NVM2 Task 1: Classification Analysis

Sina Bozorgmehr

Western Governors University

NVM2 Task 1: Classification Analysis

Introduction

One of the industries that demands a well-prepared perspective based on scientific data collected daily from the industry, especially from patients, is health care. In this industry, predicting is a key aspect of data analysis. The rate of readmission is one of the challenges that hospitals and the medical industry are dealing with. According to Hosp. Readmission Reduction | CMS (2020) readmission defines as:

"Unplanned readmissions that happen within 30 days of discharge from the index (i.e., initial) admission.

Patients who are readmitted to the same hospital, or another applicable acute care hospital for any reason".

This is such a significant issue that the Centers for Medicare and Medicaid Services (CMS) penalizes hospitals that have a high readmission rate. According to federal data, nearly half of the country's hospitals will receive decreased payments for all Medicare patients due to a history of readmitting patients, many of which are still dealing with the financial consequences from the COVID-19 pandemic (*Medicare Fines Half of Hospitals for Readmitting Too Many Patients*, 2020). Hospitals must be prepared for this issue and avoid penalties by anticipating the rate of readmission and implementing relevant actions.

Research Question

This medical facility chain with patients around the country needs to know which patients will be readmitted following their initial discharge. The chain will be able to be prepared and plan to limit the likelihood of readmission by classifying patients into two groups depending on a wide variety of metrics available from the database.

We employ a K-Nearest Neighbor (KNN) model to divide the patients into two groups: those who will be readmitted after release and those who will not. We want to build a model that can categorize new patients based on their metrics and is reliable with new data.

Method Justification

KNN is a non-parametric model, which implies that the number of parameters in the model is not fixed and may increase as the number of training instances increases. KNN is a simple model for classification and regression. The titled neighbors are metric space representations of training examples. A metric space is a feature space in which all elements of a set's distances are defined. The value of the response variable for a test instance is approximated using these neighbors. The hyperparameter k determines the number of neighbors that can be considered in the estimation (Hackeling, 2017).

KNN only has one assumption: instances that are close to one other are likely to have comparable response variable values (Hackeling, 2017). The response variable in this study is Readmission, which is a binary variable with two levels: Yes or No. “Yes” indicates that a patient was readmitted within a month of discharge from the hospital.

For the analysis, Python is utilized, with various libraries added to make coding and calculations easier. Pandas is used to manage data in data frames, and NumPy is used to do algebraic calculations. The KNN model is also created with the Sci-Kit Learn module. For data visualization, Matplotlib and Seaborn are utilized.

Data Preparation

The data was first imported into a Pandas data frame using *pandas.read_csv* (Figure 1), and the null values and total number of entries were validated. After that, a graph was made to show the number of patients who were readmitted vs those who were not (Figure 2). As seen in

Figure 3, the readmission rate is around 37%. The next step was to remove those factors from the data set that were not relevant to our investigation, primarily demographics (Figure 4). Two lists were created, one with continuous data and the other with categorical variables (Figure 5). The categorical variables must be turned into dummy variables before model fitting can begin. This was accomplished with the help of *pandas.get_dummies* (Figure 6). The cleaned dataset was then saved as *cleaned_dataset.csv* (Figure 7) available as attachment. Before beginning the data analysis, all the variables except the ReAdmis, were assigned to X as the feature set and ReAdmis was assigned to y as dependent variable (Figure 8).

Figure 1

Loading the dataset into a pandas data frame.

```
#creating the raw pandas dataframe
df = pd.read_csv("C:/Users/bozor/Documents/WGU MSDA/Data Mining I/medical_clean.csv")
df.head()
```

Figure 2

Readmission counts in the patients.

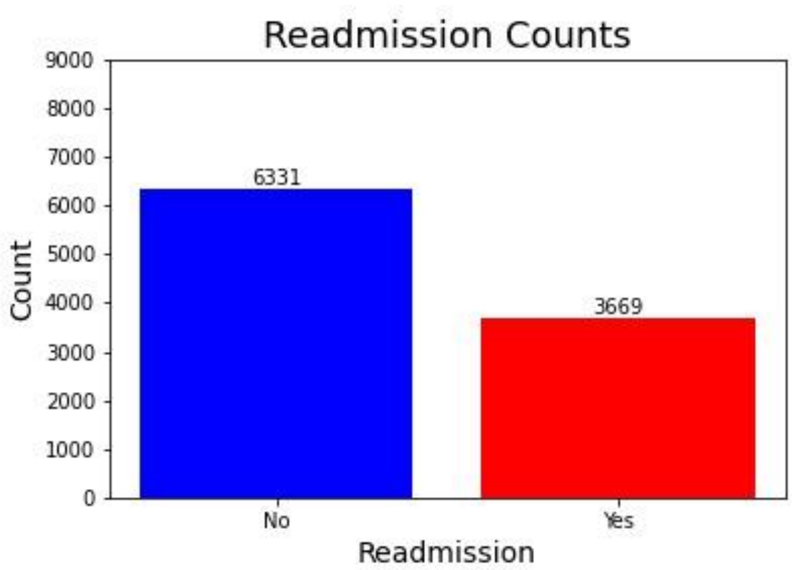


Figure 3

Readmission rate in patients.

```
# Readmission Rate

readmis_rate = round(len(df[df['ReAdmis'] == 'Yes']) / len(df), 2)
print(readmis_rate)
```

```
0.37
```

Figure 4

Removing the unimportant variable from the dataset.

```
#eliminating variables that have no importance in our analysis

column = ['CaseOrder', 'Customer_id', 'Interaction', 'UID', 'City', 'State', 'County',
          'Zip', 'Lat', 'Lng', 'Population', 'Area', 'TimeZone','Job'
         ]

new_df = df.drop(columns=column)
new_df.head()
```

Figure 5

Separating variables into continuous and categorical.

```
#Creating two lists, containing continuous and categorical variables

cont_var = ['Children', 'Age', 'Income','VitD_levels', 'Doc_visits', 'Full_meals_eaten',
            'vitD_supp', 'Initial_days', 'TotalCharge', 'Additional_charges', 'Item1',
            'Item2', 'Item3', 'Item4', 'Item5', 'Item6', 'Item7', 'Item8']
cat_var = [i for i in new_df.columns if i not in cont_var]

print('Continuous variables are:\n\n {} \n\n and Categorical variables are:\n\n {}'.format(cont_var, cat_var))
```

```
Continuous variables are:

 ['Children', 'Age', 'Income', 'VitD_levels', 'Doc_visits', 'Full_meals_eaten', 'vitD_supp', 'Initial_days', 'TotalCharge',
'Additional_charges', 'Item1', 'Item2', 'Item3', 'Item4', 'Item5', 'Item6', 'Item7', 'Item8']

 and Categorical variables are:

 ['Marital', 'Gender', 'ReAdmis', 'Soft_drink', 'Initial_admin', 'HighBlood', 'Stroke', 'Complication_risk', 'Overweight',
'Arthritis', 'Diabetes', 'Hyperlipidemia', 'BackPain', 'Anxiety', 'Allergic_rhinitis', 'Reflux_esophagitis', 'Asthma', 'Serv
ices']
```

Figure 6

Converting categorical variables to numeric variables.

```
# creating dummy variables for all categorical features

cleaned_df = pd.get_dummies(new_df, drop_first=True)
cleaned_df.rename(columns={'ReAdmis_Yes':'ReAdmis'}, inplace=True)
cleaned_df.head()
```

Figure 7

Storing the cleaned dataset into a csv file.

```
# Saving the cleaned dataset into a csv file.

cleaned_df.to_csv('cleaned_dataset.csv', index=False)
```

Figure 8

Separating dependent and independent variables.

```
#Seaparating independent and dependent variables

X = cleaned_df.drop(columns='ReAdmis')
y = cleaned_df[['ReAdmis']]
y = y.values.reshape(-1,)
print('X shape: {} \ny shape: {}'.format(X.shape, y.shape))
```

```
X shape: (10000, 43)
y shape: (10000,)
```

Analysis

Using the *train_test_split*, 30% of the data were hold out for evaluation and the training was done using the remaining 70% (Figure 9). *GridSearchCV* was used to select the best hyper-parameter, the number of neighbors, for the estimator. Also, recall chose as the scoring method since we are most interested in classifying the readmitted patients correctly than achieving an overall accuracy (Figure 10). Next, the test scores were plotted against their corresponding n_neighbors (Figure 11) to select a range with the highest recall score. As seen in the figure 11, the model achieved the highest score between 15 to 30 neighbors. Therefore, the grid search range is cut shorter to identify the best hyper-parameter for the model (Figure 12). The grid search suggests that the best recall score, 0.96, is achieved with 15 neighbors (Figure 13).

Figure 9

Holding out 30% of data for evaluation.

```
# Holding out 20% of the data for final evaluation

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=777, stratify=y )
```

Figure 10

Grid search to find the best hyper-parameter.

```
# Grid Search the K_Neighbors model and fiiting the model to the training set

from sklearn.neighbors import KNeighborsClassifier as KNN
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import make_scorer, recall_score

score = make_scorer(recall_score)

knn = KNN()
h_param = {'n_neighbors' : np.arange(1,100,5).tolist()}

cv = GridSearchCV(knn, param_grid=h_param, cv=10, scoring = score)

cv.fit(X_train, y_train)
```

Figure 11

Recall score vs. the number of neighbors.

```
#Plotting the number of neighbors aginst the score
Run this cell
mts = cv.cv_results_['mean_test_score'].tolist()
plt.plot(np.arange(1,100,5).tolist(), mts)
plt.xlabel('Neighbors', fontsize=14)
plt.ylabel('Recall Score', fontsize=14)
plt.show()
```

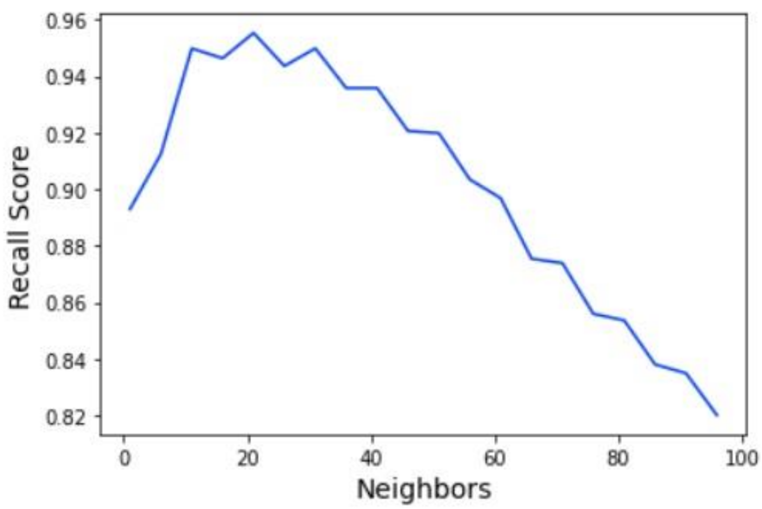


Figure 12

Repeating the grid search with a shorter range of neighbors.

```
#Selecting a shorter range of the Neighbors based on the result of last codes

knn = KNN()
h_param = {'n_neighbors' : np.arange(15,30).tolist()}

cv = GridSearchCV(knn, param_grid=h_param, cv=10, scoring=score)

cv.fit(X_train, y_train)
```

Figure 13

The best number of neighbors is 15.

```
# Getting the best n_neighbors and the corresponding score:

print('The best hyper_parameter is {} with the recall score of {:.3f}.'.format(cv.best_params_, cv.best_score_ ))
```

```
The best hyper_parameter is {'n_neighbors': 15} with the recall score of 0.958.
```

Data Summary and Implications

To evaluate the model on the unseen data, a classification report was run on the holdout data. As the results shows (Figure 14), the recall is almost the same in the unseen data that proves the model is reliable. To elaborate the power of this classification model, a confusion matrix is created in Figure 15. Area under the curve (AUC) was calculated and plotted as seen in figure 15. The resulting AUC, 0.97, completes the accuracy of the model. Using this model, the Hospital executives have a better understanding of their readmission rate and are able to predict which patients might be readmitted within a month of discharge based on their metrics. The high recall score of the model will give them an increased certainty when dealing with patients with potential readmission risk. In order to decrease the rate of readmission, hospital can implement a post-discharge monitoring of the patients classified with readmission risk. This monitoring could include daily vitals report measured and sent by patients and weekly in person check ups to make sure that the subject is recovering well.

Figure 14

Classification report on unseen data.

```
# Classification report

from sklearn.metrics import classification_report

y_pred = cv.best_estimator_.predict(X_test)
print(classification_report(y_test, y_pred, labels=[1,0]))
```

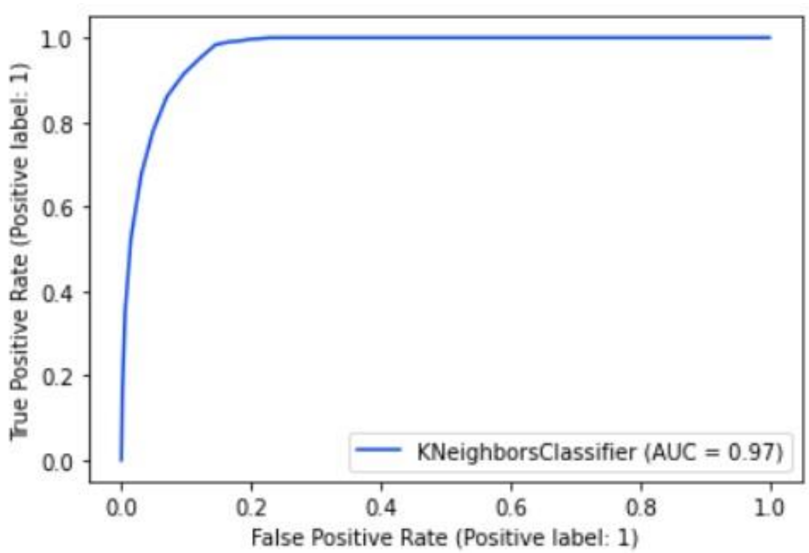
```
              precision    recall  f1-score   support

           1       0.82      0.95      0.88      1101
           0       0.97      0.88      0.92      1899

    accuracy                           0.91      3000
   macro avg       0.89      0.92      0.90      3000
weighted avg       0.91      0.91      0.91      3000
```

Figure 15

Confusion matrix.

Figure 16

AUC plot.



And the final word is that although the model accuracy in classification of readmitted patients is fantastic, but the data lacks the specific hospital that each patient was treated in. This is a huge factor, since each hospital has different policies and guidelines for discharging a patient that could heavily affect the readmission rate.

Sources

The dataset used in this analysis was acquired form WGU portal:

https://access.wgu.edu/ASP3/aap/content/d9rkejv84kd9rk30fi2l.zip

References

Hackeling, G. (2017). *Mastering Machine Learning with Scikit-Learn, Second Edition*. Van Haren Publishing.

Hosp. Readmission Reduction | CMS. (2020, August 11). Centers for Medicare and Medicaid Services. https://www.cms.gov/Medicare/Quality-Initiatives-Patient-Assessment-Instruments/Value-Based-Programs/HRRP/Hospital-Readmission-Reduction-Program

Medicare fines half of hospitals for readmitting too many patients. (2020, November 3). FierceHealthcare. https://www.fiercehealthcare.com/hospitals/medicare-fines-half-hospitals-for-readmitting-too-many-patients