NORTHEASTERN UNIVERSITY

IE 7300: STATISTICAL LEARNING FOR ENGINERRING

PREDICTIVE MODELING FOR DIABETES-RELATED READMISSIONS IN U.S HOSPITALS

GROUP 3

GROUP MEMBERS

NIRESH SUBRAMANIAM

002745713

subramanian.ni@northeastern.edu

SACHIN DEWANGAN

002771508

dewangan.s@northeastern.edu

VENKATA KRISHNAN R

002749877

ravichandran.ve@northeastern.edu

VISHNU GIRISH

002781416

nujarallaiyappan.v@northeastern.edu

INSTRUCTOR: RAMIN MOHAMMADI

DATE: 8th OCTOBER 2023

Abstract

This project endeavors to address the challenge of diabetes-related readmissions within 60 days of hospitalization by utilizing statistical learning techniques. In the initial phase, our project began with comprehensive preprocessing steps aimed at data cleaning, normalization, and feature engineering. Following this crucial step, we strategically partitioned the dataset into distinct segments for training and testing purposes. This segregation ensured that the models were trained on a designated portion of the data, allowing for learning patterns and relationships, while the untouched test set served as an unbiased yardstick to assess the models' performance on unseen data, thereby validating their predictive capabilities.

Our goals involve building a predictive model for readmission risk estimation, feature selection and engineering, model evaluation using machine learning algorithms, and ensuring interpretability for actionable insights. Our primary objectives include constructing a robust predictive model for timely interventions by healthcare providers and identifying influential features impacting readmission risk. We utilized Logistic Regression, Naive Bayes, and Neural Network models, with Neural Networks identified as the optimal choice based on the Expected Loss metric. While logistic regression achieves high sensitivity, Neural Networks outperform in overall performance metrics. Naive Bayes and Neural Networks exhibit improved accuracies compared to the baseline model, emphasizing their potential for reducing readmission rates in healthcare practices.

Introduction

In the dynamic field of healthcare analytics and predictive modeling, we address the pressing issue of diabetes related hospital readmissions. These readmissions pose a significant challenge to healthcare due to their financial burden and impact on patient well-being. Leveraging the "Diabetes 130-US hospitals for years 1999-2008" dataset from the UCI Machine Learning Repository, our project's aim is to develop a predictive model. Diabetes, a prevalent chronic condition, demands continuous care, and by understanding the factors contributing to readmissions, we can empower healthcare providers to take proactive steps to enhance care quality and mitigate the financial implications.

Data Description

The dataset, obtained from UCI's repository, consists of information from 130 U.S. hospitals spanning the years 1999 to 2008. It provides a comprehensive set of attributes, including patient demographics, clinical and treatment-related factors, and hospital operational data. The primary dataset comprises over 100,000 instances, each representing a hospital admission, and 47 features. The key information we have at our disposal includes patient demographics, admission sources, clinical attributes (e.g., diagnoses, procedures), medication information, and various hospital-related factors.

Model Description

Logistic Regression

Logistic regression is a statistical model used to predict a binary outcome (e.g., yes/no, success/failure) based on a set of independent variables. It is a widely used method in various fields, including machine learning, statistics, and economics.

Mathematical Model

The logistic regression model can be expressed mathematically as:

$$P\ (Y \mid X) = 1 / (1 + exp(-\beta 0 - \beta 1 * X1 - \ldots - \beta n * Xn))$$

where:

- P (Y | X) is the probability of the event occurring given the independent variables X.
- β0 is the intercept of the model.

- β1, ..., βn are the coefficients of the model.
- X1, ..., Xn are the independent variables.

The coefficients of the model are estimated by maximizing the log-likelihood function. The log-likelihood function measures how well the model fits the data.

Assumptions

- Binary outcome variable: The dependent variable must be binary.
- Independence of observations: Each observation in the data set should be independent of all other observations.
- No multicollinearity: The independent variables in the data set should not be highly correlated with each other.
- Linear relationship between the logit and the independent variables: The relationship between the logit of the dependent variable (the natural logarithm of the odds of the event happening) and the independent variables should be linear.
- Lack of influential outliers: Your data set should not contain any influential outliers.

Naive Bayes

Naive Bayes is a family of probabilistic classifiers based on Bayes' theorem with strong independence assumptions between features. Despite its simplicity, it can achieve high accuracy in various machine learning tasks, particularly text classification.

Mathematical Model

The Naive Bayes classifier predicts the class of a data point by calculating the posterior probability of each class and choosing the class with the highest probability. The posterior probability can be calculated using Bayes' theorem:

$$P(C \mid X) = P(X \mid C) * P(C) / P(X)$$

where:

- P (C | X) is the posterior probability of class C given data point X.
- P (X | C) is the likelihood of data point X given class C.

- P(C) is the prior probability of class C.
- P(X) is the probability of data point X.

Naive Bayes makes the strong independence assumption: all features are independent of each other given the class. This allows us to simplify the calculation of the likelihood:

$$P(X \mid C) = \Pi\_i\, P(x\_i \mid C)$$

where:

- x_i is the i-th feature of data point X.
- Π_i represents the product over all features.

Naive Bayes variants:

- Multinomial Naive Bayes: Suitable for discrete data represented by counts, like word counts in text documents.
- Gaussian Naive Bayes: Suitable for continuous data assuming Gaussian distribution.
- Bernoulli Naive Bayes: Suitable for binary data, where each feature can only take on two values.

Assumptions:

- Feature Independence: All features are independent given the class (often violated).
- Feature Distribution: Specific distribution assumed based on model variant (Multinomial, Gaussian, Bernoulli).
- No Missing Data: Missing values require imputation before use.
- Equal Class Priors: Model may be biased if priors are significantly different.
- Large Sample Size: Necessary for accurate estimates of probability distributions.

Neural Network

Neural networks are a powerful class of machine learning models inspired by the structure and function of the human brain. They consist of interconnected nodes, called neurons, arranged in layers. These neurons process information by applying activation functions to weighted sums of their inputs. The network learns by adjusting the weights of these connections based on training data.

Structure:

A neural network typically consists of three types of layers:

- Input Layer: Receives the raw input data.
- Hidden Layers: Perform the majority of the computation and information processing. There can be multiple hidden layers, each with a different number of neurons.
- Output Layer: Generates the network's predictions or outputs.

Types of Neural Networks:

There are many different types of neural networks, each with its own strengths and weaknesses. Some common types include:

- Feedforward Neural Networks: Information flows in one direction, from the input layer to the output layer.
- Recurrent Neural Networks (RNNs): Can store information over time, making them suitable for sequential data like text and speech.
- Convolutional Neural Networks (CNNs): Efficiently process grid-like data like images and videos.

Learning Process:

Neural networks learn by adjusting the weights of their connections based on training data. This process typically involves the following steps:

1. Forward Pass: The input data is propagated through the network, and the output is generated.
2. Loss Calculation: The difference between the predicted output and the actual output is calculated (e.g., mean squared error for regression, cross-entropy for classification).
3. Backward Pass: The loss is propagated back through the network, and the weights are adjusted based on their contribution to the error. This process uses algorithms like gradient descent.
4. Repeat: Steps 1-3 are repeated iteratively until the network converges on a satisfactory solution.

Exploratory Data Analysis (EDA)

Upon initial examination, our dataset comprised 101,766 rows and 48 columns. Subsequent analysis revealed missing values across several columns, notably in the fields of Race, Weight, Payer Code, Medical Specialty, and in the diagnostic columns diag_1, diag_2, and diag_3. It was observed that Weight exhibited a substantial 96.86% null values, while Payer Code accounted for approximately 40% null values. Given our domain expertise indicating that the payment method might not significantly impact the readmission variable, and similarly, recognizing that the admitting physician's specialty—represented by Medical Specialty—might not strongly correlate with the readmission variable, considering a series of tests as a better indicator, these columns were deemed less influential. With around 50% null values in Medical Specialty and the acknowledgment that nulls cannot be randomly imputed for this feature, we concluded that these columns lack substantial information to contribute effectively to predicting the target variable. Hence, they were omitted from further analysis due to their limited usefulness in predicting the target variable, stemming from both unavailable information and domain knowledge.

Regarding the 'Race' column, we detected a presence of 2.23% null values. Although imputing these missing values using the mode is a possible solution, considering the nature of medical data, this approach might not be the most suitable. There is a concern that employing mode imputation might lead to potential misclassification if this column significantly influences the target variable. Hence, instead of utilizing the mode, a more prudent strategy would involve substituting the null values with a categorical designation of 'Unknown'. This method aims to maintain data integrity without introducing potential misclassifications associated with imputing values that could skew the analysis.

Understanding the significance of ICD-9 codes in medical diagnosis is crucial. The first three digits of diag_1, diag_2, and diag_3 categorize primary, secondary, and additional secondary diagnoses, respectively. These digits provide a general classification indicating the broad group to which the specific code belongs. For instance, "401" in the ICD-9 code 401.0 represents "Essential hypertension," offering a broad classification. Codes starting with "V" (V01-V91) focus on external causes of morbidity and mortality, describing reasons like motor vehicle accidents. Meanwhile, those starting with "E" (E000-E999) detail external causes of injury and poisoning, explaining how an injury occurred. Null values in ICD-9 codes carry various interpretations. They may denote unrecorded or unavailable data, an unknown diagnosis, inapplicability in a context, potential data entry errors, or considerations for privacy and confidentiality regarding specific health information.

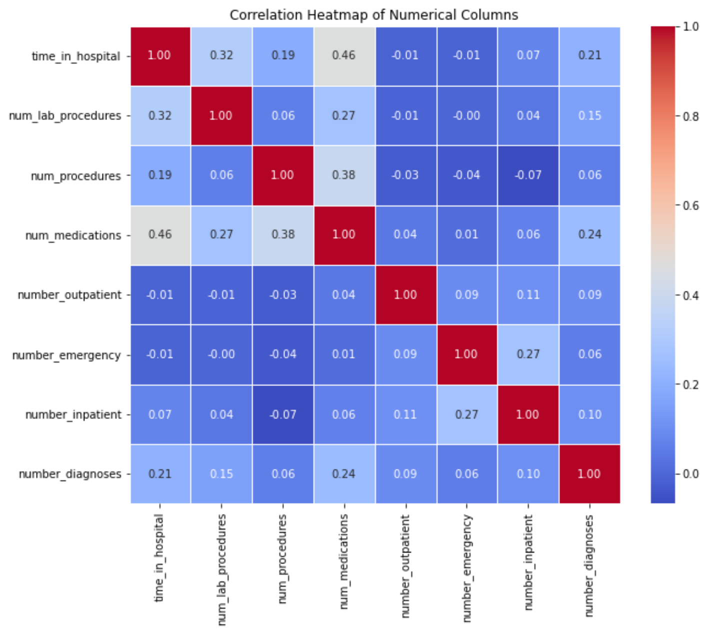
The ratio of readmitted (Target Variable) values remains consistent before and after the removal of nulls from the diagnostic columns. Upon dropping rows with nulls, approximately 98.5% of the dataset is retained. Given the minimal loss of information, we opted to proceed by eliminating records where diag_1, diag_2, or diag_3 contain null values. Consequently, our dataset was reduced to 100,244 rows and 45 columns.

We established binary categories for the target variable "readmitted." The label "No" was substituted with 0, while ">30" and "<30" were replaced with 1 to signify cases where the patient was readmitted.

We're aware that admission_type_id, discharge_disposition_id, and admission_source_id represent categorical information—each integer value corresponds to a specific category, as indicated in the data dictionary. Therefore, we converted these columns into string format.

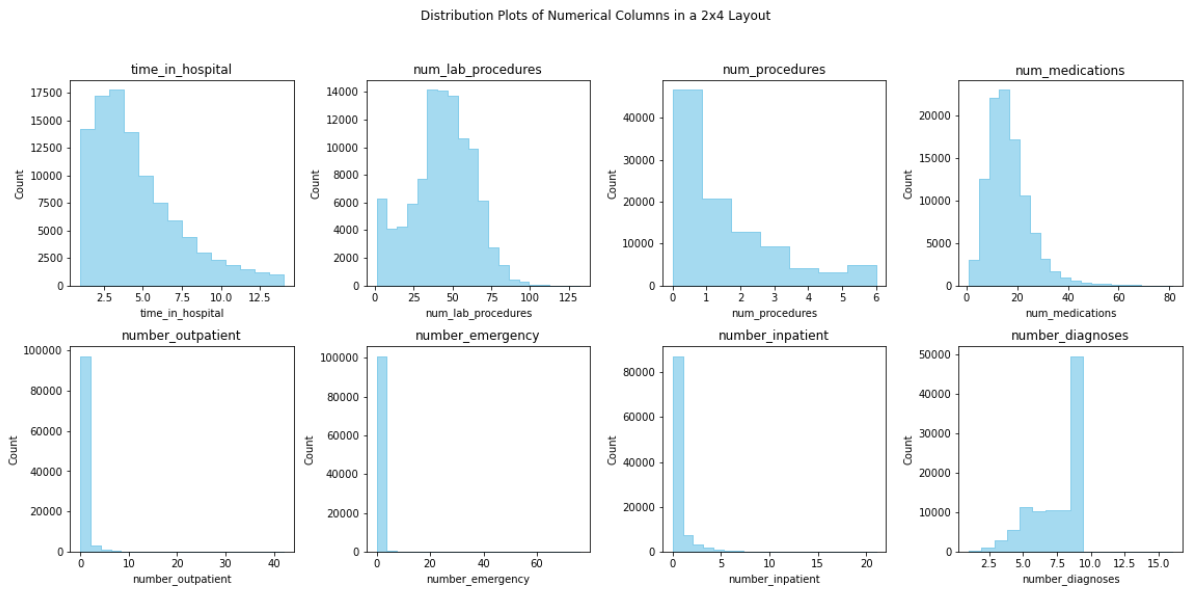
Treating Numerical Columns

There are a total of 8 numerical columns. Upon observation, these numerical columns exhibit minimal correlation among themselves, suggesting their independence and implying that they can be considered as independent variables.

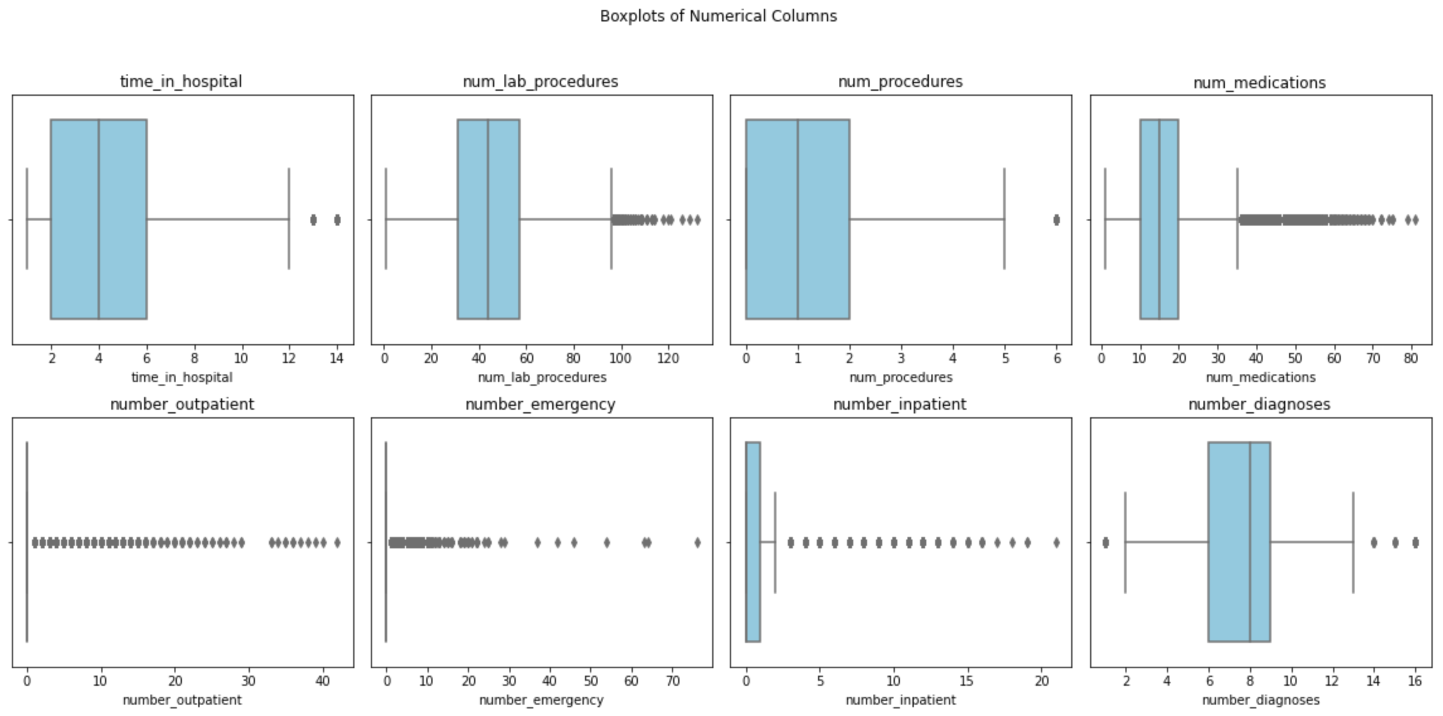


The subsequent numerical columns display particular distributions:

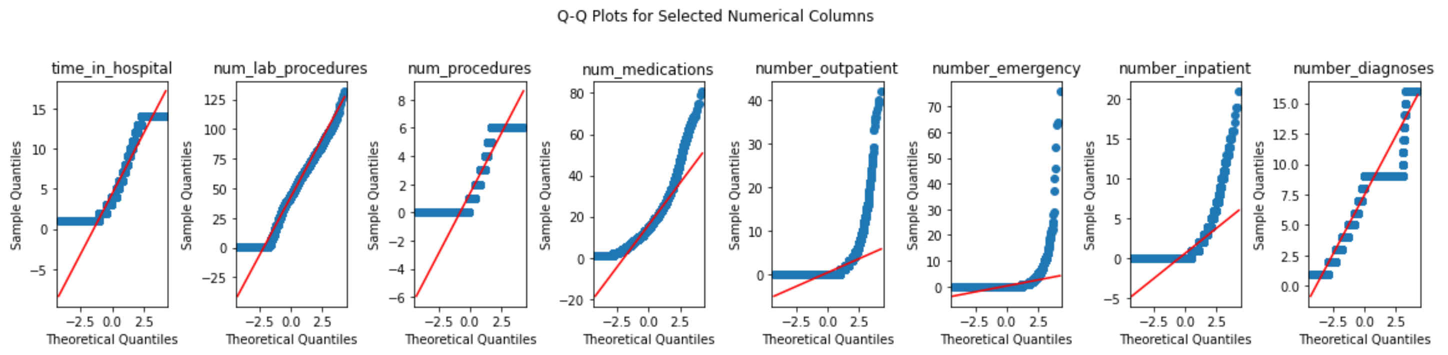
- num_lab_procedures and num_medications exhibit a normal/gaussian distribution.
- num_procedures, number_outpatient, number_emergency, number_inpatient, and time_in_hospital showcase a left-skewed distribution.
- Meanwhile, the number_diagnoses column demonstrates a right-skewed distribution.



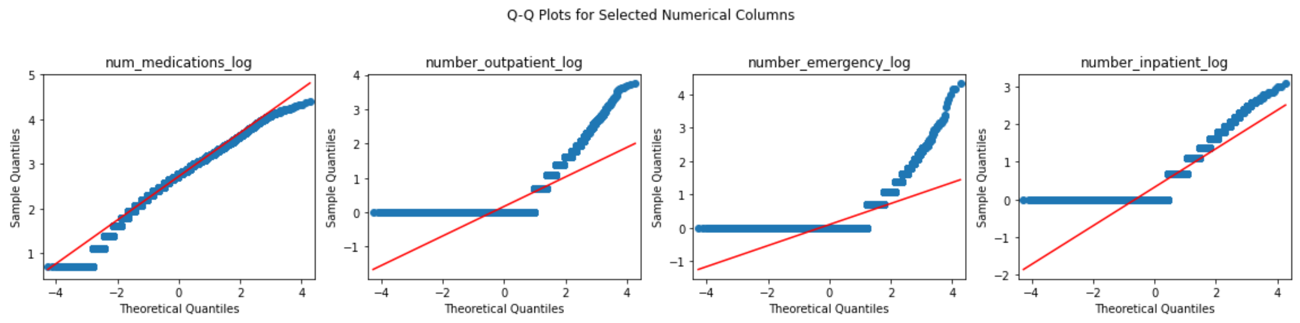
Upon examining the box plots, it's evident that num_medications, number_outpatient, number_emergency, and number_inpatient exhibit numerous outliers. Consequently, there's a necessity to explore their distribution further, particularly as the majority of outliers align with the tail of the Gaussian distribution.



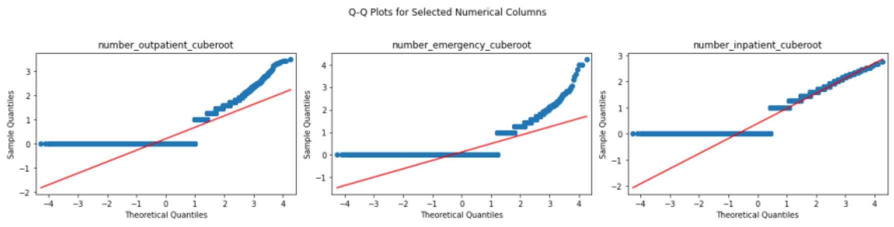
Utilizing Q-Q plots to assess distributions and determine Gaussian conformity, we find that certain columns predominantly align with the theoretical line, signifying a Gaussian distribution. These columns include time_in_hospital, num_lab_procedures, num_procedures, and number_diagnoses (although this might require enhancement). However, num_medications, number_outpatient, number_emergency, and number_inpatient exhibit a notable presence of outliers, indicating a deviation from a Gaussian distribution. Transformations are warranted to align these columns with a Gaussian distribution pattern.



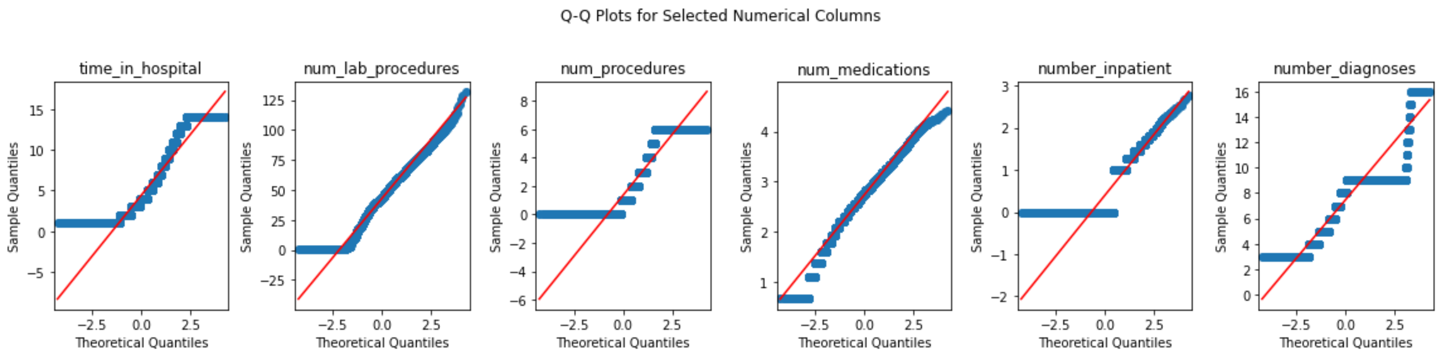
Introducing a slight constant addition aims to mitigate potential issues associated with zero values within the columns 'num_medications_log', 'number_outpatient_log', 'number_emergency_log', and 'number_inpatient_log'. Subsequently plotting Q-Q plots, it's observed that applying a logarithmic transformation successfully shifted num_medications towards a Gaussian distribution. However, while the others are progressing towards Gaussian, they still exhibit noticeable deviations.



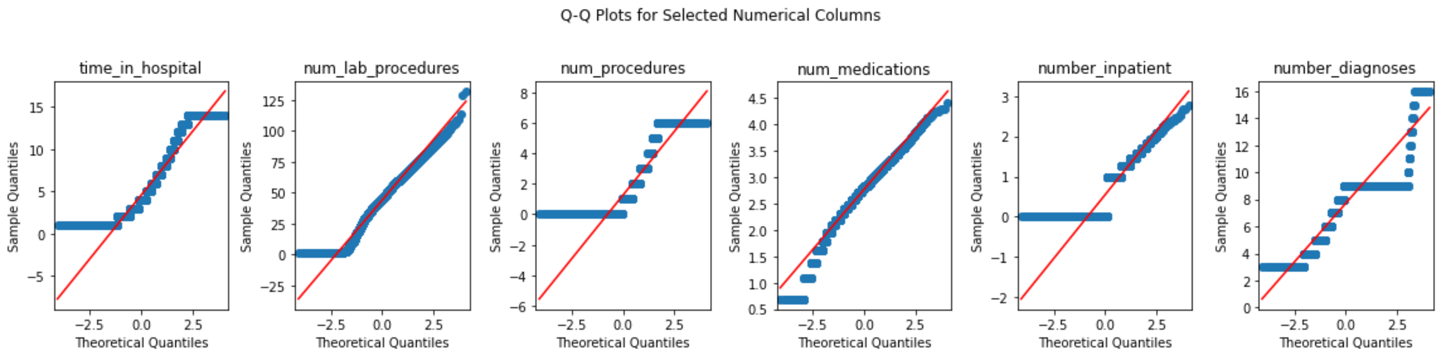
Implementing the cuberoot transformation on the remaining three columns—'number_outpatient_cuberoot', 'number_emergency_cuberoot', and 'number_inpatient_cuberoot'—resulted in successful normalization for number_inpatient. Despite attempting various transformation methods, number_outpatient and number_emergency couldn't be converted into a Gaussian distribution. Therefore, these columns might not be suitable for Gaussian-based algorithms like Naive Bayes and Logistic Regression. However, they can still be utilized with Neural Networks since such algorithms do not rely on distribution assumptions.



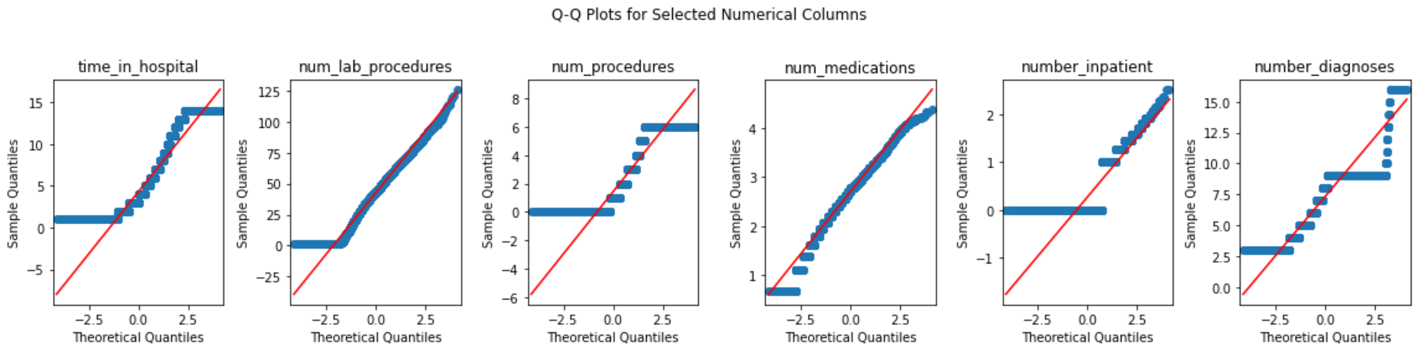
Substituted the original columns with their respective transformed versions and generated Q-Q plots for the entire dataset.



Generated a Q-Q plot specifically for instances where the target class, denoted as Readmitted equals 1, indicating cases where the patient is readmitted.



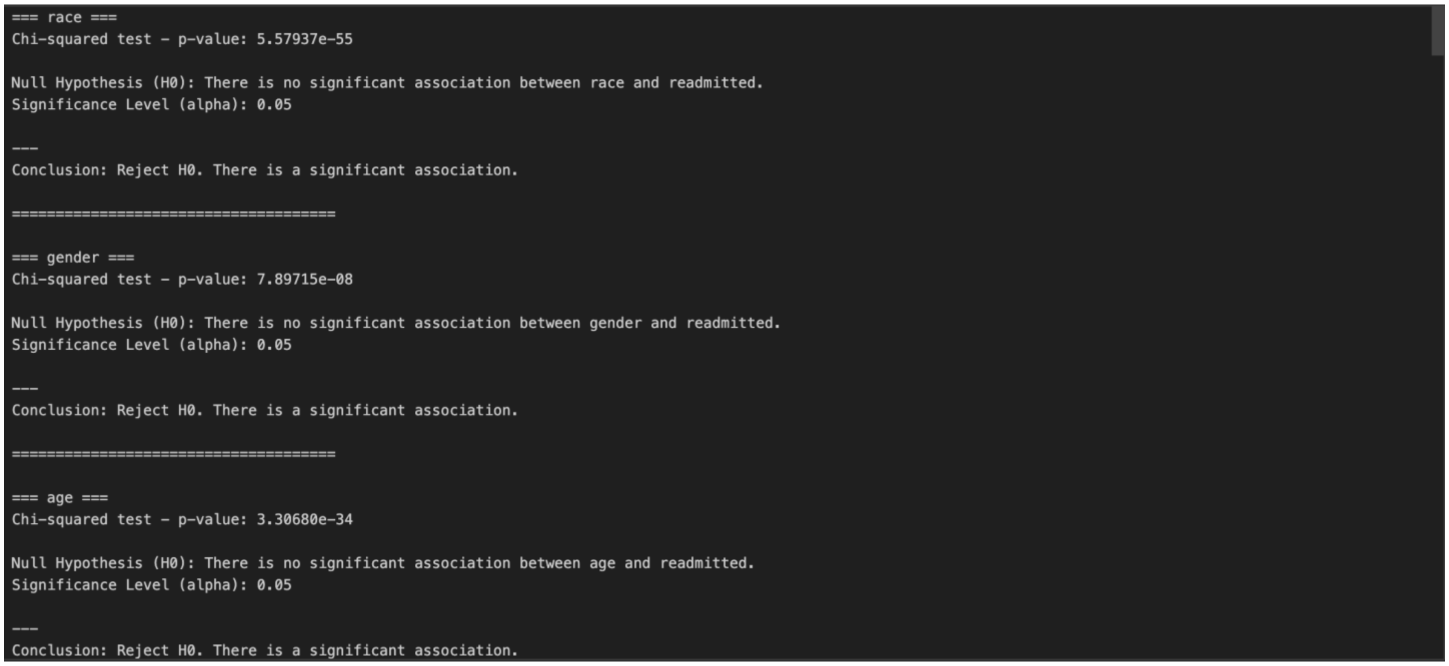
Generated a Q-Q plot specifically for instances where the target class, represented as Readmitted equals 0, indicating cases where the patient is not readmitted.



As evident from the preceding graphs, the numerical columns exhibit a Gaussian distribution for their respective target classes.

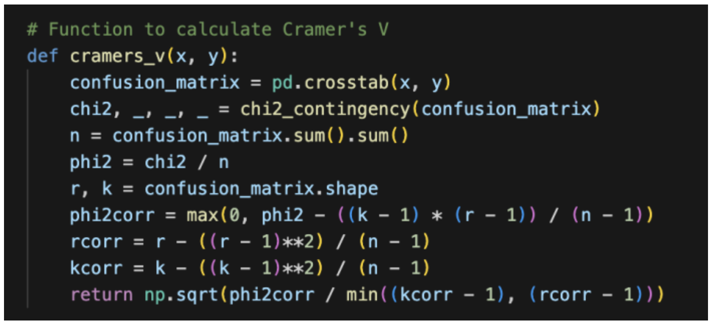
Categorical Columns

We initially conducted a chi-squared test to assess the relationship between the 36 categorical columns and the target variable. The null hypothesis assumed no significant association between a variable and the target variable, with a chosen significance level of 0.05. The image below represents a snapshot of our output; however, it does not display all 36 columns due to space limitations.



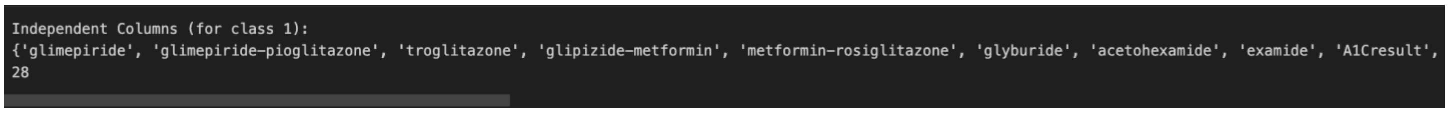
```
=== race ===
Chi-squared test - p-value: 5.57937e-55

Null Hypothesis (H0): There is no significant association between race and readmitted.
Significance Level (alpha): 0.05

---
Conclusion: Reject H0. There is a significant association.

=====================================

=== gender ===
Chi-squared test - p-value: 7.89715e-08

Null Hypothesis (H0): There is no significant association between gender and readmitted.
Significance Level (alpha): 0.05

---
Conclusion: Reject H0. There is a significant association.

=====================================

=== age ===
Chi-squared test - p-value: 3.30680e-34

Null Hypothesis (H0): There is no significant association between age and readmitted.
Significance Level (alpha): 0.05

---
Conclusion: Reject H0. There is a significant association.
```

Following the chi-square test, we utilized Cramer's V statistic, which measures the association between two categorical variables and is derived from the chi-square statistic. In sizable datasets, the chi-square test can yield statistical significance even with a weak association between variables due to the large sample size, which applies to our dataset. In such scenarios, Cramer's V offers a more standardized measure of association by considering the contingency table's size. To compute the Cramer's V statistic, we developed a function and set the threshold at 0.1. The image below illustrates the function for calculating Cramer's V statistic.



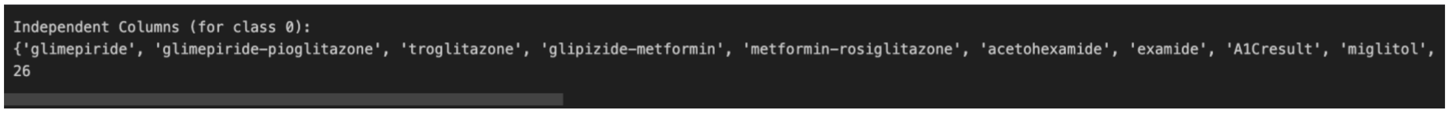
```python
# Function to calculate Cramer's V
def cramers_v(x, y):
    confusion_matrix = pd.crosstab(x, y)
    chi2, _, _, _ = chi2_contingency(confusion_matrix)
    n = confusion_matrix.sum().sum()
    phi2 = chi2 / n
    r, k = confusion_matrix.shape
    phi2corr = max(0, phi2 - ((k - 1) * (r - 1)) / (n - 1))
    rcorr = r - ((r - 1)**2) / (n - 1)
    kcorr = k - ((k - 1)**2) / (n - 1)
    return np.sqrt(phi2corr / min((kcorr - 1), (rcorr - 1)))
```

Following the application of Cramer's V statistic with the target variable set as 1 across all columns, we identified 28 columns that exhibited independence. The provided snapshot in the code output displays the results; however, it doesn't encompass all column names due to space limitations within the report.



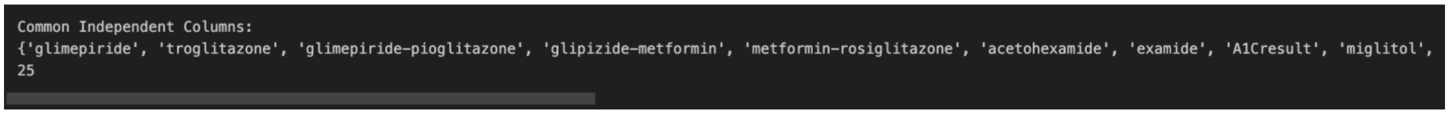
```
Independent Columns (for class 1):
{'glimepiride', 'glimepiride-pioglitazone', 'troglitazone', 'glipizide-metformin', 'metformin-rosiglitazone', 'glyburide', 'acetohexamide', 'examide', 'A1Cresult',
28
```

Following the implementation of Cramer's V statistic with the target variable set as 0 across all columns, we identified 26 columns that displayed independence. The presented snapshot in the code output illustrates the results, although it doesn't encompass all column names due to space limitations within the report.



```
Independent Columns (for class 0):
{'glimepiride', 'glimepiride-pioglitazone', 'troglitazone', 'glipizide-metformin', 'metformin-rosiglitazone', 'acetohexamide', 'examide', 'A1Cresult', 'miglitol',
26
```

We selectively retained columns that appeared as independent in both Cramer's V tests for class 1 and class 0. Consequently, we identified 25 columns that displayed independence in both scenarios. The provided snapshot in the code output demonstrates these results; however, not all column names are included due to space limitations within the report.



```
Common Independent Columns:
{'glimepiride', 'troglitazone', 'glimepiride-pioglitazone', 'glipizide-metformin', 'metformin-rosiglitazone', 'acetohexamide', 'examide', 'A1Cresult', 'miglitol',
25
```

Following this step, we proceeded to apply one-hot encoding to all categorical variables.

ML Algorithms Used

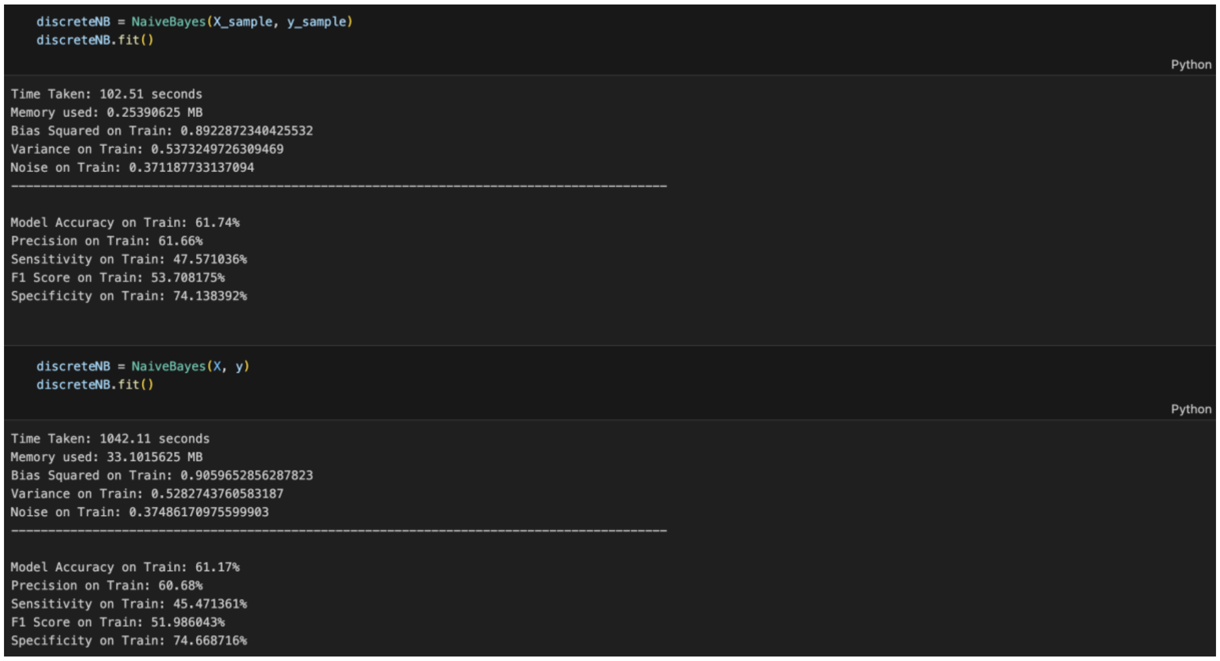
Naïve Bayes

We selected the Naive Bayes model due to the substantial size of our dataset, coupled with the prevalence of numerous categorical columns within it. Naive Bayes is particularly well-suited for such data compositions, where it tends to perform optimally. Its strength lies in its ability to handle categorical features efficiently, making it a favorable choice given our dataset's characteristics and structure.

We're considering the 25 categorical independent columns obtained from the Cramer's V test to check for independence between categorical variable pairs. As for the numerical columns, we're selecting those that exhibit a Gaussian distribution and columns that are gaussian post log and cube root transformations given the target class.

Evaluation Metrics

These metrics represent the values obtained from the subset of training data (10% of the entire dataset) and the complete main training dataset.



```python
    discreteNB = NaiveBayes(X_sample, y_sample)
    discreteNB.fit()
```

Python

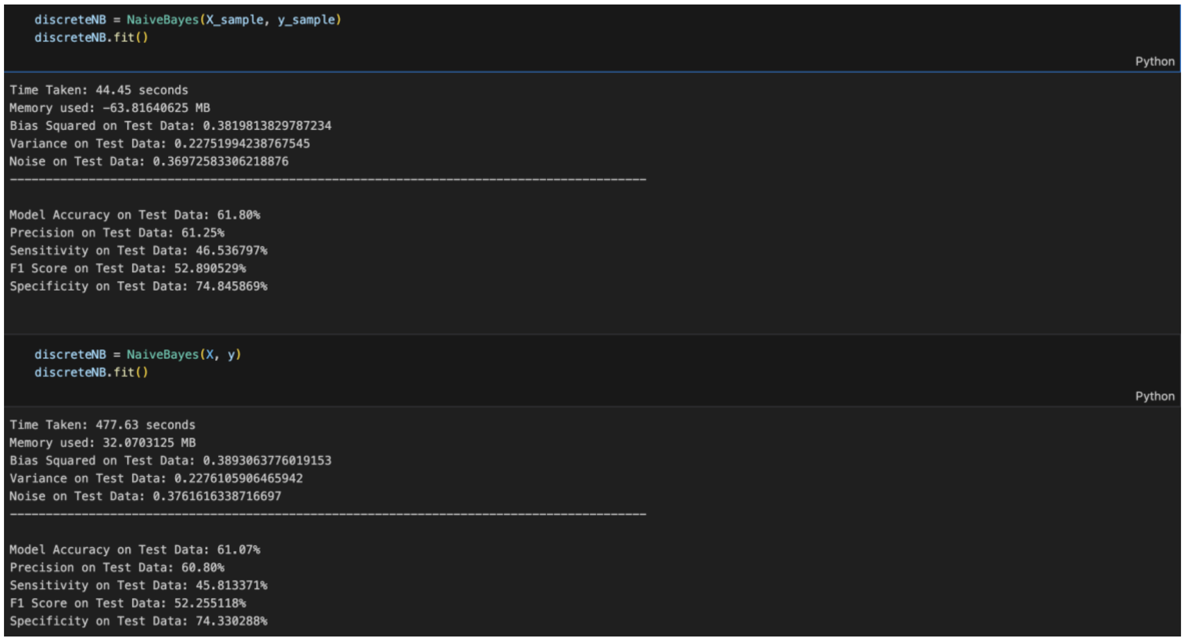
```
Time Taken: 102.51 seconds
Memory used: 0.25390625 MB
Bias Squared on Train: 0.8922872340425532
Variance on Train: 0.5373249726309469
Noise on Train: 0.371187733137094
------------------------------------------------------------------------------------------

Model Accuracy on Train: 61.74%
Precision on Train: 61.66%
Sensitivity on Train: 47.571036%
F1 Score on Train: 53.708175%
Specificity on Train: 74.138392%
```

```python
    discreteNB = NaiveBayes(X, y)
    discreteNB.fit()
```

Python

```
Time Taken: 1042.11 seconds
Memory used: 33.1015625 MB
Bias Squared on Train: 0.9059652856287823
Variance on Train: 0.5282743760583187
Noise on Train: 0.37486170975599903
------------------------------------------------------------------------------------------

Model Accuracy on Train: 61.17%
Precision on Train: 60.68%
Sensitivity on Train: 45.471361%
F1 Score on Train: 51.986043%
Specificity on Train: 74.668716%
```

Here are the metrics obtained from the subset of testing data (10% of the entire dataset) and the primary testing dataset.



```python
    discreteNB = NaiveBayes(X_sample, y_sample)
    discreteNB.fit()
```

Python

```
Time Taken: 44.45 seconds
Memory used: -63.81640625 MB
Bias Squared on Test Data: 0.3819813829787234
Variance on Test Data: 0.22751994238767545
Noise on Test Data: 0.36972583306218876
------------------------------------------------------------------------------------------

Model Accuracy on Test Data: 61.80%
Precision on Test Data: 61.25%
Sensitivity on Test Data: 46.536797%
F1 Score on Test Data: 52.890529%
Specificity on Test Data: 74.845869%
```

```python
    discreteNB = NaiveBayes(X, y)
    discreteNB.fit()
```

Python

```
Time Taken: 477.63 seconds
Memory used: 32.0703125 MB
Bias Squared on Test Data: 0.3893063776019153
Variance on Test Data: 0.2276105906465942
Noise on Test Data: 0.3761616338716697
------------------------------------------------------------------------------------------

Model Accuracy on Test Data: 61.07%
Precision on Test Data: 60.80%
Sensitivity on Test Data: 45.813371%
F1 Score on Test Data: 52.255118%
Specificity on Test Data: 74.330288%
```

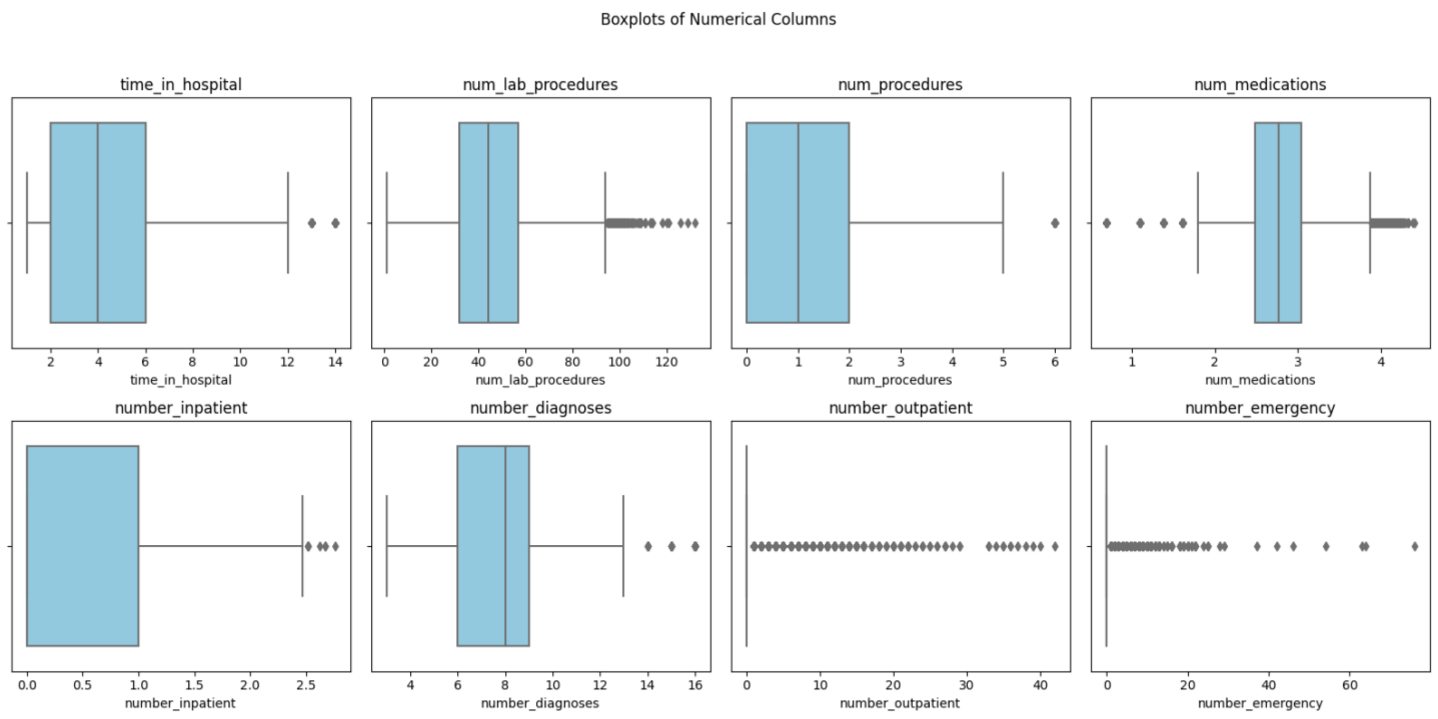
Bias squared measures how predictions differ from the actual values squared, while variance gauges how much predictions deviate from their average.

The zero-one loss (classification error) breakdown is: Expected Loss = Bias^2 + Variance + Irreducible Error (Noise), where noise represents inherent unpredictability in the data.

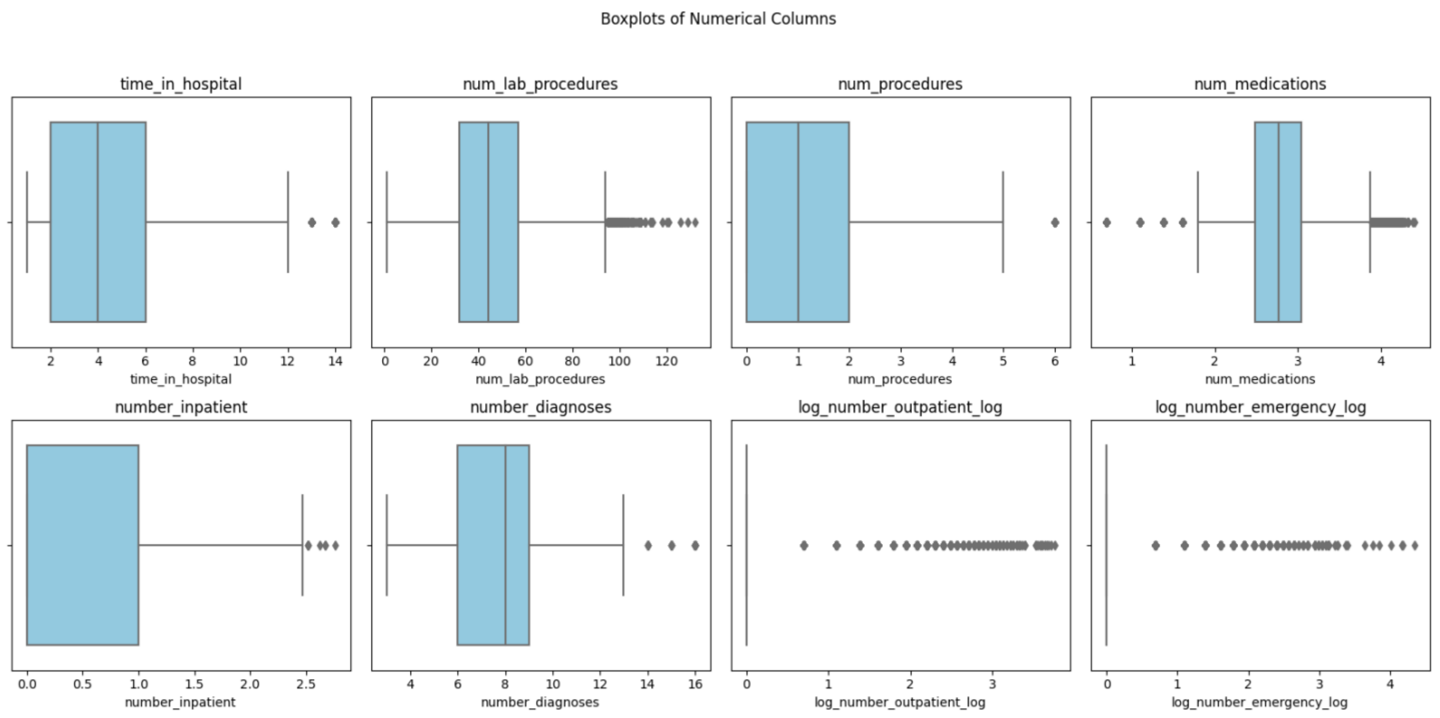
For Naive Bayes, the expected loss is calculated as 0.905 (Bias^2) + 0.227 (Variance) + 0.371 (Noise) = 1.503.

Preparing Data for Logistic Regression and Neural Network

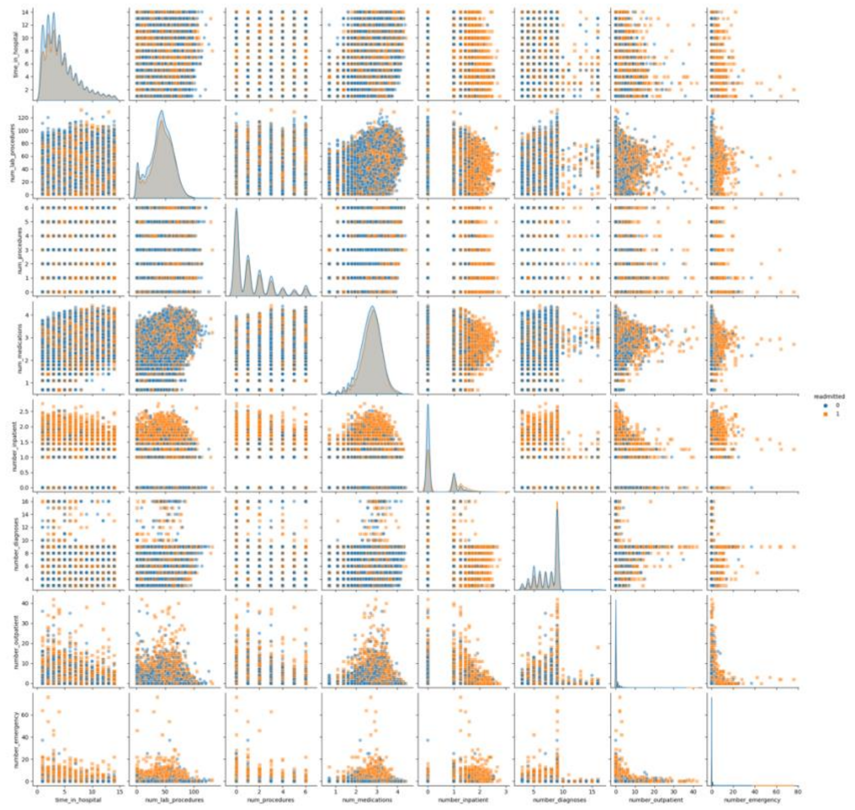
The box plots below reveal that both 'number_outpatient' and 'number_emergency' exhibit a high count of outliers. Logistic regression relies on the assumption of a linear relationship between independent variables and the log-odds of the dependent variable. The presence of numerous outliers in these variables can distort this presumed linear relationship, potentially causing several issues in the analysis.



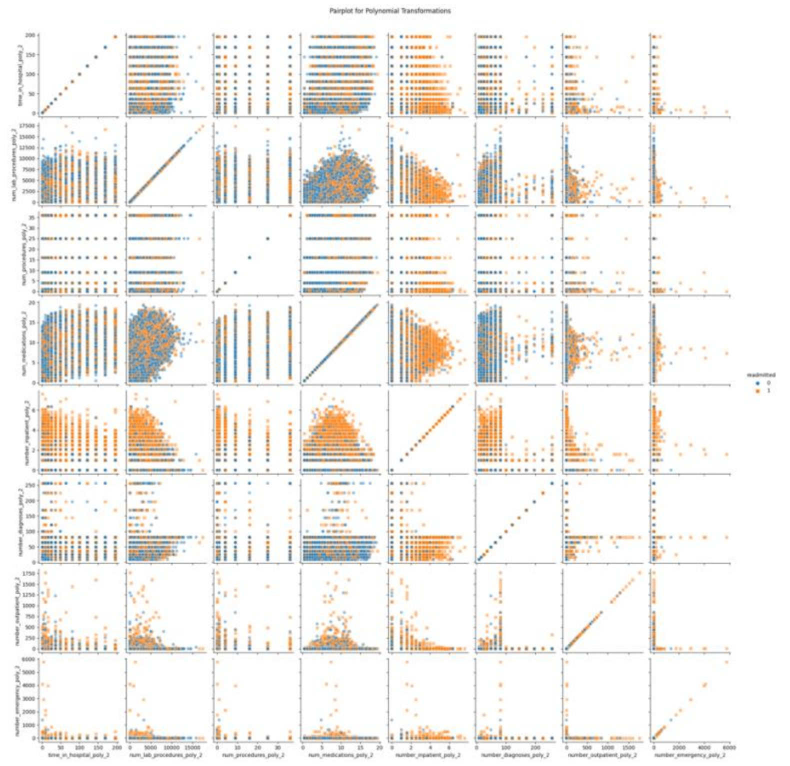
The plot below demonstrates the application of a log transformation to address outliers in 'number_outpatient' and 'number_emergency.' Unfortunately, it appears that this transformation isn't effective in resolving the issue. The transformation doesn't seem to provide the anticipated improvement in mitigating the impact of outliers in the data.



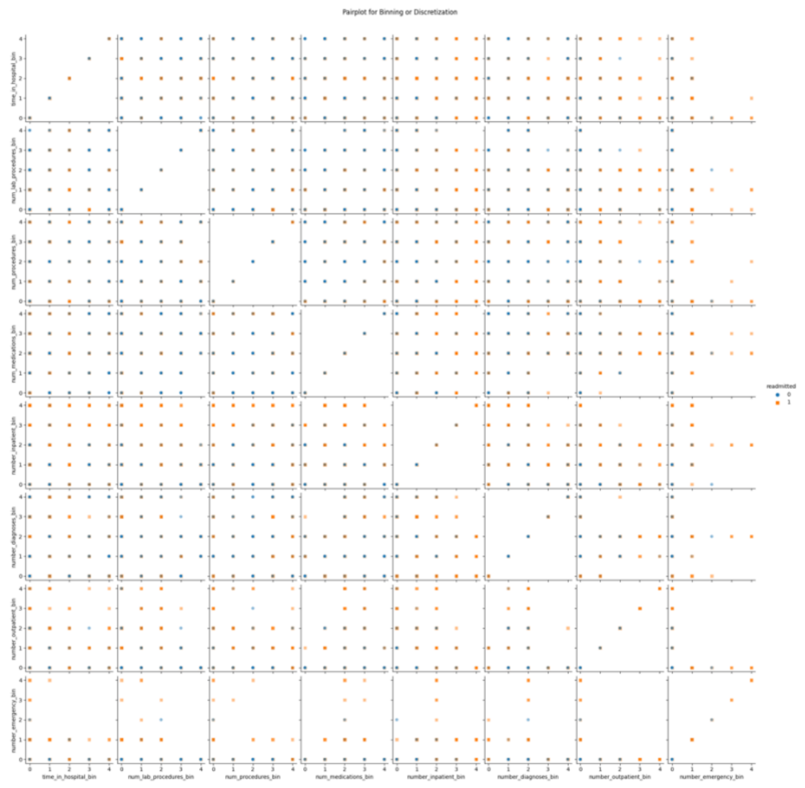
Assessing the linear separability between dependent and independent columns, it's evident from the pairplots below that they are not distinctly separable concerning the target variable. However, in the case of 'number_inpatient' and 'number_emergency,' there is a discernible degree of class distinction observed. To explore potential linear separability, transforming variables into higher dimensions might be considered as a means to enhance the separation between classes.



The pairplot depicted below illustrates that even with higher-degree polynomial transformations, class separation is observed solely in the 'number_inpatient' numerical column. Hence, considering only this specific numerical column might be sufficient for our analysis or modeling purposes.

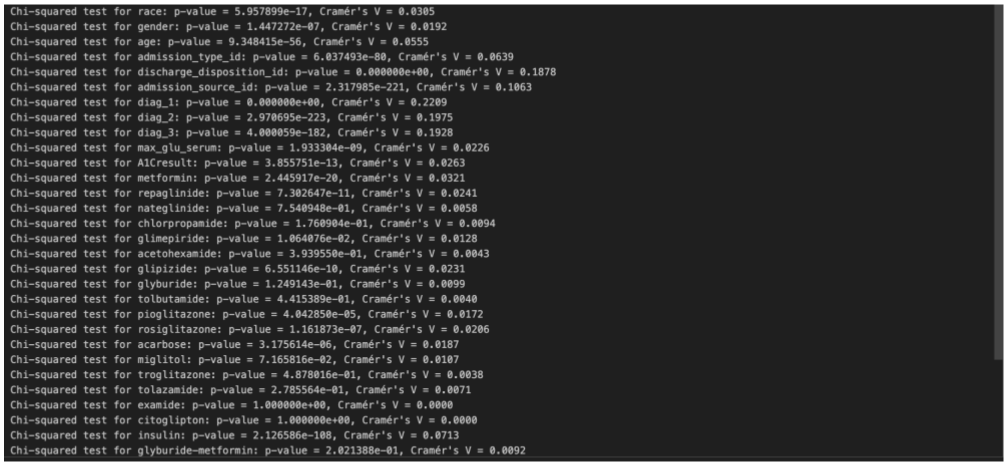


The pairplot below demonstrates the application of binning or discretization to specific numerical columns, using a set number of bins, which is defined as 'num_bins = 5'.



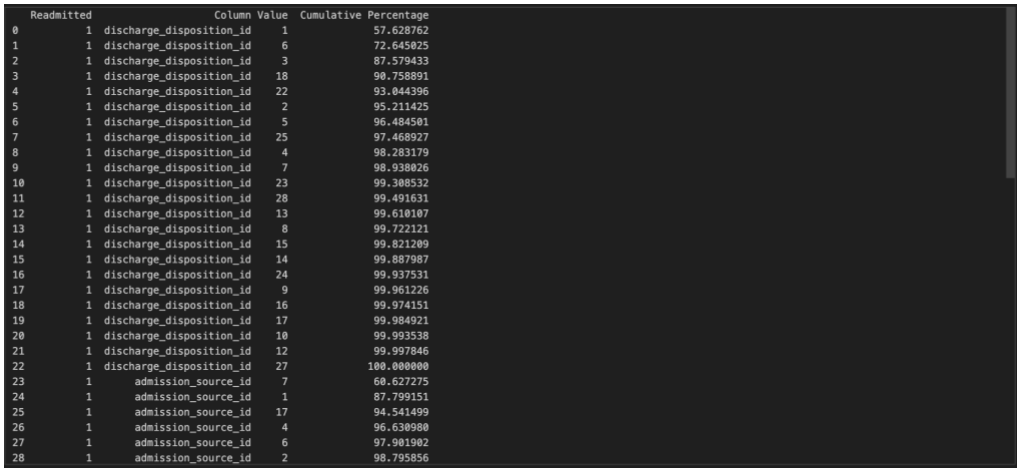
After trying different transformation techniques, we see that only number_inpatient is linearly related to readmitted. Though number_emergency shows some separation, it has many outliers. So we can keep number_inpatient and drop the rest.

We performed chi-squared and Cramer's V tests for all categorical columns concerning the target column "Readmitted." Only the columns with Cramer's V values exceeding 0.1 are retained, while the others are dropped. The code snippet below displays a partial output of the results, as the complete values are not presented in the report due to space limitations.



```
Chi-squared test for race: p-value = 5.957899e-17, Cramér's V = 0.0305
Chi-squared test for gender: p-value = 1.447272e-07, Cramér's V = 0.0192
Chi-squared test for age: p-value = 9.348415e-56, Cramér's V = 0.0555
Chi-squared test for admission_type_id: p-value = 6.037493e-80, Cramér's V = 0.0639
Chi-squared test for discharge_disposition_id: p-value = 0.000000e+00, Cramér's V = 0.1878
Chi-squared test for admission_source_id: p-value = 2.317985e-221, Cramér's V = 0.1063
Chi-squared test for diag_1: p-value = 0.000000e+00, Cramér's V = 0.2209
Chi-squared test for diag_2: p-value = 2.970695e-223, Cramér's V = 0.1975
Chi-squared test for diag_3: p-value = 4.000059e-182, Cramér's V = 0.1928
Chi-squared test for max_glu_serum: p-value = 1.933304e-09, Cramér's V = 0.0226
Chi-squared test for A1Cresult: p-value = 3.855751e-13, Cramér's V = 0.0263
Chi-squared test for metformin: p-value = 2.445917e-20, Cramér's V = 0.0321
Chi-squared test for repaglinide: p-value = 7.302647e-11, Cramér's V = 0.0241
Chi-squared test for nateglinide: p-value = 7.540948e-01, Cramér's V = 0.0058
Chi-squared test for chlorpropamide: p-value = 1.760904e-01, Cramér's V = 0.0094
Chi-squared test for glimepiride: p-value = 1.064076e-02, Cramér's V = 0.0128
Chi-squared test for acetohexamide: p-value = 3.939550e-01, Cramér's V = 0.0043
Chi-squared test for glipizide: p-value = 6.551146e-10, Cramér's V = 0.0231
Chi-squared test for glyburide: p-value = 1.249143e-01, Cramér's V = 0.0099
Chi-squared test for tolbutamide: p-value = 4.415389e-01, Cramér's V = 0.0040
Chi-squared test for pioglitazone: p-value = 4.042850e-05, Cramér's V = 0.0172
Chi-squared test for rosiglitazone: p-value = 1.161873e-07, Cramér's V = 0.0206
Chi-squared test for acarbose: p-value = 3.175614e-06, Cramér's V = 0.0187
Chi-squared test for miglitol: p-value = 7.165816e-02, Cramér's V = 0.0107
Chi-squared test for troglitazone: p-value = 4.878016e-01, Cramér's V = 0.0038
Chi-squared test for tolazamide: p-value = 2.785564e-01, Cramér's V = 0.0071
Chi-squared test for examide: p-value = 1.000000e+00, Cramér's V = 0.0000
Chi-squared test for citoglipton: p-value = 1.000000e+00, Cramér's V = 0.0000
Chi-squared test for insulin: p-value = 2.126586e-108, Cramér's V = 0.0713
Chi-squared test for glyburide-metformin: p-value = 2.021388e-01, Cramér's V = 0.0092
```

We conducted one-hot encoding on the retained categorical columns and noticed an outcome of over 2000 columns. Therefore, we opted to select the top 'n' categories that encompass more than 99% of the records, with a criterion of 75% for the 'diagnosis' column. Any remaining minority classes were grouped as 'Others,' effectively reducing the total variables to fewer than 200. The code snippet below exhibits a partial output due to space constraints in the report, showcasing only some of the values.



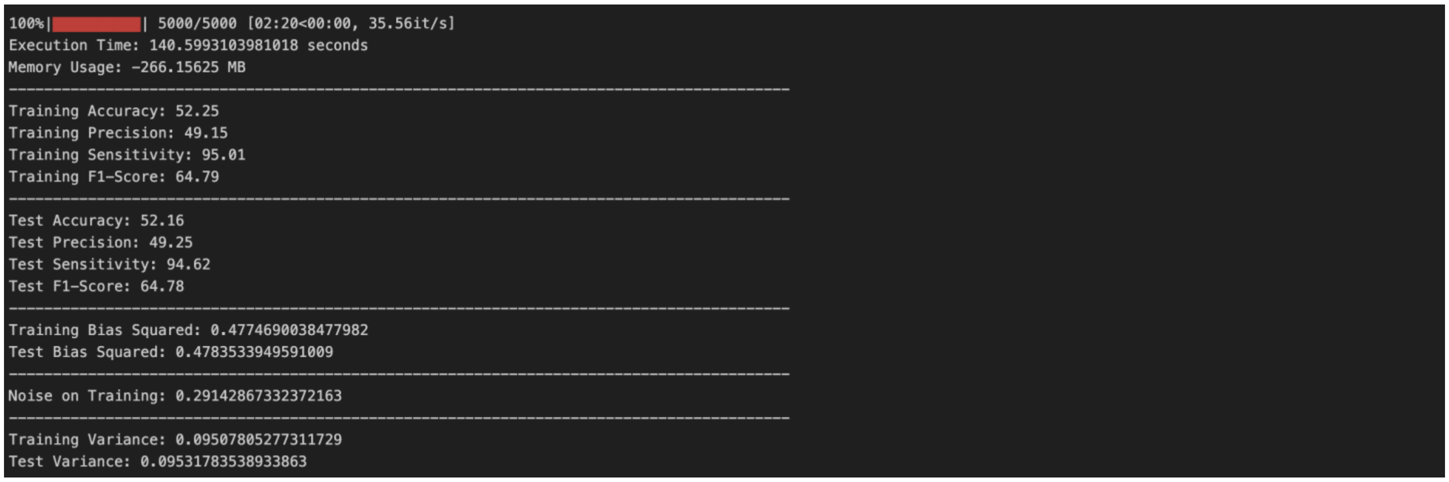
```
   Readmitted                  Column Value  Cumulative Percentage
0           1  discharge_disposition_id     1             57.628762
1           1  discharge_disposition_id     6             72.645025
2           1  discharge_disposition_id     3             87.579433
3           1  discharge_disposition_id    18             90.758891
4           1  discharge_disposition_id    22             93.044396
5           1  discharge_disposition_id     2             95.211425
6           1  discharge_disposition_id     5             96.484501
7           1  discharge_disposition_id    25             97.468927
8           1  discharge_disposition_id     4             98.283179
9           1  discharge_disposition_id     7             98.938026
10          1  discharge_disposition_id    23             99.308532
11          1  discharge_disposition_id    28             99.491631
12          1  discharge_disposition_id    13             99.610107
13          1  discharge_disposition_id     8             99.722121
14          1  discharge_disposition_id    15             99.821209
15          1  discharge_disposition_id    14             99.887987
16          1  discharge_disposition_id    24             99.937531
17          1  discharge_disposition_id     9             99.961226
18          1  discharge_disposition_id    16             99.974151
19          1  discharge_disposition_id    17             99.984921
20          1  discharge_disposition_id    10             99.993538
21          1  discharge_disposition_id    12             99.997846
22          1  discharge_disposition_id    27            100.000000
23          1       admission_source_id     7             60.627275
24          1       admission_source_id     1             87.799151
25          1       admission_source_id    17             94.541499
26          1       admission_source_id     4             96.630980
27          1       admission_source_id     6             97.901902
28          1       admission_source_id     2             98.795856
```

Logistic Regression

We opted for Logistic regression as our chosen model due to its compatibility and effectiveness with large-scale datasets. Logistic regression is particularly advantageous in handling extensive datasets efficiently, as it can manage a substantial volume of observations and variables without compromising its performance. Its computational efficiency and ability to scale well with large amounts of data make it a suitable choice for our analysis or modeling task.

Evaluation Metrics

The resulting expected loss is calculated as 0.477 for bias squared, 0.095 for variance, and 0.291 for irreducible error, totaling 0.863. The provided code snippet displays the evaluation metrics output for our Logistic Regression model.



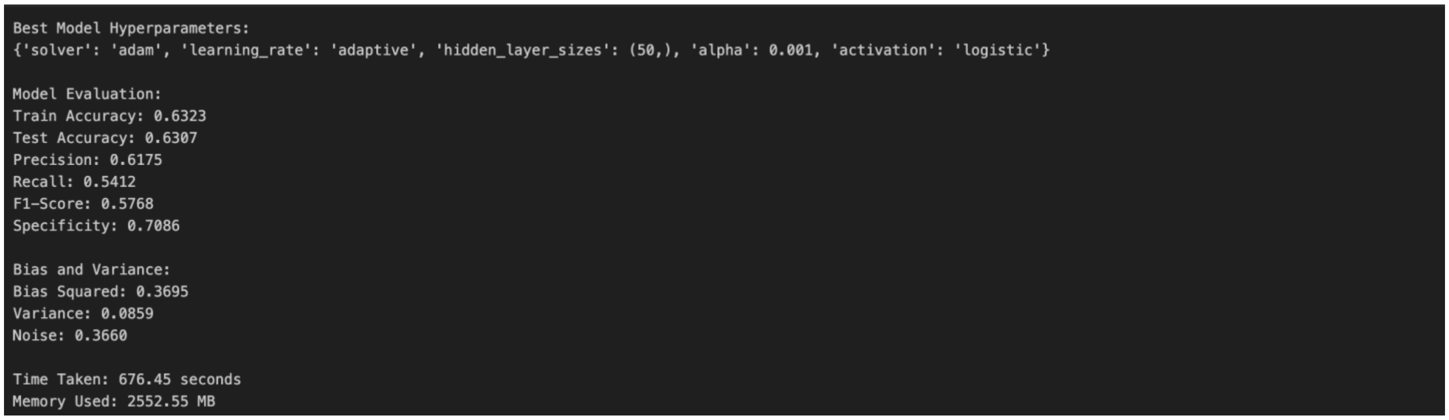
```
100%|██████████| 5000/5000 [02:20<00:00, 35.56it/s]
Execution Time: 140.5993103981018 seconds
Memory Usage: -266.15625 MB
------------------------------------------------------------------------------------------
Training Accuracy: 52.25
Training Precision: 49.15
Training Sensitivity: 95.01
Training F1-Score: 64.79
------------------------------------------------------------------------------------------
Test Accuracy: 52.16
Test Precision: 49.25
Test Sensitivity: 94.62
Test F1-Score: 64.78
------------------------------------------------------------------------------------------
Training Bias Squared: 0.4774690038477982
Test Bias Squared: 0.4783533949591009
------------------------------------------------------------------------------------------
Noise on Training: 0.29142867332372163
------------------------------------------------------------------------------------------
Training Variance: 0.09507805277311729
Test Variance: 0.09531783538933863
```

Neural Networks

Neural networks excel in capturing complex patterns due to their capacity to handle non-linear relationships in data, eliminating the need for extensive manual feature engineering. They adapt well to various data types, scale effectively with large datasets, and offer customization for specific complexities. Continual refinement through techniques like deep learning ensures iterative improvements in classification accuracy.

Evaluation Metrics

In the case of Neural Networks, the resulting expected loss sums up to 0.821, computed from bias squared (0.369), variance (0.086), and irreducible error (0.366). The best model hyperparameters identified are: {'solver': 'adam', 'learning_rate': 'adaptive', 'hidden_layer_sizes': (50,), 'alpha': 0.001, 'activation': 'logistic'}. A snapshot below presents the evaluation metrics



```
Best Model Hyperparameters:
{'solver': 'adam', 'learning_rate': 'adaptive', 'hidden_layer_sizes': (50,), 'alpha': 0.001, 'activation': 'logistic'}

Model Evaluation:
Train Accuracy: 0.6323
Test Accuracy: 0.6307
Precision: 0.6175
Recall: 0.5412
F1-Score: 0.5768
Specificity: 0.7086

Bias and Variance:
Bias Squared: 0.3695
Variance: 0.0859
Noise: 0.3660

Time Taken: 676.45 seconds
Memory Used: 2552.55 MB
```

Conclusion

Based on the Expected Loss metric, the optimal model identified is Neural Networks, specified with the following hyperparameters: {'solver': 'adam', 'learning_rate': 'adaptive', 'hidden_layer_sizes': (50,), 'alpha': 0.001, 'activation': 'logistic'}.

When comparing accuracy, sensitivity, precision, and F1 score across three models, the overall performance indicates Neural Networks as the superior performer. Although logistic regression achieves the highest sensitivity, an essential metric for this specific use case, all other measurement metrics exhibit subpar performance.

Considering the baseline model, where all values in the target variable are predicted as 0 (the majority class), the accuracy obtained is 53.69%. Notably, logistic regression displays lower accuracy (52.16%) compared to the baseline model. Conversely, Naive Bayes (61.07%) and Neural Networks (63.07%) showcase improved accuracies compared to the baseline model.