**PERFORMANCE ASSESSMENT TASK 2:**

**D209 – Data Mining I**

Michael Lawson

Western Governors University

D209 – Data Mining I

Festus Elleh

04/05/2022

Michael C. Lawson | Student ID: #001479557 | Program: MS Data Analytics (5/1/21) |

Program Mentor: Emil Stoica | 859-265-1888 (EST) | [mlaws77@wgu.edu](mailto:mlaws77@wgu.edu)

**Part I:**

A.  Purpose:

1.  Real world question and prediction method.

Can the data be used to accurately predict which category of ‘Complication\_risk’ a patient should be classified? The random forest prediction method will be used.

2.  One goal of the analysis:

The goal will be to use the random forest prediction method to classify patients by whether they are a “Low”, “Medium”, or “High” complication risk. In the medical data set, the column ‘Complication\_risk’ reveals which category patients belong to, and random forest will predict which category a patient belongs to when new data comes into the data set.

**Part II: Method Justification**

B.

1.  How the method works:

The “random forest” method is an algorithm that classifies based on the results of many decision trees. It randomly selects features and uses the bagging method to find the best results of predictions from many samples.

2.  O**ne** assumption

Random forest assumes that the sample data is representative of new data coming in.

3.  Python packages

|  |  |  |  |
| --- | --- | --- | --- |
| **Library** | **Module** | **Package** | **Notes** |
| pandas |  |  | fast, powerful data analysis & manipulation |
| numpy |  |  | math functions |
| matplotlib | pyplot |  | visualization of data |
| seaborn |  |  | visualizing data |
| sklearn | model\_selection | train\_test\_split | divide rows into training and test groups |
| sklearn | model\_selection | GridSearchCV | hyperparameter tuning |
| sklearn | ensemble | RandomForestRegressor | create the random forest model |
| sklearn | metrics | mean\_squared\_error | to calculate the mean squared error |
| sklearn | metrics | accuracy\_score | to calculate accuracy of the model |
| warnings | filterwarnings |  | to remove filter warnings |

**Part III: Data Preparation**

C.

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

For preprocessing, all object values will be changed to numeric, such as changing the “Low”, “Medium” and “High” categories of the column ‘Complication\_risk’ to “0”,”1”, and ”2”, respectively.

2.  Variables

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Complication\_risk |  | Categorical | Additional\_charges | Continuous |
| Children |  | Categorical | Item1 | Categorical |
| Age |  | Categorical | Item2 | Categorical |
| Income |  | Continuous | Item3 | Categorical |
| VitD\_levels |  | Continuous | Item4 | Categorical |
| Doc\_visits |  | Categorical | Item5 | Categorical |
| Full\_meals\_eaten |  | Categorical | Item6 | Categorical |
| vitD\_supp |  | Categorical | Item7 | Categorical |
| Soft\_drink |  | Categorical | Item8 | Categorical |
| HighBlood |  | Categorical | Services\_Blood Work | Categorical |
| Stroke |  | Categorical | Services\_CT Scan | Categorical |
| ReAdmis |  | Categorical | Services\_Intravenous | Categorical |
| Overweight |  | Categorical | Services\_MRI | Categorical |
| Arthritis |  | Categorical | Marital\_Divorced | Categorical |
| Diabetes |  | Categorical | Marital\_Married | Categorical |
| Hyperlipidemia |  | Categorical | Marital\_Never Married | Categorical |
| BackPain |  | Categorical | Marital\_Separated | Categorical |
| Anxiety |  | Categorical | Marital\_Widowed | Categorical |
| Allergic\_rhinitis |  | Categorical | Gender\_Female | Categorical |
| Reflux\_esophagitis |  | Categorical | Gender\_Male | Categorical |
| Asthma |  | Categorical | Gender\_Nonbinary | Categorical |
| Initial\_days |  | Continuous | Initial\_admin\_Elective Admission | Categorical |
| TotalCharge |  | Continuous | Initial\_admin\_Emergency Admission | Categorical |
|  |  |  | Initial\_admin\_Observation Admission | Categorical |

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

*# import all possible packages useful for multiple linear regression*

*import pandas as pd*

*import numpy as np*

*import sklearn*

*import matplotlib.pyplot as plt*

*from sklearn.model\_selection import train\_test\_split, GridSearchCV, cross\_val\_score*

*from sklearn.ensemble import RandomForestRegressor*

*from sklearn import metrics*

*from sklearn.metrics import mean\_squared\_error as MSE*

*import warnings*

*warnings.filterwarnings("ignore")*

*#Import data set from hard drive*

med **=** pd**.**read\_csv(r"C:\Users\mlaws\OneDrive - Western Governors University\Documents\WGU\D208\medical\_clean.csv", skiprows**=**0, delimiter**=**",")

*#Examine the list of variables, their data type and the shape of the data*

med**.**info()

*#print the header*

med**.**head()

*#change the values of the ordinal categorical variable to numeric ranks*

med['Complication\_risk']**.**unique()

med['Complication\_risk'] **=** med['Complication\_risk']**.**map({'Low':0,'Medium':1,'High':2})

*#change all binary categorical values to 0,1*

med['ReAdmis'] **=** med['ReAdmis']**.**map({'No':0, "Yes":1})

med['Soft\_drink'] **=** med['Soft\_drink']**.**map({'No':0, "Yes":1})

med['HighBlood'] **=** med['HighBlood']**.**map({'No':0, "Yes":1})

med['Stroke'] **=** med['Stroke']**.**map({'No':0, "Yes":1})

med['Overweight'] **=** med['Overweight']**.**map({'No':0, "Yes":1})

med['Arthritis'] **=** med['Arthritis']**.**map({'No':0, "Yes":1})

med['Diabetes'] **=** med['Diabetes']**.**map({'No':0, "Yes":1})

med['Hyperlipidemia'] **=** med['Hyperlipidemia']**.**map({'No':0, "Yes":1})

med['BackPain'] **=** med['BackPain']**.**map({'No':0, "Yes":1})

med['Anxiety'] **=** med['Anxiety']**.**map({'No':0, "Yes":1})

med['Allergic\_rhinitis'] **=** med['Allergic\_rhinitis']**.**map({'No':0, "Yes":1})

med['Reflux\_esophagitis'] **=** med['Reflux\_esophagitis']**.**map({'No':0, "Yes":1})

med['Asthma'] **=** med['Asthma']**.**map({'No':0, "Yes":1})

*#Create dummy columns for the Services column*

med **=** pd**.**get\_dummies(med, prefix**=**['Services','Marital','Gender', 'Initial\_admin'], prefix\_sep**=**'\_', dummy\_na**=False**, columns**=**["Services","Marital","Gender","Initial\_admin"])

*#Check whether dummy columns were created*

print(med**.**keys())

*#Create Data and Target selections for X and y*

X **=** med**.**drop(axis**=**1, columns**=**['CaseOrder', 'Interaction', 'Customer\_id', 'UID', 'State','County', 'Zip', 'City', 'Lat', 'Lng', 'Population', 'TimeZone', 'Area', 'Job', 'ReAdmis',])

y **=** med['ReAdmis']

4.  Cleaned data

#Export prepared data

med.to\_csv(r"C:\Users\mlaws\OneDrive - Western Governors University\Documents\WGU\D209\medical\_prepared\_D209\_T2.csv")

**Part IV: Analysis**

D.  Perform the analysis:

1.  Split data and provide the files.

#Split the data into training and test groups

SEED = 944

X\_train,X\_test,y\_train,y\_test = train\_test\_split(X,y,test\_size=0.3, random\_state=SEED, stratify=y)

#Provide split data files

Training\_file = pd.concat([X\_train,y\_train])

Test\_file = pd.concat([X\_test,y\_test])

Training\_file.to\_csv(r"C:\Users\mlaws\OneDrive - Western Governors University\Documents\WGU\D209\D209\_T2\_training\_data.csv")

Test\_file.to\_csv(r"C:\Users\mlaws\OneDrive - Western Governors University\Documents\WGU\D209\D209\_T2\_test\_data.csv")

2.  Analysis screenshots:

The random forest regressor was initiated with the following output:

Text, letter

Description automatically generated

The model is fit to the training data:

Graphical user interface, text

Description automatically generated with medium confidence

The RMSE tells how concentrated the data is around the line of best fit:



Visualize the most important explanatory variables:

Table

Description automatically generated

Perform grid search cross validation:

A screenshot of a computer

Description automatically generated with medium confidence

Find the best hyperparameters:



Run the tuned model on the test set and calculate RMSE:



3.  CODE for D2:

*#Initiate the Random Forest regressor*

rf **=** RandomForestRegressor(n\_estimators**=**300,min\_samples\_leaf**=**0.15, random\_state**=**SEED)

rf**.**get\_params()

*#Fit rf to training data*

rf**.**fit(X\_train, y\_train)

RandomForestRegressor(min\_samples\_leaf=0.15, n\_estimators=300, random\_state=944)

*#predict target and assign to y\_pred*

y\_pred **=** rf**.**predict(X\_test)

*#print the RMSE*

rmse\_test **=** MSE(y\_test,y\_pred)**\*\***(1**/**2)

print('Root mean squared error:{:.2f}%'**.**format(rmse\_test**\***100))

*#Visualize the most important explanatory variables*

importances\_rf **=** pd**.**Series(rf**.**feature\_importances\_,index**=**X**.**columns)

sorted\_importances\_rf **=** importances\_rf**.**sort\_values()

sns**.**set(rc **=** {'figure.figsize':(10,22)})

sorted\_importances\_rf**.**plot(kind**=**'barh', color**=**'lightgreen')

plt**.**show()

*#set up parameters for hypertuning*

params\_rf **=** {

'n\_estimators': [300,400,500],

'max\_depth': [4,6,8],

'min\_samples\_leaf': [0.2,0.4,0.6,0.8]

}

*#Perform grid search CV*

grid\_rf **=** GridSearchCV(estimator**=**rf, param\_grid**=**params\_rf, cv**=**3, scoring**=**'neg\_mean\_squared\_error', refit**=True**, verbose**=**1,n\_jobs**=-**1)

*#Fit grid search cv to training data*

grid\_rf**.**fit(X\_train, y\_train)

*#print the best hyperparameters*

best\_hyperparams **=** grid\_rf**.**best\_params\_

print('Best hyperparameters: \n', best\_hyperparams)

best\_model **=** grid\_rf**.**best\_estimator\_

y\_pred **=** best\_model**.**predict(X\_test)

rmse\_test **=** MSE(y\_test,y\_pred)**\*\***(1**/**2)

print('Test set RMSE of rf: {:.2f}%'**.**format(rmse\_test**\***100))

*#Export prepared data*

med**.**to\_csv(r"C:\Users\mlaws\OneDrive - Western Governors University\Documents\WGU\D209\medical\_prepared\_D209\_T2.csv")

**Part V: Data Summary and Implications**

E.  Summarize your data analysis by doing the following:

1.  Accuracy of model:

Since the target variable ‘Complication\_risk’ has more than two categories, this is not a logistic regression classification model that an accuracy score can be run on. Instead, the root mean squared error (RMSE) can be used to determine how well the model fits the data. The RMSE of the training data was an exact match to the RMSE of the test data, indicating the model is accurately predicting outcomes. The RMSE of this random forest model was 72.41% after the parameters were hyper-tuned. 72.41% RMSE indicates that this model has a strong ability to predict which category of ‘Complication\_risk’ and object belongs to.

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

72.41% RMSE indicates that the model is strong at categorizing ‘Complication\_risk’, but not perfect. The explanatory variables ‘Initial\_days’ and ‘Total\_charge’ were revealed to be the most important variables. A stronger model may be acquired by reducing the number of explanatory variables, maybe only including these two.

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

Random forest models can only predict using previously observed labels, so the predictions are limited to objects that closely match data already collected.

4.  Recommend a course of action for the real-world organizational situation from part A1 based on your results and implications discussed in part E2.

Since the RMSE of the training data was nearly an exact match to the RMSE of the test data, and since the RMSE is relatively high at 72.41%, the hospital can assume that the model predicts the complication risk of patients with a strong but not perfect confidence level. The hospital can make decisions on how to monitor patients and create warning systems that alert staff on whether they should prepare for complications.

**Part VI: Demonstration**

F. Panapto video

https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=4f029f9b-fc17-4f2c-80b9-ae6e0187323a

**References**

Machine Learning with Tree Based Models in Python. Elie Kawerk. DataCamp. https://campus.datacamp.com/courses/machine-learning-with-tree-based-models-in-python/classification-and-regression-trees?ex=1