IE 7300 Statistical Learning for Engineering

Dry Bean Classification

Group 9

Sabeen Mustafa

Shravya Vura

Mahavir Sunil Kathed

Ruthik Raj Nataraja

Under the Guidance:

Siddhartha Putti

Sachini Weerasekara

Professor:

Ramin Mohammadi

ABSTRACT

Dry bean classification plays a vital role in the agricultural industry, enabling farmers and researchers to identify and categorize different types of beans accurately. The goal of our project is to use machine learning to create a reliable and effective multi-class classification system for dry beans. First, we perform exploratory data analysis to find the characteristics of the dataset by finding summary, visualizing, and finding the correlation between columns. Mostly the dataset was clean for analysis. After that, machine learning techniques were used to analyze the dataset: Quadratic Discriminative Analysis, Logistic Regression and Gaussian Naïve Bayes. Then, we evaluate the performance of each model using metrics like accuracy, F1 score, precision, and recall.

INTRODUCTION

Dry beans are a significant crop that is a necessary self-pollinated legume that is consumed by millions of people worldwide, especially in Sub-Saharan Africa. Beans are a staple food for more than 200 million people because humans consume nearly 50% of grain legumes directly in developing nations. Bean seeds must be manually sorted and classified, but this is time-consuming and ineffective, so automatic grading and classification techniques are required. To ensure quality control and adhere to international standards for authorized seed, objective measurement systems supported using machine learning techniques are essential. Machine learning algorithms can perform precise analysis of visual characteristics, which are crucial in consumers' decisions to buy food products.

The bean industry can streamline operations, reduce labor-intensive processes, and offer consumers dependable and effective products by creating accurate and efficient automated systems.

Researchers working in the field of bean grading and classification have benefited greatly from recent technological developments. In this field, computer vision systems (CVSs) have proven to be useful tools for objective measurement and quality control. The sensory qualities of agricultural products have been evaluated using CVS technology, which is primarily based on cameras and computers. This system is made up of a light source, an image capture tool, and computer add-ons with related software. These systems digitized data can be widely disseminated and provide useful information with a variety of attributes.

A balanced dataset is essential for ensuring top performance, with an equal number of input samples representing each output class or target class. Real-time performance may suffer from the use of imbalanced training data, meaning that observations are distributed unevenly among target classes. The problems presented by unbalanced datasets have been the subject of numerous studies in this area.

In conclusion, it is crucial to use machine learning in the grading and classification of dry beans. It vastly enhances effectiveness, precision, and quality control, optimizing bean industry operations. Processes can be streamlined, global standards can be met, and better products can be offered to customers by utilizing the power of machine learning.

DATA DESCRIPTION

The data was obtained from UCI Machine Learning repository: https://archive.ics.uci.edu/dataset/602/dry+bean+dataset.

The dataset contains comprehensive data on 7 different varieties of dry beans, including Seker, Barbunya, Bombay, Cali, Dermosan, Horoz, and Sira. For each instance in the dataset, a total of 17 attributes are available. The following characteristics are among them: area, perimeter, major and minor axis lengths, aspect ratio, eccentricity, convex area, equivalent diameter, extent, solidity, roundness, compactness, and shape.

- Area - (A), The size of a bean zone and the quantity of pixels included inside it.
- Perimeter - (P), The length of a bean's border is known as its perimeter.
- MajorAxisLength - (L), The length of the line that can be traced from a bean, measured between its ends.
- MinorAxisLength - (l), The longest line that may be drawn from the bean while positioned perpendicular to the main axis is known as the minor axis length.
- AspectRatio - (K), Describes how L and l are related to one another.
- Eccentricity (Ec), the ellipse's eccentricity when its moments are equal to those of the region.
- ConvexArea - (C), The quantity of pixels in the smallest convex polygon that can accommodate a bean seed's surface area.
- EquivDiameter - (Ed): The circumference of a circle with a diameter equal to that of a bean seed.
- Extent - (Ex), The extent is the ratio of pixels in the bounding box to the bean area.
- Solidity - (S), the proportion of bean pixels to those found in a convex shell.
- Roundness - (R), Calculated using the formula: (4piA)/(P^2)
- Compactness - (CO), determines how round an object is: Ed/L
- ShapeFactor1 - (SF1)
- ShapeFactor2 - (SF2)
- ShapeFactor3 - (SF3)
- ShapeFactor4 - (SF4)
- class - The class of the bean.

Over 13,611 instances make up the dataset. The previously mentioned 7 different types of dry beans make up the primary target class. Computer vision systems were used to image these beans and then capture and analyze them after segmentation and feature extraction.

Class distribution:

DERMASON	3546
SIRA	2636
SEKER	2027
HOROZ	1928
CALI	1630
BARBUNYA	1322
BOMBAY	522

METHODS

The data is split into training and testing. The data set used to build the model, the training dataset, includes a target and well-known features. The model is validated using the test set to check the performance of the model. The dry bean categorical classes, "SIRA," "BOMBAY," "DERMASON," "BARBUNYA," "HOROZ," "CALI," and "SEKER" were transformed into the corresponding integer types 0 through 6.

Machine Learning Algorithms:

In this project we are implementing: Naïve Bayes, Logistic Regression and Quadratic Discriminative Analysis.

1. **Naïve Bayes -** A probabilistic classifier based on the Bayes theorem called Gaussian Naive Bayes assumes that the features have a Gaussian distribution. It is a variation of the Naive Bayes algorithm, which is renowned for being both straightforward and effective when used for classification problems. Continuous or real-valued characteristics are particularly well suited for Naive Bayes.

   The assumption of gaussian naïve bayes is that the dataset is normally distributed. However, our dataset is skewed so naïve bayes is not able to perform well. Additionally, given the class label, it is assumed that the features are conditionally independent of one another. The "naive" assumption, which assumes independence even though it might not always be true, makes it easier to calculate probabilities.

   Pros:

   - It is easy to implement and saves time and effort.
   - It can be efficiently used for multi class classification.

   Cons:

   - Assumes that all the features are independent.

2. **Multinomial Logistic Regression –** Logistic Regression is a statistical technique that predicts probability of an instance belonging to a particular class. It is a discriminative type of classification. In multiclass we are using the concept of one-vs-all for finding the probability. The instance will be classified in the class which has the highest probability after applying the model.
   As logistics regression is relatively easy to implement and requires less computation power, it can be considered as baseline model. This algorithm when we know that the classes can be separated by a decision boundary.

   Pros:
   - Logistic Regression is easy to implement and analyze compared to other algorithms.
   - It does not make any assumptions about distribution of classes.

   Cons:

   - It may lead to overfitting if the number of rows is less then number of columns.
   - The assumption that the dependent variable and the independent variables are linear relationships is the main drawback of logistic regression.
   -

3. **Quadratic Discriminative Analysis –** QDA is a classification algorithm with each class modelled by a multivariant Gaussian Distribution such that the decision boundary is quadratic. It works on the assumption that the correlation between each class data is not the same. QDA can capture correlation between classes better than Naïve Bayes.

   When there are an uneven number of observations in each class, an unbalanced dataset can be handled by QDA. The decision boundaries are established based on the class-specific distributions and do not presume equal class priors.

   Pros:
- It constructs a quadratic decision boundary; hence, it is better at classifying nonlinear data.

   Cons:

- The gaussian assumption has a high training time.

EXPLORATORY DATA ANALYSIS

Feature Engineering

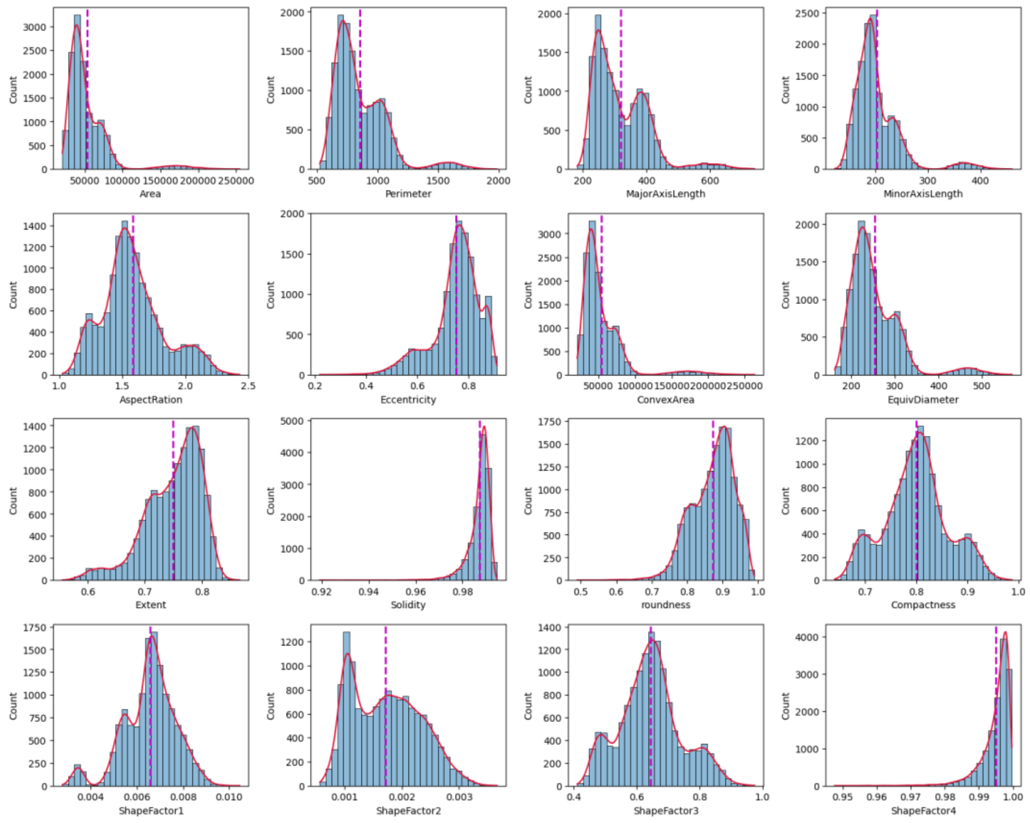
a) Null values

```
Missing values:
 Area               0
Perimeter          0
MajorAxisLength    0
MinorAxisLength    0
AspectRation       0
Eccentricity       0
ConvexArea         0
EquivDiameter      0
Extent             0
Solidity           0
roundness          0
Compactness        0
ShapeFactor1       0
ShapeFactor2       0
ShapeFactor3       0
ShapeFactor4       0
Class              0
dtype: int64
```

b) Encoding categorical data – The column class has categorical data; hence, we have utilized one hot encoding and label encoding for converting the categories into numerical.

c) Scaling and normalization - To prevent some features from dominating others during model training, numerical features were rescaled to a similar range.

d) Skew distribution -



This faceted graph shows the frequency of each variable values. Some of the variables are right skewed while others are left skewed. The shapefactor variables are almost like a normal distribution.

e) Outliers - Outliers are data points in a dataset that differ dramatically from the bulk of the data. These observations could have values that are exceptionally high or low when compared to the other data points.

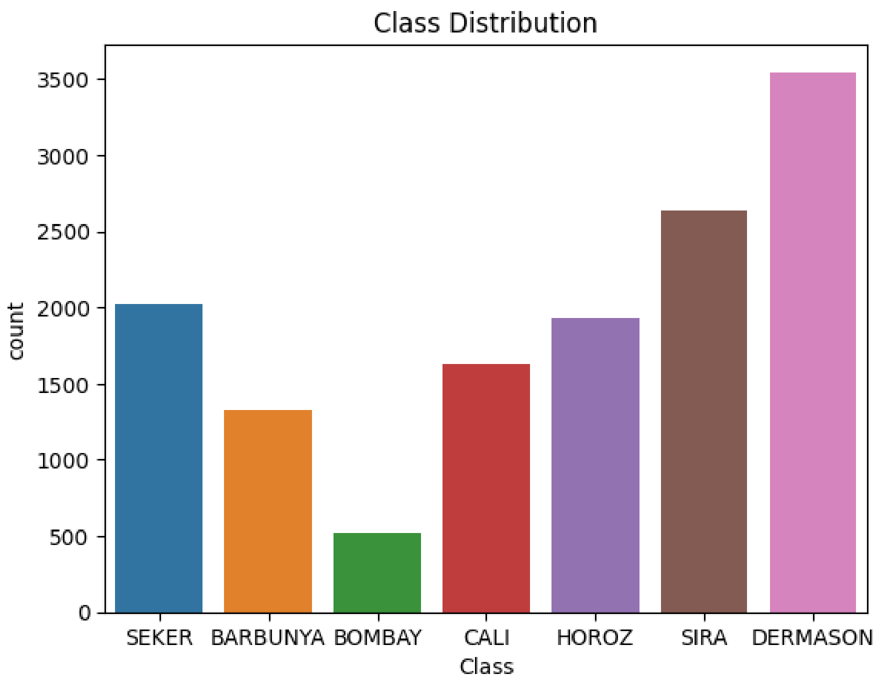
Feature selection

The correlation of a few columns was extremely high. Hence, we deleted 5 columns to reduce redundancy. The correlation of the deleted columns was around 0.95. We dropped columns: ConvexArea, EquiDiameter, Compactness, ShapeFactor3, MajorAxisLength.

Visualizations

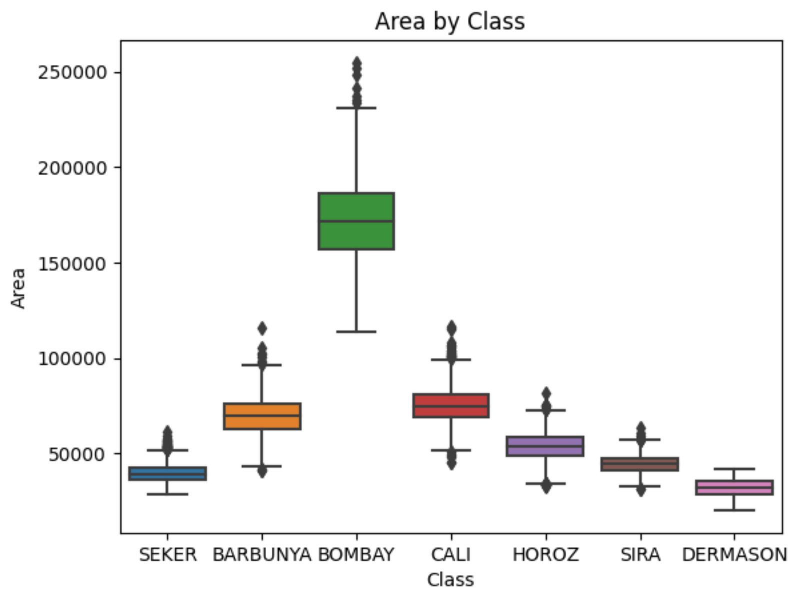
a) Bar Graph

Bar graph for each class:



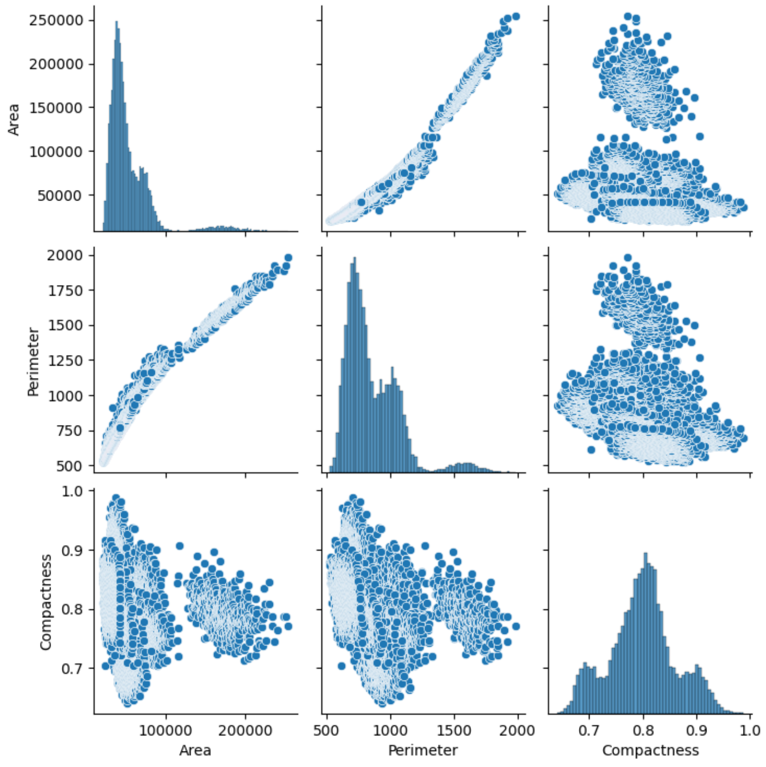
Here, we can see that Dermason has the highest instances while SIRA comes in second. Bombay has the least frequency.

b) Box Plot



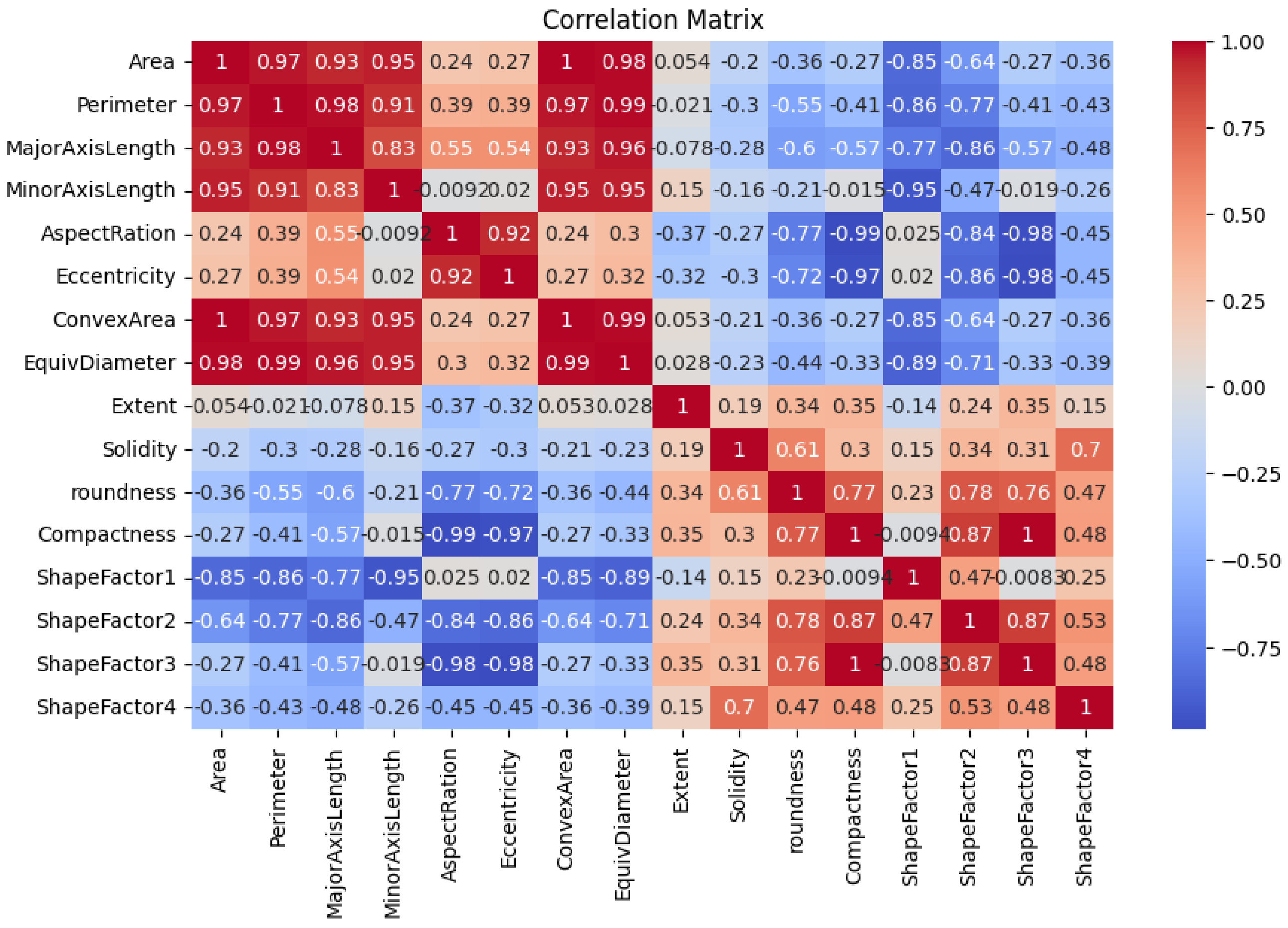
The Box plot shows the area of different classes. Bombay has the highest Area by class distribution.

c) Pairplot - The pair plot below shows the various connections between various qualities. While certain traits are distributed and associated at random, others are correlated either adversely, favorably, or both.



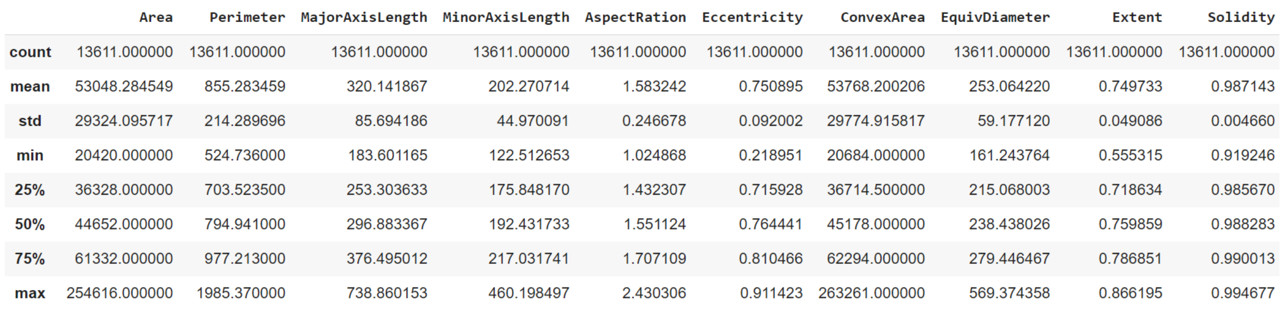
Statistical Analysis

a) Correlation - The presence and strength of a relationship between the variables is ascertained using correlation analysis.



	Area	Perimeter	MajorAxisLength	MinorAxisLength	AspectRation	Eccentricity	ConvexArea	EquivDiameter	Extent	Solidity	roundness	Compactness	ShapeFactor1	ShapeFactor2	ShapeFactor3	ShapeFactor4
Area	1	0.97	0.93	0.95	0.24	0.27	1	0.98	0.054	-0.2	-0.36	-0.27	-0.85	-0.64	-0.27	-0.36
Perimeter	0.97	1	0.98	0.91	0.39	0.39	0.97	0.99	-0.021	-0.3	-0.55	-0.41	-0.86	-0.77	-0.41	-0.43
MajorAxisLength	0.93	0.98	1	0.83	0.55	0.54	0.93	0.96	-0.078	-0.28	-0.6	-0.57	-0.77	-0.86	-0.57	-0.48
MinorAxisLength	0.95	0.91	0.83	1	-0.0092	0.02	0.95	0.95	0.15	-0.16	-0.21	-0.015	-0.95	-0.47	-0.019	-0.26
AspectRation	0.24	0.39	0.55	-0.0092	1	0.92	0.24	0.3	-0.37	-0.27	-0.77	-0.99	0.025	-0.84	-0.98	-0.45
Eccentricity	0.27	0.39	0.54	0.02	0.92	1	0.27	0.32	-0.32	-0.3	-0.72	-0.97	0.02	-0.86	-0.98	-0.45
ConvexArea	1	0.97	0.93	0.95	0.24	0.27	1	0.99	0.053	-0.21	-0.36	-0.27	-0.85	-0.64	-0.27	-0.36
EquivDiameter	0.98	0.99	0.96	0.95	0.3	0.32	0.99	1	0.028	-0.23	-0.44	-0.33	-0.89	-0.71	-0.33	-0.39
Extent	0.054	-0.021	-0.078	0.15	-0.37	-0.32	0.053	0.028	1	0.19	0.34	0.35	-0.14	0.24	0.35	0.15
Solidity	-0.2	-0.3	-0.28	-0.16	-0.27	-0.3	-0.21	-0.23	0.19	1	0.61	0.3	0.15	0.34	0.31	0.7
roundness	-0.36	-0.55	-0.6	-0.21	-0.77	-0.72	-0.36	-0.44	0.34	0.61	1	0.77	0.23	0.78	0.76	0.47
Compactness	-0.27	-0.41	-0.57	-0.015	-0.99	-0.97	-0.27	-0.33	0.35	0.3	0.77	1	-0.0094	0.87	1	0.48
ShapeFactor1	-0.85	-0.86	-0.77	-0.95	0.025	0.02	-0.85	-0.89	-0.14	0.15	0.23	-0.0094	1	0.47	-0.0083	0.25
ShapeFactor2	-0.64	-0.77	-0.86	-0.47	-0.84	-0.86	-0.64	-0.71	0.24	0.34	0.78	0.87	0.47	1	0.87	0.53
ShapeFactor3	-0.27	-0.41	-0.57	-0.019	-0.98	-0.98	-0.27	-0.33	0.35	0.31	0.76	1	-0.0083	0.87	1	0.48
ShapeFactor4	-0.36	-0.43	-0.48	-0.26	-0.45	-0.45	-0.36	-0.39	0.15	0.7	0.47	0.48	0.25	0.53	0.48	1

b) Statistical Summary –



	Area	Perimeter	MajorAxisLength	MinorAxisLength	AspectRation	Eccentricity	ConvexArea	EquivDiameter	Extent	Solidity
count	13611.000000	13611.000000	13611.000000	13611.000000	13611.000000	13611.000000	13611.000000	13611.000000	13611.000000	13611.000000
mean	53048.284549	855.283459	320.141867	202.270714	1.583242	0.750895	53768.200206	253.064220	0.749733	0.987143
std	29324.095717	214.289696	85.694186	44.970091	0.246678	0.092002	29774.915817	59.177120	0.049086	0.004660
min	20420.000000	524.736000	183.601165	122.512653	1.024868	0.218951	20684.000000	161.243764	0.555315	0.919246
25%	36328.000000	703.523500	253.303633	175.848170	1.432307	0.715928	36714.500000	215.068003	0.718634	0.985670
50%	44652.000000	794.941000	296.883367	192.431733	1.551124	0.764441	45178.000000	238.438026	0.759859	0.988283
75%	61332.000000	977.213000	376.495012	217.031741	1.707109	0.810466	62294.000000	279.446467	0.786851	0.990013
max	254616.000000	1985.370000	738.860153	460.198497	2.430306	0.911423	263261.000000	569.374358	0.866195	0.994677

PCA - Feature engineering uses Principal Component Analysis (PCA) as a dimensionality reduction method to convert a high-dimensional dataset into a lower-dimensional space. It attempts to minimize information loss while simultaneously capturing the most crucial data or patterns.

Using the variance method, we used PCA to identify the best features and variables that give us the most description.

RESULT

Naïve Bayes

An example of a generative classification method is naive bayes classification, which by default includes prior probabilities. Generative classification strategies consider both the joint distribution of features and classes, as opposed to discriminative classification techniques, which only concentrate on the relationship between features and classes. Naïve Bayes considers that the features are independent.

The following formula is used for calculating the probability:

$$P(y_i \mid x_1, x_2, \ldots, x_n) = P(x_1, x_2, \ldots, x_n \mid y_i) * P(y_i)$$

$$P(y_i) = \text{examples with } y_i \text{ / total examples}$$

In this project we have numerical columns with 6 classes, so, we are using gaussian naïve bayes.

$$P(X_k|Y=J) \sim N(\mu, \sigma\text{^}2)$$

where:

• $P(X_k|Y=J)$ denotes the probability of feature X_k given class $Y=J$.

• $N(\mu, \sigma\text{^}2)$ represents a normal distribution with mean μ and variance $\sigma\text{^}2$.

The formula for calculating the probability of feature X_k given class $Y=J$ is:

$$P(X_k|Y=J) = [1 / (2\pi)(0.5 * \sigma\text{^}2)] * e\text{^}[-((X- \mu)\text{^}2) / (2\pi * \sigma\text{^}2)\text{^}(0.5)]$$

To implement Naïve Bayes, we initially used Label Encoding to convert the class labels into numerical values. We then assessed the correlation between the columns to gain insights into the model's performance. However, we encountered a significant challenge as the model's performance was consistently low. Given the high correlation between certain columns, we decided to drop those columns in an attempt to improve the model's performance. Unfortunately, even after this step, the model did not exhibit satisfactory results.

Realizing that the remaining columns might not be independent, we decided to apply Principal Component Analysis (PCA) to address this issue. Although PCA aims to make the columns

independent, it is an unsupervised algorithm and does not take the classes into consideration. As a result, the accuracy of the model declined further.
Another approach we attempted was dropping the columns with high correlation among themselves. We retained only the columns with minimal correlation and once again applied PCA. However, despite these efforts, the model's performance remained unsatisfactory, possibly due to the data not being normally distributed.

As the data is skewed, the gaussian naïve bayes doesn’t work well and the performance is still low.

Model Naive Bayes	Accuracy
Naive Bayes	48.33
Naïve Bayes + dropping columns	41.25
Naive Bayes + PCA	42.28
Naïve Bayes + PCA + dropping columns	37.5

Logistic Regression
The goal of logistic regression is to develop a mapping function that uses the properties of the data to forecast the likelihood that a new example will correspond to one of the targets classes.

We have used one vs all method to implement logistic regression for multi class classification. The one-vs-the-rest (OvR) method involves training a distinct logistic regression model for each class, treating that class as the positive class and the other classes as the negative classes. The predicted class is determined by which model produces the highest probability during prediction. The multiclass problem is effectively split up into several binary classification problems using the OvR technique.

The sigmoid function is defined as:

$$h(t) = 1 / (1 + e^{(-t)})$$

Next, we use the cost function for logistic regression, which is given by:

$$\text{Cost}(h(t), y) = -\log(h(t)) \text{ if } y = 1 \ -\log(1 - h(t)) \text{ if } y = 0$$

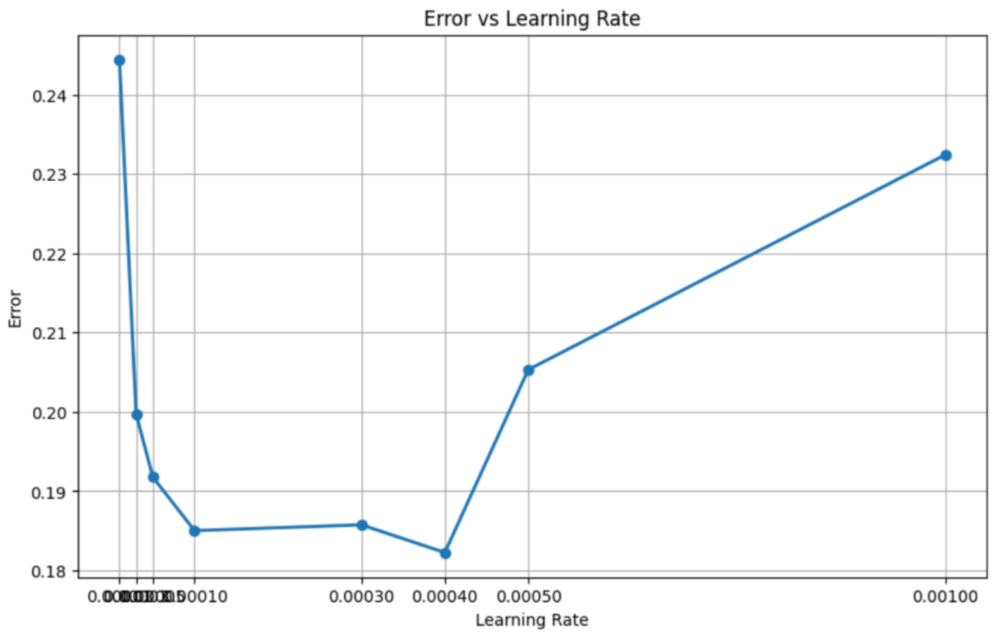
where h(t) denotes the output of the sigmoid function and y represents the actual target label.

The cost function, commonly referred to as log-loss, is intended to penalize the model for predicting the wrong class. The cost approaches zero when the model predicts that the target class will be 1.

When it came to logistic regression in our situation, our model converged and provided us with the best theta values, signaling that the learning process had come to an end. We obtained the following values for logistic regression:

Test Accuracy: 0.8993633692458374
Error_test: 0.10063663075416263
Train_Accuracy: 0.9030124908155768
Error_train: 0.09698750918442323
Total Error: 0.19762413993858585

Logistic Regression has a hyper parameter alpha. For obtaining an optimal model we are implementing Bias Variance Tradeoff.
Bias refers to the error introduced by approximating a real-world problem with a simplified model. Variation is a measure of how flexible or variable a model is. When trained on various subsets of the training data, it calculates how much the model's predictions change.



After running a bias variance tradeoff, we discovered that 0.0004 was the ideal value for learning rate. Our model's error was minimum by using optimum alpha value. After performing bias variance tradeoff, we obtained the following metrics:

Test Accuracy: 0.9071988246816847
Error_test: 0.09280117531831533
Train_Accuracy: 0.9145586228613415
Error_train: 0.08544137713865851
Total Error: 0.17824255245697385

We can observe that the accuracy of the model increased after finding the optimal alpha value.

Quadratic Discriminative Analysis
It is based on the supposition that different classes may have different class-specific covariance matrices. To divide the classes, QDA seeks to identify quadratic decision boundaries.

In order to implement Quadratic Discriminant Analysis (QDA), we first utilize Label Encoding to convert the class labels into numerical values. QDA operates by creating a decision boundary for each class using the "one vs. all" technique, since our data involves multiple classes. It establishes separate decision boundaries for each class, and when a test data point is presented, it assesses the probability of that data point belonging to each class. If the probability is high, indicating that the point is significantly distant from the decision boundary of a particular class, it is classified as belonging to that class.

Essentially, QDA examines the proximity of the test data point to the decision boundaries of each class and classifies it based on the class whose decision boundary it is farthest from. This approach allows for effective and informative classification in multiclass scenarios.

QDA can be calculated using the following formula:

$$P(Y = k|X = x) = (\pi\_k / \sqrt{((2\pi)\hat{}(p)\ |\Sigma\_k|)}) * \exp(-0.5 * (x - \mu\_k)'\ \Sigma\_k\hat{}(-1)\ (x - \mu\_k))$$

In this equation:

$P(Y = k|X = x)$ - conditional probability of class k given the input x.
π_k - prior probability of y = k.
p - number of features.
$|\Sigma\_k|$ - determinant of covariance matrix of y = k.
$\Sigma\_k\hat{}(-1)$ - inverse of covariance matrix of y = k.
μ_k - mean of y = k.

Assigning the input x to the class with the highest conditional probability is the QDA decision rule. To put it another way, the predicted class is chosen to maximize P(Y = k|X = x). After implementing QDA we obtained the following metrics:

Test_Accuracy: 0.9076885406464251
Train_Accuracy: 0.9155033063923586

RESULT:

Model	Accuracy
Naïve Bayes	48.3333 %
Logistic Regression	90.52%
Quadratic Discriminative Analysis	90.76%

CONCLUSION

Based on the obtained results, it can be concluded that QDA exhibited superior performance compared to the other algorithms. QDA achieved an accuracy of 90.76%, surpassing logistic regression which achieved an accuracy of 90.52%. Meanwhile, the accuracy of naive Bayes was notably lower at 48.33%. These findings indicate that QDA proved to be the most accurate model among the evaluated algorithms.

The choice of a model depends on the specific problem at hand and the evaluation metrics that are given priority. When accuracy metrics are used as the primary measure, QDA is typically selected as the optimal choice. However, if we are aware that the prior probability of each class remains constant and unchanging, QDA is favored as the main model. Conversely, if the prior probability is subject to change, Logistic Regression is preferred. The decision between QDA and Logistic Regression is influenced by these considerations, emphasizing the need to align the model selection with the specific characteristics and requirements of the problem being addressed.