# Big Data & Predictive Analytics Official Documentation

Jignesh Sisodiya, Owain Hughes, Jayden Sachdev

University Of Leicester

Data Cleaning and Preprocessing (Jignesh Sisodiya) ..... 2
Data Overview and Missing Value Identification ..... 2
Handling Missing Values ..... 2
Handling Outliers Using the IQR Method ..... 2
Feature normalisation using StandardScaler ..... 2
Remaining column and final dataset shape ..... 3
Data Cleansing Conclusion ..... 3
Data Visualization Using Matplotlib and Seaborn (Owain Hughes) ..... 3
Unique Classes Vs Target Variable ..... 3
ICU Cases Vs. Age ..... 4
ICU Cases Vs. Patient Classification ..... 4
Scatter Matrix & Correlation Matrices ..... 4
Additional Visualization ..... 5
Conclusion of Data Visualisation ..... 6
Basic Logistical Regression Model (Jayden Sachdev) ..... 6
Introduction and Basic Model ..... 6
Limitations and Future Improvements ..... 6
Advanced Predictive Modelling and Clustering Analysis using PySpark (Jayden Sachdev) ..... 7
Introduction – Clustering & Improved Model ..... 7
Feature Selection and Justification ..... 7
Advanced Data Cleansing ..... 7
K-Means Clustering Analysis ..... 7
Initial Model Development (Unbalanced Data) ..... 8
Handling Class Imbalance ..... 8
Improved Model and Evaluation ..... 8
Interpretation and Insights ..... 8
Technical Justification for using PySpark ..... 8
Justification for Random Forest Model Choice ..... 9
Limitations and Future Improvements ..... 9
Model Interpretability ..... 9
Static Dataset ..... 9
Conclusion ..... 9
Appendix ..... 10
Appendix A: Data Cleaning and preprocessing ..... 10
Appendix B: Data Visualisation Outputs ..... 12
Appendix C: Advanced Predictive Modelling and Clustering Analysis ..... 19
Appendix D: Github Repository ..... 20

# Data Cleaning and Preprocessing (Jignesh Sisodiya)

## Data Overview and Missing Value Identification

To begin, the original dataset was loaded, and its dimensions and structure were examined. This included displaying the first few rows of data and identifying columns with missing values. In this dataset, missing values were represented by a question mark (?), which was replaced with NaN for further processing [Figure 1.1].

## Handling Missing Values

After replacing invalid entries with NaN, rows containing missing values were dropped entirely to ensure model integrity. This process significantly reduced the dataset size but ensured completeness. All missing values were removed before any statistical imputation to prevent biases in the modelling phase [Figure 1.2].

An important exception was made for two features here: the ones regarding the date of death and pregnancy status. Regarding the former, many unknown dates were seemingly filled in with the data '9999/99/99' as opposed to null or NaN values. These were changed to all become the mean date of death instead. Additionally, the pregnancy status was marked as a *'?'* for all male patients. This led to all male patients being deleted from the dataset when we removed all patients with a *'?'* in the pregnancy columns. As a result, instead of deleting all rows with a *'?'* pregnancy status, we changed them to be 2, indicating that all males in the dataset are not pregnant.

## Handling Outliers Using the IQR Method

To detect and remove outliers, the interquartile range (IQR) method was applied to all numeric columns. This method identifies values that fall outside the acceptable range, which is calculated with the equation *(Q1 - 1.5 * IQR to Q3 + 1.5 * IQR).* Rows containing outliers were removed to ensure they don't distort model performance. This step helps refine the dataset for better accuracy in downstream analysis [Figure 1.3].

## Feature normalisation using StandardScaler

Normalization was applied to continuous numeric columns using the StandardScaler method to scale values with a mean of 0 and standard deviation of 1. The 'Age' column was the only feature in the dataset with a continuous numeric value, and as a result normalisation was applied only to the 'Age' column [Figure 1.4].

## Remaining column and final dataset shape

After completing the data cleaning and transformation process, a subset of relevant and high-quality features was retained to ensure optimal model performance and interpretability. The final dataset retained a focused set of variables that included demographic and comorbidity indicators [Figure 1.5].

The target variable, indicating whether a patient was admitted to the ICU, remained intact for supervised learning tasks. As a result of these refinements, the final dataset (189586, 20) provides a clean and structured foundation for subsequent modelling and analysis.

## Data Cleansing Conclusion

In summary, the data cleaning and transformation process allowed us to refine a complex and messy dataset into a structured and reliable foundation for analysis. By carefully removing irrelevant or problematic columns and focusing on the most meaningful features, we now have a dataset that is not only easier to work with but also better suited for building accurate models. With this cleaned version in place, we can now move forward to explore the data visually, uncovering patterns, trends, and potential insights through graphical representations that will guide our next steps in analysis.

# Data Visualization Using Matplotlib and Seaborn (Owain Hughes)

After receiving the data set post initial aggressive clean, there a handful of specification necessary visualisations to make as well as self-dictated visualisations to create to gain a better understanding of the data as well as the problem at hand. The target variable we are trying to understand better is the cases of admission into the ICU.

## Unique Classes Vs Target Variable

Since the target variable has a binary outcome (I.e., a patient is admitted to the ICU or not) a count plot was chosen to understand the data, as this graph type is useful to show the frequency of the appearance of a categorical variable [Figure 2.1].

This graph shows that most of the patients were admitted to the ICU (1 = positive, 2 = negative). It is important to note that since we are trying to find the risk of a patient being admitted to the ICU, that data is more likely to be scrutinized than the data of the non-admitted patients, but it is important to note that this bias exists in the dataset.

## ICU Cases Vs. Age

The next step was to plot the ages of the patients admitted to the ICU. A good way of showing demographical data would be through a histogram, as different age ranges can be grouped to show the value of each age group [Figure 2.2].

The graph shows a curve of normal distribution indicating that the data is not skewed towards a single age group. Here the histograms show the positive and negative ICU cases against one another, with yellows indicating ICU admission cases and purple non-ICU admission cases. This graph shows there the age of a patient had little to no correlation with the chances of being admitted to the ICU. This data does go against potential pre-existing, real-world conceptions about the severity of COVID being tied to age, but there are also explanations against that being COVID strain and vaccination status.

## ICU Cases Vs. Patient Classification

The next graph is to plot the ICU admissions versus the patient classification. Since the patient classification is a set, small number of categorical variables (presumably nominal), a bar chart was the best option [Figure 2.3].

This chart shows that patient classification might be a good way of showing whether a patient is likely to end up in the ICU. This is because just under 55% of patients admitted to the ICU were of type 3, which makes sense as this classification means the patient has the most severe form of Covid, just under 32% of them being type 7, ~9% of them being type 6, ~2% being type 5, and types 1, 2 and 4 making up just ~2% of all patients in the ICU. The red line shows the average number of ICU admissions per classification type, but no bar is close to the mean, showing that there is one or two outliers throwing off the average. Looking at this data, it is evident that patient classification is something to be further scrutinized. We will want to look at the distribution of ICU admissions for each classification types later, as bias in the data may be affecting this. For instance, is the number for Classification 3 high because it is a good indicator, or because it is the most populous data? Further analysis is required.

## Scatter Matrix & Correlation Matrices

The next plots were for the Scatter Matrix and Correlation Matrix. Due to the nature of the graphs, one scatter matrix was created for the numerical features, whereas a correlation matrix was created for both numerical and categorical features [Figure 2.4].

The scatter matrix was created using the ‘PairGrid’ function in the SeaBorn library. The hue used for the plot was the ICU variable. This means the ICU variable is used to plot features to different colours and is then excluded from the graphs in the plot. It is done to help give an understanding of how the features relate to the ICU variable [Figures 2.5, 2.6].

These matrices indicate a lack of linear relationships between these features and the ICU feature. The most correlated feature is intubation, but even that is not too far from 0 and as a result, not a strong indication of being put into the ICU.

## Additional Visualization

The following graphs were not mandatory – they were created to further explore the data in any way deemed appropriate. A lot of graphs were generated for multiple reasons. Some were to expand upon previous graphs and visualize the data in a different way, explore the same features in different ways, some were created due to real world expectations of the features one may deem important for the problem at hand [Figure 2.7].

This graph was generated to check the distribution of the genders in the dataset, to ensure the dataset is balanced in this way. To continue, the correlation matrix indicated that the most correlated feature was Intubation, so a count plot of the distribution of intubated patients was created to further explore this data. The graph shows that ~86% of all ICU admitted patients were also intubated, but this is a similar distribution to the overall distribution all patients against ICU admission, so this does not give us any insights that are unique [Figure 2.8].

A box plot was also generated to see if there were additional insights regarding the age feature, but like the histograms the distribution was as expected [Figure 2.9]. Additionally, seven graphs were generated to check the distribution of the ICU admissions for the different kinds of patient classifications. While seven were generated, they are all very similar and as a result we will only look at one [Figure 2.10].

All the count plots for the distribution of each type of patient Classification look roughly the same, with ~90% of patients being negative and just under ~10% being positive. Indicating that the distribution for each classification is the same as the distribution for the overall dataset, signifying a lack of correlation. To gain a better understanding of this data, a subsample of the ICU negative population was created with a similar population size as the whole ICU positive population. This subsampling was used to gain better insights into the distribution of ICU cases amongst each classification. The percentage of the population that was ICU positive post subsampling was:

FIGURE 2.11 - PATIENT CLASSIFICATION ICU DISTRIBUTION

| Patient Classification | Percentage / % (rounded to nearest integer) |
|---|---|
| **1** | 59 |
| **2** | 40 |
| **3** | 52 |
| **4** | 50 |
| **5** | 52 |

| | |
|---|---|
| **6** | 37 |
| **7** | 47 |

This data signifies that there is some correlation between the classification of a patient and the likelihood of them being admitted to the ICU. This is likely because a patient's classification is based off features that may include potential ICU indicators.

### Conclusion of Data Visualisation

In conclusion, it was tough to find definitive conclusions regarding causes or correlations linked to ICU admissions, however, a predictive model will be created for this purpose, so it should not be expected that purely visualising the data gives all the answers. On the other hand, visualising the data did give Some important insights regarding the dataset we have at hand. For instance, we can see that only roughly 8% of the dataset were positive ICU admissions, which is very important information to have. Additionally, we can see that the different kinds of patient classification types are not equally balanced in population either. These sorts of insights are important to understanding the information at hand. Additionally, some insights regarding minor correlations were found within the visualisations. For instance, there is minor correlation between intubation and being admitted to the ICU, and this tracks with real world conceptions of Covid-19. Overall, the visualisation gave good insights into the dataset, but did not provide us with absolute certainty of the features we should choose regarding the predictive model. As a result, feature selection may have to combine justifications from these graphs as well as pre-existing conceptions regarding their clinical significance and their presence within the dataset.

## Basic Logistical Regression Model (Jayden Sachdev)

### Introduction and Basic Model

For the basic logistical regression model, a baseline classification model was developed using logistic regression. This model was used as a baseline before the creation of the more complex Pyspark-based random forest implementation. The model provided a valuable foundation for highlighting the benefits of the advanced modelling in the later stages of the project. The python Random Forest model is used on the training data, and the baseline logistical regression model is trained with the ICU as the target and main clinical features as the input.

### Limitations and Future Improvements

One of this approach's shortcomings was the harsh data-cleaning stage, which required removing rows with missing values. Although this ensured a clean dataset, it

might have resulted in the loss of potentially valuable data points. In the future, missing values might be handled using imputation techniques rather than deletion, retaining more information for study. Furthermore, investigating other classifiers, such as Gradient Boosting Machines or Support Vector Machines, may result in additional efficiency gains. More complex clustering approaches than K-Means, such as DBSCAN or hierarchical clustering, may reveal more information about patient groups.

# Advanced Predictive Modelling and Clustering Analysis using PySpark (Jayden Sachdev)

## Introduction – Clustering & Improved Model

As part of this project, I developed an improved classification model using PySpark and did clustering analysis on the patient dataset. This part aimed to improve the baseline model's performance and uncover relevant patterns among patients using K-Means clustering. The goal was to create a fair and trustworthy model to predict better ICU admission, particularly by resolving the dataset's class imbalance. [Figure 3.1]

## Feature Selection and Justification

The revised model's characteristics were chosen based on their clinical significance and availability in the dataset. These included age, obesity, tobacco use, hypertension, and diabetes. These risk variables are typically connected with critical health conditions and have been shown to impact ICU admission rates. By focusing on these vital qualities, the model might capture more relevant patterns while staying efficient and understandable.

## Advanced Data Cleansing

To generate an even better model, deeper data cleansing must be done. While many of the initial data cleansing techniques are sound, even better techniques may be employed to generate an even better data set. Most notably would be to handle missing values through imputation as opposed to removing them altogether. While some columns such as the Pregnant column were imputed the first time around due to the nature of the feature, imputation is now implemented for every feature. The method of imputation was the K-Nearest Neighbours method, which finds the nearest neighbours value and replaces the null value with average of the neighbours' values. It should be noted that categorical variables must be encoded first before this can take place.

## K-Means Clustering Analysis

K-Means clustering was used to group comparable patients based on the chosen characteristics. A k=3 value was used to identify three separate clusters in the data.

This helped identify trends in patient characteristics and enabled additional investigation into how ICU admissions were allocated throughout these clusters. The findings revealed that specific clusters had a more significant frequency of ICU admissions, indicating that clustering effectively identified patient groupings with varying risk levels. This contributed another layer of knowledge to the whole model, potentially supporting future localised risk projections.

## Initial Model Development (Unbalanced Data)

A Random Forest classifier was first trained on an imbalanced dataset to predict ICU admission. Although the model worked quite well, it tended to underestimate ICU=1 situations due to class imbalance. This underlined the necessity for balancing measures to guarantee that the model treated both groups equitably while avoiding bias towards the majority class (ICU=2).

## Handling Class Imbalance

The sample had much fewer instances of ICU=1 cases. To overcome this, oversampling was utilised to repeat ICU=1 rows and get the class distribution closer to equilibrium. This step was necessary to guarantee that the model could learn from both classes and increase its sensitivity in spotting key situations [Figure 3.2].

## Improved Model and Evaluation

After the data was balanced, a second Random Forest model was created. The model demonstrated considerable improvement in all meaningful metrics: [Figure 3.3]

- Accuracy: 0.98
- F1 Score: 0.97
- Precision: 0.96
- Recall: 0.98

The high recall score was especially noteworthy in this hospital environment, demonstrating the model's capacity to identify genuine ICU cases correctly. This suggests that the methodology has become more equitable and reliable in identifying at-risk patients.

## Interpretation and Insights

Overall, the modified model worked well and had high predictive power. Combining clustering and balancing improved the forecast's robustness and clinical relevance. PySpark also allowed for scalability processing, making this technique appropriate for real-world healthcare datasets.

## Technical Justification for using PySpark

PySpark was utilised to create the enhanced classification model because it can handle large-scale datasets in a distributed computing environment. While the dataset employed in this research was tiny, PySpark displays a knowledge of scalable data

processing tools. PySpark allows for effective parallel processing in real-world healthcare applications with significantly more enormous data volumes, making it a more viable and industry-relevant technique for constructing strong machine-learning pipelines.

## Justification for Random Forest Model Choice

Random Forest was chosen as the enhanced classification model due to its superior performance in classification issues, mainly when dealing with datasets including various feature types. It excels at modelling complicated, non-linear connections and can manage feature interactions without considerable pre-processing. Compared to simpler models like Logistic Regression, Random Forest is more flexible and performs better when there is variability and noise in data. It also gives helpful information on the relevance of features, making the model's decision-making process more interpretable. [Figure 3.4]

## Limitations and Future Improvements

While the improved random forest model delivered a strong performance in terms of the accuracy and the recall, there are still a few limitations worth acknowledging:

### Model Interpretability

Random Forests, while accurate, are not the best models meaning that in a healthcare scenario the decision transparency is key.

### Static Dataset

The model was trained on a single static dataset, meaning patient data is often updated in real time. A main future improvement would be implementing a streaming data pipeline using the Spark Streaming to handle the flow of continuous data.

## Conclusion

Finally, this part successfully supplied an enhanced classification model utilising PySpark, resolving the constraints of the previous model via data balance, clustering analysis, and sophisticated assessment approaches. Predictions became more accurate and meaningful after picking clinically significant characteristics and using Random Forest modelling.

# Appendix

## Appendix A: Data Cleaning and preprocessing

FIGURE 1.1 - DATASET SAMPLE BEFORE CLEANING AND MISSING VALUE IDENTIFICATION

```
First 5 rows of the dataset:
   index  USMER  MEDICAL_UNIT  SEX  PATIENT_TYPE   DATE_DIED INTUBED  \
0      2      2             1    2             2  09/06/2020       1
1      5      2             1    1             2  9999-99-99       2
2      8      2             1    1             2  9999-99-99       2
3      9      2             1    1             2  9999-99-99       2
4     11      2             1    2             2  9999-99-99       2

  PNEUMONIA  AGE PREGNANT  ... ASTHMA INMSUPR HIPERTENSION OTHER_DISEASE  \
0         2   55        ?  ...      2       2            2             2
1         1   40        2  ...      2       2            2             2
2         2   37        2  ...      2       2            1             2
3         2   25        2  ...      2       2            2             2
4         2   24        ?  ...      2       2            2             2

  CARDIOVASCULAR OBESITY RENAL_CHRONIC TOBACCO CLASIFFICATION_FINAL  ICU
0              2       2             2       2                    3  2.0
1              2       2             2       2                    3  2.0
2              2       1             2       2                    3  2.0
3              2       2             2       2                    3  2.0
4              2       2             2       2                    3  2.0

[5 rows x 22 columns]
Dataset Shape: (200031, 22)
```

FIGURE 1.2 - DATASET SHAPE BEFORE AND AFTER REMOVING MISSING VALUES

```
Missing values before dropping rows:
USMER                      0
MEDICAL_UNIT               0
SEX                        0
DATE_DIED                  0
INTUBED                 7325
PNEUMONIA               5144
AGE                        0
PREGNANT                   0
DIABETES                1195
COPD                    1064
ASTHMA                  1066
INMSUPR                 1280
HIPERTENSION            1106
OTHER_DISEASE           2074
CARDIOVASCULAR          1143
OBESITY                 1113
RENAL_CHRONIC           1074
TOBACCO                 1126
CLASIFFICATION_FINAL       0
ICU                     7488
dtype: int64
Dataset shape before dropping nulls: (200031, 20)
```

```
Missing values after dropping rows:
USMER                   0
MEDICAL_UNIT            0
SEX                     0
DATE_DIED               0
INTUBED                 0
PNEUMONIA               0
AGE                     0
PREGNANT                0
DIABETES                0
COPD                    0
ASTHMA                  0
INMSUPR                 0
HIPERTENSION            0
OTHER_DISEASE           0
CARDIOVASCULAR          0
OBESITY                 0
RENAL_CHRONIC           0
TOBACCO                 0
CLASIFFICATION_FINAL    0
ICU                     0
dtype: int64
Dataset shape after dropping nulls: (189586, 20)
```

FIGURE 1.3 - SUMMARY STATISTICS AND COLUMNS USED FOR OUTLIER DETECTION

```
Numeric columns for outlier detection: Index(['USMER', 'MEDICAL_UNIT', 'SEX', 'AGE', 'CLASIFFICATION_FINAL'], dtyp
e='object')
        USMER  MEDICAL_UNIT  SEX  AGE  CLASIFFICATION_FINAL
0           2             1    2   55                     3
1           2             1    1   40                     3
2           2             1    1   37                     3
3           2             1    1   25                     3
4           2             1    2   24                     3
...       ...           ...  ...  ...                   ...
200026      2            13    1   61                     7
200027      2            13    1   63                     7
200028      1            13    1   23                     7
200029      1            13    1   56                     7
200030      1            13    2   51                     7

[189586 rows x 5 columns]
Shape after removing outliers: (183673, 20)
0    55
1    40
2    37
3    25
4    24
```

FIGURE 1.4 - AGE COLUMN AFTER NORMALISATION

```
Name: AGE, dtype: int64
Data after normalization:
0    0.096959
1   -0.674738
2   -0.829077
3   -1.446434
4   -1.497881
Name: AGE, dtype: float64
```

FIGURE 1.5 - REMAINING COLUMNS AND FINAL DATASET SHAPE

```
Remaining columns:
Index(['USMER', 'MEDICAL_UNIT', 'SEX', 'DATE_DIED', 'INTUBED', 'PNEUMONIA',
       'AGE', 'PREGNANT', 'DIABETES', 'COPD', 'ASTHMA', 'INMSUPR',
       'HIPERTENSION', 'OTHER_DISEASE', 'CARDIOVASCULAR', 'OBESITY',
       'RENAL_CHRONIC', 'TOBACCO', 'CLASIFFICATION_FINAL', 'ICU'],
      dtype='object')
Final dataset shape: (189586, 20)
```

## Appendix B: Data Visualisation Outputs

FIGURE 2.1 - DISTRIBUTION OF ICU ADMISSIONS

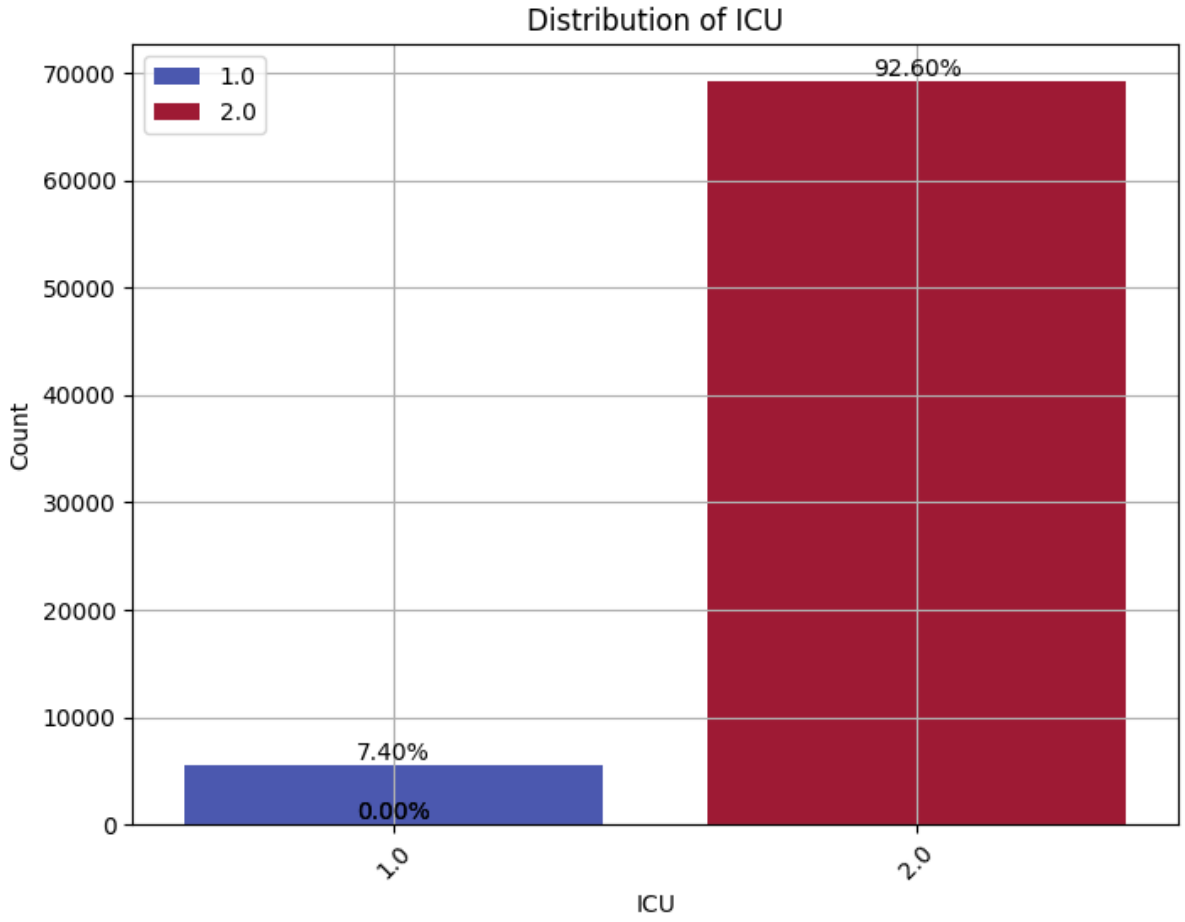


FIGURE 2.2 - ICU CASE VS. AGE

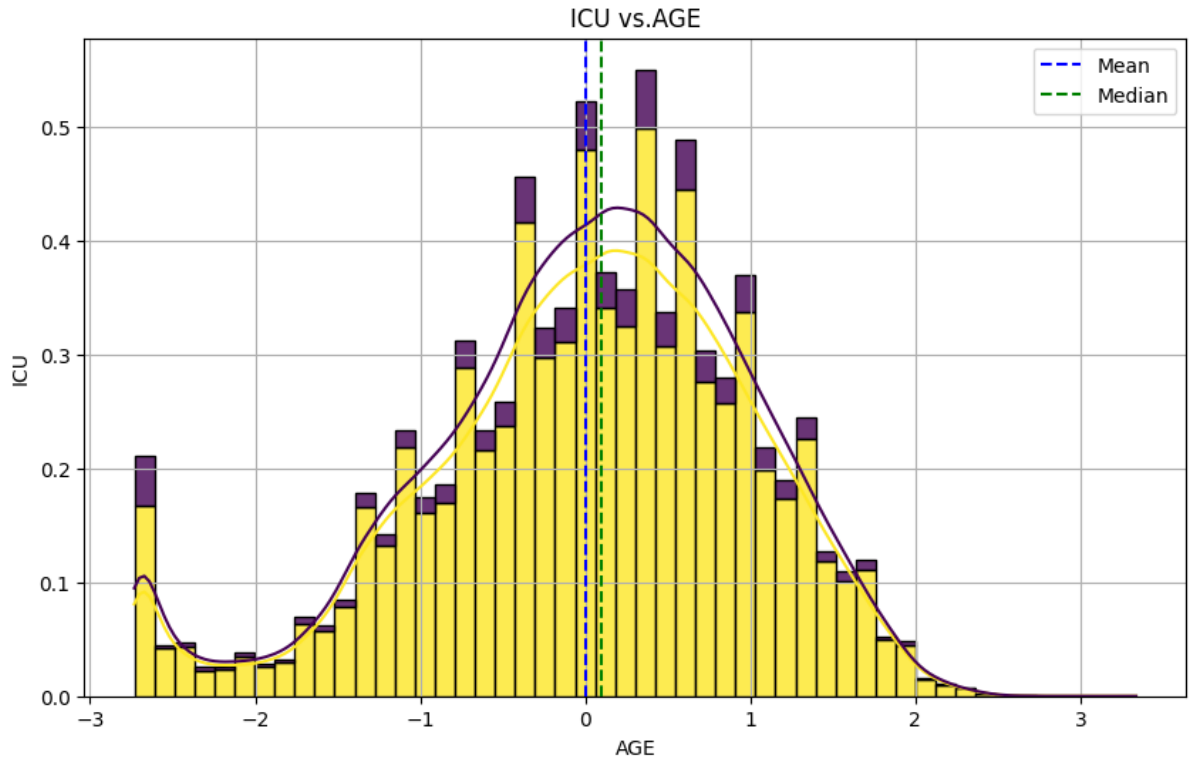



FIGURE 2.3 - PATIENT CLASSIFICATION BAR CHART

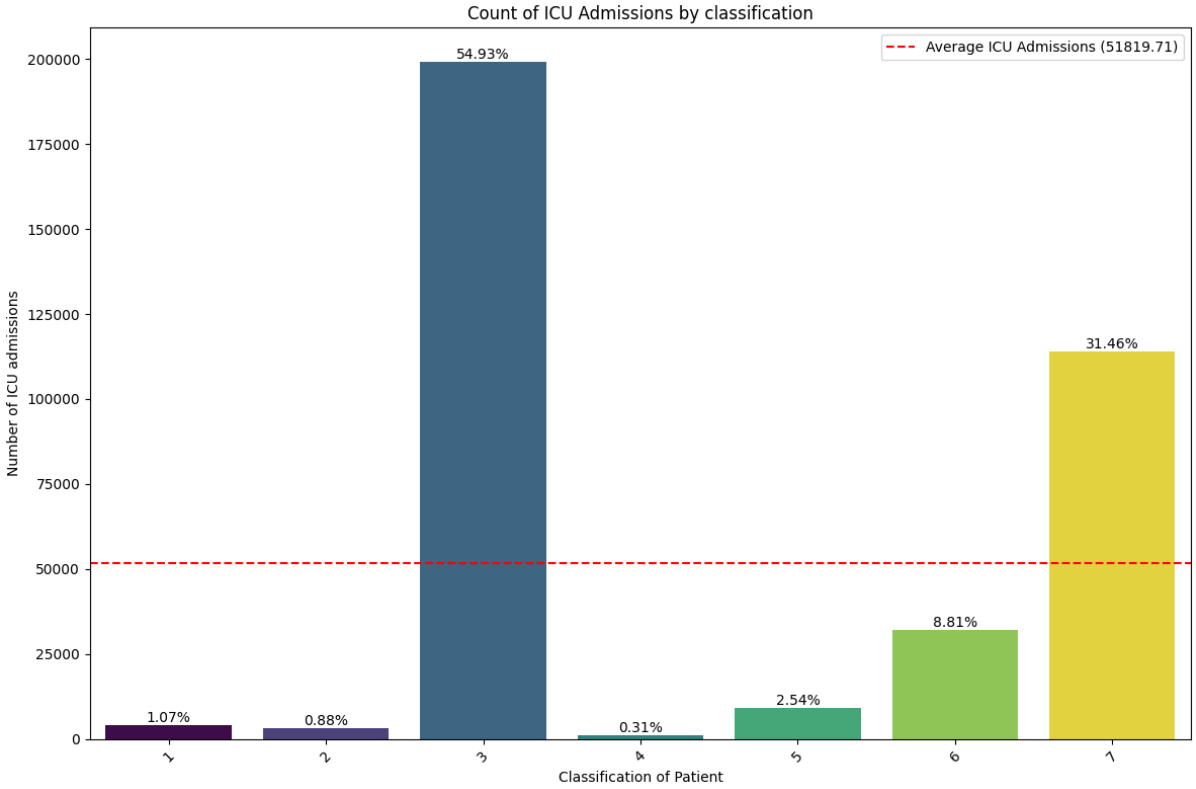
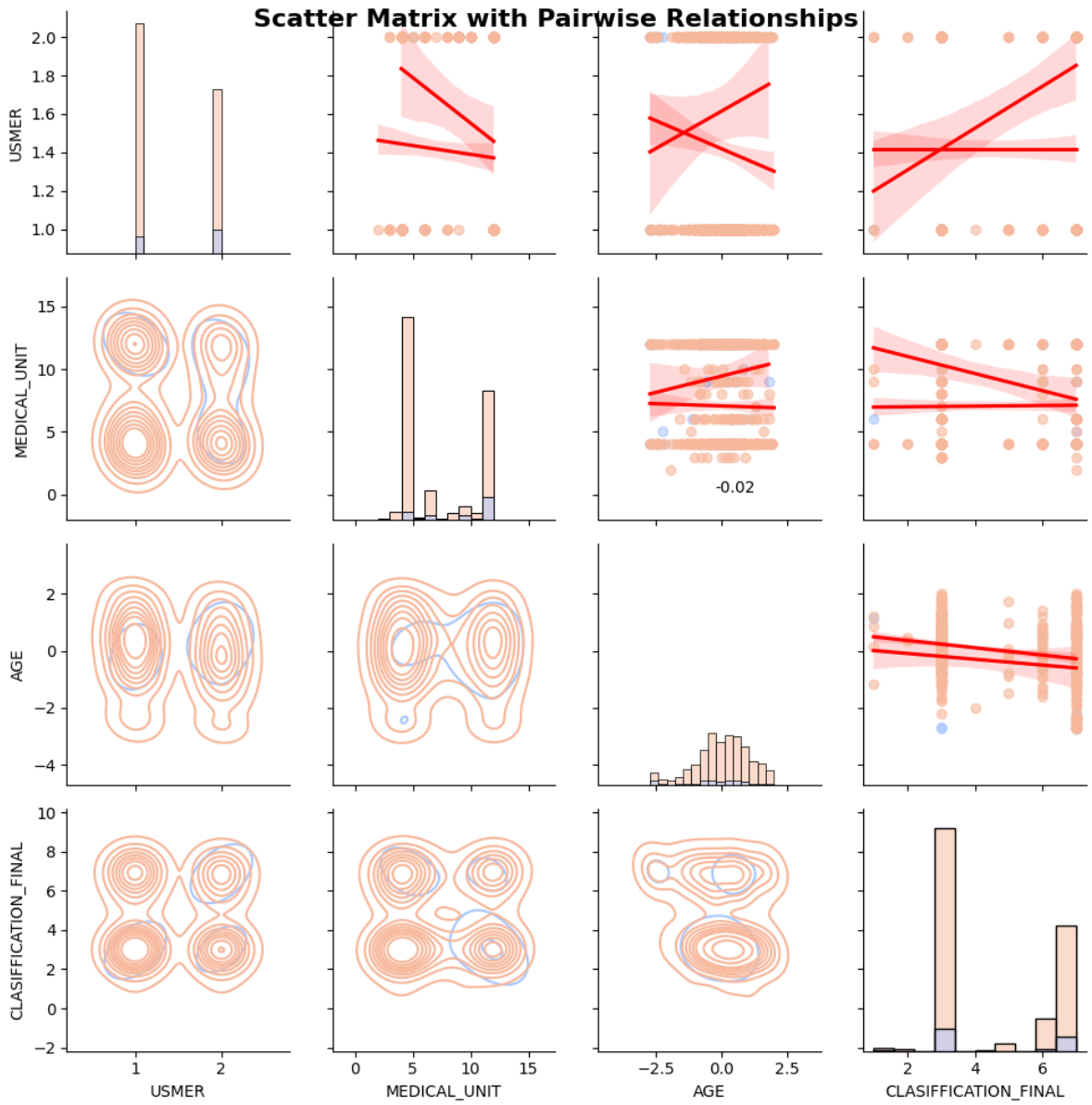
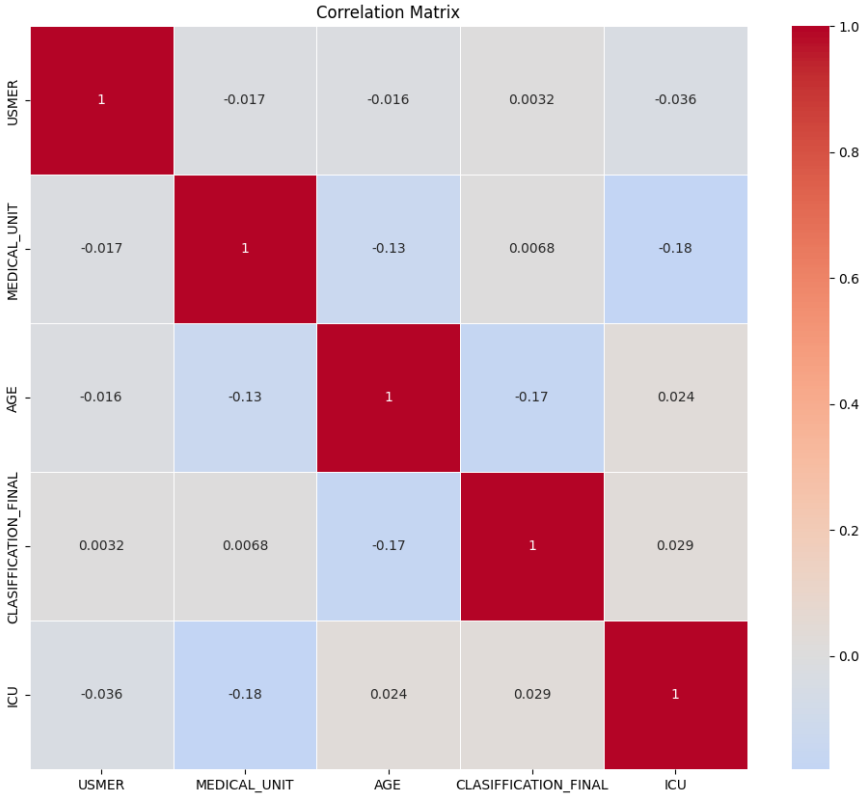


FIGURE 2.4 - SCATTER MATRIX WITH PAIRWISE RELATIONSHIPS



## Figure 2.5 - Numerical Feature Correlation Matrix




## Figure 2.6 - Categorical Feature Correlation Matrix

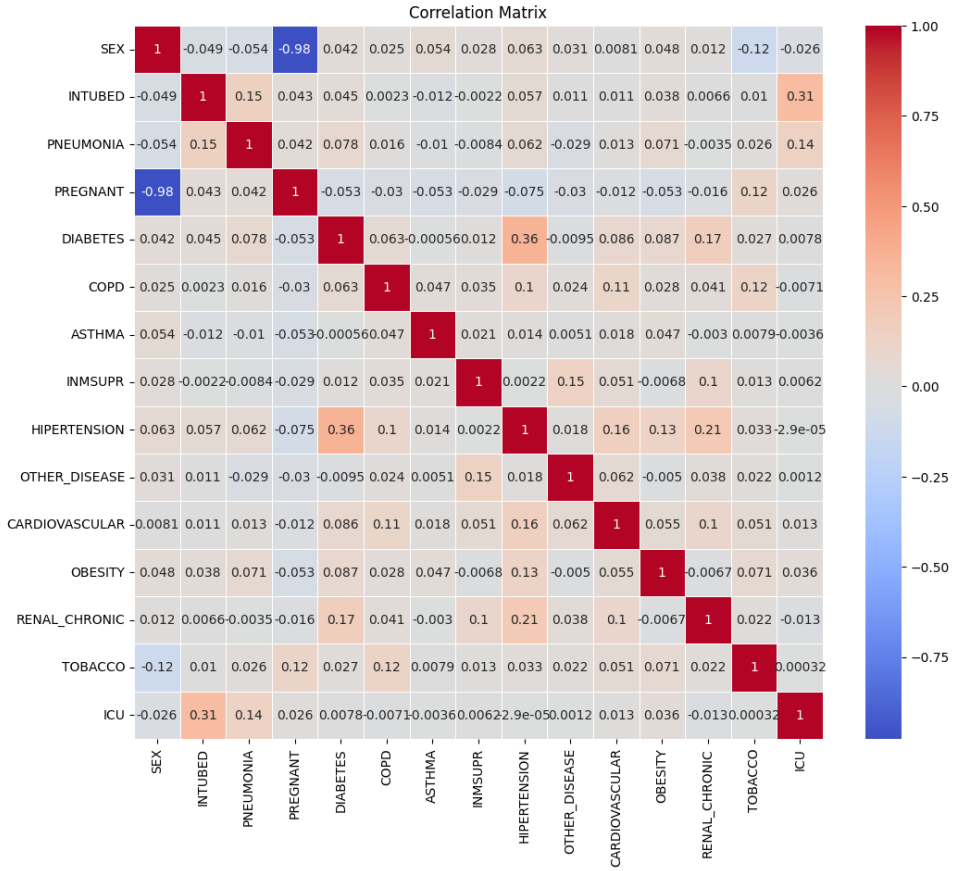


FIGURE 2.7 - GENDER BREAKDOWN

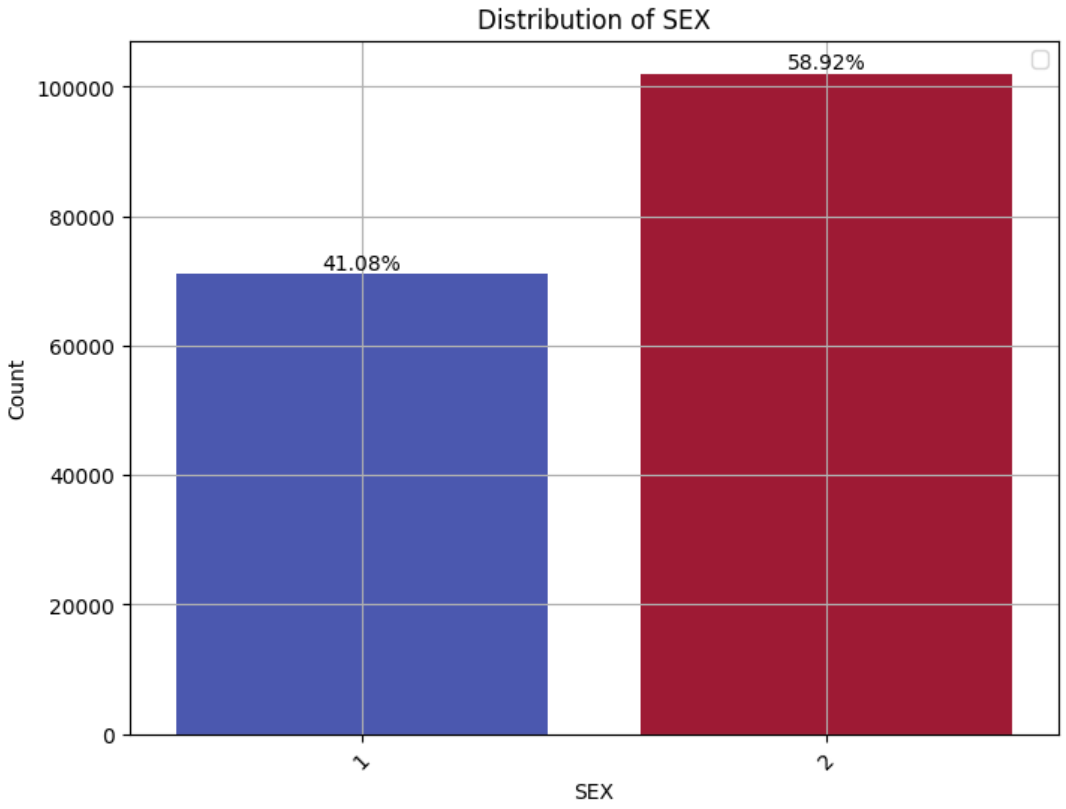



FIGURE 2.8 - DISTRIBUTION OF INTUBED PATIENTS

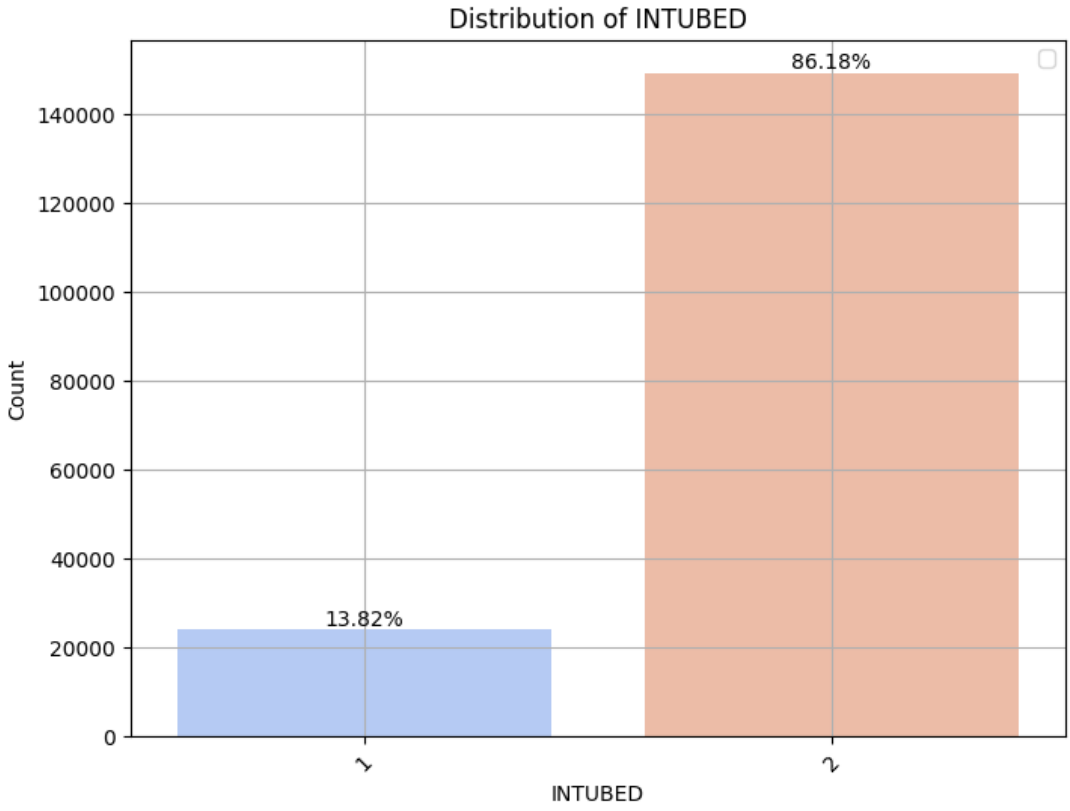


FIGURE 2.9 - AGE FEATURE BOX PLOT

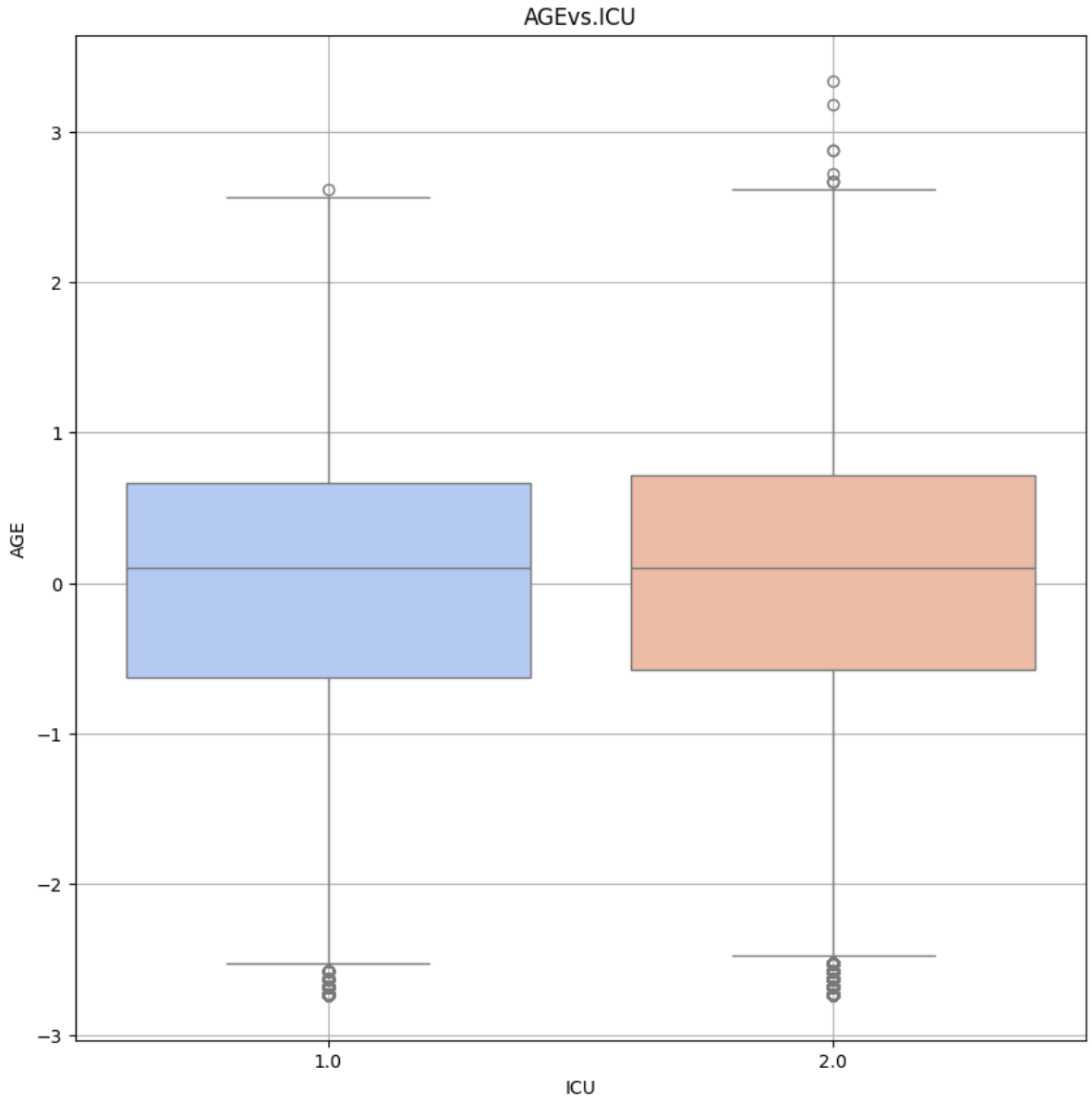
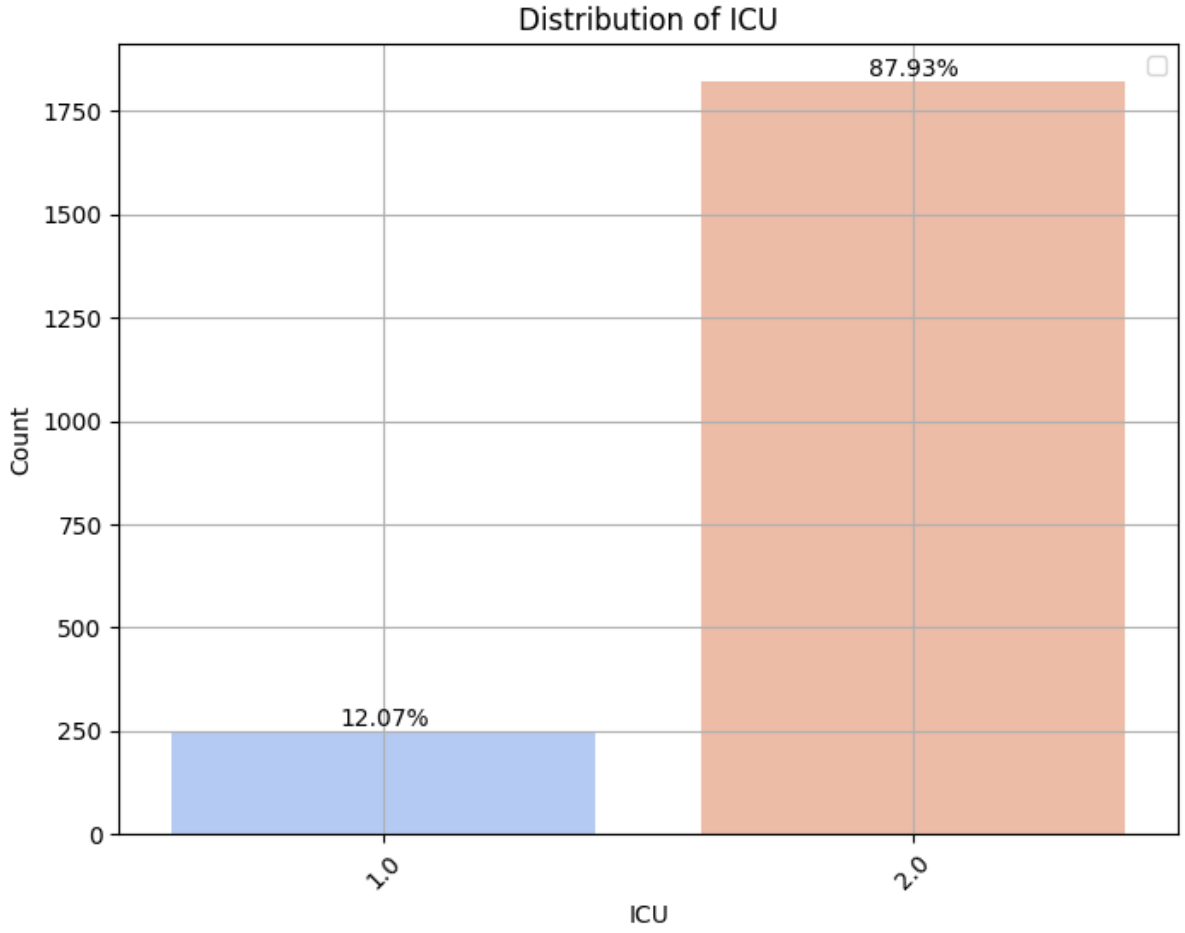
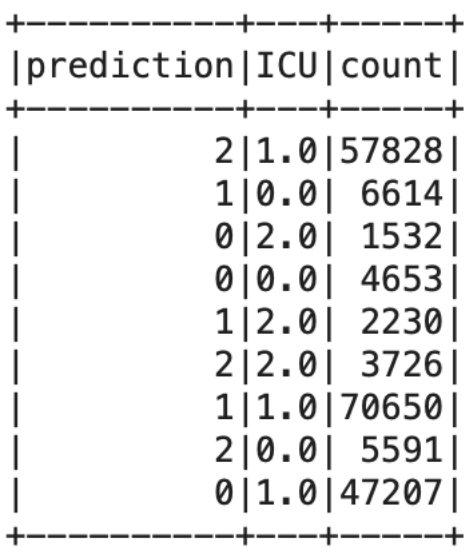


FIGURE 2.10 - DISTRIBUTION OF CLASSIFICATIONS IN ICU ADMISSIONS



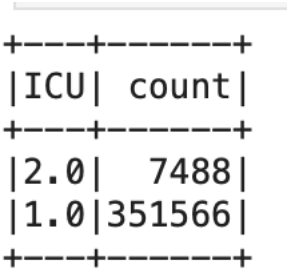
## Appendix C: Advanced Predictive Modelling and Clustering Analysis

FIGURE 3.1- CLUSTER BASED ICU PREDICTION COUNT



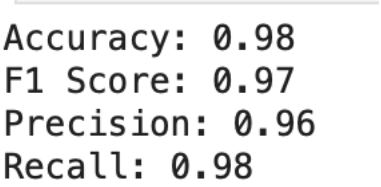
| prediction | ICU | count |
|---|---|---|
| 2 | 1.0 | 57828 |
| 1 | 0.0 | 6614 |
| 0 | 2.0 | 1532 |
| 0 | 0.0 | 4653 |
| 1 | 2.0 | 2230 |
| 2 | 2.0 | 3726 |
| 1 | 1.0 | 70650 |
| 2 | 0.0 | 5591 |
| 0 | 1.0 | 47207 |

FIGURE 3.2- CLASS IMBALANCE IN ICU LABELS BEFORE BALANCING



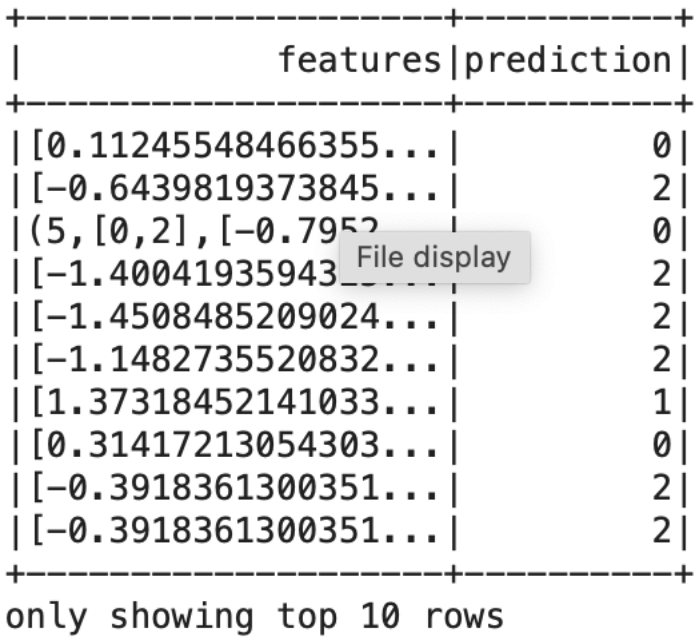
| ICU | count |
|---|---|
| 2.0 | 7488 |
| 1.0 | 351566 |

FIGURE 3.3- PYSPARK LOGISTIC REGRESSION MODEL PERFORMANCE METRICS



```
Accuracy: 0.98
F1 Score: 0.97
Precision: 0.96
Recall: 0.98
```

FIGURE 3.4- LOGISTIC REGRESSION MODEL PREDICTIONS ON FEATURE VECTORS




## Appendix D: Github Repository

https://github.com/owainjhughes/Big-Data-And-Predictive-Analytics