Predictive Analytics [PGR304]

2023 Autumn

Candidate: 2002

Abstract — The study is to explore the given dataset by using various statistical concepts and techniques for predictive analytics from the perspective of data scientist. Based on the comprehensive understanding of the data, the paper detects the data science problem by selecting appropriate predictive analytics methods and tools. In addition, the study evaluates the results and models in terms of the methods and techniques employed. Furthermore, the paper critically reflects on the tradeoffs in the design and implementation of predictive analytics solutions.

I INTRODUCTION

The given dataset is from an industrial or manufacturing environment, where a factory is conducting controlled experiments to understand how different conditions affect the outcome or quality of a product. It demonstrates 480 experiments. Each individual experiment includes 25 recorded features. Feature 1 to Feature 22 are statistical calculations derived from various sensors during the experiment, measuring aspects like pressure, density, and so on. Feature 23 and Feature 24 are input variables that could influence the experiment's results, similar to environment factors like temperature at the start and end of the process. The target variable is an integer between 1 and 5 that rates the product's quality.

II UNDERSTANDING THE DATA

The given dataset is a high-dimensional dataset. All the variables are numerical data. Below data visualization graphs illustrate various exploring approaches to understand the data.

A. *Data Distribution*

Figure 1 displays the general data distribution [1] overview in a series of histograms. Each histogram represents the distribution of values for different features in the dataset. A right-skewed distribution often has a 'hill' on the left and a long tail stretching to the right, like Feature1, 3, 4, 5, 6, 7, 12, 13, 15,16,17,18, 20, 21, 22, where most of the data is concentrated at the lower end of the value range with tails extending towards higher values. And a left-skewed distribution is on the contrast. Feature2 Has a bimodal distribution with two distinct peaks, which may indicate two different groups or behaviors within this feature. Feature8 has a very strong peak at a lower value and a tail stretching to the right, indicating a highly skewed distribution. Feature9 shows a distribution with a single, sharp peak and a decline towards the higher values, which is indicative of skewness. Feature10 appears to be an exception with a more normal, symmetric distribution. Feature11 to Feature24 display various degrees of right-skewness, with most of the data concentrated on the left and tails extending to the right. Some of these features exhibit potential outliers, as seen by the isolated bars at the higher and end of the scale.

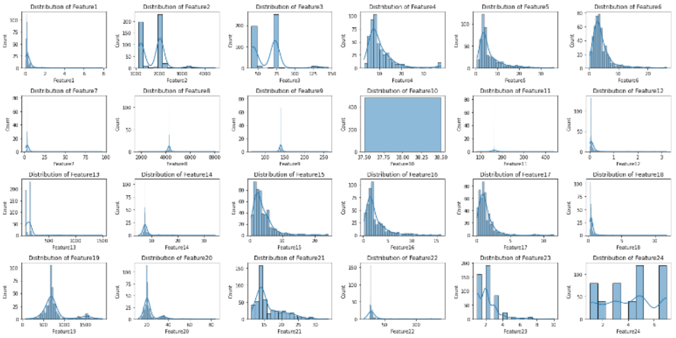
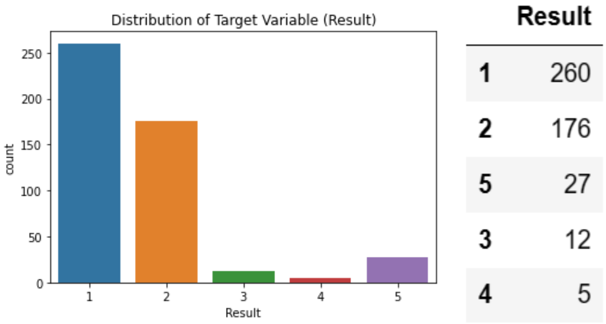


Figure 1

B. *Result Distribution*

Figure 2 demonstrates a bar chat representing the distribution of a categorical target variable for column 'Result' in the dataset. Category1 has the highest count, with 260 occurrences. Category2 is the second most common with 176 occurrences. Category5 has 27 occurrences, which suggests it's less common than Categories 1 and 2. Categories3 and 4 are at the least common results, with 12 and 5 occurrences respectively.



	Result
1	260
2	176
5	27
3	12
4	5

Figure 2

The distribution illustrates the dataset is with unbalanced classification, which could affect the performance of machine learning models trained on this data. Because they may become biased towards predicting the more common classes.

C. *Data Correlation*

The correlation heatmap [8] in Figure3 is a graphical representation of a Pearson correlation matrix that measures the linear relationships between variables. It provides a quick identifying how each feature is related to the others. The variables (features and result) are usually represented on both the x-axis and the y-axis. The correlation heatmap can be interpreted as below:

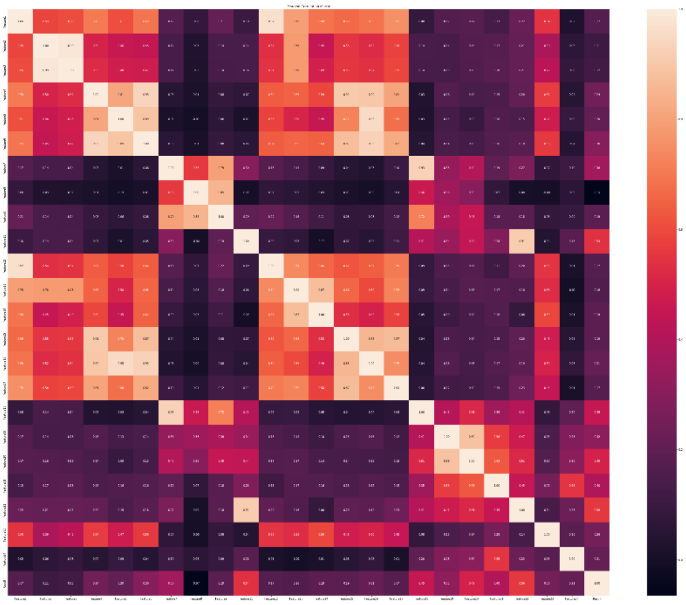


Figure 3

1. Color Scheme

Typically, heatmap use a color gradient to represent the strength and direction of the correlation. According to the color bar on the most right-hand side of the heatmap, the cooler color, the stronger positive correlation is indicated as the correlation coefficient is closer to ‘1’. In contrast, the warmer color displays the weaker correlation, which is at a correlation coefficient is closer to ‘o’ or even a negative index.

2. Correlation Coefficient

The cells in the heatmap contain correlation coefficients that range from -1 to 1. A value close to 1 implies a strong positive correlation: as one feature increase, the other feature also increases. A value close to -1 implies a strong negative correlation: as one feature increases, the other feature decreases. A value close to 0 means that there is no linear relationship between the features. A value at 1 means the feature itself.

D. *Outlier Data*

Understanding the outlier data is likewise important, outlier may need to be examined to determine if they should be kept, adjusted, or removed, depending on their nature and the goals of the analysis. A boxplot visualization is typically used to display the distribution of a dataset and to identify potential outliers.

The boxplot (Figure4) visualizes several features have a significant number of outliers, as evidenced by the points that fall outside the whiskers. For instance, Feature2, Feature8, and Feature22 have many data points that are distant from the bulk of the data, which are visualized as separate points above the top whisker.

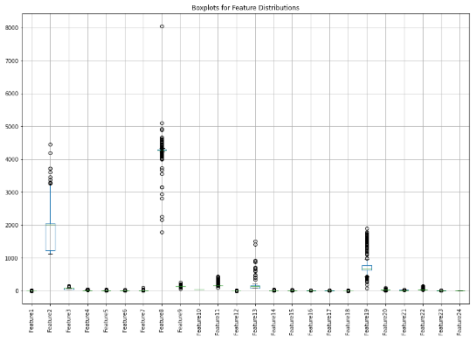


Figure 4

E. *Skewness*

Skewness [1, 13] measures the asymmetry of the probability distribution of a real-valued random variable about its mean.

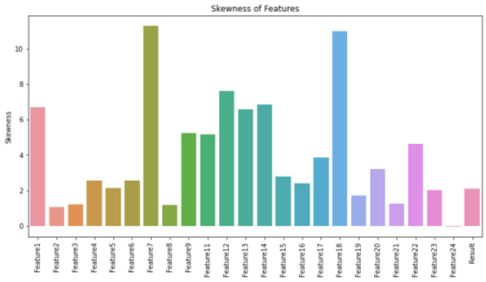


Figure 5

The skewness bar chart (Figure 5) displays the skewness of each feature. Bars above zero on the y-axis indicate positive skewness, where the tail of the distribution is longer on the right side. Bars below zero would indicate negative skewness where the tail is longer on the left side. Features with bars extending further from the zero line have more severe skewness. For instance, Feature7 and Feature18 with a skewness value of over 10, they have a highly skewed distribution.

F. *Kurtosis*

Kurtosis [13] describes the extreme values (outliers) in one versus the other tail. This includes both the frequency and the magnitude of these extreme deviations from the mean.

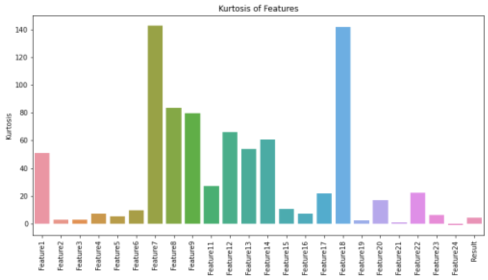


Figure 6

Kurtosis chart (Figure 6) displays Feature2, 3, 19 have kurtosis value close to 3, suggesting a distribution shape that is similar to a normal distribution (mesokurtic). Feature1, 7, 8, 9, 11, 12, 13, 14, 17, 18, 20, 22 have very high kurtosis values, indicating highly leptokurtic and a significant presence of outliers. Feature 21 has a kurtosis value less than 3, which is platykurtic and indicates fewer outliers. Feature24 has a slightly negative kurtosis value indicates a distribution with very light tails (platykurtic), which could mean that data is clustered around the mean with fewer and less significant outliers. Other features have kurtosis values that are not noticeably higher than 3 but not as extreme as some others. They indicate moderately heavy tails and potential outliers.

III UTILITY VALUE

Based on the understanding of the data, the dataset can be used for several applications, particularly because it contains experimental results from a production process with various measurements:

A. *Quality Control and Improvement*

By predicting the quality of the product, the factory can proactively identify and correct issues during the production process, leading to a more consistent product quality. In the given dataset, the target variable in the ‘Result’ column can classify the product quality from ‘class 1’ to ‘class 5’.

B. *Process Optimization*

Analysis of the dataset could reveal which features (process parameters) are most influential on product quality, enabling process engineers to fine-tune the manufacture line for optimal performance.

C. *Predictive Maintenance*

If the features include data from machine sensors, predictive models could help anticipate equipment failures before they occur by correlating sensor readings with experiment outcomes that had poor quality results.

D. *Cost Reduction*

By understanding which features and variables impact the quality, the factory can focus resources on monitoring and controlling critical factors, potentially reducing the cost associated with over-processing or excessive quality assurance testing.

However, the given dataset might not be ready for analysis. Before proceeding with further analysis and modeling, the following data pre-processing steps are considered:

1. Missing values need to be detected and replaced by median values.

2. Separate features and target.

The dataset is separated in features part represented by ‘X’ and the target variable part represented by ‘y’.

3. Split the dataset.

To put the data in machine learning model for training and validate the model learning performance with the data that the model has never seen, the dataset is split to train dataset and validate dataset. Both on the features part and the target variable part.

4. Feature Scaling and standardization

Machine learning algorithms like KNN and SVM are sensitive to the scale of the data. Standardization (z-score normalization) or Min-Max scaling might be required.

5. Dimensionality Reduction

There are 24 features in the given dataset, especially correlated ones. Techniques like PCA could be used to reduce the number of features while retaining most of the information.

6. Handling Class Imbalance

From the distribution of target variable ‘Result’, the variable is imbalanced. Hence, techniques like SMOTE for oversampling the minority class or stratified sampling during train-test split can be considered.

7. Feature Selection

Although the correlations are detected, more sophisticated feature selection methods can be applied to retain only the most relevant features, or the most important features with the target variable to implement the modeling task.

IV ANALYSIS, MODELING AND PREDICTION

A. *Data Analysis Methods*

To analyze the data for the prediction that the owner is interested in Prediction of the results of each experiment, we can employ a combination of machine learning, statistical methods, and visualizations. Here is a breakdown of the methods and their justifications:

1. Machine Learning Methods

This kind of dataset is very common in the field of process optimization and manufacturing. The goal is often to create a predictive model that can guide the settings of the process variables (like temperature, pressure, etc.) to ensure the best quality output. Machine learning models, such as regression or classification algorithms, are typically applied to such data to predict the 'Result' based on the various features.

Logistics Regression (Logit) [6]: Logistics regression is a statistical method used for classification problems, where the output variable is categorical. In this case, the output variable is the quality of the product, which can be categorized as ‘good’ or ‘bad’. Logistic regression is a suitable choice for this case due to its simplicity, effectiveness, and interpretability.

2. Statistical Methods

2.1 Correlation analysis: Correlation analysis [9] can be used to assess the relationships between the input variable (Feature1 to Feature 24) and the output variable ‘Result’. This can help identify which input variables are most relevant to predicting the quality of the product.

Based on the correlation heatmap Figure3, it can be observed that Feature2 and Feature3 have the most highly correlation with a coefficient at 0.99. The pair followed are Feature1 and Feature12 have a correlation coefficient of 0.97. Next pair is Feature5 and Feature16 at a correlation coefficient 0.95. Feature4 and Feature6 have a correlation coefficient of 0.93. Figure 4 displays four pairs of features that have strong positive correlation, the scatter plot demonstrates that scatter points are in a linear relationship. These four pairs of features have linear correlation.

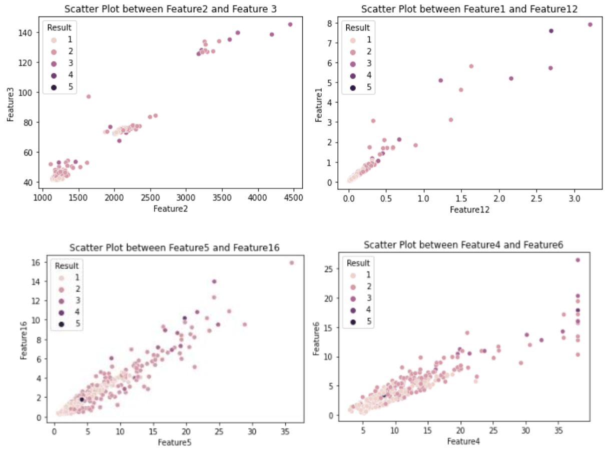


Figure 7

Feature11 and Feature22 do not have strong positive correlation with each other in Figure5, their scatters are discrete compared with above four pairs of features. However, they have individually positive correlation with the target variable (‘Result’). Feature22 has a correlation coefficient 0.55 with the target variable, which is the most highly correlated to influence the result. And Feature 11 follows with a correlation coefficient at 0.54, also has positive correlation with the result.

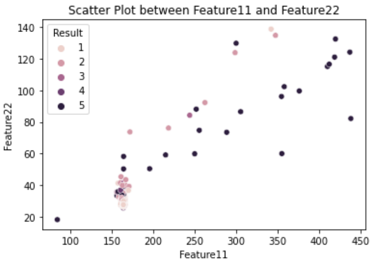


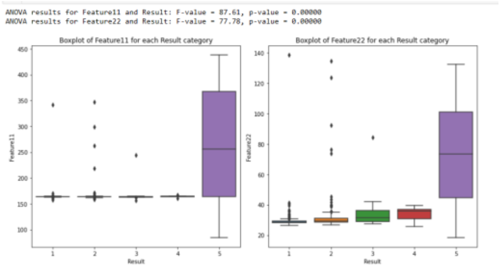
Figure 8

2.2 Analysis of Variance

ANOVA [3] is a statistical method used to compare the means of three or more groups to see if at least one group mean is statistically different from the others. When the target variable has more than two categories (as is the case with ‘Result’),

A low p-value (typically < 0.05) would indicate that there is a statistically significant difference in the mean values of 'Feature11' or 'Feature22' across the different categories of 'Result'. A high p-value would suggest that any mean differences are likely due to random chance, and the feature may not be as useful for discriminating between 'Result' categories.

The selection of Feature11 and 12 to implement ANOVA is because of their individual correlation with the target variable are higher than other features. In Figure 9, both Feature11 and Feature22, the F-value is quite high (87.61 for Feature11 and 77.78 for Feature22), and the p-value zero. This suggests that there are significant differences in the means of these features across the different result categories.



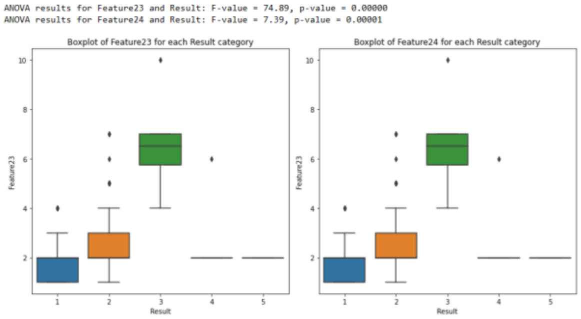
```
ANOVA results for Feature11 and Result: F-value = 87.61, p-value = 0.00000
ANOVA results for Feature22 and Result: F-value = 77.78, p-value = 0.00000
```

Boxplot of Feature11 for each Result category
Feature11
450
400
350
300
250
200
150
100
1
2
3
4
5
Result
Boxplot of Feature22 for each Result category
Feature22
140
120
100
80
60
40
20
1
2
3
4
5
Result

Figure 9

Figure 9 displays the spread and central tendency of Feature11 and 22 with individual Result category. For Feature11, the boxplot indicates a higher median and greater variability for the highest category, which could be class ‘5’ on the boxplot, suggesting that Feature11 could be a significant predictor for the particular outcome class ‘5’. For Feature22, the median values seem to be more consistent across the results, but the highest category shows a higher spread, which is also reflected in the significant F-value from the ANOVA.

Another selection of Feature23 and 24 is due to their input variable property. In fact, ANOVA can be particularly useful for assessing whether different groups (in this case, defined by the target variable 'Result') have different mean levels of these input variables. This can help determine if there's a statistically significant effect of these input variables on the outcome. Figure10 shows that Feature23 has a F-value 74.89 that is much higher than Feature24 (7.30), and their P-value is very close to zero. This also suggests that there are significant differences in the means of these features across the different result categories. The boxplot displays both input variables are more trends to predict outcome class 1, 2, 3.



```
ANOVA results for Feature23 and Result: F-value = 74.89, p-value = 0.00000
ANOVA results for Feature24 and Result: F-value = 7.39, p-value = 0.00001
```

Boxplot of Feature23 for each Result category
Feature23
10
8
6
4
2
1
2
3
4
5
Result
Boxplot of Feature24 for each Result category
Feature23
10
8
6
4
2
1
2
3
4
5
Result

Figure 10

3. Visualizations

Scatter plots: Scatter plots can be used to visualize the relationship between pairs of variables. This can help identify potential correlations between the input variables and the output variable.

Histograms: Histograms can be used to visualize the distribution of a single variable. This can help identify any non-normal distributions or outliers.

B. Data Partitioning

To evaluate the performance of our models, we will partition the dataset into two sets:

1. Training set: This set will be used to train the machine learning models. It should be large enough to capture the underlying patterns in the data, but not so large that it becomes computationally intractable.
2. Validation set: This set will be used to evaluate the performance of the trained models. It should be small enough to be manageable, but large enough to provide a reliable estimate of the model’s performance on unseen data.

In this case, 15 rows experiments (about 3% of the data) will be assigned to validation set and the rest 97% data will be assigned to the training set.

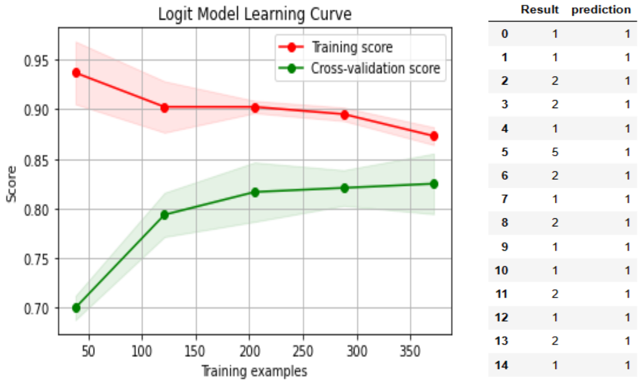
C. Application of Logit Model for Classification

Applying logit model is a suitable choice for multi-classification problems such as the given dataset. The logistic regression model is designed to predict categorical outcomes. For instance, the multinomial logistic regression is an extension of logistic regression for multi-class classification.

- Define a logit model with parameter: ‘multinomial’ and choose a solver ‘ibfgs’ to support the multi-classification classes.
- The logit model is fitted in the training set with feature scaled for training, and the prediction is implemented in the validation set.
- Reset the index in validation set and remove original index.
- Add the prediction to the test data frame.

- Gradually increase the training sample from 10%, 32%, 55%, 77% till 100% in the training set to see the model learning performance.
- Calculate mean and standard deviation for train set scores.
- Calculate mean and standard deviation for validation set scores.
- Plot the model learning curve, print out the model prediction and the actual result for comparison.

The left-hand side on Figure11 displays a learning curve for the model, which indicate the relationship between the number of training examples and the model's performance, measured by training score and cross-validation score. The red line represents the training score, which is the model's performance on the training set. It starts high and stabilizes as the number of training examples increases, suggesting that the model fits the training data well. The green line represents the cross-validation score, which is the model's performance on a separate validation set that it has not seen during training. The score increases with more data and seems to be converging towards the training score, which is a good sign. The shaded area around each line represents the variability (standard deviation) of the scores across different cross-validation splits. A narrow-shaded area suggests consistent model performance across different splits. The convergence of the two lines suggests that adding more training data may not significantly improve the model's performance, and the model is generalizing well to unseen data.



	Result	prediction
0	1	1
1	1	1
2	2	1
3	2	1
4	1	1
5	5	1
6	2	1
7	1	1
8	2	1
9	1	1
10	1	1
11	2	1
12	1	1
13	2	1
14	1	1

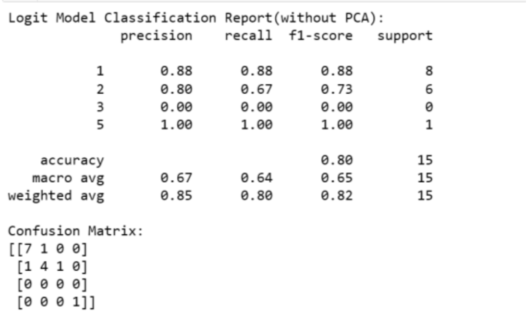
Figure 11

The table on the right-hand side on Figure11 displays a comparison of the actual results (target variable 'Result') and the model's predictions. It is observed the model appears to have high accuracy for class '1', but poor recall for other classes. This could be a result of class imbalance, where the model is biased towards the most frequent class in the training data, which seems to be class '1' based on previous context. Given the misclassification of other classes, it would be beneficial to look at more comprehensive performance metrics such as the confusion matrix, precision, recall, F1 score for each class, and overall accuracy to better understand the model's performance. This will be further evaluated in the next chapter.

Next is to implement the hyperparameter tuning the logit model by importing GridSearchCV from the scikit-learn library:

- Define the parameter grid.
- Initialize the logistic regression classifier.
- Initialize GridSearchCV
- Fit the model in the training set.
- Evaluate the model in a validation set with feature scaled where the data have never been seen.

Now the model has been fine-tuned for the multi-classification problem, it exhibits the classification metrics obtained from the validation set in the following report.



```
Logit Model Classification Report(without PCA):
              precision    recall  f1-score   support

           1       0.88      0.88      0.88         8
           2       0.80      0.67      0.73         6
           3       0.00      0.00      0.00         0
           5       1.00      1.00      1.00         1

    accuracy                           0.80        15
   macro avg       0.67      0.64      0.65        15
weighted avg       0.85      0.80      0.82        15

Confusion Matrix:
[[7 1 0 0]
 [1 4 1 0]
 [0 0 0 0]
 [0 0 0 1]]
```

Above classification report provides precision, recall, and F1-score for each class, along with the support which indicates the number of true instances for each label. Notably, after tuning the logit model, and apply the model in the small validation set that has not been used in the training/tuning, the model achieves high precision and recall for class '1' and class '5'. This may be indicative of a good fit for these classes. And the model also can predict Class '2' even if it has a moderate recall. However, the absence of class '3' predictions suggests that the model fails to identify this class altogether. Anyway, the model displays obvious ability to predict more classes in an unseen validation set after being training and tunning.

The 'Confusion Matrix' further elucidates the model's performance, revealing a tendency to classify instances into class '1', which could be due to a class imbalance. The perfect scores for class '5' are noteworthy, but given that it only represents a single instance, this result should be treated with caution.

Overall, the logit model demonstrates an 80% accuracy rate, which seemingly high, but does not tell the full story. The weighted average F1-score of 0.82 is more reflective of the model's true performance, accounting for class imbalance by weighting the F1-score of each class by its support. The model's tendency to favor the majority class (class '1') suggests the need for strategies to address class imbalance, such as resampling methods or the use of class weights in model training.

D. *Dimensionality Reduction with PCA*

Principal Component Analysis (PCA) [2] is a dimensionality reduction technique that can be used to reduce the number of features in the dataset. This can be beneficial if the dataset is high-dimensional, as it can simplify the analysis and improve the performance of machine learning models.

PCA is performed to reduce the 24 features space to two principal components (PCs), which were then used for visualization and as inputs to the logit model. This transformation aims to capture the most significant variance in the data with fewer features, reducing computational complexity and potentially improving model performance. The dimension reduction by using PCA is addressed as below steps:

- Initialize PCA and fit the model to the scaled training set.
- Initialize the logit model.
- Apply PCA transformation to the validation dataset.
- Train the model using the PCA-transformed training dataset.
- Predict the results on the PCA-transformed validation dataset.

- Print out the evaluation report.

The scatter plot visualizes how the observations are distributed in the space defined by the first two principal components. The 'PC1' values on x-axis and 'PC2' values on y-axis are the scores of the first and second principal components, which are linear combinations of the original feature. Observations that are close together are similar in terms of the underlying features, which were transformed into principal components. Different colors or symbols are used to show the true class ('Result') of each observation, providing insight into how well-separated the classes are in the reduced feature space. The visualization displays the scatters represent class '1' and class '2' are more concentrated at the low score area, while the other scatters are distributed discretely score area.

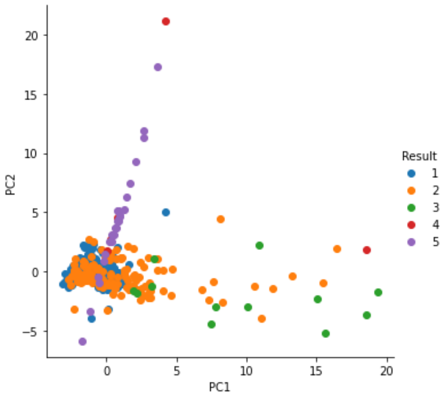
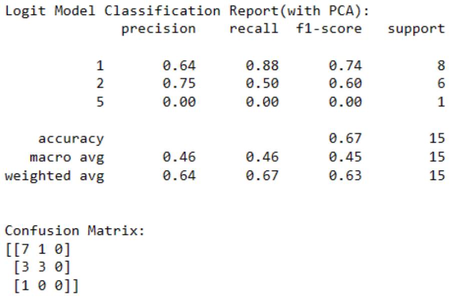


Figure 12

The evaluation report includes key metrics such as precision, recall, and F1-score for each class, as well as overall accuracy. The report shows the model with PCA-transformed achieves moderate precision for but low recall for class '2', while low precision but high recall for class '1'.



```
Logit Model Classification Report(with PCA):
              precision    recall  f1-score   support

           1       0.64      0.88      0.74         8
           2       0.75      0.50      0.60         6
           5       0.00      0.00      0.00         1

    accuracy                           0.67        15
   macro avg       0.46      0.46      0.45        15
weighted avg       0.64      0.67      0.63        15


Confusion Matrix:
[[7 1 0]
 [3 3 0]
 [1 0 0]]
```

Figure 13

Overall, the logit model with PCA-transformed performances an 67% accuracy rate, which is just moderate. The weighted average F1-score at 0.63 that reflects a lower model performance level after PCA-transformed. The model's tendency to favor the class '2', followed by class '1'. This indicates again the logit model with PCA-transformed do not performs as well as the model without PCA-transformation, but just impact on different classification of the target variable.

E. Bonus Task

1. Decision Tree

Decision trees [4, 12] are excellent choice for cluster interpretation because they are supervised models known for their ease of interpretation. Figure14 displays a decision tree with high-depth and complex. The decision tree predicts the quality of a product based on two input variables: Feature23 and 24. The tree starts by splitting the data into two nodes based on the value of Feature23. More details in the decision tree are described as following:

- Root node: The root node of the tree represents the entire dataset. At this node, the tree splits the data into two nodes based on the value of Feature23. This split is likely because Feature23 is a strong predictor of the quality of the product.
- Left node: The left node represents the subset of data where Feature23 is less than or equal to 0.5. At this node, the tree splits the data again based on the value of Feature24. This split is likely because Feature24 is also a predictor of the quality of the product, but it is not as strong as Feature23.
- Right node: The right node represents the subset of data where Feature23 is greater than 0.5. At this node, the tree does not split the data any further, because Feature23 is a strong enough predictor of the quality of the product on its own.
- Based on the bottom nodes, the decision tree seems to be able to cluster class1,2,3, which coincides with the results of ANOVA.

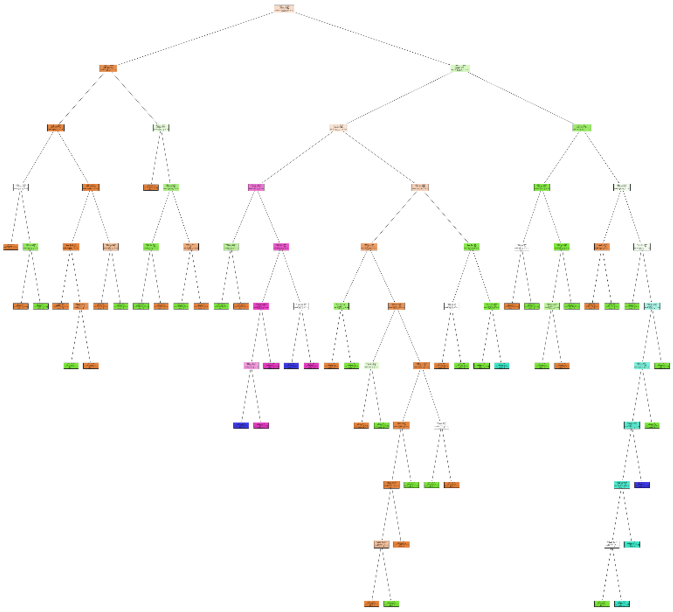


Figure 14

Overall, the decision tree is well-balanced, with a similar number of data points in each node. This suggest that the tree is not overfitting the data. In addition, the decision tree can be interpreted as a set of rules for predicting the quality of a product. For instance, the following rule can be inferred from the tree:

- If Feature23 > 0.5, then the product is good quality.
- If Feature23 <= 0.5, and Feature 24 > 0.75, then the product is of good quality.

Above rules can be used to predict the quality of new products by simply plugging in the values of Feature23 and Feature24.

2. KNN decision boundary

The K-Nearest Neighbor (KNN) [5, 11] model is a non-parametric algorithm that is among the simplest of algorithms available for classification. The KNN algorithm classifies a point based on the majority class among its closest neighbors. In this case the k is 5, which means the neighbor quantity is 5.

The KNN decision boundary (Figure15) shows very clear boundaries for the classifications in five different regions with different colors, each color represents the correspond class that the model predicts. The x-axis represents Feature 11 and y-axis represents Feature22. The points plotted over the decision boundaries represent the training data and their color indicates the true class label.

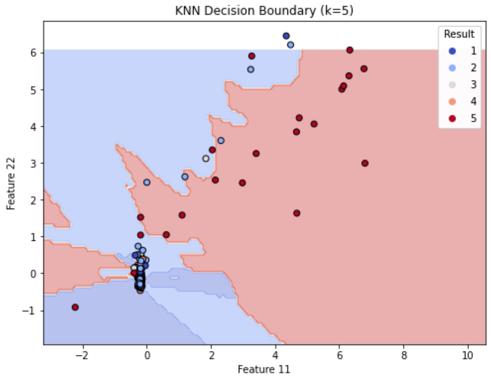


Figure 15

It appears to be some overlap between classes, as indicated by differently colored points within the same decision region. For example, there are points labeled for Class 1 within the decision boundary for Class 2. The model seems to be more accurate to detect class 1, 2, 5. While there are some mis-predictions, for instance, there are some data points of class 1, 2 are predicted to class 3.

3. LDA Decision Boundary

Linear Discriminant Analysis (LDA) [7, 10] is a technique for multi-class classification that can be used to automatically perform dimensionality reduction. Figure 16 shows the decision boundaries created by a LDA model in a two-dimensional feature space. The x-axis and y-axis likely represent the two most discriminative features (Feature 11 and 12) extracted by LDA, which are linear combinations of the original features.

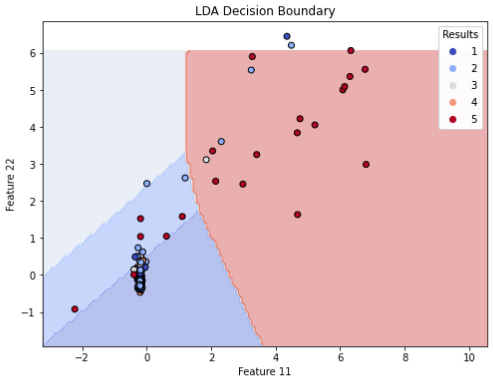


Figure 16

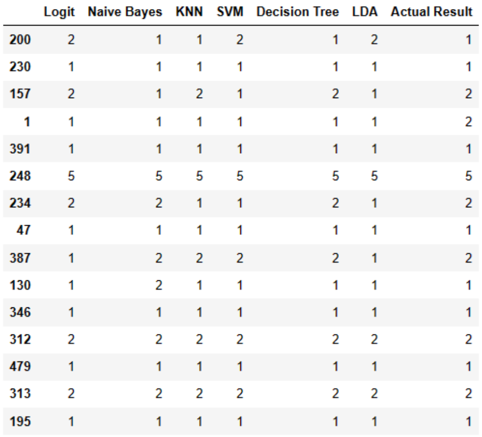
It appears that the LDA model has found a linear decision boundary that attempts to best separate the different classes based on the training data. There are clear overlaps between the classes, especially between classes 1, 2, and 3, which indicates that these classes are not perfectly linearly separable. This might suggest the need for either a more complex model or additional feature engineering to improve separation.

V RESULTS AND EVALUATION

Further exploration of the data on different machine learning models, which assists to evaluate the learning and prediction capabilities of different models for a given dataset. This will provide a comprehensive overview to select the most appropriate model to trained on the dataset and achieve the most accurate predictions, particularly when aiming at a certain class prediction.

A. *Models predictions and Actual Result table*

The table (Figure 17) displays the predictions made by six machine learning models for a set of experiments, which are used to compare with the actual result. The decision tree model appears to have a high degree of accuracy, as its predictions match the actual results quite frequently. The naive bayes model also shows high accuracy, matching the actual results in most experiments shown. There is a discrepancy in prediction for experiment 157, where logit model, KNN and decision tree predict class 2, while naive bayes, SVM and LDA predict class 1. The actual result is class 2, which makes the predictions by logit model, KNN and decision tree correct for this experiment. For experiment 248, all models correctly predict class 5, indicating a possible strong signal or clear class definition for this particular case. The LDA model's predictions do not align with the actual results in several cases (experiment 200, 157, 1, 234, 387), which could be suggested its lower performance metrics.



	Logit	Naive Bayes	KNN	SVM	Decision Tree	LDA	Actual Result
200	2	1	1	2	1	2	1
230	1	1	1	1	1	1	1
157	2	1	2	1	2	1	2
1	1	1	1	1	1	1	2
391	1	1	1	1	1	1	1
248	5	5	5	5	5	5	5
234	2	2	1	1	2	1	2
47	1	1	1	1	1	1	1
387	1	2	2	2	2	1	2
130	1	2	1	1	1	1	1
346	1	1	1	1	1	1	1
312	2	2	2	2	2	2	2
479	1	1	1	1	1	1	1
313	2	2	2	2	2	2	2
195	1	1	1	1	1	1	1

Figure 17

B. *Models Evaluation*

1. Model Learning Curve

The curve plot in Figure 18 displays six different learning curves for six different models, typically used to evaluate the performance of machine learning models as more training data is used. Each graph represents a model's performance on the training set (red line) and on cross-validation (green line), with the shaded areas indicating the variability (standard deviation) of the scores across different cross-validation folds.

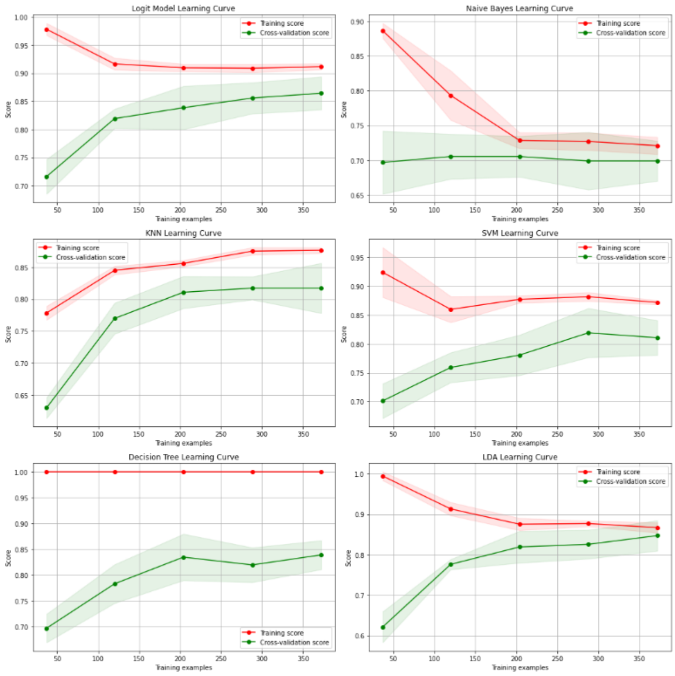


Figure 18

Logit model starts with high training score and decreases slightly as more data is introduced, while the cross-validation score increases with more data. The model is likely exhibiting slight overfitting with small amounts of data but generalizes better as more data is provided. The convergence of the training and validation scores suggests that adding more data will not significantly improve performance.

KNN model: The model training score gradually increases while the cross-validation score increases with more data. The model might be slightly overfitting with fewer data points but is generally well-performing.

Decision tree: The training and cross-validation scores converge as more data is added, with the training score decreasing and the cross-validation score increasing. This indicates that the model generalizes well and might not benefit much from additional data.

Naive Bayes model starts with a high training score that drops significantly as more data is added, and a cross-validation score that increases but then plateaus. This could indicate a model with high variance, which overfits the training data with smaller sample sizes.

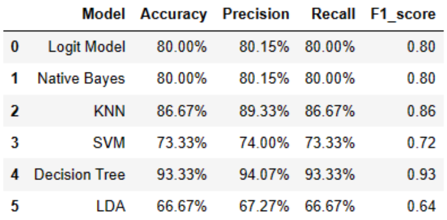
SVM model: The training score slightly decreases while the cross-validation score increases with more data. The model might be slightly overfitting with fewer data points but is generally well-performing.

LDA model: The training score is high and decreases as more data is introduced, while the cross-validation score increases but with a lower slope. The model might be slightly overfitting and could benefit from regularization or simpler model complexity.

In terms of learning curve, model generalize and model fitting, KNN and decision tree demonstrate stronger learning capabilities on the given dataset compared with other models.

2. Model Evaluation Table

The table (Figure19) is a concise summary of the model's performance intuitively with result data. It can be used to compare and select the best model for further development or application in real case. The model evaluation mainly covers several aspects:



	Model	Accuracy	Precision	Recall	F1_score
0	Logit Model	80.00%	80.15%	80.00%	0.80
1	Native Bayes	80.00%	80.15%	80.00%	0.80
2	KNN	86.67%	89.33%	86.67%	0.86
3	SVM	73.33%	74.00%	73.33%	0.72
4	Decision Tree	93.33%	94.07%	93.33%	0.93
5	LDA	66.67%	67.27%	66.67%	0.64

Figure 19

Accuracy: Measures the percentage of total correct predictions. The decision Tree model has the highest accuracy (93.33%), indicating it correctly predicted the most instances. Followed is KNN with 86.87%. LDA has the lowest accuracy (66.67%), making it the least accurate model for the dataset.

Precision: Indicates the ratio of true positive predictions to the total positive predictions made. It shows how precise a model is in predicting positive instances. Again, decision tree has the highest precision (94.07%), which leads almost 5% of the followed KNN (89.33%). The lowest precision is LDA (67.27%).

Recall (Sensitivity): Measures the ratio of true positive predictions to all actual positive instances. It shows how well the model can find all the positive instances. Similar to precision, the decision tree model leads with a recall of 93.33%, and LDA is at the bottom with 66.67%.

F1_score: The harmonic mean of precision and recall, this metric considers both false positives and false negatives. It is particularly useful when the class distribution is uneven. The decision tree model scores the highest F1-score (0.93), reflecting a balanced precision and recall. LDA has the lowest F1_score (0.64), indicating it is not as effective in balancing recall and precision.

Overall, whether in terms of model learning ability, or comprehensive assessment of accuracy, precision, recall or F1_score, it can conclude that decision tree and KNN can be the model option for further training or development in the given dataset. While LDA shows lowest performance metrics as seen in all the previous analysis. However, each model's performance might vary with changes in hyperparameters, feature engineering, or if cross-validation is applied, so further tuning and validation are recommended before finalizing a model for deployment.

VI TOOLS AND METHOD OVERVIEW

Visualization Tools:

- Histogram
- Heatmap
- Scatterplot
- Boxplot
- Curve plot
- Tables
- Count plot
- Tree plot
- Decision boundary plot

Methods:

- Normal Distribution
- Pearson correlation
- ANOVA (boxplot)
- Logit Model
- Dimension Reduction (PCA, LDA)
- Model performance comparation: Logit model, Naïve Bays, KNN, SVM, Decision tree, LDA

BIBLIOGRAPHY

[1] Abbott. D., Komp. W.J. (2014). *Applied Predictive Analytics*. Chapter 3, Page 46-50

[2] Abbott. D., Komp. W.J. (2014). *Applied Predictive Analytics*. Chapter 4, Page 113-114

[3] Abbott. D., Komp. W.J. (2014). *Applied Predictive Analytics*. Chapter 7, Page 204-205

[4] Abbott. D., Komp. W.J. (2014). *Applied Predictive Analytics*. Chapter 8, Page 214-229

[5] Abbott. D., Komp. W.J. (2014). *Applied Predictive Analytics*. Chapter 8, Page 254-263

[6] Brownlee. J. (2020). *Multinomial Logistic Regression With Python*. [online] Available at: https://machinelearningmastery.com/multinomial-logistic-regression-with-python/

[7] Brownlee. J. (2020). *Linear Discriminant Analysis for Dimensionality Reduction in Python*. [online] Available at: https://machinelearningmastery.com/linear-discriminant-analysis-for-dimensionality-reduction-in-python/

[8] Geeksforgeeks. (). *How to create a seaborn correlation heatmap in Python?* [online] Available at: https://www.geeksforgeeks.org/how-to-create-a-seaborn-correlation-heatmap-in-python/

[9] Turney.S. (2021). *Pearson Correlation Coefficient (r) | Guide & Examples*. [online] Available at: https://www.scribbr.com/statistics/pearson-correlation-coefficient/

[10] Scikit Learn (2007-2023). *Linear and Quadratic Discriminant Analysis with covariance ellipsoid*. [online] Available at: https://scikit-learn.org/stable/auto_examples/classification/plot_lda_qda.html

[11] Scikit Learn (2007-2023). *Nearest Neighbors Classification*. [online] Available at: https://scikit-learn.org/stable/auto_examples/neighbors/plot_classification.html

[12] Scikit Learn (2007-2023). *Decision Trees*. [online] Available at: https://scikit-learn.org/stable/modules/tree.html

[13] Siegrist.K. (2022). *Skewness and Kurtosis*. [online] Available at: https://stats.libretexts.org/Bookshelves/Probability_Theory/Probability_Mathematical_Statistics_and_Stochastic_Processes_(Siegrist)/04%3A_Expected_Value/4.04%3A_Skewness_and_Kurtosis