|  |
| --- |
| Modeling Inpatient Readmissions |
| Western Governors University |
| Heather Rauch |

Table of Contents

[A: Research Question 2](#_Toc116064431)

[A1: Research Question 2](#_Toc116064432)

[A2: Objectives and Goals 2](#_Toc116064433)

[B: Method Justification 2](#_Toc116064434)

[B1: Summary of Assumptions 2](#_Toc116064435)

[B2: Tool Benefits 2](#_Toc116064436)

[B3: Appropriate Technique 3](#_Toc116064437)

[C: Data Preparations 3](#_Toc116064438)

[C1: Data Goals 3](#_Toc116064439)

[C2: Summary Statistics 3](#_Toc116064440)

[C3: Steps to Prepare the Data 5](#_Toc116064441)

[C4: Visualizations 10](#_Toc116064442)

[C5: Prepared Data Set 10](#_Toc116064443)

[D: Model Comparison 11](#_Toc116064444)

[D1: Initial Model 11](#_Toc116064445)

[D2: Justification of Model Reduction 11](#_Toc116064446)

[D3: Reduced Multiple Regression Model 12](#_Toc116064447)

[E: Model Analysis 13](#_Toc116064448)

[E1: Model Comparison 13](#_Toc116064449)

[E2: Output and Calculations 14](#_Toc116064450)

[E3: Code 19](#_Toc116064451)

[F: Data Summary and Implications 21](#_Toc116064452)

[F1: Results 21](#_Toc116064453)

[F2: Recommendations 23](#_Toc116064454)

[G: Panopto Recording 23](#_Toc116064455)

[H: Sources of Third-Party Code 23](#_Toc116064456)

[I: Sources 23](#_Toc116064457)

# A: Research Question

## A1: Research Question

This analysis aims to answer the question: What factors predict whether a patient will be readmitted to the hospital.

## A2: Objectives and Goals

The primary goal and supporting objectives of this data analysis are as follows:

* Goal: Gain a better understanding of hospital readmissions.
  + Objective 1: Determine which features predict with the most statistical significance whether a patient will be readmitted to the hospital.
  + Objective 2: Develop a logistic model of hospital readmissions.
  + Objective 3: Identify real-world scenarios which could prevent hospital readmissions.

# B: Method Justification

## B1: Summary of Assumptions

There are six assumptions of multiple regression modeling (Bobbitt, 2020):

1. The dependent variable must only have two possible outcomes.
2. Each observation must be independent from one another.
3. There must not be multicollinearity between the independent variables.
4. No extreme outliers exist.
5. Each independent variable has a linear relationship with the logit of the dependent variable.
6. There must be enough observations in the dataset to make valid conclusions.

## B2: Tool Benefits

To complete all phases of the analysis Python was used. For data cleaning, the numpy and pandas libraries allow Structured Query Language style operations. These make data cleaning far easier than using a tool like spreadsheet software, which can be problematic with large data sets.

For the more advanced operations like data wrangling, and the analysis itself python has available statistical libraries like scipy, statsmodels, and scikit-learn. Between the three of them, just about any analysis is not only possible, but simple to implement. Just like with the data cleaning process, python is superior to spreadsheet software. To perform a regression analysis in such software, many intermediate steps must be performed. This is contrary to the one-line statements available in the python libraries.

To produce visualizations, python was also used with the matplotlib and seaborn libraries. There are several benefits to this, with the top being simplifying the process. Having already completed the other two portions using python, it was simpler to use python for everything. The data was readily available and matplotlib is an excellent visualization library.

## B3: Appropriate Technique

The question being researched aims to discover which factors best predict whether a patient will be readmitted to the hospital. Regression analysis is used to understand the relationship between a single response variable and many predictor variables. More specifically, multiple logistic regression analysis is designed to handle a response variable that is categorical rather than numeric. Because the response variable, ReAdmis, is categorical, multiple logistic regression is the most practical analysis for the question being asked.

# C: Data Preparations

## C1: Data Goals

The goals of the data preparation process were to clean, explore, and wrangle the data in preparation for multiple logistic regression analysis.

During the cleaning phase, the data set will be checked for duplicate observations, missing values, and outliers. If any duplicate observations are identified, they will be removed. If any missing values are identified, they will either be removed or imputed. If any outliers are identified, they will be investigated individually to determine if any further action is required.

During the exploration phase, univariate and bivariate analyses will be performed. Each variable will be assessed individually to understand its respective distribution. Then, each predictor variable will be assessed with the response variable to gain a better understanding of the individual relationships.

During the last phase, data wrangling, three activities will take place. First, all categorical variables will be re-coded to numeric either by mapping values to the categories or by creating dummy variables to represent the categories. The second activity will be scaling the numeric variables. This will ensure the magnitude of each variable does not skew the model. The last step in the data wrangling process will be to check the variance influence factors of each of the variables. This is to ensure that no multicollinearity exists within the dataset. If any variables are found to have a VIF of 10 or more, one will be removed to reduce their effect on the model.

## C2: Summary Statistics

Among the quantitative variables, Income has the widest range while vitD\_supp has the smallest range. TotalCharge and Additional\_charges are also variables with quite large numeric values. These will need to be scaled during the data wrangling phase to prevent them from skewing the regression analysis.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Variable** | **count** | **mean** | **std** | **min** | **25%** | **50%** | **75%** | **max** |
| Lat | 10000 | 38.7511 | 5.4031 | 17.9672 | 35.2551 | 39.4194 | 42.0442 | 70.5610 |
| Lng | 10000 | -91.2431 | 15.2060 | -174.2097 | -97.3530 | -88.3972 | -80.4381 | -65.2902 |
| Children | 10000 | 2.0972 | 2.1637 | 0.0000 | 0.0000 | 1.0000 | 3.0000 | 10.0000 |
| Age | 10000 | 53.5117 | 20.6385 | 18.0000 | 36.0000 | 53.0000 | 71.0000 | 89.0000 |
| Income | 10000 | 40490.4952 | 28521.1533 | 154.0800 | 19598.7750 | 33768.4200 | 54296.4025 | 207249.1000 |
| VitD\_levels | 10000 | 17.9643 | 2.0172 | 9.8065 | 16.6264 | 17.9511 | 19.3480 | 26.3944 |
| Doc\_visits | 10000 | 5.0122 | 1.0457 | 1.0000 | 4.0000 | 5.0000 | 6.0000 | 9.0000 |
| Full\_meals\_eaten | 10000 | 1.0014 | 1.0081 | 0.0000 | 0.0000 | 1.0000 | 2.0000 | 7.0000 |
| vitD\_supp | 10000 | 0.3989 | 0.6285 | 0.0000 | 0.0000 | 0.0000 | 1.0000 | 5.0000 |
| Initial\_days | 10000 | 34.4553 | 26.3093 | 1.0020 | 7.8962 | 35.8362 | 61.1610 | 71.9815 |
| TotalCharge | 10000 | 5312.1728 | 2180.3938 | 1938.3121 | 3179.3740 | 5213.9520 | 7459.6998 | 9180.7280 |
| Additional\_charges | 10000 | 12934.5286 | 6542.6015 | 3125.7030 | 7986.4878 | 11573.9777 | 15626.4900 | 30566.0700 |

The qualitative or categorical variables are largely made up of yes and no values. These will lend themselves well to encoding during the wrangling phase. Area, Marital, Gender, Initial\_admin, Complication\_risk, and Services all have three or more levels which will require different handling than the yes and no values.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Variable** | **count** | **unique** | **top** | **freq** | **categories** |
| ReAdmis | 10000 | 2 | No | 6331 | No, Yes |
| HighBlood | 10000 | 2 | No | 5910 | Yes, No |
| Stroke | 10000 | 2 | No | 8007 | No, Yes |
| Overweight | 10000 | 2 | Yes | 7094 | No, Yes |
| Arthritis | 10000 | 2 | No | 6426 | Yes, No |
| Diabetes | 10000 | 2 | No | 7262 | Yes, No |
| Hyperlipidemia | 10000 | 2 | No | 6628 | No, Yes |
| BackPain | 10000 | 2 | No | 5886 | Yes, No |
| Anxiety | 10000 | 2 | No | 6785 | Yes, No |
| Allergic\_rhinitis | 10000 | 2 | No | 6059 | Yes, No |
| Reflux\_esophagitis | 10000 | 2 | No | 5865 | No, Yes |
| Asthma | 10000 | 2 | No | 7107 | Yes, No |
| Area | 10000 | 3 | Rural | 3369 | Suburban, Urban, Rural |
| Marital | 10000 | 5 | Widowed | 2045 | Divorced, Married, Widowed, Never Married, Separated |
| Gender | 10000 | 3 | Female | 5018 | Male, Female, Nonbinary |
| Initial\_admin | 10000 | 3 | Emergency Admission | 5060 | Emergency Admission, Elective Admission, Observation Admission |
| Complication\_risk | 10000 | 3 | Medium | 4517 | Medium, High, Low |
| Services | 10000 | 4 | Blood Work | 5265 | Blood Work, Intravenous, CT Scan, MRI |

## C3: Steps to Prepare the Data

The python code provided below was used to prepare the medical data for final analysis. This and all other code used for the analysis can be found attached to the submission as task2.py.

To configure the coding environment for the rest of the analysis, several libraries were imported, an output directory was created, and the data set was loaded into a pandas DataFrame.

import os  
import numpy as np  
import pandas as pd  
import matplotlib.patches as mpatches  
import matplotlib.pyplot as plt  
import seaborn as sns  
from scipy import stats  
from sklearn.preprocessing import StandardScaler  
from statsmodels.stats.outliers\_influence import variance\_inflation\_factor  
from statsmodels.formula.api import logit  
from statsmodels.genmod.families import Binomial  
from statsmodels.genmod.generalized\_linear\_model import GLM  
from statsmodels.tools.tools import add\_constant  
from itertools import product

os.makedirs('./output', exist\_ok=True)

medical\_clean = pd.read\_csv(r"C:\Users\user\OneDrive\Documents\Education\Western Govenors University"  
 r"\MS - Data Analytics\D208 - Predictive Modeling\medical\_clean.csv")

Next, the dependent variable, ReAdmis, and the independent variables were selected. The independent variables included Lat, Lng, Children, Age, Income, VitD\_levels, Doc\_visits, Full\_meals\_eaten, vitD\_supp, Initial\_days, TotalCharge, Additional\_charges, HighBlood, Stroke, Overweight, Arthritis, Diabetes, Hyperlipidemia, BackPain, Anxiety, Allergic\_rhinitis, Reflux\_esophagitis, Asthma, Area, Marital, Gender, Initial\_admin, Complication\_risk, and Services.

columns = ['ReAdmis', 'Lat', 'Lng', 'Children', 'Age', 'Income', 'VitD\_levels', 'Doc\_visits', 'Full\_meals\_eaten',  
 'vitD\_supp', 'Initial\_days', 'TotalCharge', 'Additional\_charges', 'HighBlood', 'Stroke', 'Overweight',  
 'Arthritis', 'Diabetes', 'Hyperlipidemia', 'BackPain', 'Anxiety', 'Allergic\_rhinitis', 'Reflux\_esophagitis',  
 'Asthma', 'Area', 'Marital', 'Gender', 'Initial\_admin', 'Complication\_risk', 'Services']  
df = medical\_clean[columns]

Once the variables of interest were selected, the data was cleaned. This was accomplished by checking for duplicate observations, missing values, and outliers.

print(f'Duplication check:\n\tNumber of duplicated rows: {df.duplicated().sum()}')  
print(f'Null detection:\n{df.isna().sum()}')  
  
num\_cols = [column for column in columns if df[column].dtype in ['float64', 'int64']]  
df\_num = df[num\_cols]  
df\_zscores = df\_num.apply(stats.zscore)  
df\_outliers = df\_zscores.apply(lambda x: (x > 3) | (x < -3))  
print(f'Outlier counts:\n{df\_outliers.sum().to\_string()}')

There were no duplicate observations or missing values. However, there were several variables with outliers. Each outlier, or set of outliers where it made sense, was investigated to ensure the data contained no errors. The Lat and Lng outliers were reviewed together since they were somewhat meaningless without one another.

When the Lat and Lng outliers were grouped by State, it became apparent that the outliers represented Alaska, Hawaii, and Puerto Rico. All three of these represent locations not part of the continental United States. Since they are still part of the United States as a whole, they were not excluded from the final analysis.

location\_outliers = medical\_clean.loc[df\_outliers.Lat | df\_outliers.Lng, ['State', 'Lat', 'Lng']]  
location\_outliers.columns = ['State', 'Lat\_mean', 'Lng\_mean']  
location\_outlier\_groups = location\_outliers.groupby("State", as\_index=False).mean()  
print(f'Location Outliers:\n{location\_outlier\_groups.to\_string()}')

The Children outliers were investigated next.

children\_outliers = medical\_clean.loc[df\_outliers.Children, ['Children', 'Customer\_id']]  
children\_outliers.columns = ['Children', 'count']  
children\_outliers = children\_outliers.groupby("Children", as\_index=False).count()  
print(f'Children Outliers:\n{children\_outliers.to\_string()}')

While having nine or 10 children was certainly not the norm in this data, the values did not appear to be erroneous. For that reason, the observations were not excluded.

Next, the Income outliers were reviewed along with the Job variable for context.

income\_outliers = medical\_clean.loc[df\_outliers.Income, ['Job', 'Income']]  
income\_outliers.columns = ['Job', 'Income\_mean']  
income\_outliers = income\_outliers.groupby("Job", as\_index=False).mean(). \  
 sort\_values("Income\_mean", ascending=False)  
print(f'Income Outliers:\n{income\_outliers.iloc[:10].to\_string()}')

Most of the job titles were those known to receive well above the average compensation (e.g., Chief Executive Officer, Software engineer, etc.). It would not make sense to exclude completely legitimate income values.

The VitD\_levels and vitD\_supp outliers were investigated together since vitamin D supplements get prescribed for vitamin D deficiency.

vitd\_outliers = df.loc[:, ['VitD\_levels', 'vitD\_supp']]  
vitd\_outliers['VitD\_levels\_outlier'] = df\_outliers.VitD\_levels  
vitd\_outliers['vitD\_supp\_outlier'] = df\_outliers.vitD\_supp  
vitd\_outliers = vitd\_outliers.groupby(['VitD\_levels\_outlier', 'vitD\_supp\_outlier'], as\_index=False). \  
 mean().sort\_values(['VitD\_levels\_outlier', 'vitD\_supp\_outlier'], ascending=False)  
print(f'Vitamin D Outliers\n{vitd\_outliers.to\_string()}')

In both the VitD\_levels and vitD\_supp outliers, the mean vitamin D levels and mean vitamin D supplements were above the mean for those who were not outliers for either variable. Since the goal of this analysis was to determine what impacts the length of inpatient admissions, it would not be logical to exclude variables directly relevant to the patient's health.

The Doc\_visits outliers were investigated next.

doc\_visits\_outliers = medical\_clean.loc[df\_outliers.Doc\_visits, ['Doc\_visits', 'Customer\_id']]  
doc\_visits\_outliers.columns = ['Doc\_visits', 'count']  
doc\_visits\_outliers = doc\_visits\_outliers.groupby('Doc\_visits', as\_index=False).count()  
print(f'Doc\_visits Outliers:\n{doc\_visits\_outliers.to\_string()}')

There were only eight outliers in the Doc\_visits variable, to begin with. The outliers were one and nine doctor visits. Depending on how long the patients were hospitalized, both could be completely valid. None of them were excluded.

The last set of outliers investigated was Full\_meals\_eaten.

full\_meals\_eaten\_outliers = medical\_clean.loc[df\_outliers.Full\_meals\_eaten, ['Full\_meals\_eaten', 'Customer\_id']]  
full\_meals\_eaten\_outliers.columns = ['Full\_meals\_eaten', 'count']  
full\_meals\_eaten\_outliers = full\_meals\_eaten\_outliers.groupby('Full\_meals\_eaten', as\_index=False).count()  
print(f'Full\_meals\_eaten Outliers:\n{full\_meals\_eaten\_outliers.to\_string()}')

The values for the Full\_meals\_eaten outliers ranged from five to seven with almost three-quarters of them having a value of five. Considering the length of some of the stays in this data set, it was surprising that the values were not higher. A total of seven full meals seemed low for someone hospitalized for 60 days. Without knowing the medical conditions, for instance, whether the patient was conscience during their stay, it is not possible to determine whether the outliers were erroneous. To not unintentionally skew the data, the outliers were not excluded.

Once the data cleaning process was complete, exploratory data analysis was performed. Each of the selected variables in the cleaned data was plotted to view their distributions.

s, e = 0, 6  
grid = []  
for i in range(int(np.ceil(df.shape[1] / e))):  
 grid.append([x for x in df.columns.values[s:e]])  
 s = e  
 e += 6  
fig, ax = plt.subplots(len(grid), 6, figsize=(15, 8))  
fig.set\_tight\_layout(True)  
fig.suptitle('Univariate Analysis')  
for r, row in enumerate(grid):  
 for c, col in enumerate(row):  
 if df[col].dtype in ('float64', 'int64'):  
 ax[r, c].hist(df[col])  
 ax[r, c].vlines(df[col].median(), 0, 4000, color='black', label='median')  
 ax[r, c].vlines(df[col].mean(), 0, 4000, color='red', linestyles='--', label='mean')  
 else:  
 df\_grp = df.groupby(col).count()  
 if col == 'Initial\_admin':  
 df\_grp.index = df\_grp.index.str.replace(' Admission', '')  
 ax[r, c].bar(df\_grp.index, df\_grp.values[:, 0])  
 plt.setp(ax[r, c].get\_xticklabels(), rotation=20, ha='right', rotation\_mode='anchor')  
 ax[r, c].set\_xlabel(col)  
plt.show()

All of the variables were displayed in a single figure, each having its subplot. Quantitative variables were displayed as histograms with solid black lines marking the median and dashed red lines marking the mean. The addition of median and mean lines allowed the assessment of skew in each variable. Qualitative variables were displayed as bar charts.

Following the univariate analysis, each independent variable was plotted against the dependent variable, ReAdmis, to assess their relationships.

s, e = 0, 6  
grid = []  
for i in range(int(np.ceil(len(df.columns) / e))):  
 grid.append([x for x in df.columns[1:].values[s:e]])  
 s = e  
 e += 6  
fig, ax = plt.subplots(len(grid), 6, sharey='row', figsize=(15, 8))  
fig.set\_tight\_layout(True)  
fig.suptitle('Bivariate Analysis')  
for r, row in enumerate(grid):  
 for c, col in enumerate(row):  
 if df[col].dtype in ('float64', 'int64'):  
 readmis\_yes = df.loc[df.ReAdmis.eq('Yes'), col].values  
 readmis\_no = df.loc[df.ReAdmis.eq('No'), col].values  
 ax[r, c].boxplot([readmis\_yes, readmis\_no], labels=['Yes', 'No'], vert=False, sym='', widths=0.5)  
 ax[r, c].set\_xlabel(col)  
 if c == 0:  
 ax[r, c].set\_ylabel('ReAdmis')  
 else: *# True* readmis\_yes = df.loc[df.ReAdmis.eq('Yes'), ['ReAdmis', col]].groupby(col).count()['ReAdmis'].values  
 readmis\_no = df.loc[df.ReAdmis.eq('No'), ['ReAdmis', col]].groupby(col).count()['ReAdmis'].values  
 labels = df.loc[:, col].str.replace(' Admission', '').unique()  
 x = np.arange(len(labels))  
 width = 0.35  
 bar1 = ax[r, c].bar(x - width/2, readmis\_yes, width, label='ReAdmis=Yes')  
 bar2 = ax[r, c].bar(x + width/2, readmis\_no, width, label='ReAdmis=No')  
 ax[r, c].set\_xlabel(col)  
 ax[r, c].set\_xticks(x, labels)  
 if c == 0:  
 ax[r, c].set\_ylabel('count')  
 plt.setp(ax[r, c].get\_xticklabels(), rotation=20, ha='right', rotation\_mode='anchor')  
blue = mpatches.Patch(color='#1f77b4', label='Yes')  
orange = mpatches.Patch(color='#ff7f0e', label='No')  
ax[4, 5].legend(title='ReAdmis', handles=[blue, orange], loc='center')  
plt.show()

Once again, all of the plots were displayed in a single figure using subplots. Quantitative independent variables were plotted against ReAdmis using box plots. Qualitative independent variables were plotted against ReAdmis using color-coded bar charts.

Next, the data wrangling process took place. This consisted of recoding categorical variables, scaling numeric variables, and removing variables causing multicollinearity in the data set.

During the re-coding process, Yes and No values were replaced with 1 and 0, respectively. The remaining categorical variables were replaced with dummy variables. One new variable was produced for every category minus one to prevent multicollinearity in the data. The spaces in the newly created column names were also replaced with underscores to allow a more programmatic approach during the model reduction process.

After re-coding the categorical variables, the numeric variables were scaled to prevent the magnitude of each variable from skewing the analysis.

*# Recode categorical variables*num\_cols = [col for col in df.columns if df[col].dtype in ('int64', 'float64')]  
df = df.replace({'Yes': 1, 'No': 0})  
cat\_cols = [col for col in df.columns if df[col].dtype not in ('int64', 'float64')]  
df = pd.get\_dummies(df, columns=cat\_cols, drop\_first=True)  
  
*# Fix column names*df.columns = [col.replace(' ', '\_') for col in df.columns]  
print(f'Recoded data:\n{df.head().to\_string()}')

The last step before building the model was checking for multicollinearity since it was one of the assumptions of multiple linear regression.

x = df.loc[:, df.columns[1:]]  
vif\_data = pd.DataFrame()  
vif\_data['feature'] = x.columns  
vif\_data['vif'] = [variance\_inflation\_factor(x.values, i) for i in range(x.shape[1])]  
print(f'Variance Inflation Factor:\n{vif\_data.to\_string()}')

scaler = StandardScaler()  
df.loc[:, df\_num.columns] = scaler.fit\_transform(df\_num)  
print(f'Scaled Data:\n{df.head().to\_string()}')

To check for multicollinearity, the variance influence factor for each variable was assessed.

*# Remove high VIFs*x.pop('TotalCharge')  
x.pop('Additional\_charges')  
  
*# Collinearity check again*vif\_data = pd.DataFrame()  
vif\_data['feature'] = x.columns  
vif\_data['vif'] = [variance\_inflation\_factor(x.values, i) for i in range(x.shape[1])]  
print(f'Variance Inflation Factor:\n{vif\_data.to\_string()}')

Once the highest two values, Initial\_days at 2555.87 and Additional\_charges at 16.32, were removed and the VIF was rechecked, no additional multicollinearity was present.

At this time, the final dataset was saved.

*# Save prepared data set*df.to\_csv('./output2/prepared\_data\_set.csv')

## C4: Visualizations

Because all of the variables were visualized in a single plot, making them difficult to view as figures in a document such as this, they were also included as attachments with the task submission. The attached files are titled univariate\_analysis.png and bivariate\_analysis.png.

Diagram

Description automatically generated

Diagram

Description automatically generated

## C5: Prepared Data Set

The prepared dataset has been included with the task submission as prepared\_data\_set.csv.

# D: Model Comparison

## D1: Initial Model

Graphical user interface, table

Description automatically generated

## D2: Justification of Model Reduction

Variable selection will be performed using each variable’s p-value. One at a time, the variable with the highest p-value will be removed until none above .05 remains. This will ensure that only variables whose impact on ReAdmis is statistically significant will remain. Overall model evaluation will be assessed using Pseudo R-squared and LLR p-value, as well as the output of the confusion matrix.

## D3: Reduced Multiple Regression Model

Table

Description automatically generated

# E: Model Analysis

## E1: Model Comparison

Variable selection was performed by looping over a user-defined function. The function looked at the p-values for each independent variable and removed the variable with the highest p-value. It then rebuilt and fit the model before returning it. Each time the model was returned the new p-values were checked. If any of them were over 0.05, the function was called again until no more insignificant independent variables remained. Once the model was reduced, the reduced model’s assumptions were evaluated, resulting in the removal of the Children variable from the final reduced model. The gross and reduced models were then compared using the Pseudo R-squared and the LLR p-value. Referencing the output from each model, the LLR p-value did not change, but the pseudo R-squared dropped slightly. Referencing the confusion matrix and associated metrics from each model, the reduced model had slightly lower accuracy and sensitivity, but slightly higher specificity. Because the reduced model produced such similar performance with fewer variables, the reduced model was determined to be superior.

|  |  |  |
| --- | --- | --- |
| **Metric** | **Gross Model** | **Reduced Model** |
| Pseudo R-squared | 0.9501 | 0.9490 |
| LLR p-value | 0.000 | 0.000 |

Gross Model Confusion Matrix:

|  |  |  |
| --- | --- | --- |
|  | **Predicted False** | **Predicted True** |
| **Actual False** | 6261 | 70 |
| **Actual True** | 64 | 3605 |

Reduced Model Confusion Matrix:

|  |  |  |
| --- | --- | --- |
|  | **Predicted False** | **Predicted True** |
| **Actual False** | 6261 | 70 |
| **Actual True** | 66 | 3603 |

Metrics:

|  |  |  |
| --- | --- | --- |
| **Metric** | **Gross Model** | **Reduced Model** |
| Accuracy | 0.9866 | 0.9864 |
| Sensitivity | 0.9825565549195966 | 0.982011447260834 |
| Specificity | 0.9889432948981204 | 0.9889432948981204 |

## E2: Output and Calculations

Removing Doc\_visits with a p-value of 0.965562637785445

Optimization terminated successfully.

Current function value: 0.032781

Iterations 14

Removing Services\_Intravenous with a p-value of 0.9551157868887363

Optimization terminated successfully.

Current function value: 0.032782

Iterations 14

Removing Marital\_Separated with a p-value of 0.7842293333851279

Optimization terminated successfully.

Current function value: 0.032785

Iterations 14

Removing Area\_Suburban with a p-value of 0.7761446289247053

Optimization terminated successfully.

Current function value: 0.032789

Iterations 14

Removing Area\_Urban with a p-value of 0.799346580416658

Optimization terminated successfully.

Current function value: 0.032793

Iterations 14

Removing Gender\_Nonbinary with a p-value of 0.6518865683946828

Optimization terminated successfully.

Current function value: 0.032803

Iterations 14

Removing Age with a p-value of 0.595505595206742

Optimization terminated successfully.

Current function value: 0.032817

Iterations 14

Removing VitD\_levels with a p-value of 0.5836933168822631

Optimization terminated successfully.

Current function value: 0.032832

Iterations 14

Removing Income with a p-value of 0.5616442394730123

Optimization terminated successfully.

Current function value: 0.032849

Iterations 14

Removing Full\_meals\_eaten with a p-value of 0.5667840913156185

Optimization terminated successfully.

Current function value: 0.032865

Iterations 14

Removing vitD\_supp with a p-value of 0.46763529381209046

Optimization terminated successfully.

Current function value: 0.032892

Iterations 14

Removing Gender\_Male with a p-value of 0.4574724569925702

Optimization terminated successfully.

Current function value: 0.032919

Iterations 14

Removing Marital\_Widowed with a p-value of 0.3611008523802378

Optimization terminated successfully.

Current function value: 0.032961

Iterations 14

Removing Lng with a p-value of 0.3904379004757337

Optimization terminated successfully.

Current function value: 0.032998

Iterations 14

Removing Marital\_Married with a p-value of 0.30767558613491497

Optimization terminated successfully.

Current function value: 0.033050

Iterations 14

Removing Marital\_Never\_Married with a p-value of 0.30184466932987797

Optimization terminated successfully.

Current function value: 0.033104

Iterations 14

Removing Overweight with a p-value of 0.24171563911595595

Optimization terminated successfully.

Current function value: 0.033173

Iterations 14

Removing Lat with a p-value of 0.1999602447706358

Optimization terminated successfully.

Current function value: 0.033255

Iterations 14

Verifying model assumptions:

Model assumption 1: binary response variable:

ReAdmis unique values: 2

Model assumption 2: independent observations:

Data contains only unique patients

Model assumption 3: no multicollinearity

Variance Inflation Factor:

feature vif

0 Children 1.001823

1 TotalCharge 1.023971

2 HighBlood 1.596162

3 Stroke 1.219590

4 Arthritis 1.490174

5 Diabetes 1.335211

6 Hyperlipidemia 1.442117

7 BackPain 1.609960

8 Anxiety 1.417876

9 Allergic\_rhinitis 1.552548

10 Reflux\_esophagitis 1.599088

11 Asthma 1.364331

12 Initial\_admin\_Emergency\_Admission 2.334365

13 Initial\_admin\_Observation\_Admission 1.638303

14 Complication\_risk\_Low 1.487435

15 Complication\_risk\_Medium 2.043917

16 Services\_CT\_Scan 1.135550

17 Services\_MRI 1.042598

Model assumption 4: no outliers

Several outliers were detected and kept during data cleaning

Model assumption 5: linearity

Generalized Linear Model Regression Results

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

Dep. Variable: ReAdmis No. Observations: 10000

Model: GLM Df Residuals: 9995

Model Family: Binomial Df Model: 4

Link Function: Logit Scale: 1.0000

Method: IRLS Log-Likelihood: -977.32

Date: Wed, 12 Oct 2022 Deviance: 1954.6

Time: 23:26:00 Pearson chi2: 2.84e+03

No. Iterations: 14 Pseudo R-squ. (CS): 0.6734

Covariance Type: nonrobust

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

coef std err z P>|z| [0.025 0.975]

-------------------------------------------------------------------------------------------------

Children 0.3535 0.120 2.949 0.003 0.119 0.588

TotalCharge -0.0233 0.048 -0.486 0.627 -0.117 0.071

Children\_x\_Log\_Children -0.1589 0.055 -2.892 0.004 -0.267 -0.051

TotalCharge\_x\_Log\_TotalCharge 0.0029 0.005 0.603 0.547 -0.007 0.012

const -18.1670 33.699 -0.539 0.590 -84.216 47.882

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

No linear relationship exists between ReAdmis and Children\_x\_Log\_Children

Optimization terminated successfully.

Current function value: 0.033490

Iterations 14

Children variable removed from reduced model

Linearity was evaluated using the Box-Tidwell test, which required the following calculations (Leung, 2021):

Model assumption 6: sufficient sample size

Min sample per group: 10, Variables: 17, Min Probability: 0.3669

Minimum sample size: 463.34150994821476

Actual sample size: 10000

Sample size was evaluated using the following formula (Bobbitt, 2020):

Gross model confusion matrix:

predicted false predicted true

actual false 6261.0 70.0

actual true 64.0 3605.0

Reduced model confusion matrix:

predicted false predicted true

actual false 6261.0 70.0

actual true 66.0 3603.0

Model metrics were calculated using the following formulas:

Gross Model Metrics:

accuracy: 0.9866

sensitivity: 0.9825565549195966

specificity: 0.9889432948981204

Reduced Model Metrics:

accuracy: 0.9864

sensitivity: 0.982011447260834

specificity: 0.9889432948981204Predictions:

count mean std min 25% 50% 75% max

TotalCharge 458752.0 5208.902824 2180.396214 1.938312e+03 3.028509e+03 5.208903e+03 7389.296662 8479.49358

HighBlood 458752.0 0.500000 0.500001 0.000000e+00 0.000000e+00 5.000000e-01 1.000000 1.00000

Stroke 458752.0 0.500000 0.500001 0.000000e+00 0.000000e+00 5.000000e-01 1.000000 1.00000

Arthritis 458752.0 0.500000 0.500001 0.000000e+00 0.000000e+00 5.000000e-01 1.000000 1.00000

Diabetes 458752.0 0.500000 0.500001 0.000000e+00 0.000000e+00 5.000000e-01 1.000000 1.00000

Hyperlipidemia 458752.0 0.500000 0.500001 0.000000e+00 0.000000e+00 5.000000e-01 1.000000 1.00000

BackPain 458752.0 0.500000 0.500001 0.000000e+00 0.000000e+00 5.000000e-01 1.000000 1.00000

Anxiety 458752.0 0.500000 0.500001 0.000000e+00 0.000000e+00 5.000000e-01 1.000000 1.00000

Allergic\_rhinitis 458752.0 0.500000 0.500001 0.000000e+00 0.000000e+00 5.000000e-01 1.000000 1.00000

Reflux\_esophagitis 458752.0 0.500000 0.500001 0.000000e+00 0.000000e+00 5.000000e-01 1.000000 1.00000

Asthma 458752.0 0.500000 0.500001 0.000000e+00 0.000000e+00 5.000000e-01 1.000000 1.00000

Initial\_admin\_Emergency\_Admission 458752.0 0.500000 0.500001 0.000000e+00 0.000000e+00 5.000000e-01 1.000000 1.00000

Initial\_admin\_Observation\_Admission 458752.0 0.500000 0.500001 0.000000e+00 0.000000e+00 5.000000e-01 1.000000 1.00000

Complication\_risk\_Low 458752.0 0.500000 0.500001 0.000000e+00 0.000000e+00 5.000000e-01 1.000000 1.00000

Complication\_risk\_Medium 458752.0 0.500000 0.500001 0.000000e+00 0.000000e+00 5.000000e-01 1.000000 1.00000

Services\_CT\_Scan 458752.0 0.500000 0.500001 0.000000e+00 0.000000e+00 5.000000e-01 1.000000 1.00000

Services\_MRI 458752.0 0.500000 0.500001 0.000000e+00 0.000000e+00 5.000000e-01 1.000000 1.00000

ReAdmis 458752.0 0.302747 0.452119 1.542043e-46 2.100708e-27 2.525223e-12 0.999671 1.00000

Graphical user interface

Description automatically generated with medium confidence

Chart, line chart

Description automatically generated

## E3: Code

*# Gross logistic model*formula = f'ReAdmis ~ {" + ".join([x for x in df.columns[1:]])}'  
model1 = logit(formula, df).fit()  
print(f'Gross Logistic Model:\n{model1.summary()}\n')

*# Feature Selection*def reduce\_model(model: logit, data: pd.DataFrame) -> logit:  
 features = model.params.index.to\_list() *# Get the list of features* features.pop(0) *# Remove the intercept from the list of features* highest\_p = model.pvalues.argmax() - 1 *# Get the index of the highest p-value* print(f'Removing {features[highest\_p]} with a p-value of {float(model.pvalues[highest\_p + 1])}')  
 features.pop(highest\_p) *# Remove the item with the highest p-value* formula = f'ReAdmis ~ {" + ".join(features)}' *# Rewrite the formula* model = logit(formula=formula, data=data).fit() *# Rebuild the model* return model  
  
model2 = model1  
while model2.pvalues.max() >= 0.05:  
 model2 = reduce\_model(model2, df)

*# Model Assumptions*print(f'Verifying model assumptions:\n\n'  
 f'Model assumption 1: binary response variable:\n'  
 f'ReAdmis unique values: {df.ReAdmis.nunique()}')

print(f'Model assumption 2: independent observations:\n'  
 f'Data contains only unique patients')

print(f'Model assumption 3: no multicollinearity')  
x = df.loc[:, model2.params.index.to\_list()[1:]]  
vif\_data = pd.DataFrame()  
vif\_data['feature'] = x.columns  
vif\_data['vif'] = [variance\_inflation\_factor(x.values, i) for i in range(x.shape[1])]  
print(f'Variance Inflation Factor:\n{vif\_data.to\_string()}')

print(f'Model assumption 4: no outliers\n'  
 f'Several outliers were detected and kept during data cleaning')

print(f'Model assumption 5: linearity')  
*# Box-Tidwell test*test\_data = medical\_clean.loc[:, ['Children', 'TotalCharge']] test\_data.loc[:, 'Children\_x\_Log\_Children'] = test\_data.Children \* np.log1p(test\_data.Children)  
test\_data.loc[:, 'TotalCharge\_x\_Log\_TotalCharge'] = test\_data.TotalCharge \* np.log1p(test\_data.TotalCharge)  
test\_data = add\_constant(test\_data, prepend=False)  
bt\_results = GLM(df.ReAdmis, test\_data, family=Binomial()).fit()  
print(bt\_results.summary())  
print('No linear relationship exists between ReAdmis and Children\_x\_Log\_Children')  
params = model2.params.index.to\_list()  
params.pop(params.index('Intercept'))  
params.pop(params.index('Children'))  
formula = f'ReAdmis ~ ' + ' + '.join(params)  
model2 = logit(formula, df).fit()  
print('Children variable removed from reduced model')

print('Model assumption 6: sufficient sample size')  
min\_sample = 10  
n\_vars = len(params)  
min\_prob = df[df.ReAdmis.eq(1)].shape[0] / df.shape[0]  
print(f'Min sample per group: {min\_sample}, Variables: {n\_vars}, Min Probability: {min\_prob}')  
print(f'Minimum sample size: {(min\_sample \* n\_vars) / min\_prob}')  
print(f'Actual sample size: {df.shape[0]}')

*# Confusion Matrix*index = ['actual false', 'actual true']  
columns = ['predicted false', 'predicted true']  
conf\_matrix1 = pd.DataFrame(model1.pred\_table(), index=index, columns=columns)  
conf\_matrix2 = pd.DataFrame(model2.pred\_table(), index=index, columns=columns)  
print(f'Gross model confusion matrix:\n{conf\_matrix1.to\_string()}\n')  
print(f'Reduced model confusion matrix:\n{conf\_matrix2.to\_string()}')

*# Gross Model Metrics*tn1 = conf\_matrix1.loc['actual false', 'predicted false']  
fp1 = conf\_matrix1.loc['actual false', 'predicted true']  
fn1 = conf\_matrix1.loc['actual true', 'predicted false']  
tp1 = conf\_matrix1.loc['actual true', 'predicted true']  
accuracy1 = (tn1 + tp1) / (tn1 + fn1 + fp1 + tp1)  
sensitivity1 = tp1 / (fn1 + tp1)  
specificity1 = tn1 / (tn1 + fp1)  
print(f'Gross Model Metrics:\n'  
 f'\taccuracy: {accuracy1}\n'  
 f'\tsensitivity: {sensitivity1}\n'  
 f'\tspecificity: {specificity1}\n')  
  
*# Reduced Model Metrics*tn2 = conf\_matrix2.loc['actual false', 'predicted false']  
fp2 = conf\_matrix2.loc['actual false', 'predicted true']  
fn2 = conf\_matrix2.loc['actual true', 'predicted false']  
tp2 = conf\_matrix2.loc['actual true', 'predicted true']  
accuracy2 = (tp2 + tn2) / (tn2 + fn2 + fp2 + tp2)  
sensitivity2 = tp2 / (fn2 + tp2)  
specificity2 = tn2 / (tn2 + fp2)  
print(f'Reduced Model Metrics:\n'  
 f'\taccuracy: {accuracy2}\n'  
 f'\tsensitivity: {sensitivity2}\n'  
 f'\tspecificity: {specificity2}')

*# Make data*TotalCharge = np.arange(medical\_clean.TotalCharge.min(),  
 medical\_clean.TotalCharge.max() + 1,  
 medical\_clean.TotalCharge.std() / 2)  
HighBlood = np.array([1, 0], int)  
Stroke = np.array([1, 0], int)  
Arthritis = np.array([1, 0], int)  
Diabetes = np.array([1, 0], int)  
Hyperlipidemia = np.array([1, 0], int)  
BackPain = np.array([1, 0], int)  
Anxiety = np.array([1, 0], int)  
Allergic\_rhinitis = np.array([1, 0], int)  
Reflux\_esophagitis = np.array([1, 0], int)  
Asthma = np.array([1, 0], int)  
Initial\_admin\_Emergency\_Admission = np.array([1, 0], int)  
Initial\_admin\_Observation\_Admission = np.array([1, 0], int)  
Complication\_risk\_Low = np.array([1, 0], int)  
Complication\_risk\_Medium = np.array([1, 0], int)  
Services\_CT\_Scan = np.array([1, 0], int)  
Services\_MRI = np.array([1, 0], int)  
p = product(TotalCharge, HighBlood, Stroke, Arthritis, Diabetes, Hyperlipidemia, BackPain, Anxiety, Allergic\_rhinitis,  
 Reflux\_esophagitis, Asthma, Initial\_admin\_Emergency\_Admission, Initial\_admin\_Observation\_Admission,  
 Complication\_risk\_Low, Complication\_risk\_Medium, Services\_CT\_Scan, Services\_MRI)  
prediction\_data = pd.DataFrame(p, columns=model2.params.index.to\_list()[1:])  
*# Scale TotalCharges*scaler = StandardScaler()  
prediction\_data.loc[:, ['TotalCharge', 'HighBlood\_']] = \  
 scaler.fit\_transform(prediction\_data.loc[:, ['TotalCharge', 'HighBlood']])  
prediction\_data.drop('HighBlood\_', axis=1, inplace=True)  
*# Make Predictions*prediction\_data['ReAdmis'] = model2.predict(prediction\_data)  
*# Unscale TotalCharges*prediction\_data.loc[:, ['TotalCharge', 'HighBlood\_']] = \  
 scaler.inverse\_transform(prediction\_data.loc[:, ['TotalCharge', 'HighBlood']])  
prediction\_data.drop('HighBlood\_', axis=1, inplace=True)  
print(f'Predictions:\n{prediction\_data.describe().T.to\_string()}')

*# Plot Predictions*

sns.regplot(x=medical\_clean.TotalCharge, y=df.ReAdmis, logistic=True, label='observations')  
sns.scatterplot(x=prediction\_data.TotalCharge, y=np.round(prediction\_data.ReAdmis), color='red', label='predictions')  
plt.title('Gross Logistic Regression')  
plt.legend()  
plt.show()

# F: Data Summary and Implications

*# Calculate and plot odds ratio*prediction\_data['odds\_ratio'] = prediction\_data.ReAdmis / (1 - prediction\_data.ReAdmis)

sns.lineplot(x='TotalCharge', y='odds\_ratio', data=prediction\_data)  
plt.axhline(y=1, linestyle='dotted')  
plt.yscale('log')  
plt.title('TotalCharge vs ReAdmis odds ratio')  
plt.show()

*# Reduced model*print(model2.summary())

## F1: Results

The regression equation for the reduced model is:

Each coefficient can be interpreted as follows:

* **TotalCharge**: With all other variables held constant, for each one-unit increase in TotalCharge, there is a 39.1384 unit increase in the logit of ReAdmis.
* **HighBlood**: With all other variables held constant, having high blood pressure results in a 1.1249 unit decrease in the logit of ReAdmis compared to those who do not have high blood pressure.
* **Stroke**: With all other variables held constant, having a history of stroke results in a 1.6529 unit increase in the logit of ReAdmis compared to those who do not have a history of stroke.
* **Arthritis**: With all other variables held constant, having arthritis results in a 2.6284 unit decrease in the logit of ReAdmis compared to those who do not have arthritis.
* **Diabetes**: With all other variables held constant, having diabetes results in a 0.8926 unit decrease in the logit of ReAdmis compared to those who do not have diabetes.
* **Hyperlipidemia**: With all other variables held constant, having hyperlipidemia results in a 1.4407 unit decrease in the logit of ReAdmis compared to those who do not have hyperlipidemia.
* **BackPain**: With all other variables held constant, having back pain results in a 1.2508 unit decrease in the logit of ReAdmis compared to those who do not have back pain.
* **Anxiety**: With all other variables held constant, having anxiety results in a 2.5685 unit decrease in the logit of ReAdmis compared to those who do not have anxiety.
* **Allergic\_rhinitis**: With all other variables held constant, having allergies results in a 1.4165 unit decrease in the logit of ReAdmis compared to those who do not have anxiety.
* **Reflux\_esophagitis**: With all other variables held constant, having acid reflux results in a 1.4465 unit decrease in the logit of ReAdmis compared to those who do not have acid reflux.
* **Asthma**: With all other variables held constant, having asthma results in a 1.3569 unit decrease in the logit of ReAdmis compared to those who do not have asthma.
* **Initial\_admin\_Emergency\_Admission**: With all other variables held constant, being admitted from the emergency department results in a 6.7132 unit decrease in the logit of ReAdmis compared to those who were admitted for elective procedures.
* **Initial\_admin\_Observation\_Admission**: With all other variables held constant, being admitted for observation results in a 0.7649 unit increase in the logit of ReAdmis compared to those who were admitted for elective procedures.
* **Complication\_risk\_Low**: With all other variables held constant, having a low risk of complication results in a 5.5850 unit increase in the logit of ReAdmis compared to those who have a high risk of complications.
* **Complication\_risk\_Medium**: With all other variables held constant, having a medium risk of complications results in a 7.0545 unit increase in the logit of ReAdmis compared to those who have a high risk of complications.
* **Services\_CT\_Scan**: With all other variables held constant, receiving a CT scan while admitted results in a 1.5515 unit increase in the logit of ReAdmis compared to those who receive intravenous or blood work while admitted.
* **Services\_MRI**: With all other variables held constant, receiving an MRI while admitted results in a 2.6991 unit increase in the logit of ReAdmis compared to those who receive intravenous or blood work while admitted.

The reduced model is statistically significant. During the model comparison, it was revealed that the Pseudo R-squared value was 0.9490, which is good. The LLR p-value was 0.000, indicating results did not happen by chance. Looking at each independent variable individually, the p-values were all well under 0.05, indicating that each of them was also statistically significant and not just contributing by chance.

The reduced model is also practically significant. It predicts, with decent accuracy, whether a patient is likely to be readmitted given the total charges for the patient’s hospital stay, whether or not the patient has any of the following: high blood pressure, a history of stroke, arthritis, diabetes, hyperlipidemia, back pain, anxiety, allergies, acid reflux, or asthma; whether the patient was admitted from the emergency department, for observation, or for another reason; whether the patient has a high, medium, or low risk of complication; and whether the patient received a CT scan, MRI, or some other service while admitted. Many of these likely don’t directly impact whether a patient will be readmitted, however they do give insight to why patients get readmitted. For instance, having high blood pressure is associated with a lower probability of being readmitted than not having high blood pressure. This could mean that when all of a patient’s conditions are documented, they can be better cared for, making them less likely to be readmitted. Uncovering lurking information such as this is valuable to a hospital because it can uncover larger problems like documentation or training issues.

The data analysis did have several limitations, starting with the data cleaning process. During the outlier investigation, all outliers were retained. Regression analysis is not resilient to outliers, which can weaken the model’s ability to extrapolate beyond the sample. Another limitation was introduced during the data wrangling phase by scaling the data. Any transformations to the data make it more complex to interpret. This also means that any data fed into the model for practical use will also need to be scaled for predictions and unscaled for interpretation. The third limitation is that no causation can be implied from the correlations observed. There are clear patterns present that indicate there *could be* causation. However, that can’t be confirmed without performing a properly controlled experiment. The fourth limitation is that the data was not split for testing. The entire data set was used for both training and testing the model. Ideally, the data would have been split in a stratified manner to allow separate data sets to be used for training and testing. The last and most significant limitation of this analysis is that hospital charges are incurred during a patient’s stay. This means that, unless a large number of charges are incurred at the beginning of a patient's stay, for instance, due to a surgical procedure, the patient may appear to have a poor chance of being readmitted until closer to the end of their hospital stay.

## F2: Recommendations

Based on these findings, there are a couple courses of action that should be taken. First, at least one controlled experiment should be designed and implemented to determine which, if any, of the independent variables cause or contribute to increased odds of readmission. Likely lurking variables are interacting with the model which could make better sense of the information. For that reason, the second course of action which should be taken is further analysis. Some indication of the state of the patient's health, like admitting diagnosis or chief complaint, as well as whether the patients had any surgical procedures during their stay would be great information to include.

# G: Panopto Recording

[The Panopto recording can be accessed at the following address: <https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=3e264f8f-a4ef-4ff4-b202-af30012ab3b2>](https://d.docs.live.net/116baf7bed3a0c05/Documents/Education/Western%20Govenors%20University/MS%20-%20Data%20Analytics/D208%20-%20Predictive%20Modeling/The%20Panopto%20recording%20can%20be%20accessed%20at%20the%20following%20address:%20https:/wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=506470ba-068c-4392-86b0-af2701704a9b)

# H: Sources of Third-Party Code

Only library reference documentation was used to produce the included code. This includes the following Python libraries: NumPy, pandas, Matplotlib, Seaborn, Scikit-learn, and Statsmodels.

# I: Sources

Bobbitt, Z. (2020, October 13). *The 6 Assumptions of Logistic Regression (With Examples).* Retrieved from Statology: https://www.statology.org/assumptions-of-logistic-regression/

Leung, K. (2021, October 4). *Assumptions of Logistic Regression, Clearly Explained | by Kenneth Leung | Towards Data Science*. Retrieved from towardsdatascience.com: https://towardsdatascience.com/assumptions-of-logistic-regression-clearly-explained-44d85a22b290