The evaluation of a Logistic Regression model and a Random Forest Classification model

Dataset Explanation:

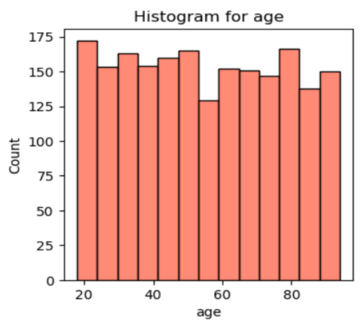
This dataset is collection of anonymous individuals whose information was collected with the objective of detecting which applicant was readmitted to the hospital within 30 days. It comprises of 10 variables with hospital readmission being the target variable. The variables are age, sex, residence_type, systolic_bp, diastolic_bp, cholesterol, hermoglobin, past_visits, length_of_stay and insurance_status. The details of the variables will be explained further during the exploratory data analysis phase of the report. The purpose of this report is to evaluate which model between a logistic regression model and random forest classifier model can best predict the outcome of the target variable using the above-mentioned variables. These models will also help in detecting which features are the most important which will decrease the amount of information to scan as well as help the hospitals with being able to detect the number of patients that will be admitted to the hospital within 30 days. This will help hospitals with making space for individuals and decrease waiting times for individuals by streamlining various processes.

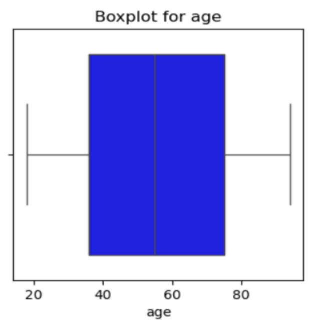
Data Profiling:

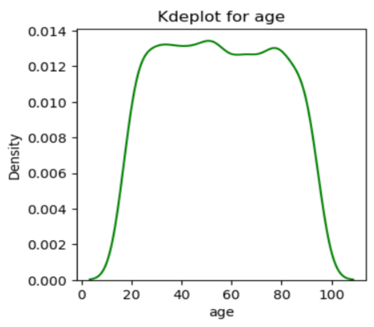
The dataset has 2000 rows; this means that 2000 people were used for this dataset. There are 4 categorical variables in the dataset namely, sex which consists of 2 unique values, residence_type consists of 3 unique values, insurance_status which consists of 3 unique values and readmitted_within_30days which indicates 1='Yes' and 0='No. The other variables are all numerical data types. Upon further investigation it was detected that cholesterol, hemoglobin, and past_visits had missing values. Missing values amounted to 213 rows which is just over 10% of the data. There were no duplicates in the dataset, which is also evidence that it was 2000 different people that were observed.

Exploratory Data Analysis (Variables with no missing Values):

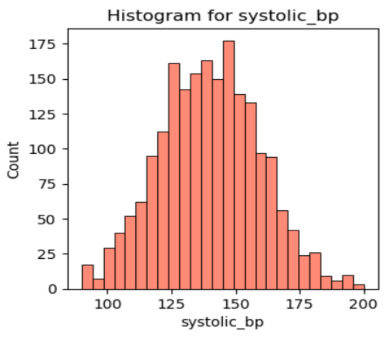
Numerical Variables

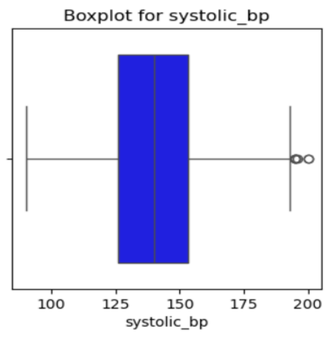


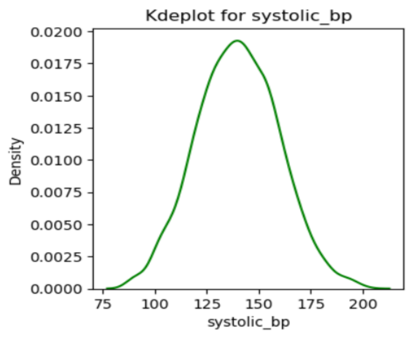




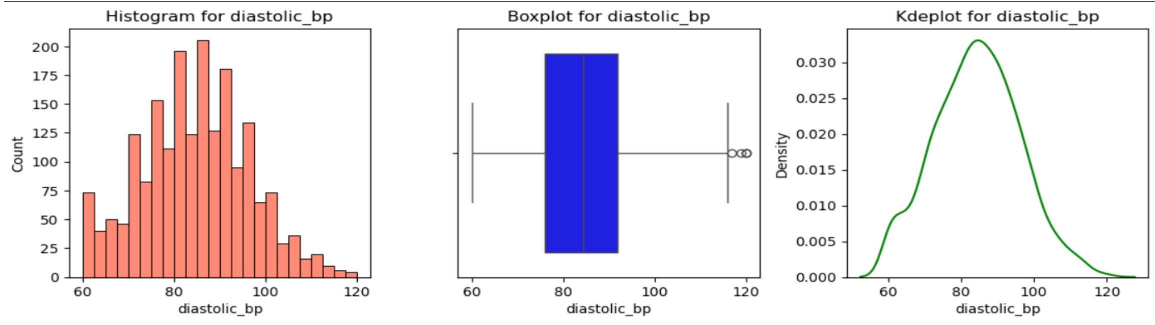
Age – We see here from the histogram that different age groups were assessed, and the youngest age was younger than 20 and the oldest age is over 80 years old. The boxplot tells us that there were no outliers in this dataset and that they tried to get profiles on all age groups. The standard deviation for the age variable is 22.23 and from the kdeplot we can see the most frequent age is in the 50's.



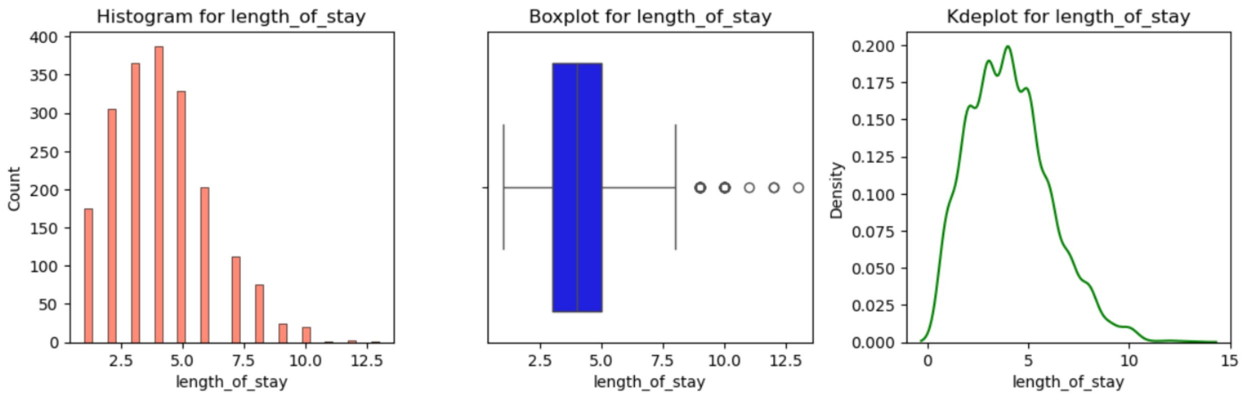




Systolic_bp – The histogram shows us most blood pressures were in the 150 range. The boxplot tells us that the median is about 140 and that there are few outliers. The most frequent blood pressure here would be in the 140 range as well. The average here is about 140 blood pressure



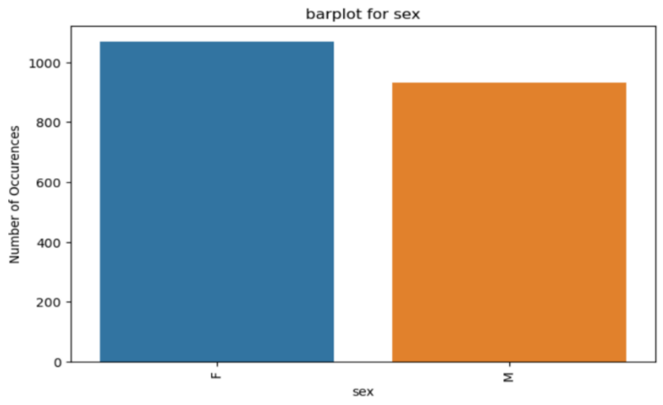
Diastolic_bp - The histogram shows us most blood pressures were in the 85 range. The boxplot tells us that the average is about 85 and that there are few outliers. The most frequent blood pressure here would be in the 85 range as well and the average is 84.5 blood pressure.

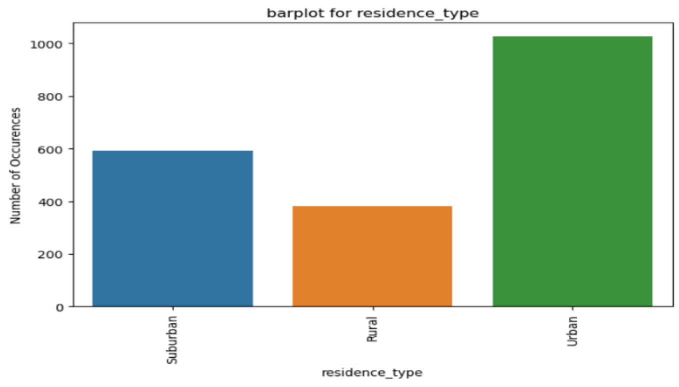


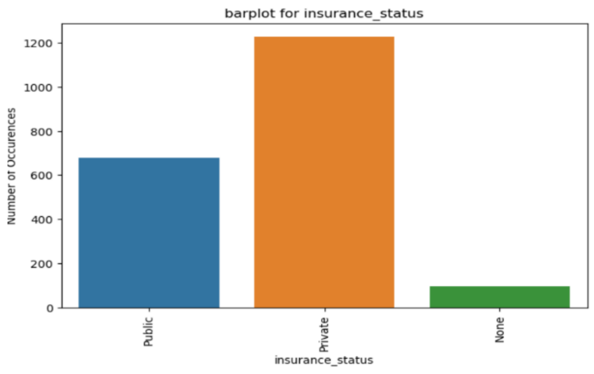
Length_of_stay - - The histogram shows us most people stayed in the hospital for about 4 days. The boxplot tells us that the average stay is about 4 days and that there are outliers. The most frequent days in the hospital is also 4 days.

The other numerical variables will be mentioned later when we show how we dealt with missing values.

Categorical Variables



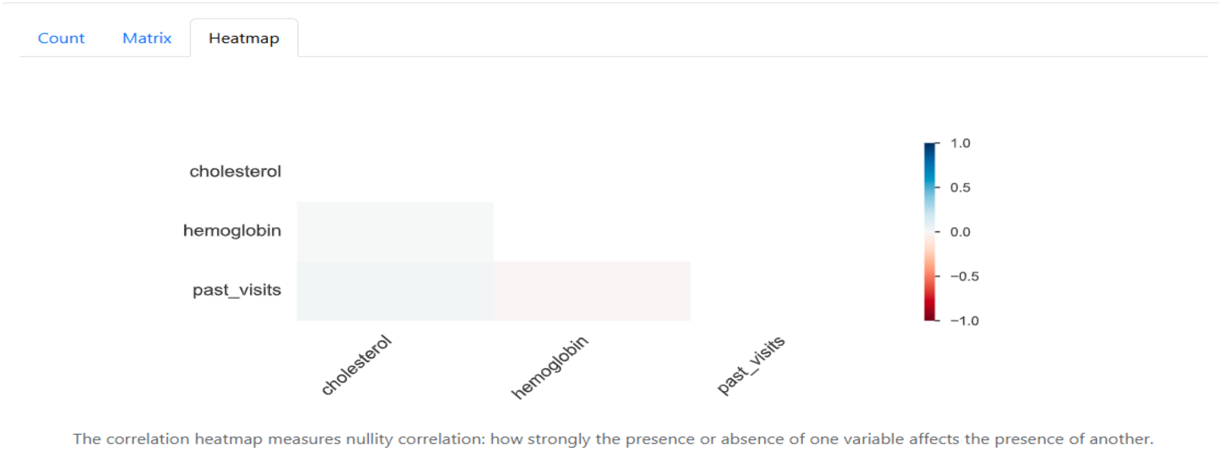




A) The gender that was observed the most was females.
B) The residence_type that was observed the most is urban, and the least is rural
C) The insurance status that was observed the most is the private type, and the least is the individuals whose value is ‘None’.

Missing Values and handling

As mentioned earlier in the data profiling section of the report. There were a total of 213 missing values missing from the dataset, that is just above 10% of the data missing that's a lot of data to drop and can influence the predictive ability of the models. We need to first identify the type of missing values in the dataset. This heatmap gives us more information regarding the columns with missing values. The image tells us that the values are missing completely at random as the strength of the heatmap is very light. This indicates that the missingness of one variable is not related to the missingness of the other variable

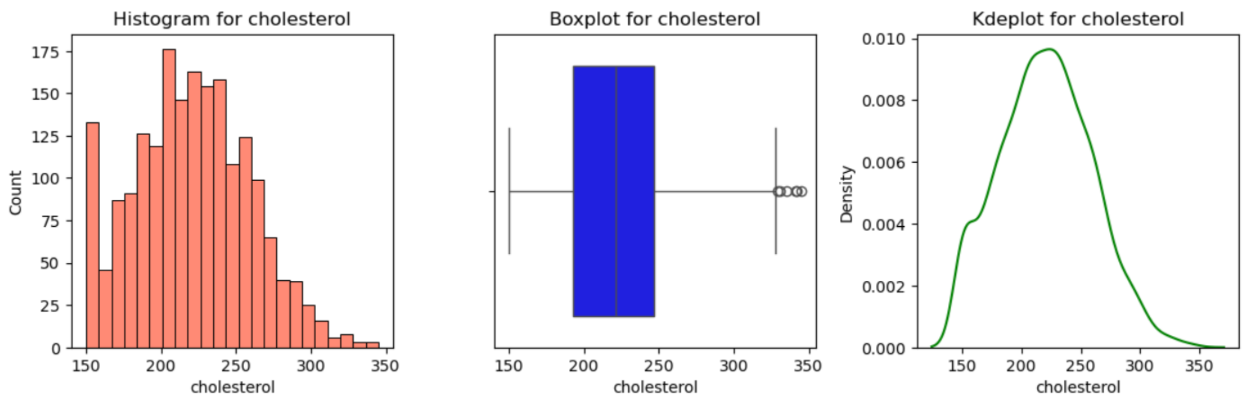


The correlation heatmap measures nullity correlation: how strongly the presence or absence of one variable affects the presence of another.

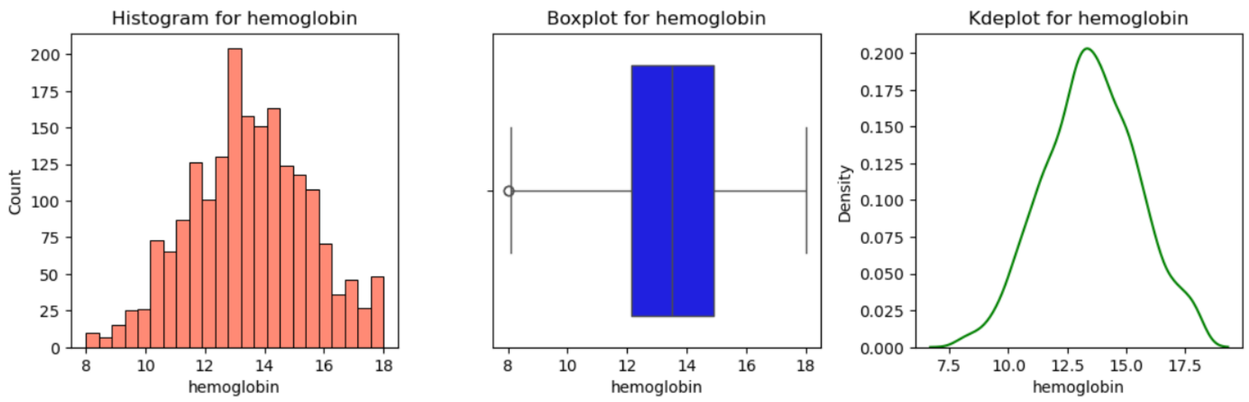
Because of the outliers in all the features, the method I chose to fill in the missing columns was by using the median. I first separated the dataset by gender because I didn't want to use a global median for both males and females. Below you'll see the histogram before imputation and after imputation. The shapes do change but the changes in standard deviations are not significant therefore there wasn't a radical change in data spread

Before Imputation

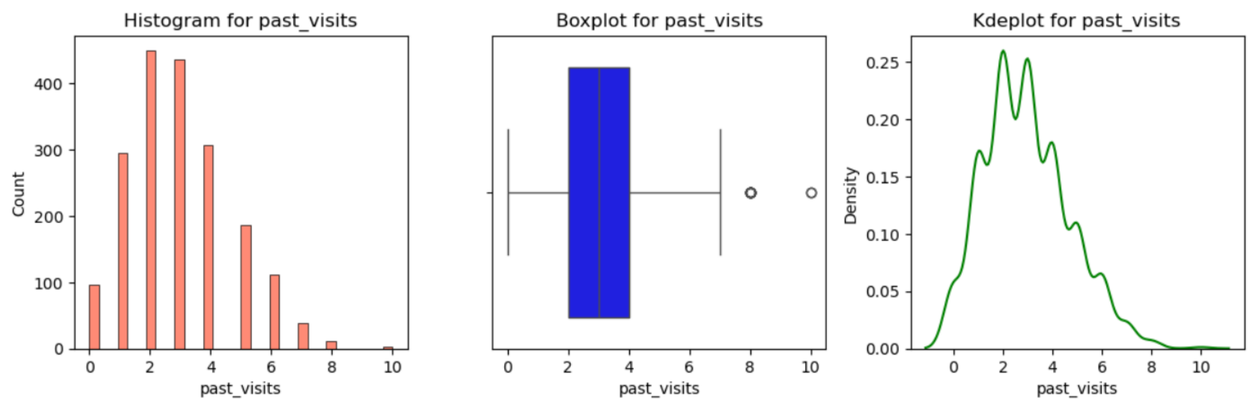
Standard deviation: 38.73



Standard deviation: 1.98

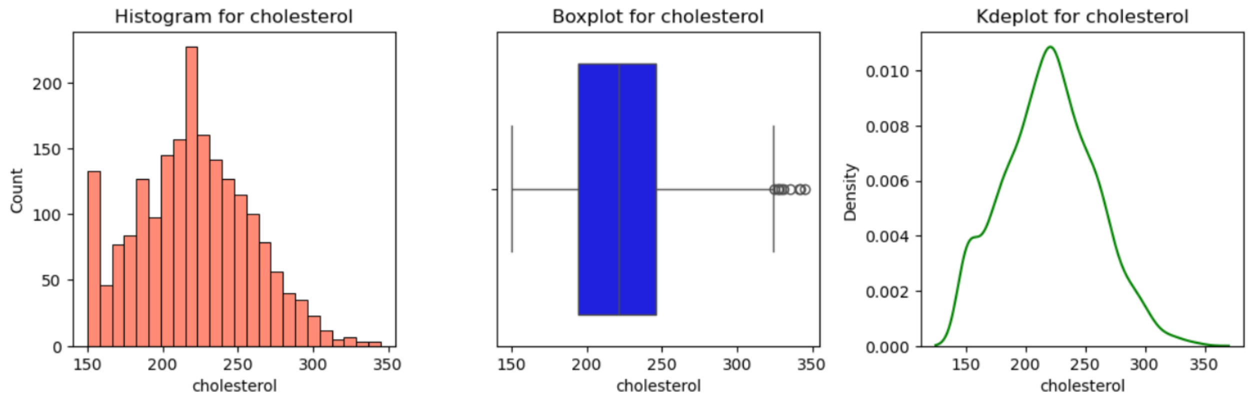


Standard deviation: 1.67

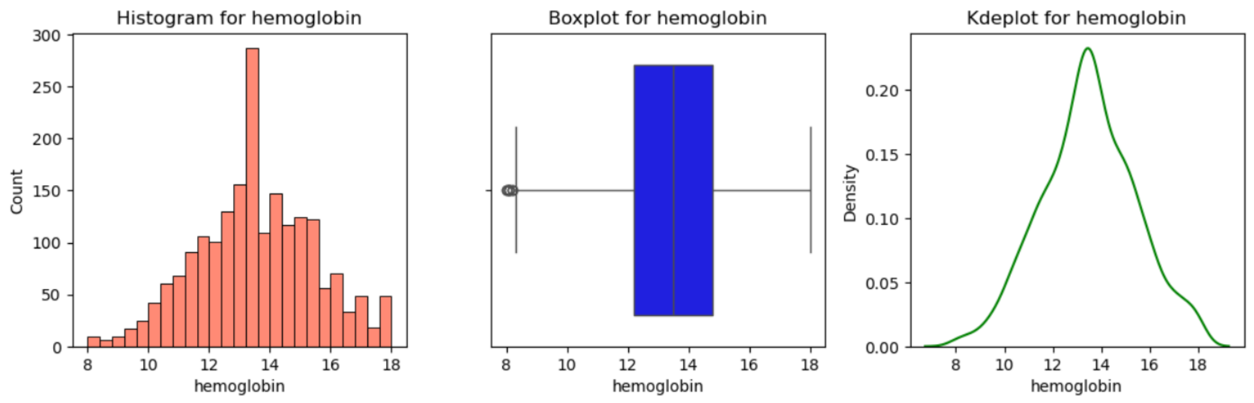


After Imputation

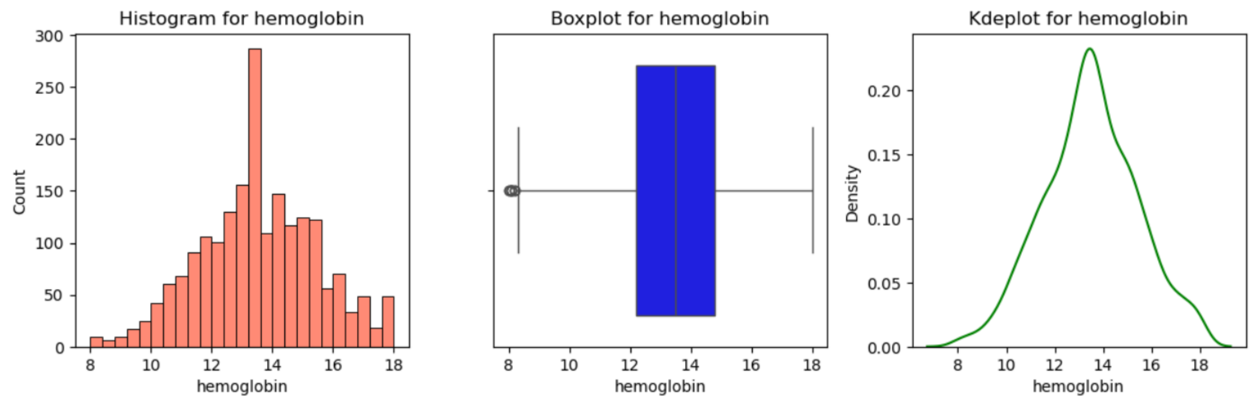
Standard deviation: 38.10



Standard deviation: 1.94



Standard deviation: 1.67



<u>Identifying Feature Importance using two methods and discussion of differences:</u>

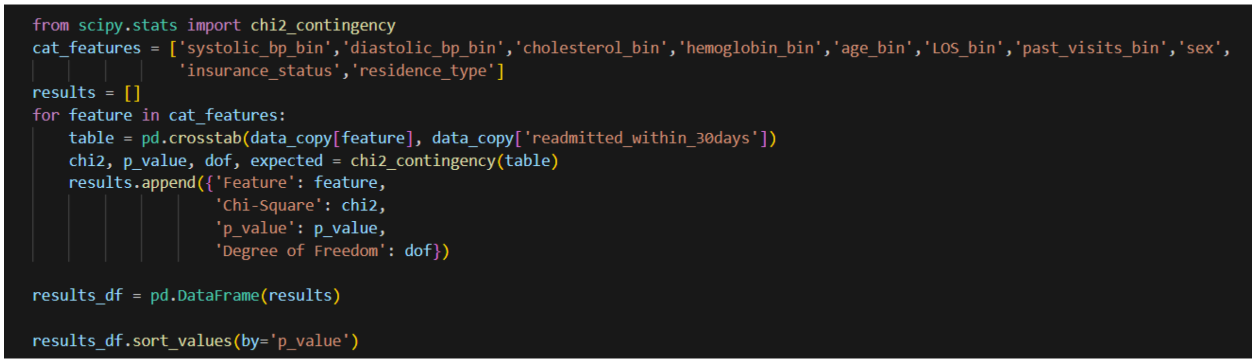
Chi-Square Test

This is a test used to determine feature importance based on the relationship between a feature and a target variable. This test determines the relationship between two categorical variables; with this being said numerical data types need to be transformed into categorical variables. It calculates the actual data counts and the expected data counts. This results in a Chi-Score. The Chi-Score tells us if the variables are dependent or independent of each other. A high Chi-Score results in a high dependency therefore revealing importance when detecting features for when training a model.

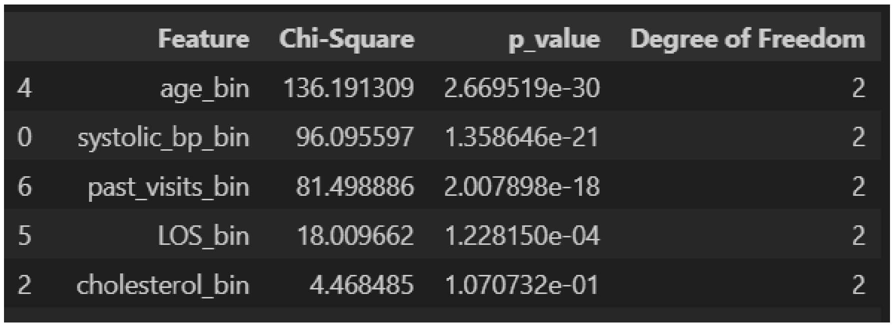
To get the numerical data in categorical data I had to put the numerical data in bins. This table provides a description of how the data was binned and aligns with the scales of the numerical data. This table shows that all the numerical data was binned into 3 categories. Bio_categories is a variable which consists of systolic_bp, diastolic_bp, cholesterol and hemoglobin. The other variables are all already in categorical data and nothing would need to be changed.

Numerical_data	Scale
Bio_categories	Low, Medium and High
Age_bin	Young adults, Middle Aged, Senior
LOS_bin	Short, Medium, Long
Past_visits	Low, Medium, High

This is the code for the Chi-Square test and the results.



```python
from scipy.stats import chi2_contingency
cat_features = ['systolic_bp_bin','diastolic_bp_bin','cholesterol_bin','hemoglobin_bin','age_bin','LOS_bin','past_visits_bin','sex',
                'insurance_status','residence_type']
results = []
for feature in cat_features:
    table = pd.crosstab(data_copy[feature], data_copy['readmitted_within_30days'])
    chi2, p_value, dof, expected = chi2_contingency(table)
    results.append({'Feature': feature,
                    'Chi-Square': chi2,
                    'p_value': p_value,
                    'Degree of Freedom': dof})

results_df = pd.DataFrame(results)

results_df.sort_values(by='p_value')
```



	Feature	Chi-Square	p_value	Degree of Freedom
4	age_bin	136.191309	2.669519e-30	2
0	systolic_bp_bin	96.095597	1.358646e-21	2
6	past_visits_bin	81.498886	2.007898e-18	2
5	LOS_bin	18.009662	1.228150e-04	2
2	cholesterol_bin	4.468485	1.070732e-01	2

These are the most important features ranked by the Chi-Square test. Age could influence hospital readmission because with different age groups, the body changes. Systolic_bp refers to blood against the artery walls of the heart, anything to do with the heart would need to be treated at a hospital. Past_visits refers to how many times you were at the hospital. A high number of past_visits could mean that you probably have a chronic illness. Length_of_stay refers to how many days you stayed in the hospital. This has a relationship with hospital readmissions because it could mean that the reason for your long stay in the hospital is the same reason as you are coming back. Cholesterol also has to do with blood and high cholesterol can lead to heart attacks which would mean that the individual would need to go to a hospital.

Forward Selection Wrapper Method

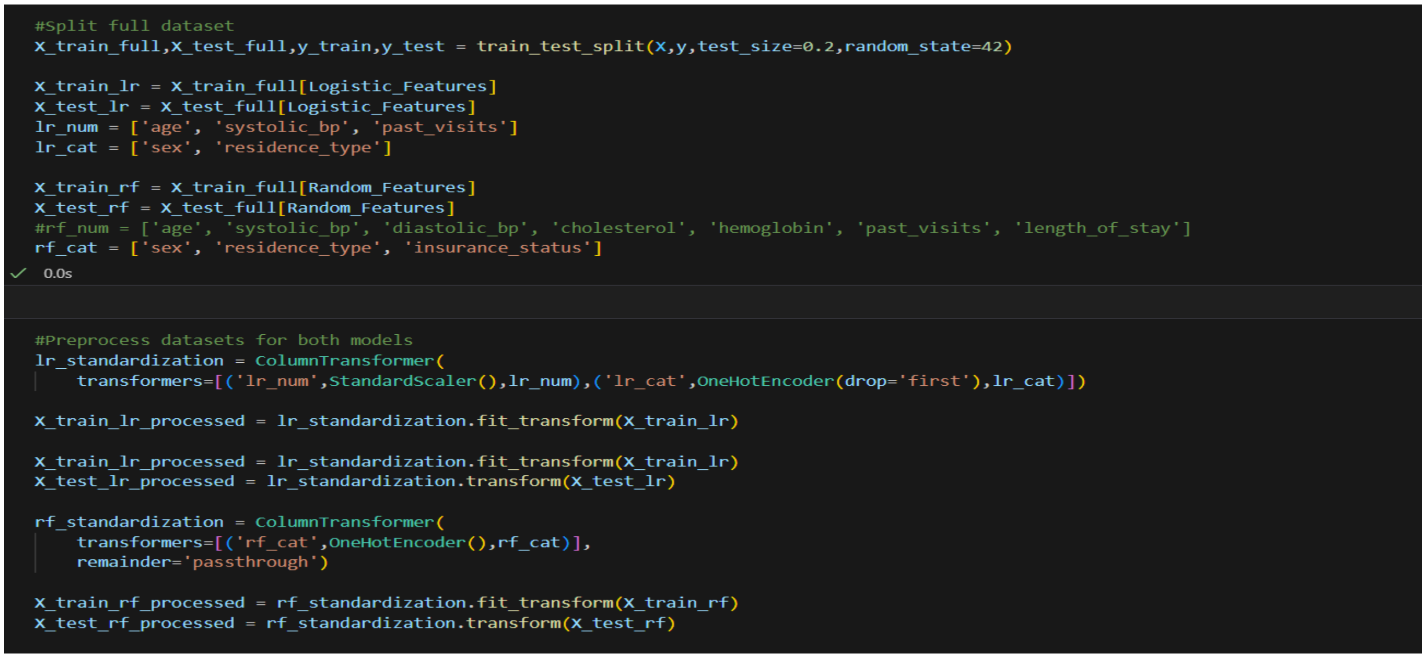
This is a machine-learning algorithm that selects and presents the best features by starting with one feature and then iteratively adding the next feature. This method stops when it sees the model's performance isn't improving anymore. The purpose of this report is to evaluate two classification models, so I developed two forward selection wrapper methods: one for Logistic Regression and the other for Random Forest Classifier. The result of this was I got two different subsets of features based on the model. In the table below you will find the results of both models as well as the performance of the models with the resulting subset.

Model	Subset	Performance
Logistic Regression	['age', 'systolic_bp', 'past_visits', 'sex_M', 'residence_type_Suburban', 'residence_type_Urban']	0.8935
Random Forest Classifier	['age', 'systolic_bp', 'diastolic_bp', 'cholesterol', 'hemoglobin', 'past_visits', 'length_of_stay', 'sex_M', 'residence_type_Suburban', 'residence_type_Urban', 'insurance_status_Private', 'insurance_status_Public']	0.883

We see here that both models removed some categories due to multicollinearity. The random forest classification model has all the variables contributing to predicting the outcome of hospital readmissions whilst the logistic model only has 5 of the variables contributing to predicting the outcome of the target variable and according to the performance score the logistic regression model performs better at predicting the outcome. To conclude this part, I am going to be speaking about the differences of the subset of these two feature selection methods. The Chi-Square function doesn’t encode the categorical variables like the forward selection wrapper method does, this will help during the scaling and sub setting of data for the predictive model. Random forest classification brings in all the features, but Logistic regression forward selection method brings in two features that the Chi-Square method doesn’t which is sex and residence_type whereas the two the Chi-Square regards as important is length_of_stay and cholesterol. If we look at the p-value in the Chi-Square section it is greater than 0.05 which means that it is not statistically significant so if we were to use the features from the Chi-Square filter method, we could drop that feature from the features predicting the outcome variable.

<u>Building the Logistic Regression and Random Forest Classification</u>

Process: I first split the dataset into a training and testing sample. I then identified the most important features from the forward selection wrapper method and created variables. I then created separate datasets for my two models and created variables for my categorical and numerical columns and prepared them for scaling. For random forest classification I didn’t scale my numerical columns as it’s from the ensemble library it uses thresholds, but I did do one -hot encoding on my categorical columns. For my logistic regression model, I used one-hot encoding for my categorical columns and then I used z-scale scoring to scale my numerical features.



```
#Split full dataset
X_train_full,X_test_full,y_train,y_test = train_test_split(X,y,test_size=0.2,random_state=42)

X_train_lr = X_train_full[Logistic_Features]
X_test_lr = X_test_full[Logistic_Features]
lr_num = ['age', 'systolic_bp', 'past_visits']
lr_cat = ['sex', 'residence_type']

X_train_rf = X_train_full[Random_Features]
X_test_rf = X_test_full[Random_Features]
#rf_num = ['age', 'systolic_bp', 'diastolic_bp', 'cholesterol', 'hemoglobin', 'past_visits', 'length_of_stay']
rf_cat = ['sex', 'residence_type', 'insurance_status']
```

```
0.0s
```

```
#Preprocess datasets for both models
lr_standardization = ColumnTransformer(
    transformers=[('lr_num',StandardScaler(),lr_num),('lr_cat',OneHotEncoder(drop='first'),lr_cat)])

X_train_lr_processed = lr_standardization.fit_transform(X_train_lr)

X_train_lr_processed = lr_standardization.fit_transform(X_train_lr)
X_test_lr_processed = lr_standardization.transform(X_test_lr)

rf_standardization = ColumnTransformer(
    transformers=[('rf_cat',OneHotEncoder(),rf_cat)],
    remainder='passthrough')

X_train_rf_processed = rf_standardization.fit_transform(X_train_rf)
X_test_rf_processed = rf_standardization.transform(X_test_rf)
```

Model Train – I then took the relevant datasets to fit onto my Logistic and Random Forest Classification models. The fitting was successful and both models are now working and can be used on unseen data. Below you'll see a confusion matrix that was performed on unseen data which provides as evidence that the models were built and how they are performing.

True Positives – In this case this is where 'readmitted_within_30days' is equal to 1 and the model predicted correctly that it is 1

True Negatives - – In this case this is where 'readmitted_within_30days' is equal to 0 and the model predicted correctly that it is 0

False Positives – This is where the model predicted 0 but the actual value was 1

False Negatives -This where the model predicted 1 and the actual value is 0

Measurement	**Logistic Regression**	**Random Forest Classification**
True Positive	7	5
True Negative	359	363
False Positive	6	2
False Negative	28	30

Model Evaluation using these measurements

Accuracy – This is a proportion of all predictions that were correct

Precision – measures how many instances that were predicted positive were positive

Recall – How many actual positives were captured

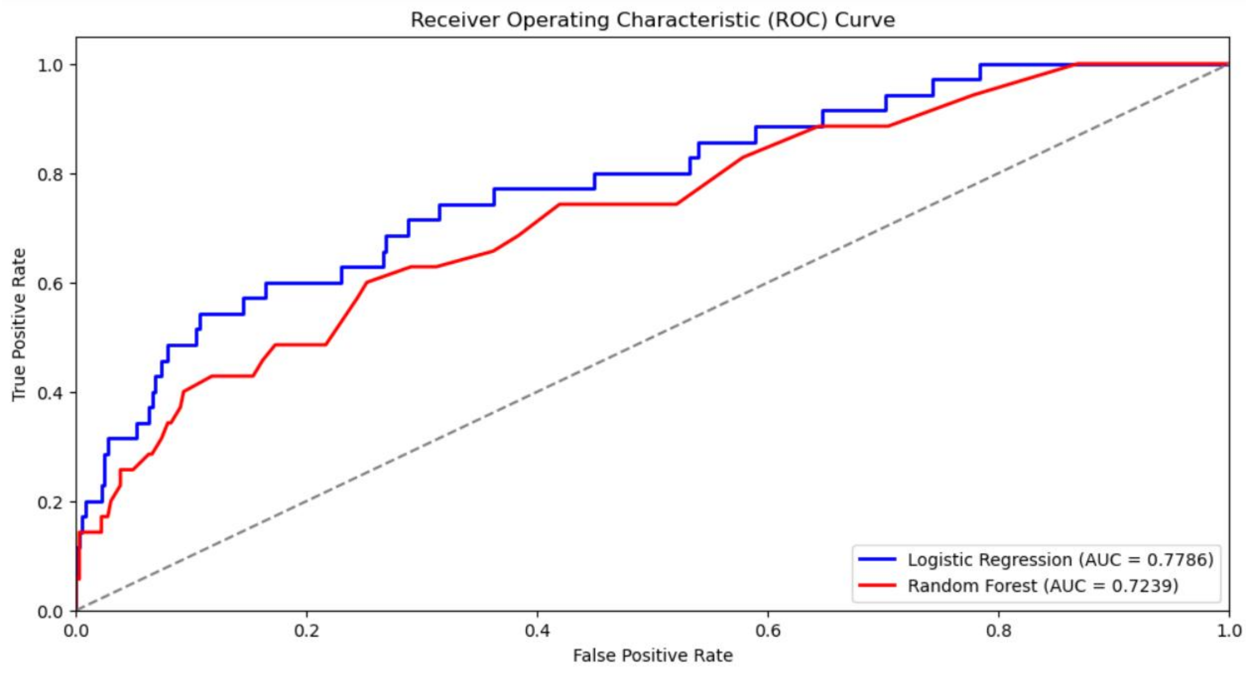
F1-Score – This refers to the balance between Precision and Recall

ROC-AUC – Measures the model's ability to distinguish the target variable's classes

Measurement	**Logistic Regression**	**Random Forest Classifier**
Accuracy	0.91	0.92
Precision	0.54	0.71
Recall	0.2	0.14
F1-Score	0.29	0.24
ROC-AUC	0.78	0.72

For accuracy the Random Forest model performs just a bit better. In the unseen data this model got more predictions this is also evidence of the imbalanced dataset because 0 as the negative dominates the dataset. In terms of precision the Random Forest model is better at detecting false positives, by sub setting the data the hospital would do better detecting individuals that would be admitted in 30 days with this model. Both models have significantly low recall measures, but Logistic regression has a better score which means that it's better at identifying actual positives. Logistic regression has a better F1-score which means that it has better balance for precision and recall. Logistic Regression has a better ROC-AUC score which means that it's better at distinguishing the target variables classes.

The table above aligns with the ROC curve plots. The y-axis measures the rate of True Positives, this rate is known as the sensitivity. Therefore, the logistic regression model has a higher True positive rate, this means that this model is better at catching actual positives which in this case is the class = 1. Specificity is 1 minus the False Positive rate that is plotted on the x-axis. Therefore, wherever the False profit rates are the same the logistic regression model has a higher sensitivity. The Area Under the Curve for the Random Forest is less than the logistic regression model which means that the logistic model can distinguish between the two classes better than the random forest classification model.

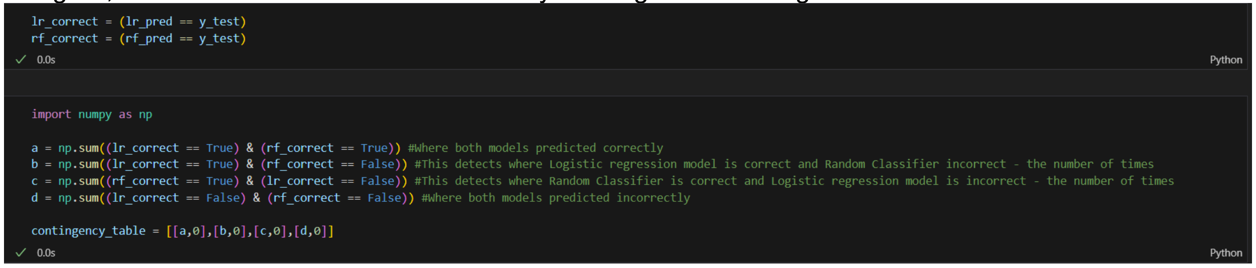


Statistical Significance Test

This section of the report we are going to use Mcnemar's test to determine the statistical significance of the models, in other words we are going to be testing the gap of performance between the two models. This test focuses on where both models disagree. This means that it isolates where the one model was correct and the other one wasn't.

Methodology

I created two variables whereby it stores are the values where the predicted values were equal to the true values. 1 for Logistic Regression and the other for Random Forest Classification. I then created a matrix where it sums up where both models were correct, both models were incorrect and where one models was correct and the other was incorrect. I then proceeded to store the amounts in a contingency table that I'm going to use for McNemar's test even though this test focuses only on where the models disagree, I still stored the values for when they both agree and disagree.



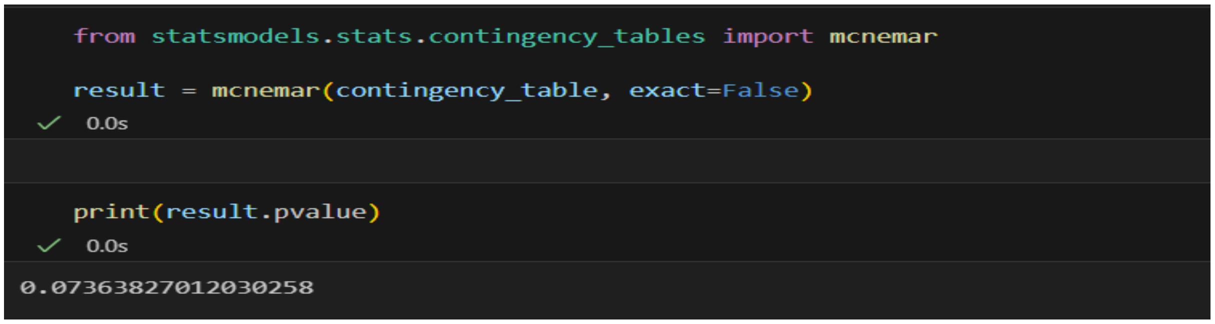
```
lr_correct = (lr_pred == y_test)
rf_correct = (rf_pred == y_test)
✓ 0.0s                                                                                   Python
```

```
import numpy as np

a = np.sum((lr_correct == True) & (rf_correct == True)) #Where both models predicted correctly
b = np.sum((lr_correct == True) & (rf_correct == False)) #This detects where Logistic regression model is correct and Random Classifier incorrect - the number of times
c = np.sum((rf_correct == True) & (lr_correct == False)) #This detects where Random Classifier is correct and Logistic regression model is incorrect - the number of times
d = np.sum((lr_correct == False) & (rf_correct == False)) #Where both models predicted incorrectly

contingency_table = [[a,0],[b,0],[c,0],[d,0]]
✓ 0.0s                                                                                   Python
```

Contingency table results

Both Agree	Logistic Correct	Random Forest Correct	Both Disagree
362	5	7	27

I then used the statsmodel library to import mcnemar's test to get the p-value that will measure the statistical difference between both models.



```
from statsmodels.stats.contingency_tables import mcnemar

result = mcnemar(contingency_table, exact=False)
✓ 0.0s
```

```
print(result.pvalue)
✓ 0.0s
```

```
0.07363827012030258
```

The p-value is 0.07. This value is greater than 0.05 which indicates that the gap of performance between the two models isn't huge. In machine learning terms – it is not statistically significant. This means that both models work relatively the same and that there's not enough information to definitively choose a model that works the best. The table above also provides evidence regarding this because there were 5 times the logistic regression model was correct, and the random forest regression model was incorrect and then there were 7 times the random forest classification model was correct, and the logistic regression model was incorrect. The difference is very small.

<u>**Conclusion**</u>

The case is that the hospital wants to better predict hospital readmissions within 30 days to be able to apply resources better, therefore I would suggest the logistic regression model because it has a higher recall value meaning that it predicts the 1 value better. It has a better F1-score meaning that it has a better balance between precision and recall also has a higher ROC-AUC score meaning that it knows how to distinguish the classes in the target variable better. There is also an imbalance on the negative class this could be the reason why there is no significant performance difference in the models.