Eric Yarger

Logistic Regression Analysis using

Statsmodels Logit and Sklearn KNeighborsRegressor

Research Question

Logistic regression is a versatile tool for statistical analysis. The purpose of this project is to use logistic regression to analyze a realistic organizational situation, and to provide the results of the analysis. The question I'll address is: **What variables from the data set are the most effective at predicting if a patient will be Readmitted?** This analysis will be performed with the medical_clean data set provided for the course. The variable 'ReAdmis' is selected as the target variable for logistic regression.

The relevancy of being able to identify a patient's odds of readmission from data gathered at their initial visit is substantial. High readmission rates are fined and penalized by governing bodies in the USA. By identifying variables that can potentially identify patients who are at risk of readmission, our organization can save time and money, while also providing more effective treatment to our patients.

Objectives and Goals

The goal for analysis is to find what features we can use to best predict readmission rates of patients. This will be addressed by cleaning, preparing, and performing logistic regression on data from the medical_clean dataset provided for this course. A broad net will be cast to include all independent features and their respective dummy variables that show a correlatory value greater than .02 with our target variable, ReAdmis. This ensures that bias and assumption into potential variable correlation is negated, and makes sure that all variable selection is performed in a statistically sound manner. Next, Initial model features will be statistically analyzed with logistic regression. Model features will be reduced, if viable, and logistic regression will be performed again, resulting in identifying the predictive variables and their corresponding coefficients that answer the research question.

Summary of Assumptions

From Statology.org (Zach, 2020) the assumptions of logistic regression for this analysis are:

- The response variable is binary. The response variable can only take on two possible outcomes.
- The observations are independent.
- It is based on Bernoulli Distribution because the target variable is binary.
- There is no multicollinearity among the independent variables.
- There are no extreme outliers in the data set.
- There's a linear relationship between independent variables and the logit of the dependent variable.

- The sample size is large enough to draw valid conclusions from the model. From researching best practices regarding logistic regression sample size (Bujang, M., Sa'at, N., Sidik, T., Joo, L., 2018), with a pre-cleaned size of 10,000 cases our data set size is sufficient. After removal of outliers from the dataset I've set the goal of staying above 9000 cases for this analysis. This is still sufficiently sized for logistic regression with the amount of chosen independent variables for our analysis.

Tool Benefits

JupyterLabs IDE using Python3 as the language was used to perform this analysis. The Python environment is rich in libraries and functions that allow analysts to clean, prepare, and analyze large datasets. Python has a wide variety of open-source tools that analysts can import into their codebase to perform logistic regression. JupyterNotebooks are beneficial for organizing the necessary code to analyze and visualize the dataset. This is a huge benefit over less structured programming environments where altering and organizing code can be a hassle.

The tools used to perform this analysis are both built into Python as well as imported from various open-source libraries. From preparing, cleaning, to analyzing these tools are beneficial across the data analysis lifecycle, saving time and potential coding errors while improving preparation and analysis accuracy. These libraries include:

- Matplotlib - free and comprehensive library for creating visualizations in python. Example - Useful for visualizing histograms and the shape of variable data during cleaning.
- Pandas - powerful open source data analysis tool. Example - useful for creating two-dimensional data frames during the preparation stage.
- Seaborn - visualization tool for creating beautiful statistical graphics. Example - useful for visualizing data in the Cleaning and Analysis stages.
- Numpy - Allows Python to process larger arrays than it could normally handle. Example - used in scientific computing using arrays during the analysis stage.
- Missingno - used to visualize missing data in data sets. Example - used to visualize missing data in the cleaning and preparation stages.
- Scipy - Open source Python library for statistical analysis and scientific computing. Example, used for calculating z-scores in the data cleaning and preparation stage.

    - Scipy.stats.zscore - Used for calculating z-scores for independent feature analysis and normalization in the data preparation stage.
- Statsmodels.formula.api logit
    - Creates a Model from a formula and a dataframe. Used in the analysis stage of the project to perform Logistic regression for both our initial and reduced model.
- Sklearn libraries used in the analysis and model building phases.
    - Confusion_matrix, used in the analysis stage of the project and computes a confusion matrix to evaluate and visualize the accuracy of our classification from logistic regression.
    - accuracy _score, used in the analysis stage of the project and computes subset accuracy of our logistic regression.
    - Classification_report, used to build a text report showing the main classification metrics
    - preprocessing.StandardScaler - standardizes features by removing the mean and scaling to unit variance.
    - Pipeline.make_pipeline - constructs pipeline from given estimators
    - model_selection.KFold - provides test/ train indices to split data into sets. Splits into specified # of folds, which are then used for analysis validation.
    - neighbors.KNeighborsRegressor - Regression based on k-nearest neighbors. The target is predicted by interpolation of targets associated with the nearest neighbors in the training set.
    - Metrics.mean_squared_error, r2_score - calculation of Root Mean Square Error and R_squared.
    - Model_selection.cross_val_predict - Generates cross-validated estimates for each input data point.
- Patsy dmatrices constructs a single design matrix given a formula and data used in the feature selection phase.

- Statsmodels.stats.outliers_influence variance_inflation_factor calculates VIF (variance inflation factor) for our independent variables. Used in the analysis and preparation stages for determining independent variable admission into the model.

Appropriate Technique

Logistic regression is appropriate for this analysis because all of the assumptions for a successful model are affirmed. The dependent variable in our analysis, ReAdmis, our target variable, is categorical, and during the preparation phase will become a dummy variable with binary response. The observations in the data set are independent of one another. There's a linear relationship between the explanatory variables and the logit of the dependent variable. Multicollinearity between predictive variables is assessed and taken care of during preparation and initial feature selection to ensure there is no severe multicollinearity among independent variables. Outliers in the data set are addressed before logistic regression analysis. The sample size after cleaning and preparation is sufficiently large to draw valid conclusions from the regression model.

With all assumptions of a logistic regression model being checked and affirmed, this ensures that it is an appropriate technique for analyzing our research question.

Data Goals

According to software developer educational site EDUCBA, The goal of data preparation and manipulation efforts is to make interpreting and analyzing the insights from the data more structured and better designed (*Data Manipulation with Python, N.d.)*. The preparation and manipulations were undertaken to align our data goals with the goal of performing more effective logistic regression analysis. These actions included:

- Examining feature variance.
- Renaming applicable features for better insight and utility.
- Plotting univariate boxplots and histograms to analyze the shape and distribution of features.
- Splitting the data set into numerical and categorical feature sets. This allowed me to run Z-score analysis on all numerical data and remove cases with z-scores of more than 3. This removes outlier cases from the dataset, which creates a more uniform distribution within features for Logistic Regression.

- Plotting bivariate scatterplots for analysis of independent features in their relation to the dependent feature.
- Creating dummy variables for categorical variables, ensuring k-1 dummy variable creation in the process.
- Statistically selecting the most viable features for logistic regression analysis.
- VIF calculated for predictor variables selected for the initial model to look for multicollinearity. Variables with VIF > 3 are removed from the predictor variable list to ensure no multicollinearity.

Each of these preparation and manipulations of the data set provide a more cohesive, structured, and viable data set for logistic regression, aligning our data quality with our data goals. This ensures that the analysis returns the best possible results.

Summary Statistics

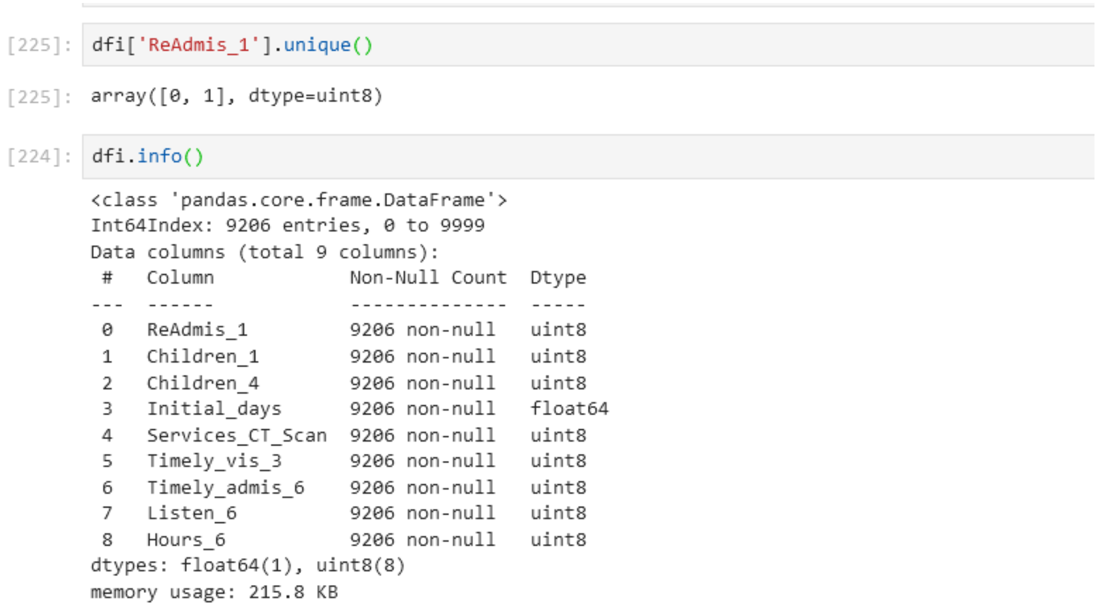
Summary statistics help us to identify the shape, distribution, correlation, meaning, and size of the data. For instance, data set size provides us with knowledge that our data set will be large enough to provide meaningful and valid insight from analysis. With 9206 observations for each variable, we know our set is large enough for regression analysis. Knowing if a variable is categorical or continuous helps us decide how to clean, prepare, and utilize each feature. We can see that our Target Variable, ReAdmis_1 is binary categorical by using df.unique() to see an array of unique values in ReAdmis_1. This satisfies the requirement for our logistic regression target variable. The predictor variables are made up of 1 continuous ('Initial_days'), and 7 categorical variables.

Mean, Median, and Mode help to visualize the feature's distribution and size. This helps us see the distribution and average observation size for ReAdmis_1 in comparison to all predictor variables. Knowing features percentiles, minimum, and maximum quantities helps to identify shape, distribution, and outliers.

Being able to see correlation between our target variable and each predictor variable helps to identify which features will be viable for analysis. From the correlation table below we can see that predictor variables' correlational relationship with each other, as well as with our target variable.

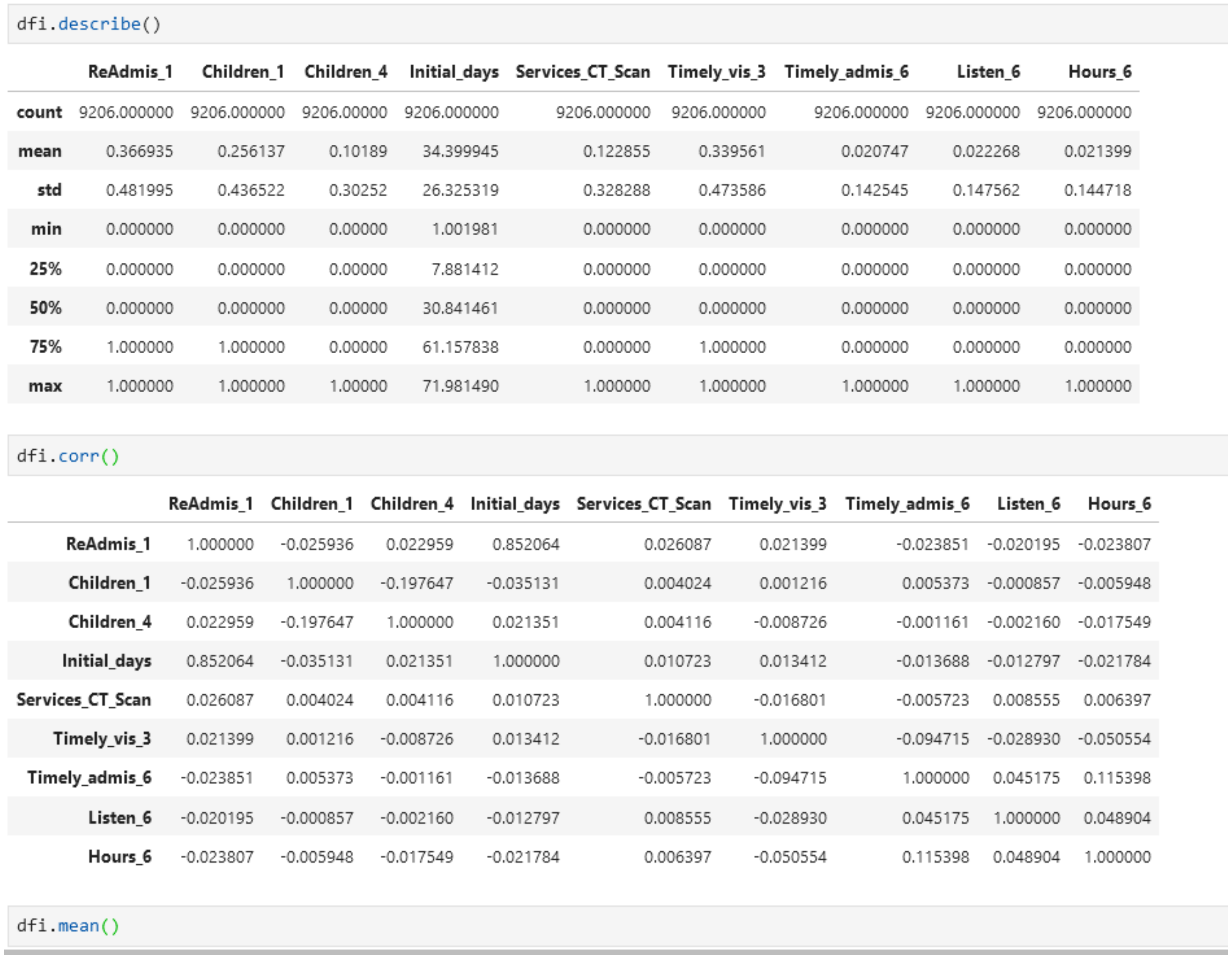
The tables below provide the summary statistics relevant to our target variable, ReAdmis_1, and all predictor variables. This information helps us to quantify and visualize each variable, ensuring that each variable is relevant for our analysis. The summary statistics presented in this document, in conjunction with the univariate and bivariate visualizations created and provided in the supplemental code PDF in this submission paint an accurate depiction of each variable to ensure our features are viable for logistic regression..

This table identifies the target variable, ReAdmis_1, and outputs an array identifying all unique column entries. This verifies that the target variable is binary categorical. Underneath is a table identifying the target variable and all prediction variables. Each variable has 9206 Non-Null cases. Datatype is listed for each variable under column 'Dtype'. Initial_days is continuous, as shown by Dtype 'float64', all other variables are categorical.



```
[225]: dfi['ReAdmis_1'].unique()

[225]: array([0, 1], dtype=uint8)

[224]: dfi.info()

<class 'pandas.core.frame.DataFrame'>
Int64Index: 9206 entries, 0 to 9999
Data columns (total 9 columns):
 #   Column            Non-Null Count  Dtype  
---  ------            --------------  -----  
 0   ReAdmis_1         9206 non-null   uint8  
 1   Children_1        9206 non-null   uint8  
 2   Children_4        9206 non-null   uint8  
 3   Initial_days      9206 non-null   float64
 4   Services_CT_Scan  9206 non-null   uint8  
 5   Timely_vis_3      9206 non-null   uint8  
 6   Timely_admis_6    9206 non-null   uint8  
 7   Listen_6          9206 non-null   uint8  
 8   Hours_6           9206 non-null   uint8  
dtypes: float64(1), uint8(8)
memory usage: 215.8 KB
```

Pandas Dataframe.describe() method is used to output statistical details for count of observations, mean value, standard deviation, min & max values, and percentiles (25th, 50th, and 75th) for target variable and all predictor variables. Dataframe.corr() method is used to output the correlation between each variable. The screenshot below displays this information for our target variable and all predictor variables.



```
dfi.describe()
```

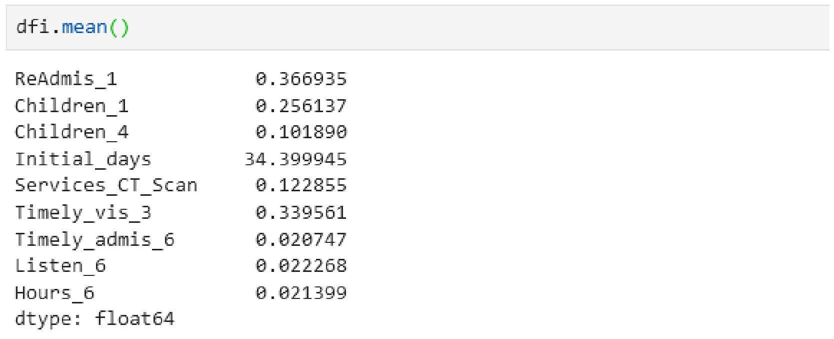
	ReAdmis_1	Children_1	Children_4	Initial_days	Services_CT_Scan	Timely_vis_3	Timely_admis_6	Listen_6	Hours_6
count	9206.000000	9206.000000	9206.00000	9206.000000	9206.000000	9206.000000	9206.000000	9206.000000	9206.000000
mean	0.366935	0.256137	0.10189	34.399945	0.122855	0.339561	0.020747	0.022268	0.021399
std	0.481995	0.436522	0.30252	26.325319	0.328288	0.473586	0.142545	0.147562	0.144718
min	0.000000	0.000000	0.00000	1.001981	0.000000	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.00000	7.881412	0.000000	0.000000	0.000000	0.000000	0.000000
50%	0.000000	0.000000	0.00000	30.841461	0.000000	0.000000	0.000000	0.000000	0.000000
75%	1.000000	1.000000	0.00000	61.157838	0.000000	1.000000	0.000000	0.000000	0.000000
max	1.000000	1.000000	1.00000	71.981490	1.000000	1.000000	1.000000	1.000000	1.000000

```
dfi.corr()
```

	ReAdmis_1	Children_1	Children_4	Initial_days	Services_CT_Scan	Timely_vis_3	Timely_admis_6	Listen_6	Hours_6
ReAdmis_1	1.000000	-0.025936	0.022959	0.852064	0.026087	0.021399	-0.023851	-0.020195	-0.023807
Children_1	-0.025936	1.000000	-0.197647	-0.035131	0.004024	0.001216	0.005373	-0.000857	-0.005948
Children_4	0.022959	-0.197647	1.000000	0.021351	0.004116	-0.008726	-0.001161	-0.002160	-0.017549
Initial_days	0.852064	-0.035131	0.021351	1.000000	0.010723	0.013412	-0.013688	-0.012797	-0.021784
Services_CT_Scan	0.026087	0.004024	0.004116	0.010723	1.000000	-0.016801	-0.005723	0.008555	0.006397
Timely_vis_3	0.021399	0.001216	-0.008726	0.013412	-0.016801	1.000000	-0.094715	-0.028930	-0.050554
Timely_admis_6	-0.023851	0.005373	-0.001161	-0.013688	-0.005723	-0.094715	1.000000	0.045175	0.115398
Listen_6	-0.020195	-0.000857	-0.002160	-0.012797	0.008555	-0.028930	0.045175	1.000000	0.048904
Hours_6	-0.023807	-0.005948	-0.017549	-0.021784	0.006397	-0.050554	0.115398	0.048904	1.000000

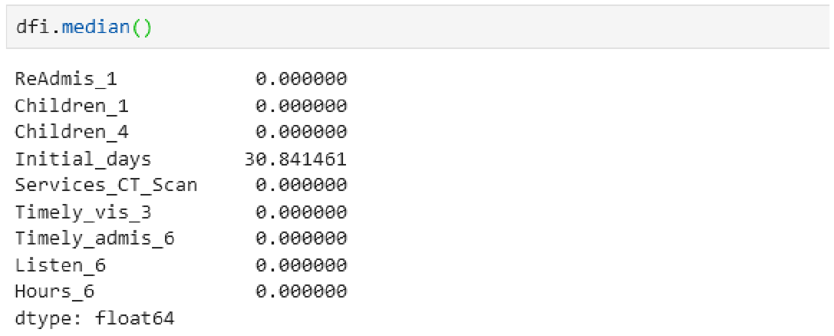
```
dfi.mean()
```

Measures of central tendency are single values that represent the center point of the dataset. There are three common measures of central tendency: the mean, median, and mode (Zach, 2018). Measures of central tendency are detailed for the target variable and all predictor variables in the table below.



```
dfi.mean()
```

```
ReAdmis_1           0.366935
Children_1          0.256137
Children_4          0.101890
Initial_days       34.399945
Services_CT_Scan    0.122855
Timely_vis_3        0.339561
Timely_admis_6      0.020747
Listen_6            0.022268
Hours_6             0.021399
dtype: float64
```



```
dfi.median()
```

```
ReAdmis_1           0.000000
Children_1          0.000000
Children_4          0.000000
Initial_days       30.841461
Services_CT_Scan    0.000000
Timely_vis_3        0.000000
Timely_admis_6      0.000000
Listen_6            0.000000
Hours_6             0.000000
dtype: float64
```

Table showing Mode for Target and all Predictor variables, identified using Google Sheets Column Stats.

Variable	**Mode**
ReAdmis_1 (Target Variable)	0
Children_1	0
Children_4	0
Initial_days	67.42139, frequency = 2 70.32542, frequency = 2 65.54432, frequency = 2
Services_CT_Scan	0
Timely_visit_3	0
Timely_admis_6	0
Listen_6	0
Hours_6	0

Steps to Prepare Data

All code used to clean and prepare the data for analysis can be found in the included PDF 'D208_2.3'. All coded inputs and their generated outputs are shown executed without errors. Detailed below are the steps taken to prepare the data for Logistic Regression.

Step 1: Environment setup

Libraries are imported into the development environment in preparation for the data preparation and analysis. These include analytical libraries such as pandas, scipy, and numpy, as well as visual generation libraries like seaborn, pyplot, and missingno. The data is loaded into a JupyterNotebook.

Step 2: Cleaning and Preparation

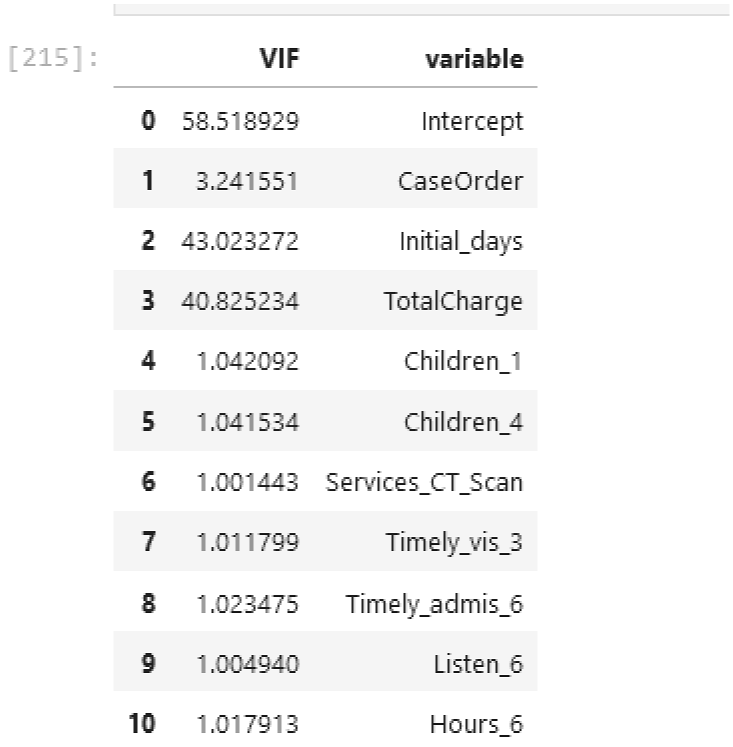
The dataset is analyzed as a whole to look at data types, if there's missing values, and if any values need to be altered or deleted. Correlation tables are mapped. Variable 'CaseOrder' is dropped due to correlation with ReAdmis that violates the assumption of case independence. Z-scores are calculated and outliers z-score >3 are removed. Visualization tools such as missingno's matrix, seaborn's boxplots, and pyplot's histograms are used to plot Univariate and Bivariate graphs.

Dummy variables are created using pd.get_dummies(), with drop_first parameter set = True to ensure that the first variable is dropped. This ensures that each dummy set has k-1 number of features. All applicable variables are renamed for utility and ease of use. Heatmaps and correlation tables are drawn looking for which variables most closely correlate with our target dependent variable, ReAdmis_1 (the dummy variable of ReAdmis).

At this point the data set is cleaned, dummied, and ready for initial feature selection for Logistic Regression. The original dataset contained 50 columns/variables, our prepared data set contains 98 columns/features. The increase in size is due to the creation of dummy variables for categorical features that have multiple responses, such as 'Doc_visits' and all of the survey response variables.

Our target variable is quantified and dummied, resulting in a change from 'ReAdmis' to 'ReAdmis_1' as our target variable. Initial independent features are chosen from this data set through statistical feature selection. This was performed by selecting all independent features that have a correlation > .02 with our Target variable, ReAdmis_1. This left us with the 10 predictive variables.

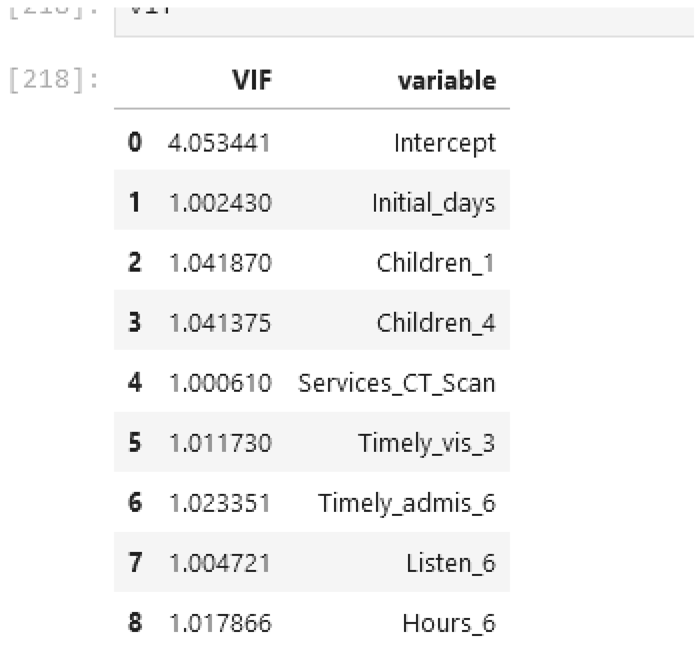
VIF is calculated for our predictive variables, outputting the following table:



[215]:

	VIF	variable
0	58.518929	Intercept
1	3.241551	CaseOrder
2	43.023272	Initial_days
3	40.825234	TotalCharge
4	1.042092	Children_1
5	1.041534	Children_4
6	1.001443	Services_CT_Scan
7	1.011799	Timely_vis_3
8	1.023475	Timely_admis_6
9	1.004940	Listen_6
10	1.017913	Hours_6

Variable CaseOrder and TotalCharge are removed from the selection. VIF is rerun to ensure no multicollinearity between the predictive variables for the initial Logistic Regression. VIF table #2 is shown below:



[218]:

	VIF	variable
0	4.053441	Intercept
1	1.002430	Initial_days
2	1.041870	Children_1
3	1.041375	Children_4
4	1.000610	Services_CT_Scan
5	1.011730	Timely_vis_3
6	1.023351	Timely_admis_6
7	1.004721	Listen_6
8	1.017866	Hours_6

At this stage the target variable and all predictive variables are prepared for logistic regression.

Visualizations

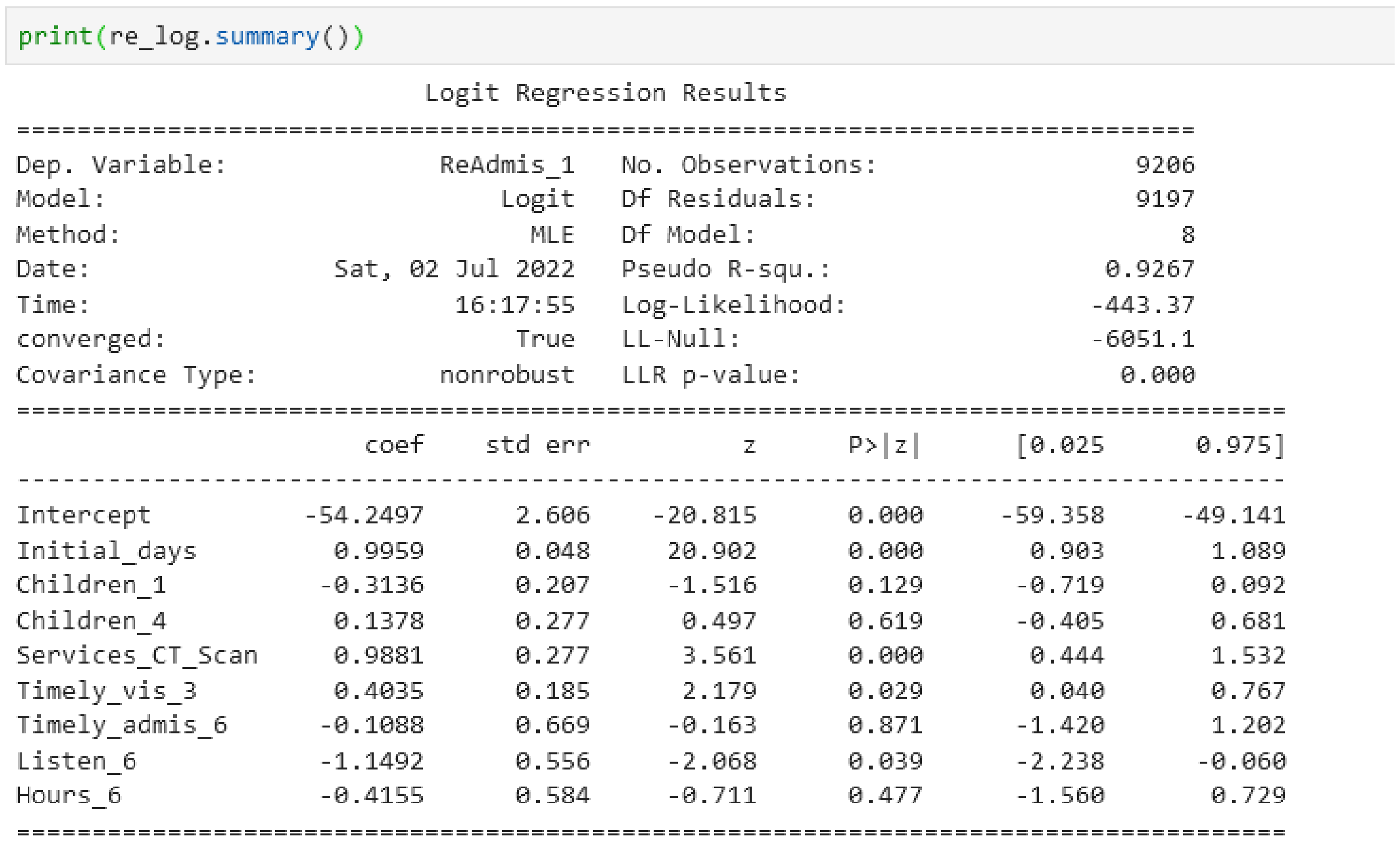
Univariate and bivariate visualizations have been generated for variables in the cleaned data set. Visualizations are included in the supplemental PDF 'D208_2.3' of the Jupyter Notebook used for all code for this PA. For univariate visualization, histograms and boxplots were created for all necessary variables. All bivariate visualizations (pairplots) are mapped with the target variable on the X-Axis, and the independent variables on the Y-Axis.

Prepared Data Set

I was unsure if the PA rubric requirement read as providing a copy of the fully prepared data set (all variables, pre-selection of variables for multiple regression), or a copy of the fully prepared data set (post-variable selection preparation). I've included both in the submission.

Initial Model

The table below provides the results of the initial logistic regression model. The target variable is ReAdmis_1 and the model contains all predictive variables identified in section C2.



```
print(re_log.summary())
```

```
                           Logit Regression Results                           
==============================================================================
Dep. Variable:              ReAdmis_1   No. Observations:                 9206
Model:                          Logit   Df Residuals:                     9197
Method:                           MLE   Df Model:                            8
Date:                Sat, 02 Jul 2022   Pseudo R-squ.:                  0.9267
Time:                        16:17:55   Log-Likelihood:                -443.37
converged:                       True   LL-Null:                       -6051.1
Covariance Type:            nonrobust   LLR p-value:                     0.000
=====================================================================================
                      coef    std err          z      P>|z|      [0.025      0.975]
-------------------------------------------------------------------------------------
Intercept         -54.2497      2.606    -20.815      0.000     -59.358     -49.141
Initial_days        0.9959      0.048     20.902      0.000       0.903       1.089
Children_1         -0.3136      0.207     -1.516      0.129      -0.719       0.092
Children_4          0.1378      0.277      0.497      0.619      -0.405       0.681
Services_CT_Scan    0.9881      0.277      3.561      0.000       0.444       1.532
Timely_vis_3        0.4035      0.185      2.179      0.029       0.040       0.767
Timely_admis_6     -0.1088      0.669     -0.163      0.871      -1.420       1.202
Listen_6           -1.1492      0.556     -2.068      0.039      -2.238      -0.060
Hours_6            -0.4155      0.584     -0.711      0.477      -1.560       0.729
=====================================================================================
```

Logistic regression was performed using statsmodels.formula.api logit function. Full code input and output is provided in the accompanying PDF.

Justification of Model Reduction

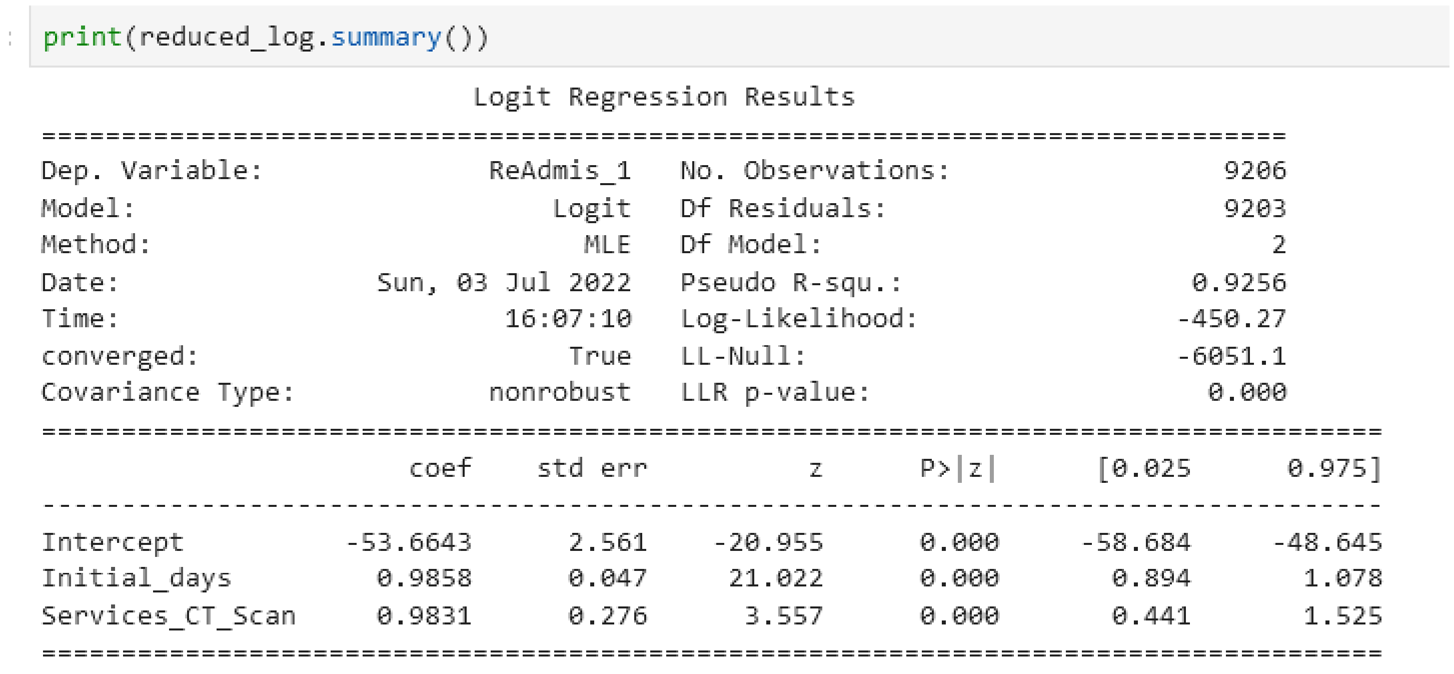
The statistically based variable selection procedure used for model reduction was performed by analyzing variable P-values, and by selecting features that reduce root mean square error (RMSE) . First, features with P(t)-values > .05 from the initial model fail to reject the null hypothesis, and are dropped from the model. This leaves Iniital_days, Services_CT_Scan, Timely_vis_3, and Listen_6 as predictive variables selected from this step in variable selection.

The second step in statistically selecting variables is to assess (RMSE) values, and minimize RMSE for variable selection. To do this, K-nearest neighbor, a non-parametric supervised learning method, and cross_val_predict perform and generate RMSE estimates for the initial model's predictive variables. Initial_days, Children_1, and Services_CT_Scan are identified at the minimized RMSE. Children_1 is dropped from the aspect of variable selection for having a P value of .129, as specified in the first step of our feature selection technique. This statistically based procedure selects Initial_days and Services_CT_Scan as the predictive variables for the reduced model.

The model evaluation metrics used to reduce the initial model are metrics from the Classification Report, Confusion Matrix, and Accuracy Score. The classification report is generated using sklearn's classification_report function, the confusion matrix is calculated using sklearn's confusion_matrix function, and the accuracy score is calculated using sklearn's accuracy_score function. The evaluation metrics generated from each of these tools ensures that the reduced model still retains its accuracy and precision in predicting Readmission rates. Evaluation metrics also serve as a double-check that the variable selection procedure used was appropriate for the model in regard to aligning with our research question.

Reduced Logistic Regression Model

Provided below is a screenshot of the reduced Logit Regression model. All code inputs and outputs necessary to create the reduced logit regression model are included in the supplemental PDF copy of the code used for this analysis.



```
print(reduced_log.summary())
```

```
                           Logit Regression Results
==============================================================================
Dep. Variable:              ReAdmis_1   No. Observations:                 9206
Model:                          Logit   Df Residuals:                     9203
Method:                           MLE   Df Model:                            2
Date:                Sun, 03 Jul 2022   Pseudo R-squ.:                  0.9256
Time:                        16:07:10   Log-Likelihood:                -450.27
converged:                       True   LL-Null:                       -6051.1
Covariance Type:            nonrobust   LLR p-value:                     0.000
====================================================================================
                       coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------------
Intercept          -53.6643      2.561    -20.955      0.000     -58.684     -48.645
Initial_days         0.9858      0.047     21.022      0.000       0.894       1.078
Services_CT_Scan     0.9831      0.276      3.557      0.000       0.441       1.525
====================================================================================
```

Predictive variables Initial_days and Services_CT_Scan are chosen for the model and are in alignment with the justification from part D2.

Model Comparison

The logic of the variable selection technique for the initial model was to statistically select predictive variables based on their correlation with the target variable, and to ensure there was minimal multicollinearity between them. Specifically, any variable that showed a correlation value >= .02 was retained as a potential predictive variable. Before the logistic model was run multicollinearity among predictive variables was assessed using VIF. Variable TotalCharge was dropped from the predictive variables to address multicollinearity with variable Initial_days.

The statistically based variable selection procedure used for model reduction was performed by analyzing variable P-values, and by selecting features that reduce root mean square error (RMSE) . First, features with P(t)-values > .05 from the initial model fail to reject the null hypothesis, and are dropped from the model. This leaves Iniital_days, Services_CT_Scan, Timely_vis_3, and Listen_6 as predictive variables selected from this step in variable selection.

The second step in statistically selecting variables is to assess (RMSE) values, and minimize RMSE for variable selection. To do this, K-nearest neighbor, a non-parametric supervised learning method, and cross_val_predict perform and generate RMSE estimates for the initial model's predictive variables. Initial_days, Children_1, and Services_CT_Scan are identified at the minimized RMSE. Children_1 is dropped from the aspect of variable selection for having a P value of .129, as specified in the first step of our feature selection technique. This statistically based procedure selects Initial_days and Services_CT_Scan as the predictive variables for the reduced model.

The table below compares the model's predictive variables and their coefficients.

Feature	**Initial Model**	**Reduced Model**	**Result**
Intercept	-54.2497	-53.6643	Higher intercept in reduced model
Initial_days	.9959	.9858	Lower in reduced model
Services_CT_Scan	.9881	.9831	Lower in reduced model
Children_1	-.3136	Variable not included in reduced model	Removal of variable reduced model complexity at minimal loss of model predictive accuracy.

Children_4	.1378	Variable not included in reduced model	Removal of variable reduced model complexity at minimal loss of model predictive accuracy.
Timely_vis_3	.4035	Variable not included in reduced model	Removal of variable reduced model complexity at minimal loss of model predictive accuracy.
Timely_admis_6	-.1088	Variable not included in reduced model	Removal of variable reduced model complexity at minimal loss of model predictive accuracy.
Listen_6	-1.1492	Variable not included in reduced model	Removal of variable reduced model complexity at minimal loss of model predictive accuracy.
Hours_6	-.4155	Variable not included in reduced model	Removal of variable reduced model complexity at minimal loss of model predictive accuracy.

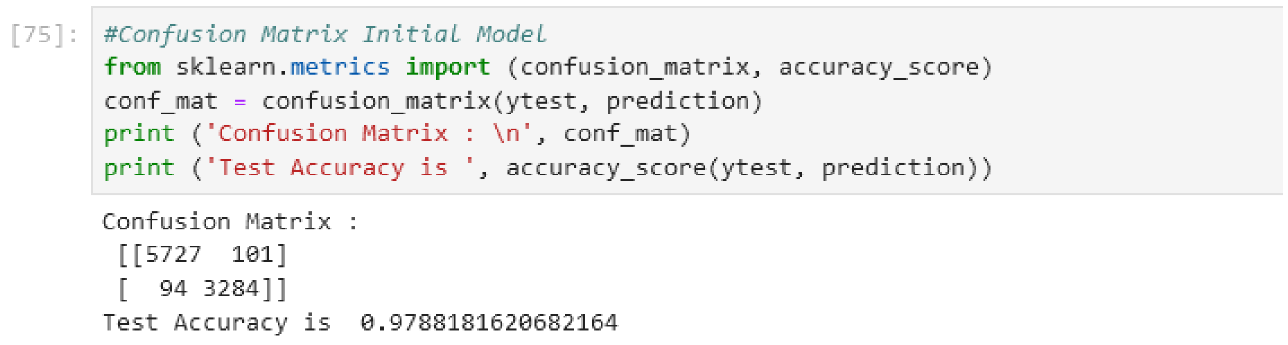
The table below compares the results of the Logit Regression summary tables.

Item	**Initial Model**	**Reduced Model**	**Result**
No. of Observations	9206	9206	Same # of cases
DF Residuals	9197	9203	Drop in reduced model representative of # of features reduced
DF Model	8	2	Drop in reduced model representative of # of features reduced
Pseudo R-squ	.9267	.9256	.0011 drop in reduced model. The reduced complexity of dropping variables more than justifies the drop in pseudo R-square in the reduced model.
Log-Likelihood	-443.37	-450.27	Negative increase of 6.9

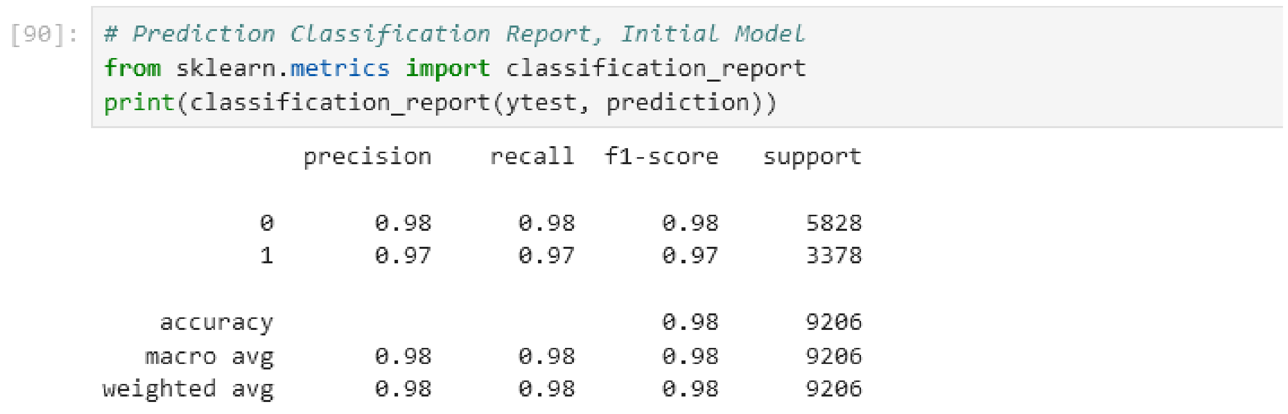
			in reduced model. Indicates the initial model offers a slightly better model fit.
LL-Null	-6051.1	-6051.1	Indicates that models **with** independent variables offer better goodness of fit than if there were none. Both models are equal in this regard.
LLR p-value	0.000	0.000	In comparison both models' LLR p-value is representative that we can reject the null hypothesis.

The model evaluation metrics used to reduce the initial model are the Confusion Matrix, Accuracy Score, and the metrics generated by the Classification Report. The generated model evaluation metrics are shown below.

Model Evaluation Metrics: Initial Model



```
[75]: #Confusion Matrix Initial Model
      from sklearn.metrics import (confusion_matrix, accuracy_score)
      conf_mat = confusion_matrix(ytest, prediction)
      print ('Confusion Matrix : \n', conf_mat)
      print ('Test Accuracy is ', accuracy_score(ytest, prediction))

      Confusion Matrix :
       [[5727  101]
       [  94 3284]]
      Test Accuracy is  0.9788181620682164
```



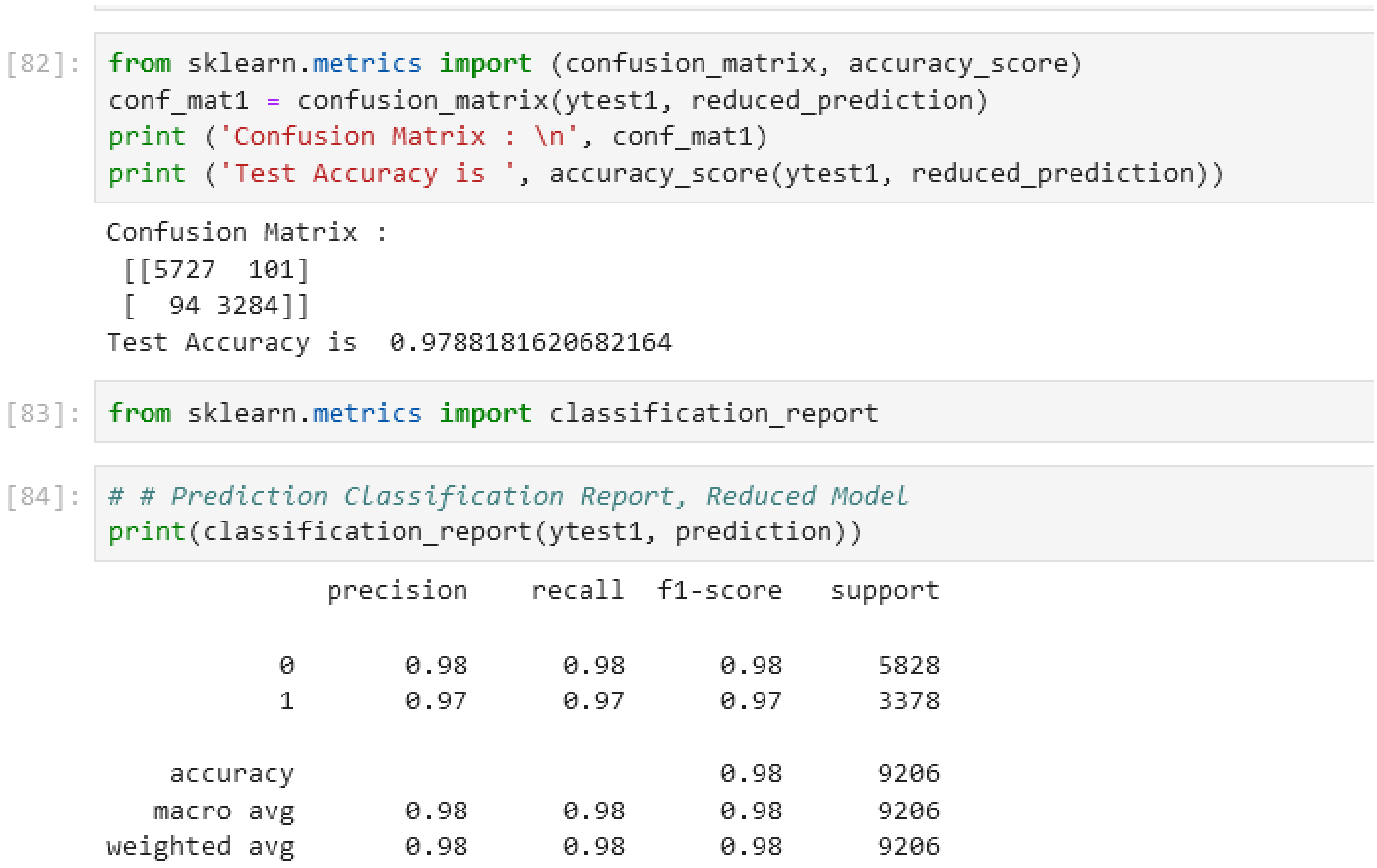
```
[90]: # Prediction Classification Report, Initial Model
      from sklearn.metrics import classification_report
      print(classification_report(ytest, prediction))

                    precision    recall  f1-score   support

                 0       0.98      0.98      0.98      5828
                 1       0.97      0.97      0.97      3378

          accuracy                           0.98      9206
         macro avg       0.98      0.98      0.98      9206
      weighted avg       0.98      0.98      0.98      9206
```

Confusion matrix, detailed, for the Initial Model

True Positive Count: 5727	False Negative Count: 101
False Positive Count: 94	True Negative Count: 3284

Test accuracy for the initial model is .9788181620682164

Model Evaluation Metrics: Reduced Model



```
[82]: from sklearn.metrics import (confusion_matrix, accuracy_score)
      conf_mat1 = confusion_matrix(ytest1, reduced_prediction)
      print ('Confusion Matrix : \n', conf_mat1)
      print ('Test Accuracy is ', accuracy_score(ytest1, reduced_prediction))
```

```
Confusion Matrix :
 [[5727  101]
 [  94 3284]]
Test Accuracy is  0.9788181620682164
```

```
[83]: from sklearn.metrics import classification_report
```

```
[84]: # # Prediction Classification Report, Reduced Model
      print(classification_report(ytest1, prediction))
```

```
              precision    recall  f1-score   support

           0       0.98      0.98      0.98      5828
           1       0.97      0.97      0.97      3378

    accuracy                           0.98      9206
   macro avg       0.98      0.98      0.98      9206
weighted avg       0.98      0.98      0.98      9206
```

Confusion matrix, detailed, for the Reduced Model

True Positive Count: 5727	False Negative Count: 101
False Positive Count: 94	True Negative Count: 3284

Test Accuracy for the reduced model is .9788181620682164

The confusion matrix of the initial model was compared to the confusion matrix of the reduced model. The model evaluation metrics for the models are identical. The code used to generate both models was checked multiple times to ensure there wasn't an error causing this.

The metrics identical figures for precision and test accuracy, and the quadrants of the confusion matrix showing identical values shows that the reduced model is as accurate as the initial model. True Positives, False Negatives, False Positives, and True Negatives are predicted from the data as accurately in the reduced model, with the added benefit of reducing six predictive variables.

Results

Regression equation for the reduced model from the analysis, Reference (FAQ How do I interpret odds ratios in logistic regression, N.d.) is:

logit(p) = -53.6643 +.9858*(Initial_days) + .9831*(Services_CT_Scan)

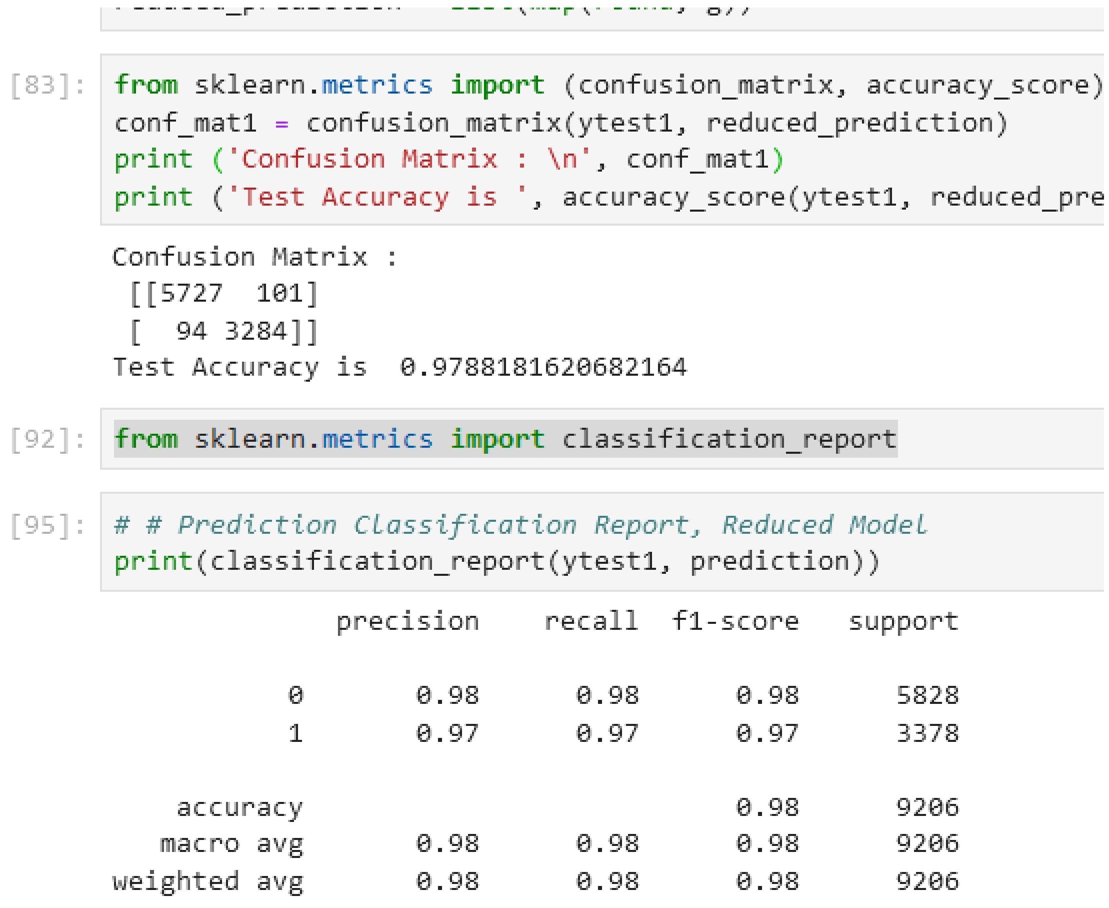
Where p = probability of ReAdmis_1 = 1

Interpreting the coefficients for the reduced model. -53.6643 is the Intercept. Services_CT_Scan is categorical. If Services_CT_Scan is 1, then .9831*(1). If Services_CT_Scan is 0, then .9831*(0). A quick look at the count for Services_CT_Scan shows that in our data set 1131 cases are '1', 8075 cases are '0'.

Initial_days is continuous, and as such the coefficient, .9858, is multiplied by the # of initial days for each specific case. This means that Initial_days on average has a much greater impact on the equation. The results of this analysis yield that the vast majority of the model's predictive ability comes from one variable, Initial_days.

The statistical significance of the model is derived from looking at model metrics. The p-values from the variables in the reduced model is .000 for both Initial_days and Services_CT_Scan. This proves statistical significance for each variable. The LLR p-value of 0.000 suggests that we can reject the null hypothesis for the model and be ensured of its' statistical significance.

The practical significance of the model is validated by deciding if the effect is large enough to be significant if applied in the real world. This can be decided by looking at figures gathered from the classification report and test accuracy of the reduced model, shown below:



```
[83]: from sklearn.metrics import (confusion_matrix, accuracy_score)
      conf_mat1 = confusion_matrix(ytest1, reduced_prediction)
      print ('Confusion Matrix : \n', conf_mat1)
      print ('Test Accuracy is ', accuracy_score(ytest1, reduced_pre

      Confusion Matrix :
       [[5727  101]
       [  94 3284]]
      Test Accuracy is  0.9788181620682164

[92]: from sklearn.metrics import classification_report

[95]: # # Prediction Classification Report, Reduced Model
      print(classification_report(ytest1, prediction))
                    precision    recall  f1-score   support

                 0       0.98      0.98      0.98      5828
                 1       0.97      0.97      0.97      3378

          accuracy                           0.98      9206
         macro avg       0.98      0.98      0.98      9206
      weighted avg       0.98      0.98      0.98      9206
```

The precision column lists the accuracy of our test vs actual figures. To interpret this, out of all cases that the model predicted would not be readmitted, 98% were not. Out of all cases the model predicted would be readmitted, 98% were. These results are further validated by the model Test Accuracy score of .978818620682164. The Recall column above shows the correct positive predictions relative to the total actual positives. With this being tested on a model with 9206 cases, we can therefore determine that the model is practically significant and produces meaningful and accurate results.

Limitations of this analysis are:

- Correlation does not imply causation.
    - Initial_days has a correlation with ReAdmis of .85, while the next highest correlation for a predictive variable in our data set was .026 for Services_CT_Scan. This was a premonition at the beginning of the analysis that Initial_days was going to account for a large share of the models' predictive capacity. Even though this looks great in the model we must be careful to not imply that the # of Initial days a patient is at the hospital is the cause of Readmission within 30 days..

- Quasi Separation in Logistic Regression model
    - Both the initial model (value = .74) and reduced model (value = .73) included a message regarding quasi-separation. Quasi separation happens in logistic regression when outcome variables separate a predictor variable or a combination of predictor variables (*Faq What is Complete or Quasi-Complete Separation In Logistic/Probit Regression and How Do We Deal With Them,* N.d.). This can cause the validity of the model to be questionable, which is a major limitation.
- Low correlative values of meaningful predictive variables.
    - I tried to cast a wide net to include many predictive variables in the initial model. The limitation is that the vast majority of the data set's variables show very little correlation with the target variable, ReAdmis. By setting a statistical threshold of .02 for the correlation for initial feature selection the meaningfulness of the initial model predictive features was diluted.
- Model preparation.
    - When preparing the model I would like to note that the variable CaseOrder showed a high correlation with ReAdmis, which can be seen by looking at the bivariate plots. The first 5000 cases all have a 'No' response to ReAdmis. This indicates that the data set provided for the course is not random.
- Logistic regression assumptions.
    - To build on the response to 'Model preparation' above, the assumption that the observations in the data set are independent may be violated. The definition for variable 'CaseOrder' from the data set download for the course reads 'A placeholder variable to preserve the original order of the raw data file.' This means that the original data set's first 5000 cases all have a 'No' response to ReAdmis. There's virtually no probability that this occurred randomly in the data set. To mitigate the effects of this CaseOrder was removed from the data set for the purpose of this analysis - effectively ignoring the original order and resolving this issue.
- Reliance on a minimal number of predictive variables.
    - The reduced model relies heavily on Initial_days for its predictive ability. This limits the model's robustness and practicality in real world application.

Recommendations

Based on the results of our analysis, the number of initial days, represented in the data set as variable Initial_days, should be weighted heavily when our organization is assessing what can help us identify patients that will be readmitted in 30 days from their initial visit. That isn't to say that initial days spent by the patient in the hospital is the causation of readmittance. We should be aware, though, that the longer the patient is in the hospital, measured in days, the more likely they are to be readmitted within 30 days.

It is recommended that the initial and reduced model be independently evaluated, with the possible removal of variable 'Initial_days'. Judging from the identical results from our model evaluation metrics, Initial_days is responsible for nearly all of the predictive capacity of the model. I recommend that it is not wise to rely so heavily on a model of any number of predictive variables that has one variable that it relies on for commercial applications. Depending on the importance of this analysis' outcome for the organization, more resources should (or shouldn't, depending on the value to our organization) be allocated to developing a more robust model.

It is recommended that Quasi separation be addressed in the model. Before the model is used in a commercial application the validity of its results needs to be assessed and tested. Techniques for dealing with quasi–complete separation are:

- Make sure the target variable is not a near-match for a predictive variable in the model.
- Identify and remove the predictive variable from the model. This can potentially lead to biased estimates for other predictor variables.
- Do nothing. The maximum likelihood for other variables is still valid. The drawback of this strategy is that the overall model loses the ability to predict the target variable with statistical and practical utility.

The overall recommended course of action is to use the results of this analysis as just one predictive tool in the organizations' toolkit for assessing readmittance rates. Our organization needs to use multiple models to make certain that bias is minimized and statistical and practical significance are maximized for making business decisions regarding assessing Readmittance rates.

From an operational perspective, one use of this analysis is that our organization can set up a system to contact patients that spent more time at the hospital on their initial visit, and are more likely to be readmitted due to this. One course of action we could perform based on our results from this analysis is to conduct a test on a group of patients, and compare it to our general patient population to see if there's a statistically significant decrease in Readmittance rates. A simple and cost effective test would be if we contacted the customers in the top 25% by Initial_days at the hospital 10 days after they are released. These patients would have a chance to express any issues or complications they are experiencing, without having to be readmitted. This would give our organization the chance to address minor issues for the patient without them coming back to the hospital, potentially reducing readmission rates and providing better care to our patients.

Panopto

I've included a Panopto recording with this submission. It includes all necessary elements, specifically::

- Screen and I are recorded for presentation duration.
- Code used for the analysis is demonstrated to be functional.
- Program version and environment are identified.
- Both logistic regression models are compared.
- Model coefficients are interpreted.

Third-Party Code Sources

Bushmanov, Sergey. (January 28, 2019). *How to remove Outliers in Python?.* Stackoverflow. https://stackoverflow.com/questions/54398554/how-to-remove-outliers-in-python

Pandas.get_dummies. (N.d.). pandas. https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.get_dummies.html

Seaborn.heatmap. (N.d.). seaborn. https://seaborn.pydata.org/generated/seaborn.heatmap.html

Zach. (July 20, 2020). *How to Calculate VIF in Python.* Statology.org. https://www.statology.org/how-to-calculate-vif-in-python/

Cosine1509. (May 18, 2022). *Logistic Regression using Statsmodels.* Geeksforgeeks.org. https://www.geeksforgeeks.org/logistic-regression-using-statsmodels/

Feely, Ciara. [Ph.D. and Productivity]. (May 21, 2020). *Feature Selection in Python | Machine Learning Basics | Boston Housing Data* [Video]. YouTube. https://www.youtube.com/watch?v=iJ5c-XoHPFo&ab_channel=PhDandProductivity

Sklearn.metrics.confusion_matrix. (N.d.) Scikit Learn. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html

Sklearn.metrics.accuracy_score. (N.d.) Scikit Learn. https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html

Zach. (September 1, 2021). *How to Create a Confusion Matrix in Python.* Statology.org. https://www.statology.org/confusion-matrix-python/

Sources

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

Bujang, M., Sa'at, N., Sidik, T., Joo, L.. (2018, Aug 30). *Sample Size Guidelines for Logistic Regression from Observational Studies with Large Population: Emphasis on the Accuracy Between Statistics and Parameters Based on Real Life Clinical Data.* PubMed Central. https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6422534/

Data Manipulation with Python. *(*N.d.). EDUCBA.com. https://www.educba.com/data-manipulation-with-python/

Zach. (September 18, 2018). *Measures of Central Tendency: Definition and Examples.* Statology.org. https://www.statology.org/measures-central-tendency/

Faq What is Complete or Quasi-Complete Separation In Logistic/Probit Regression and How Do We Deal With Them. (N.d.). UCLA Advanced Research Computing, Statistical Methods and Data Analytics. https://stats.oarc.ucla.edu/other/mult-pkg/faq/general/faqwhat-is-complete-or-quasi-complete-separation-in-logisticprobit-regression-and-how-do-we-deal-with-them/

FAQ How do I interpret odds ratios in logistic regression? (n.d.). UCLA Statistical Methods and Data Analytics. https://stats.oarc.ucla.edu/other/mult-pkg/faq/general/faq-how-do-i-interpret-odds-ratios-in-logistic-regression/