Splitting

# Data Pre-processing:

# Handling Missing Values: Missing values were addressed by filling them with either the mean or median of their respective columns.

# Feature Standardization: StandardScaler from sci-kit-learn was used to standardize the features, ensuring they have a mean of 0 and a standard deviation of 1.

# Data Visualization: Histograms, heatmaps, and boxplots were employed to visualize the data, providing insights into its distribution, correlations, and presence of outliers.

# Data Splitting:

# Train-Test Split: The dataset was split into a training set and a testing set using train\_test\_split from scikit-learn. This technique allows for the evaluation of model performance on unseen data, helping to assess the generalization ability of the models.

# Model Description

# Logistic Regression: We chose logistic regression because it's straightforward and good at predicting binary outcomes like defective or non-defective items. It gives us probabilities for each prediction, making it easier to understand how confident the model is. Logistic regression works well for our data because it can handle linear relationships between features, which is common in defect detection scenarios.

* **Decision Tree Classifier:** We also used a decision tree classifier because it can capture complex patterns in the data and create simple decision rules. Decision trees are helpful for numerical and categorical data, which fits our dataset well. They're easy to understand and visualize, making it easier to see how the model makes decisions.

**Model Selection**

* **Resampling Method:** We used SMOTE to deal with imbalanced data where one class is much more common. SMOTE creates new synthetic samples for the minority class to balance things out.
* **Performance Metrics:** To see how well our models perform, we used accuracy, precision, recall, and the AUC-ROC score. Accuracy tells us how often the model is correct overall, while precision shows the proportion of correctly predicted positive cases. Recall measures how many positive cases were rightly expected, and the AUC-ROC score tells us how well the model can distinguish between classes.

# Results

# Confusion Matrix

The confusion matrix reveals the model's performance.

* True Positives (TP): 45,910 non-defective units were correctly identified.
* False Positives (FP): Only three defective units were mistakenly classified as non-defective.
* False Negatives (FN): 14 non-defective units were erroneously labeled defective.
* True Negatives (TN): 73 defective units were correctly identified.

# ROC Curve and AUC

* The high accuracy of 99.96% implies that the ROC curve likely closely follows the ideal curve.

A graph of a function

Description automatically generated with medium confidence

* A remarkably low false positive rate of 0.0065% indicates minimal misclassification of defective units.

# Accuracy and Other Performance Metrics

* + Accuracy: 99.96% accuracy signifies precise classification of almost all units.
  + Precision: A perfect precision score of 1.00 ensures all predicted non-defective units are non-defective.
  + Recall: A recall of 0.84 suggests that 84% of defective units are accurately identified.
  + F1-score: With an F1-score of 0.90, the model achieves a harmonious balance between precision and recall.

# Conclusion

In conclusion, the model demonstrates remarkable performance, achieving an accuracy of 99.96%, a precision of 96%, and an F1 score of 90%. While these metrics signify the model's proficiency in accurately classifying manufacturing defects, the recall rate of 84% suggests that there is room for improvement, as it indicates that 16% of defective units may be misclassified.

Moving forward, we will explore additional algorithms that will be studied in subsequent classes to refine the model's performance further and address the existing limitations.