MTH 522: Advanced Mathematical Statistics

Predicting Delay Until Patient Seeks Medical Treatment from Heart Health Dataset

03/19/2023

Issues

This report discusses using a logistic model to predict whether a patient will seek medical treatment within a certain number of days. Unlike linear regression models, logistic models predict the probability of an outcome, determining whether something is true or false. The report uses heart health medical data from anonymous patients, including information about their marital status, living situations, age, ethnicity, and symptoms rated on a scale. The data contains 18 factors or variables to help predict whether a patient seeks medical treatment before or after a certain number of days. However, there were issues in determining the most valuable variables to use in the predictions and choosing an appropriate delay time to maximize the use of the dataset. These were challenges encountered in developing the prediction model using this heart health dataset.

Findings

The dataset provided contains 406 rows and 20 columns, with only 5 missing values. These missing values were replaced with the mean value of their corresponding column. The accuracy of the model for the first task is 0.59 or 59%, indicating that people seeking medical attention within 2 days are about two-fifths compared to those who do not seek medical attention within 2 days.

For the second task, the mean delay days were calculated and used in a logistic regression model, resulting in an accuracy of 0.71 or 71%. This indicates that people seeking medical attention within the mean delay days are more than two-thirds compared to those who do not seek medical attention within the mean delay days.

In the final task, a logistic regression method was used to find the accuracy. The accuracy was found to be 0.62 or 62%, indicating that people seeking medical attention within 1 day are more than two-fifths compared to those who do not seek medical attention within 1 day.

Appendix A: Method

The dataset contains 18 factors (variables), including the predicted factor. The missing values were checked and replaced with their corresponding mean values using the "fillna()" function. A new column was added to the dataset, which based on the "delaydays" column, with a threshold of 2 (i.e., <=2=1 & >2=0). The "describe()" method was used to compute and display the

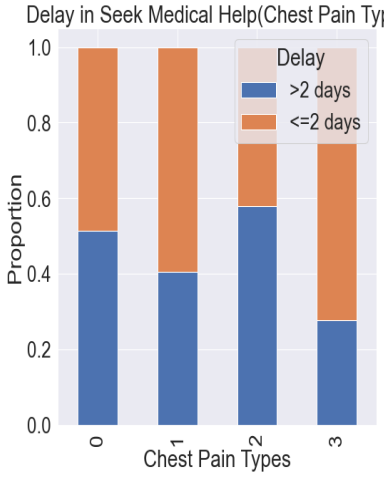
summary statistics for the dataset, and the "corr()" method was applied to find the correlation of each column. The dataset was then split into a training part and a testing part, with a train size of 0.75 and a test size of 0.25. The logistic regression method was applied, and the accuracy of this model was determined.

In Task 2, a new column was added to the dataset based on the mean value of the "delaydays" column, with a threshold of the mean value (i.e., <=meanvalue=1 & >meanvalue=0). The dataset was again split into a training part and a testing part, and the logistic regression method was applied to determine the accuracy of this model.

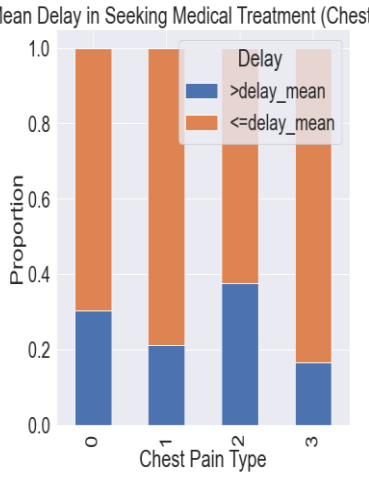
In the final task, a new column was added to the dataset based on the "delaydays" column, with a threshold of 1 (i.e., <=1=1 & >1=0). The dataset was again split into a training part and a testing part, and the logistic regression method was applied to determine the accuracy of this model.

Appendix B: Result

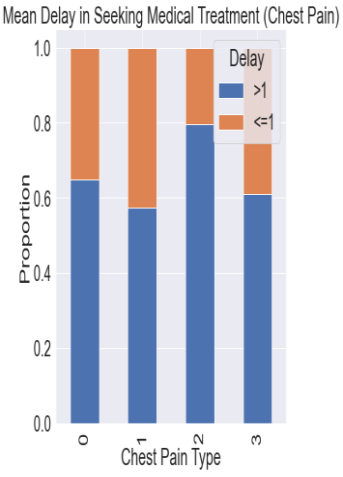
Bar Graphs for Chest pain



Task-1

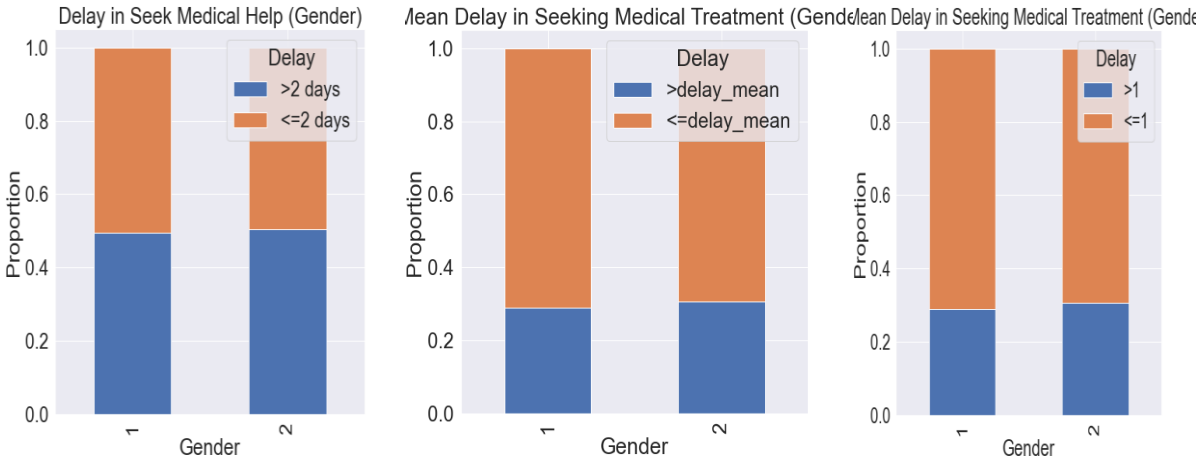


Task-2



Task-2

Bar Graph for Gender



Task-1 Task-2 Task-3

Appendix C: Code

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report

pd.set_option('display.max_columns', None)
data = pd.read_excel('Heart Health Data.xlsx')
data

data.info()

data.head()

missing_values = data.isnull().sum()
missing_values

data.mean()

data_new = data.fillna(data.mean())

missing_values = data_new.isnull().sum()
```

```
missing_values

data_new['Categorical Delay'] = np.where(data_new['delaydays'] <= 2,1,0)
data_new


data_new['Categorical Delay'].value_counts()

data_new.describe()

data_new.corr()

plt.figure(figsize=(50,50))
sns.set(font_scale=2.2)
sns.heatmap(data_new.corr(), cmap='Blues', annot=True, linewidths=1)
plt.show()


#Task 1
#Split Dataframe into training and testing parts

train_data, test_data = train_test_split(data_new, test_size=0.25,
random_state=43)

predictors = data_new.columns[:-2]
predictors

logistic_regression = LogisticRegression()
logistic_regression.fit(train_data[predictors], train_data['Categorical Delay'])
test_pred = logistic_regression.predict(test_data[predictors])

#Logistic Model Coefficients
coefficients = pd.DataFrame(logistic_regression.coef_, columns=predictors)
coefficients


#Classification Report
print(classification_report(test_data['Categorical Delay'], test_pred))


fig, ax = plt.subplots(figsize=(8,8))
```

```
data_new.groupby(['chestpain'])['Categorical
Delay'].value_counts(normalize=True).unstack().plot(kind='bar', stacked=True,
ax=ax)
plt.xlabel('Chest Pain Types')
plt.ylabel('Proportion')
plt.title('Delay in Seek Medical Help(Chest Pain Type)')
plt.legend(title='Delay', loc='upper right', labels=['>2 days', '<=2 days'])
plt.savefig('task1_heart.png')
plt.show()


fig, ax = plt.subplots(figsize=(8,8))
data_new.groupby(['Gender'])['Categorical
Delay'].value_counts(normalize=True).unstack().plot(kind='bar', stacked=True,
ax=ax)
plt.xlabel('Gender')
plt.ylabel('Proportion')
plt.title('Delay in Seek Medical Help (Gender)')
plt.legend(title='Delay', loc='upper right', labels=['>2 days', '<=2 days'])
plt.savefig('task1_gender.png')
plt.show()


# Task 2
delay_mean = data_new['delaydays'].mean()
delay_mean

data_new['categorical_delay_mean'] = np.where(data_new['delaydays'] <=
delay_mean, 1, 0)

data_new.info()

plt.figure(figsize=(50, 50))
sns.set(font_scale=2.2)
sns.heatmap(data_new.corr(), cmap='Blues', annot=True, linewidths=1)
plt.show()

train_data, test_data = train_test_split(data_new, test_size=0.25,
random_state=43)

predictors_1 =data_new.columns[:-3]
```

```
predictors_1

logistic_regression_1 = LogisticRegression()
logistic_regression_1.fit(train_data[predictors_1],
train_data['categorical_delay_mean'])

coefficients_1 = pd.DataFrame(logistic_regression_1.coef_,
columns=predictors_1)
coefficients_1

test_pred_1 = logistic_regression_1.predict(test_data[predictors_1])
print(classification_report(test_data['categorical_delay_mean'], test_pred_1))


fig, ax = plt.subplots(figsize=(8,8))
data_new.groupby(['chestpain'])['categorical_delay_mean'].value_counts(norm
alize=True).unstack().plot(kind='bar', stacked=True, ax=ax)
plt.xlabel('Chest Pain Type')
plt.ylabel('Proportion')
plt.title('Mean Delay in Seeking Medical Treatment (Chest Pain)')
plt.legend(title='Delay', loc='upper right', labels=['>delay_mean',
'<=delay_mean'])
plt.savefig('task2_heart.png')
plt.show()

fig, ax = plt.subplots(figsize=(8,8))
data_new.groupby(['Gender'])['categorical_delay_mean'].value_counts(normali
ze=True).unstack().plot(kind='bar', stacked=True, ax=ax)
plt.xlabel('Gender')
plt.ylabel('Proportion')
plt.title('Mean Delay in Seeking Medical Treatment (Gender)')
plt.legend(title='Delay', loc='upper right', labels=['>delay_mean',
'<=delay_mean'])
plt.savefig('task2_gender.png')
plt.show()

# Task 3
data_new['categorical_delay_oneday'] = np.where(data_new['delaydays'] <= 1,
1, 0)
data_new
```

```
plt.figure(figsize=(50, 50))
sns.set(font_scale=2.2)
sns.heatmap(data_new.corr(), cmap='Blues', annot=True, linewidths=1)
plt.show()

train_data, test_data = train_test_split(data_new, test_size=0.25,
random_state=43)

predictors_2 = data_new.columns[:-4]
predictors_2

logistic_regression_2 = LogisticRegression()
logistic_regression_2.fit(train_data[predictors_2],
train_data['categorical_delay_oneday'])

coefficients_2 = pd.DataFrame(logistic_regression_2.coef_,
columns=predictors_2)
coefficients_2
test_pred_2 = logistic_regression_2.predict(test_data[predictors_2])
print(classification_report(test_data['categorical_delay_oneday'], test_pred_2))

fig, ax = plt.subplots(figsize=(8,8))
data_new.groupby(['chestpain'])['categorical_delay_oneday'].value_counts(nor
malize=True).unstack().plot(kind='bar', stacked=True, ax=ax)
plt.xlabel('Chest Pain Type')
plt.ylabel('Proportion')
plt.title('Mean Delay in Seeking Medical Treatment (Chest Pain)')
plt.legend(title='Delay', loc='upper right', labels=['>1', '<=1'])
plt.savefig('task3_heart.png')
plt.show()

fig, ax = plt.subplots(figsize=(8,8))
data_new.groupby(['Gender'])['categorical_delay_mean'].value_counts(normali
ze=True).unstack().plot(kind='bar', stacked=True, ax=ax)
plt.xlabel('Gender')
plt.ylabel('Proportion')
plt.title('Mean Delay in Seeking Medical Treatment (Gender)')
plt.legend(title='Delay', loc='upper right', labels=['>1', '<=1'])
plt.savefig('task3_gender.png')
plt.show()
```