Performance Assessment 2 – Revision 1

D209 – Data Mining I

Jessica Hosey

MSDA, College of Information Technology

Western Governors University

July 26th, 2024

Part I: Research Question

A.1. Propose one question relevant to a real-world organizational situation that you will answer using one of the following prediction methods: decision trees, random forests, or advanced regression.

Using decision trees, can we predict whether a patient will be readmitted to the hospital a month later based on the other factors within the dataset?

A.2. Define one goal of the data analysis.

As stated in my previous submission for D209 Task1 (Hosey 2024), Knowing why a patient returns to the hospital a month later might help public health officials plan and provide information about ailments affecting people in the local community. In addition, hospitals knowing what precursors may label a patient as needing more care later would allow them to be more thorough in their current hospital visit versus having the patient return with the same or similar issue. Obviously, in this dataset, there are people with children, so parents might visit the hospital more often if they have children than if they didn't. However, this dataset does not contain information about patients under 18. Ultimately, this research project aims to see if any factors (Diabetes, Stroke, High Blood Pressure, etc.) could label or predict whether a patient will be readmitted to the hospital within a month of their last visit.

Part II: Method Justification

B.1. Explain how the prediction method you chose analyzes the selected data set. Include expected outcomes.

Decision Trees were used for how easy it is to employ on a dataset and to understand for most individuals. This method is easy to understand as it creates a tree structure with nodes and responds to questions with an if or else response (Analytics Vidhya 2021). The SelectKBest will select the best features to help us start our decision tree. After that, it will create nodes off that if or else question as the branches. Each new branch will start a new section of data that will be asked again and form new branches or subsets of the data (Analytics Vidhya 2021). The algorithm will continue creating new sections of the "tree" until there are no more features to ask questions. So essentially, without any hyperparameter tuning or changes to the model, the decision tree will continue grouping the data until it runs out. Hyperparameter tuning lets you decide when the tree will stop "branching" out. This method is perfect for this dataset as most variables are categorical and have a Yes or No response. We can create a simple tree that shows which features or variables are responsible for patient readmissions. The expected outcomes for this will be a 2-3 level decision tree with two responses (yes or no) after the nodes, giving us the average probabilities for those possible outcomes. Ultimately, it will allow us to gain insights into why patients might be readmitted one month after their previous visit to the hospital.

B.2. Summarize one assumption of the chosen prediction method.

Superficial relationships must exist between the dependent and independent variables. This means a question must be answered with simple responses, going left or right, not complex ones requiring more analysis (KDnuggets 2020).

B.3. List the packages or libraries you have chosen for Python and justify how each item supports the analysis.

- Pandas: CategoricalDtype and loading the dataset to the anaconda jupyter environment.

- NumPy: to perform general mathematical calculations

- Matplotlib and Seaborn for visualizations

- Sklearn:

    train_test_split – splits data into training and testing datasets

    Preprocessing (used MinMaxScaler() to standardize the numeric values) and OneHotEncoder (to encode the categorical data)

    GridSearchCV – model selection

    confusion_matrix – metrics on model

    roc_auc_score – the area below the curve score

    roc_curve – visualization of the curve

    classification_report for data manipulation to perform several Machine Learning Algorithms and to fine-tune/evaluate our model.

    DecisionTreeClassifier – Decision Tree model creation

    Accuracy_Score – calculated accuracy score of the model

    Mean_Squared_Error – calculation of mean squared error of the model

Part III: Data Preparation

C.1. Describe one data preprocessing goal relevant to the prediction method from part A1.

Many preprocessing methods were used to prepare the dataset for analysis. One method needed to complete the decision tree classification was to encode the categorical data to represent 1 for a Yes response and 0 for a No response. This allows the human response to be represented as a numerical value. This allows the decision tree to see the data, understand what a Yes or No response entails, and direct the model to the simple responses and the following question (if there is another one).

C.2. Identify the initial data set variables used to analyze the prediction question from part A1 and group each variable as numeric or categorical.

Variables	Classification
Area (Rural, Suburban, Urban)	Categorical
Number of Children	Numeric
Age	Numeric
Gender	Categorical

Readmission (Dependent Variable)	Categorical
Vitamin D levels	Numeric
Number of Dr Visits	Numeric
Meals Eaten Prior	Numeric
Soft Drink	Categorical
Initial Admission Reason (Observation, Elective, Emergency)	Categorical
High Blood Pressure	Categorical
Stroke	Categorical
Complication Risk (Low, Medium, High)	Categorical
Overweight	Categorical
Arthritis	Categorical
Diabetes	Categorical
Hyperlipidemia	Categorical
Anxiety	Categorical
Allergic Rhinitis	Categorical
Asthma	Categorical
Days hospitalized	Numeric

C.3. Explain the steps to prepare the data for the analysis. Identify the code segment for each step.

The same steps were used to prepare the data for D208 tasks for this dataset and D209 Task 1 (Hosey 2024). Most of this preparation includes remapping the ordinal data and the binary categories to numerical values needed to complete this task's decision tree classification method.

Once the data types were changed and the ordinal and binary data remapped were completed, I explored the data. The use of `describe()` and `value_counts()` was to make sure that the data was ready to be used for the classification method that I chose (decision tree). Then, dummy columns will be created using sklearn's `OneHotEncoder()`. This is a different method than my D208 submissions, as `GetDummies` was not the best for KNN Classification, and I assumed it would not be suitable for decision tree classification either. These dummy values and the columns I am interested in will remain in the dataset, and the columns I am not interested in will be dropped. After this, the data frame will be rechecked to ensure the columns I selected remain the same and have the correct data type to move forward.

After creating dummy values, I attempted to complete the next step; however, the code brought back errors stating that Nan values were present. This issue did not need to be resolved, but I felt the dataset would not be clean, and the decision tree classification might be invalid with null values. The use of `dropna()`, `iloc`, and `concat()` were used to drop those rows with Nan values to move forward. I did not use imputer as it would have messed with the data and created values that were not true to the dataset, so I chose to drop the rows instead.

C.4. Provide a copy of the cleaned data set.

See the attached files in my submission.

Part IV: Analysis

D.1. Split the data into training and test data sets and provide the file(s).

See the attached files in my submission.

D.2. Describe the analysis technique you used to analyze the data appropriately. Include screenshots of the intermediate calculations you performed.

Before creating the model, a few things need to be done, or the decision tree classifier from sklearn will not work. The first thing I did was define the parameters using `max_depth` and `min_samples_leaf`. The `max_depth` will determine the number of steps the decision tree will grow to (the number of questions asked to the model before stopping). The higher the steps, the more complex the decision tree could lead to overfitting the model. `Min_samples_leaf` tells the model how many "leaf" nodes are to split up (yes or no, left or right, etc.).

After the parameters are set for the model, I will instantiate the `DecisionTreeClassifier` followed by `GridSearchCV` to calculate the performance of the decision tree model. After deploying this, an AUC score and accuracy score were calculated to see how the model accurately predicted the model. This model had a high AUC score, so I felt that any more fine-tuning for this model would result in the same results. Therefore, no additional steps were used, like adaptive boosting, as the model performed well. Screenshots are on the next page.

```
#Hyper Parameter Tuning
#Define params_dt
params_dt = {
    'max_depth' : [1, 2, 3, 4, 5, 6, 7, 8, 9, 10],
    'min_samples_leaf' : [0.04, 0.06, 0.08, 0.10, 0.12, 0.14, 0.16, 0.18, 0.20, 0.22]
}

#Instantiate dt
initial_dt = DecisionTreeClassifier(random_state=42)

#Instantiate grid_dt
grid_dt = GridSearchCV(estimator=initial_dt,
                       param_grid=params_dt,
                       scoring='roc_auc',
                       cv=5,
                       n_jobs=-1)

#Fit grid search to training model
grid_dt.fit(X_train, y_train)

#Extract the best estimator
best_model = grid_dt.best_estimator_

#Predict values for test set
initial_y_pred = best_model.predict(X_test)

#Generate accuracy report for this model
acc_test = accuracy_score(y_test, initial_y_pred)
print('Test set accuracy of best decision tree: {:.2f}'.format(acc_test))

#Predict the test set probabilities of the positive class
initial_y_pred_proba = best_model.predict_proba(X_test)[:,1]

#Compute test_roc_auc
initial_roc_auc = roc_auc_score(y_test, initial_y_pred_proba)

#Print test_roc_auc
print('Test set ROC AUC score: {:.3f}'.format(initial_roc_auc))
```

```
Test set accuracy of best decision tree: 0.98
Test set ROC AUC score: 0.998
```

```
best_model
```

```
DecisionTreeClassifier(max_depth=4, min_samples_leaf=0.04, random_state=42)
```

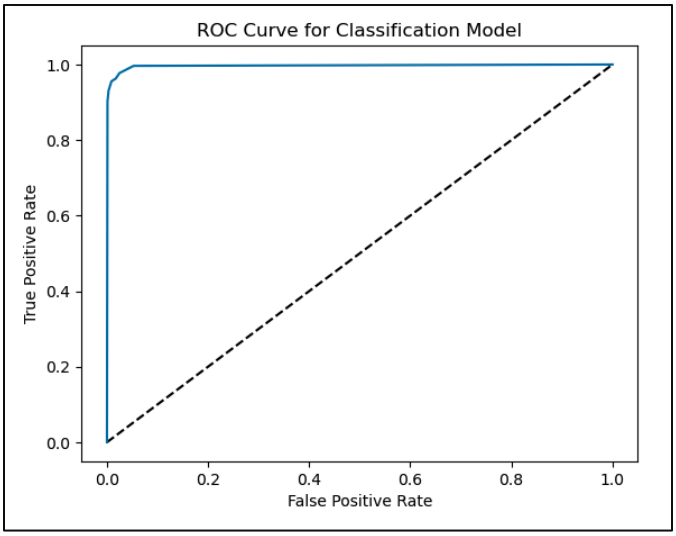
D.3. Provide the code used to perform the prediction analysis from part D2.

See the attached files in my submission.

Part V: Data Summary and Implications

```
#Predict test set labels
final_y_pred = initial_dt.predict(X_test)

# Evaluate acc_test
final_acc_test = accuracy_score(y_test, final_y_pred)
print('Test set accuracy of decision tree model: {:.2f}'.format(final_acc_test))

# Compute the probabilities of obtaining the positive class
final_y_pred_proba = initial_dt.predict_proba(X_test)[:,1]

# Evaluate test-set roc_auc_score
final_roc_auc = roc_auc_score(y_test, final_y_pred_proba)

# Generate Confusion Matrix
final_matrix = confusion_matrix(y_test, final_y_pred)
print("\nThe confusion matrix for this Decision Tree model:")
print("Predicted Not Readmitted | Predicted Readmitted")
print(f"                   {final_matrix[0]} Actual Not Readmitted")
print(f"                   {final_matrix[1]} Actual Readmitted")

# Generate mean_squared_error and root mean squared error
mse = mean_squared_error(y_test, final_y_pred)
root_mse = mse**(1/2)
print(f"The mean squared error of this model is: {mse}")
print(f"The root mean squared error of this model is: {round(root_mse, 2)}\n\n")

# Generate classification report
print(classification_report(y_test, final_y_pred))

print('\nThe Area Under the Curve (AUC) score of the decision tree model is: {:.2f}'.format(final_r
```

```
Test set accuracy of decision tree model: 0.98

The confusion matrix for this Decision Tree model:
Predicted Not Readmitted | Predicted Readmitted
                   [1866   33] Actual Not Readmitted
                   [  41 1060] Actual Readmitted
The mean squared error of this model is: 0.024666666666666667
The root mean squared error of this model is: 0.16


              precision    recall  f1-score   support

         0.0       0.98      0.98      0.98      1899
         1.0       0.97      0.96      0.97      1101

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


The Area Under the Curve (AUC) score of the decision tree model is: 1.00
```



E.1. Explain your prediction model's accuracy and mean squared error (MSE).

Screenshots of the results are provided above.

The accuracy score is computed; the higher the score, the better the model's predictions are. For my model, the accuracy score was 0.98 or 98%. If my accuracy score were lower than 80%, I would have completed changes in hyper-tuning or adaboost to approve the score.

Mean Squared Error is best when the MSE is near zero (Analytics Vidhya 2021). The higher the number, the higher the error in the model. For my model, the MSE was very low, 0.02467. This means that the relative errors of the model were also very low and did not require further analysis or hyper-tuning.

E.2. Discuss the results and implications of your prediction analysis.

Screenshots of results are provided above in Part E1.

The model results are highly accurate, with a 98% accuracy score and an AUC of 100%. The ROC curve does show some variation in the data in the top left of the curve, so it is not a 100% perfect model, but it is remarkably close. However, remember that the dataset is only from one day of operation. This leaves out many data points from the day before, the day after, or even the weeks around that time. This analysis is not helpful unless there is more data from previous days, weeks, months, or years (more ideal) of hospital operations.

E.3. Discuss one limitation of your data analysis.

As stated in previous tasks in D209 and D208, the main limitation of these analyses is that this dataset is from one day of hospital operation. This is not ideal, as many individuals could arrive at the hospital after a random car accident and might have seen a doctor for their yearly physical a week prior. So, if there were more than one day of data, this analysis would be more effective for hospitals in predicting insights into their patient base.

E.4. Recommend a course of action for the real-world organizational situation.

The organization needs to provide more data for this analysis to remain valid. The next steps would be to access more data on the hospital's operations of the last 3-5 years to predict when a patient might be readmitted accurately.

Part VI: Demonstration

F. Panopto Video

See the attached files in my submission.

G. Code Sources

Bowne-Anderson, Hugo. DataCamp. (n.d.). Supervised Learning with Scikit-Learn: Classification [Online course]. DataCamp. https://campus.datacamp.com/courses/supervised-learning-with-scikit-learn/classification-1?ex=1

Glen, Stephanie. Data Science Central. (2019, June 19). Comparing classifiers: Decision trees, KNN, Naive Bayes. Data Science Central. https://www.datasciencecentral.com/comparing-classifiers-decision-trees-knn-naive-bayes/#:~:text=Naive%20Bayes%20is%20a%20linear,Naive%20Bayes%20over%20K%2DNN

Hosey, Jessica. (2024). Performance Assessment 2, D208 – Predictive Modeling. [Unpublished manuscript]. WGU.

Hosey, Jessica. (2024). Performance Assessment 1, D209 – Data Mining I. [Unpublished manuscript]. WGU.

Stats Wire. (2021 March 13). Data Preprocessing 06: One Hot Encoding Python | Scikit Learn | Machine Learning [Video]. YouTube. https://www.youtube.com/watch?v=InZ0n2knz1E

H. Content Sources

KDnuggets. (2020, January). Decision Tree Algorithm, Explained. KDnuggets. https://www.kdnuggets.com/2020/01/decision-tree-algorithm-explained.html

Analytics Vidhya. (2021, October). Evaluation Metric for Regression Models. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/10/evaluation-metric-for-regression-models/

Analytics Vidhya. (2021, August). Decision Tree Algorithm. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/08/decision-tree-algorithm/