Data Mining I

Jason Willis

College of Information Technology,

Western Governors University

Dr. Eric Straw

February 23rd, 2021

Table of Contents for Each Rubric

Hospital Readmission Problem

For our chain of hospitals to lower readmission concerns, we need to identify patients who have increased risk of rehospitalization within a month of their release. According to Schuller (2020), non-obese adults were 21% less likely to be readmitted than obese adults. A readmission study by Gert, et. al. (2002) showed a correlation between longer initial hospital stays and readmission. Within the provided dataset, I'm leveraging these studies to help create my hypothetical question and shape my approach in finding potential patient groups with a statistically significant chance for readmission outcomes.

After viewing the provided medical_clean.csv data set and accompanying data dictionary, there seems to be some patient groupings which are aligned with the research mentioned above. For instance, the following patient data fields: Initial patient admin days, Total Charges, and Initial Says (inpatient) both caught my attention and were underscored by the research mentioned above. While my initial feelings towards these variables might make them feel related, are they?

A1 - Research Question

Given the medical dataset provided, can we classify (label) if a patient will be readmitted or not?

A2 - Defined Goal

The goal of our analysis is to logically investigate the provided data set and, with evidence, support or reject the hypothesis. Some data will need to be converted from categorical

to numerical data types prior to processing. Our objective is to see how, if at all, any patient's data correlate with potential readmission.

B1 – Explanation of Classification Method

According to Bruce (2020) "When if comes to prediction, however, harnessing the results from multiple trees is typically more powerful than using just a single tree. In particular, the random forest and boosted tree algorithms almost always provide superior predictive accuracy and performance." While decision trees can be easier to understand, random forests are usually a better choice since they use an amalgamation of multiple tree analysis to create an ensemble of better predictions by averaging the probability of individual trees. Not only does a random forest sample the records but it also samples the variables.

B2 – Summary of Method Assumption

An assumption in random forests, as in decision trees, are that the sampling is representative. Another assumption, according to Vishalmendekarhere (2021) is that random forests are known as a non-parametric model. This means data distributions are assumed that they can't be defined in finite terms. By creating dimensions (permutations) closer to infinite, they are assumed to move closer to a more defined state.

B3 – Packages or Libraries List

The following Python libraries were used followed by their corresponding reason for use:

- Pandas – Used to import dataset and data analysis tasks.
- Numpy – Used for describing the data set and computing distances in KNN.
- Matplotlib – Used for viewing the testing and actual data as a scatter plot.

- Seaborn – Used for creating a heatmap when looking for null values in the original dataset and ggplot style graph matrix to help visualize univariate data.
- Sklearn – Used for preprocessing, model splitting, classification, and random forest tasks.

C1 – Data Preprocessing

Not much preprocessing was needed for the random forest. I did use Pandas '.get_dummies' method to convert categorical data to numerical. (Figure 1)

```
predictors = ['Age', 'Gender', 'VitD_levels', 'Doc_visits', 'vitD_supp', 'Initial_admin', \
                  'HighBlood', 'Stroke', 'Complication_risk', 'Overweight', 'Arthritis', \
                  'Diabetes', 'Hyperlipidemia', 'BackPain', 'Anxiety', 'Allergic_rhinitis', \
                  'Reflux_esophagitis', 'Asthma', 'Services', 'Initial_days', 'TotalCharge', \
                  'Additional_charges']
outcome = 'ReAdmis'

X = pd.get_dummies(pruned_df[predictors], drop_first=True)
y = pruned_df[outcome]

rf_all = RandomForestClassifier(n_estimators=n_estimators_selected, random_state=1)
rf_all.fit(X, y)

rf_all_entropy = RandomForestClassifier(n_estimators=n_estimators_selected, random_state=1,
                                        criterion='entropy')
print(rf_all_entropy.fit(X, y))
```

```
RandomForestClassifier(criterion='entropy', n_estimators=2000, random_state=1)
```

Figure 1 - Preprocessing Goal: Dummy Variables

C2 – Data Set Variables

- Age, VitD_levels, Doc_visits, vitD_supp, Initial_days, TotalCharge, Additional_charges, Initial_days – continuous
- ReAdmis, Gender, Initial_admin, HighBlood, Stroke, Complication_risk, Overweight, Arthritis, Diabetes, Hyperlipidemia, BackPain, Anxiety, Allergic_rhinitis, Reflux_esophagitis, Asthma, Services - categorical (converted later by '.get_dummies()' method)

C3 Steps for Analysis

Initially, the dataset was loaded using pd.read_csv('medical_clean.csv') and a data frame was created. Some exploratory data analysis was performed to familiarize myself to the data, look for missing values and view data statistics using df.describe(). I viewed univariate data using a ggplot style matrix and matplotlib. Next the data was split to train and test a decision tree model. This model was quite impressive (98% accurate) and provided insight into important variables (Figure 2).

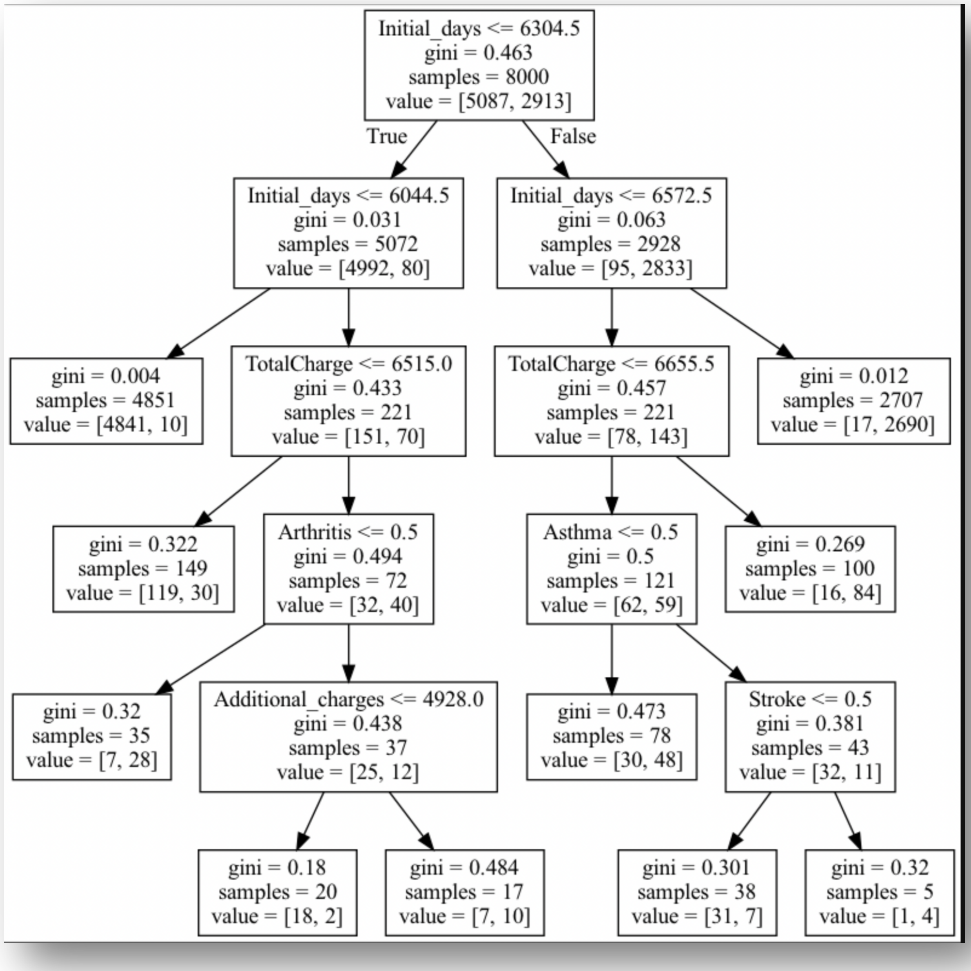

Figure 2 - Decision Tree

A random forest model was also created. With the predictors and outcome the same, I wanted to test against the simpler decision tree. I set my n_estimators and calculated an out-of-bag score (Figure 3) at approximately (98%), which correlates to the previous decision tree.

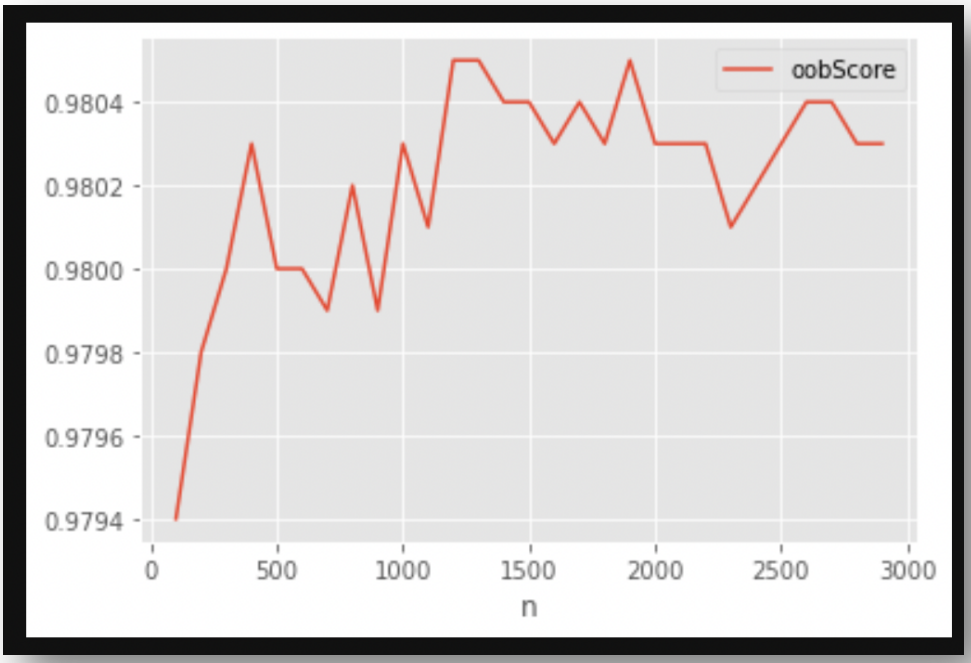

Figure 3 - Out-Of-Bag (OOB) Score

Since my target variable was initially a categorical data series, I converted this column (and others) to integers (0,1) using the Pandas '.get_dummies()' method. Next, the variables were sorted by score, showing an effect they had on random chance, 'Initial_days' being the most significant. Overall, this model also performed very well at 98 using the 'gini' criterion. A display of accuracy and gini decrease displays comparisons of the predictor variables used. The values are logged to show a more normalized comparison.

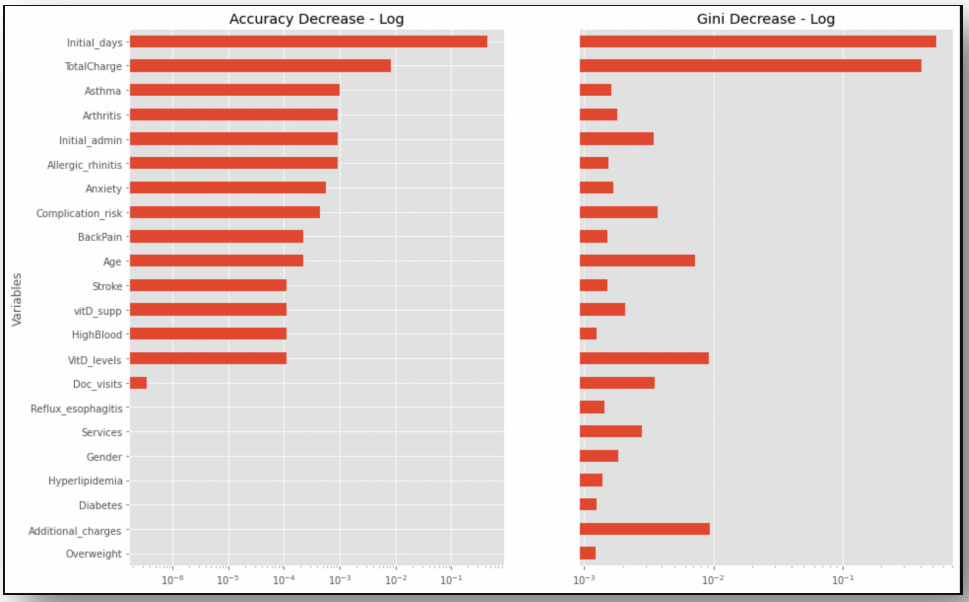

Figure 4 - Feature Importance – Log

C4 – Cleaned Data Set

The cleaned and reduced data set was saved to "final_cleaned_dataset.csv".

D1 – Splitting the Data

Next, the data was split into training (80%) and testing sets (20%) to evaluate the decision tree model while the random forest split was 70% and 30% respectively.

The following data sets are provided:

- medical_clean.csv (original format)
- final_cleaned_dataset.csv
    - These three columns have one target and two predictor series, this file is in a pre-split format

- Decision tree data sets:
    - d_tree_test.csv
    - d_tree_train.csv
- Random Forest data sets:
    - r_forest_train_X.csv
    - r_forest_train_y.csv
    - r_forest_valid_X.csv
    - r_forest_valid_y.csv

D2 – Output and Intermediate Calculations

Initial accuracy of the decision tree and random forest models were 98%. The only tweaking I did to the data was with the random forest when adjusting the n_estimator. The data seemed to flatten out at 1500 but I kept the 3000 estimations to verify. Both models seemed to rely heavily on the 'Initial_days' feature series.

D3 – Code Execution

Code is located in the "JWillis_D209_Data_Mining_PA2.ipynb" document.

E1 – Accuracy and MSE

The Accuracy Score of the decision tree was scored at 98%. Using a random forest and manipulating the n_estimators, I was able to match the decision tree score of 98% as well. The OOB Accuracy Score was 97.8%. A combined display of a confusion matrix, classification report and accuracy score helps measure performance of our classification model (Figure 5).

```
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score

print(confusion_matrix(y_test,y_pred))
print(classification_report(y_test,y_pred))
print(accuracy_score(y_test, y_pred))
```

```
[[1862   37]
 [  33 1068]]
              precision    recall  f1-score   support

           0       0.98      0.98      0.98      1899
           1       0.97      0.97      0.97      1101

    accuracy                           0.98      3000
   macro avg       0.97      0.98      0.97      3000
weighted avg       0.98      0.98      0.98      3000

0.9766666666666667
```

Figure 5 - Classification Model Accuracy

E2 – Results and Implications

The model is highly accurate. Looking at Figure 3 and 4, you can see that this model reaches 98% accuracy and is heavily correlated to a patients initial stay in days. As the patient's initial stay is longer and their total charge is higher, the chance that they are readmitted is higher.

E3 – Limitation

One limitation of random forests is that results can be difficult to explain when a more simple model, like decision trees, are much easier. Random forests blur understanding by the sheer number of permutations, making it difficult to understand "why" an outcome is accurate. This black box understanding can put off stakeholders who might not understand how random forests are calculated.

E4 – Course of Action

Our model is highly predictive of patient readmissions rates. My recommendation would be to focus on researching and identifying patient's threshold criteria's for their initial

administration stay as these predictor variables influenced our prediction model to predict readmission with a high accuracy rate. By focusing on this predictor, we should be able to anticipate if a patient will have a higher likelihood of readmission within 30 days of their initial inpatient stay.

F – Panopto Demonstration

Panopto video Will be uploaded once report is returned.

G – Sources for Third-Party Code

- Help using Markdown: https://www.markdownguide.org/basic-syntax/
- Help to see ALL columns: https://stackoverflow.com/questions/24524104/pandas-describe-is-not-returning-summary-of-all-columns
- Help to create a better histogram design: https://mode.com/example-gallery/python_histogram/
- Matplotlib Help: https://matplotlib.org/2.1.2/api/_as_gen/matplotlib.pyplot.plot.html
- Multiple ways to conduct ANOVA: https://www.marsja.se/four-ways-to-conduct-one-way-anovas-using-python/
- Numpy Help: https://numpy.org/doc/stable/
- Pandas Help: https://pandas.pydata.org/docs/user_guide/index.html#user-guide
- Python Help: https://docs.python.org/3.9/library/index.html
- Scipy.stats Help: https://docs.scipy.org/doc/scipy/reference/tutorial/stats.html

References

Bruce, P., C., & Bruce, A (2020). Practical Statistics for Data Scientists: 50+ Essential Concepts Using R and Python. O'Reilly Media. https://learning.oreilly.com/library/view/practical-statistics-for/9781492072935/titlepage01.html

Gert P Westert, Ronald J Lagoe, Ilmo Keskimäki, Alastair Leyland, Mark Murphy, An international study of hospital readmissions and related utilization in Europe and the USA, Health Policy, Volume 61, Issue 3, 2002, Pages 269-278, ISSN 0168-8510, https://doi.org/10.1016/S0168-8510(01)00236-6. (https://www.sciencedirect.com/science/article/pii/S0168851001002366)

Larose, D., C., & Larose, D., T. (2019). Data Science Using Python and R. Wiley. https://www.wiley.com/en-us/Data+Science+Using+Python+and+R-p-9781119526810

Schuller K. A. (2020). Is obesity a risk factor for readmission after acute myocardial infarction? *Journal of healthcare quality research*, *35*(1), 4–11. https://doi.org/10.1016/j.jhqr.2019.09.002

Vishalmendekarhere (2021). It's All About Assumptions, Pros & Cons. Medium.com https://medium.com/swlh/its-all-about-assumptions-pros-cons-497783cfed2d