

**ANL252**

**Python for Data Analytics**

**End-of-course Assignment**

**July 2023 Presentation**

**Submitted by:**

|  |  |
| --- | --- |
| **Name** | **PI No.** |
| **Tan Pei Ling** | **W2211367** |

**Tutorial Group:**  **T03**

**Instructor’s Name:**  **Munish Kumar**

**Submission Date:**  **3/11/2023**

**Question 1**

The 3 pre-processing steps to clean and prepare dataset for use in Python:

1) First, we want to determine if there is any **duplicate data** and to decide if it should be deleted or treated. Duplicate data created by error can affect the reliability and results if they are included in data analysis.

# to import pandas library   
import pandas as pd  
MedCosts = pd.read\_csv("ECA.csv")  
MedCosts

# to find duplicates  
MedCosts.duplicated()

# to check the number of rows that are duplicates  
num\_duplicates = len(duplicates)

# to display the duplicated rows of data  
duplicates

A screenshot of a cell phone

Description automatically generated

We can to remove row 100 and 101 as they are the duplicated data of PersonID 100.

# to remove duplicate data in the entire file  
MedCosts.drop\_duplicates()

MedCosts

Row 100 and 101 have been removed.

# to check cleaned up DataFrame length

("Original DataFrame length:", len(MedCosts))  
  
MedCosts = MedCosts.drop\_duplicates()  
  
("DataFrame length after removing duplicates:", len(MedCosts))

('DataFrame length after removing duplicates:', 1338)

A screenshot of a computer screen

Description automatically generated

2) Secondly, it is common to find **missing values** in a dataset. Therefore, we want to find the missing values as they are ignored by statistical functions and may skew the results of analysis.

# to check existence of missing values  
MedCosts.isnull().any(axis = 0)

A screenshot of a computer

Description automatically generated

In the ‘age’ column, there are missing values.

# to confirm the number of rows with age missing  
missrow = MedCosts.isnull().any(axis =1)  
MedCosts.loc[missrow[missrow == True].index]

A screenshot of a table

Description automatically generated

Since the age ranges from teens to over 60, which is a wide range, one way to treat the missing values will be to delete them. Also, to get an average for the age will not provide a reliable analysis. The deleted number of values constitute about 9% of the total data, which is an acceptable percentage

# delete rows with missing values  
MedCosts.drop(index = missrow[missrow == True].index)

MedCosts.dropna(axis=0, inplace=True)

123 rows have been deleted. We have now, 1215 rows after removing duplicates and missing ‘age’ rows of data.

A screenshot of a data

Description automatically generated

3) Format ‘F’ and ‘M’ values in the ‘sex’ column to ‘female’ and ‘male’ respectively for consistency

MedCosts["sex"] = MedCosts["sex"].str.replace('F', 'female')  
MedCosts["sex"] = MedCosts["sex"].str.replace('M', 'male')

MedCosts

A screenshot of a medical data

Description automatically generated

Finally, the cleaned data will be saved to a new csv file for data visualisation and analysis.

# save cleaned data to a new file

MedCosts.to\_csv('Final\_MedCosts.csv', index=False)

(365 words)

Question 2

**First Plot**

# Create a histogram for the 'charges' column

import pandas as pd

Final\_MedCosts = pd.read\_csv('Final\_MedCosts.csv')

import matplotlib.pyplot as plt

plt.figure(figsize=(10, 6))

plt.hist(Final\_MedCosts['charges'], bins= range(0,70001,5000), rwidth = 0.8, color='orange', edgecolor='black')

plt.xlabel('Charges')

plt.ylabel('Frequency')

plt.title('Charges Distribution Histogram')

plt.xticks(ticks = range(0,70000,5000), labels = range(0,70000,5000))

plt.show()

A graph of orange bars

Description automatically generated  
 Figure 2.1 Histogram of Medical Costs

The histogram shows the distribution of 1215 individual’s medical costs from these four regions southeast, southwest, northeast and northwest of the US, with age group ranging between teenage years and up to age sixties It is positively-skewed and that tells us that the distribution of the medical costs is shifted to the left, indicating the majority of the costs billed by the health insurance occurring below $15000.

**Second Plot**

# Create a bar chart comparing smokers and non-smokers with charges by region

import matplotlib.pyplot as plt

# Group the data and calculate mean charges by region and smoker

bar\_data = Final\_MedCosts.groupby(['region', 'smoker'])['charges'].mean().unstack()

# Create the bar chart

plt.figure(figsize=(12, 6))

bar\_data.plot(kind='bar', colormap='Set3')

plt.xlabel("Region")

plt.ylabel("Mean Charges")

plt.title("Medical Costs by Region and Smoking Status")

plt.xticks(rotation=0)

plt.legend(title='Smoker', loc='upper left')

plt.show()

**A graph of different colored bars

Description automatically generated**

Figure 2.2 Bar Chart of Mean Medical Costs by Region and Smoking Status

The chart shows a distinctive difference of the mean medical costs billed by the health insurance to the smokers and non-smokers. The smokers by all regions paid more than 60% of medical costs compared to the non-smokers. This reflects the medical risks and higher medical costs associated with smoking (Truth Initiative, 2023). There is a higher percentage of about slightly more than 10% of smokers in the southeast compared with the rest of the regions. One of the reasons for the higher occurrence of smoking in this region may be linked to the lower tobacco tax there than in other states.

**Third Pot**

# Create a scatter plot comparing BMI vs. Charges by age group

Final\_MedCosts = pd.read\_csv('Final\_MedCosts.csv')

import matplotlib.pyplot as plt

# Create age groups

age\_bins = [20, 40, 60]

age\_labels = ['20-39', '40-59']

Final\_MedCosts['age\_group'] = pd.cut(Final\_MedCosts['age'], bins=age\_bins, labels=age\_labels)

# Create a scatter plot of BMI vs. Charges by age group

plt.figure(figsize=(12, 6))

colors = ['blue', 'red',]

for i, group in enumerate(age\_labels):

subset = Final\_MedCosts[Final\_MedCosts['age\_group'] == group]

plt.scatter(subset['bmi'], subset['charges'], label=f'Age Group: {group}', alpha=0.5, c=colors[i])

plt.xlabel('BMI')

plt.ylabel('Charges')

plt.title('Scatter Plot of BMI vs. Charges by Age Group')

plt.legend()

plt.show() **A diagram of red and blue dots

Description automatically generated**

Figure 2.3A scatter plot comparing BMI vs. Charges by age group

Two age groups’ BMI were used to construct this scatter plot. The younger age group paid lower medical costs as compared to the age group between 40 and 59. And majority of the individuals incurred health expenses below $1000.

From the second plot – bar chart – where we determine that smokers paid higher costs than non-smokers, we can extrapolate that the higher costs of $20000 and above incurred are paid by the smokers.

There is a positive correlation between BMI and medical costs. As BMI climbed, the medical costs also increased, indicating individuals with higher BMI may reach obesity range which could result in medical interventions and costs.

(495 words)

Question 3

In the second plot in question 2, a bar chart showing the medical costs paid by smokers versus non-smokers, we can see that the medical costs is a clear indication of the difference in the costs incurred by these two groups of people. Therefore, treating smoker as the dependent variable, we want to look most closely at the charges, and also the age and bmi of the group, to predict if the person is a smoker.

We will import libraries of pandas and matplotlib before analysing the data.

And after defining dependent features and dependent variable, we use the train\_test\_split function which has the training data set at 80% and testing data at 20%. Next, for simplicity purpose, we set hyperparameters to create a decision tree of max depth of 3, after which we predict the test data and evaluate the model performance. Accuracy score is used to calculate the accuracy.

# import modules

import pandas as pd

import matplotlib.pyplot as plt

# read data

data = pd.read\_csv('Final\_MedCosts.csv')

# Define features (x) and the dependent variable (y)

x = data[['age', 'bmi', 'charges']]

y = data['smoker']

# Preprocessing

from sklearn.model\_selection import train\_test\_split

x\_train, x\_test, y\_train, y\_test = train\_test\_split(x, y, test\_size=0.2, stratify=y, random\_state=40)

# Create a Decision Tree Classifier with specified hyperparameters

from sklearn.tree import DecisionTreeClassifier

model = DecisionTreeClassifier(max\_depth=3,)

# Train the classifier on the training data

model.fit(x\_train, y\_train)

# Make predictions on the test data

y\_pred = model.predict(x\_test)

# Calculate the accuracy of the model

from sklearn.metrics import accuracy\_score

accuracy = accuracy\_score(y\_true=y\_test, y\_pred=y\_pred)

(f'Accuracy: {accuracy}')

'Accuracy: 0.9300411522633745'

(256 words)

Question 4

# plot decision tree graph

from sklearn.tree import plot\_tree

plt.figure(figsize=(12, 7))

plot\_tree(model, feature\_names=x.columns.tolist(), class\_names=['non-smoker', 'smoker'], filled=True)

plt.show()

A diagram of a diagram

Description automatically generated

Question 5

Decision tree can be used to predict data in their traditional role and to be used effectively by selecting important features to make better predictions as they are more informative. We can explore the variables relationship and ascertain the dependency between different variable and how they affects the dependent variable for example, the smoker status.

In business, we may use decision tree to perform cost-benefit analysis by assessing the decision paths and their projected cost and benefits beyond just making predictions.

One useful result we can apply from the medical costs decision tree is to not just predict but to further better determine the cause of high and lower medical costs based on either bmi, ages, smoking status. This helps the medical professionals to look at ways to improve the lives of people.

However, there can be the problem of overfitting when it can perform well with training data but becomes unreliable when capturing key indications of the test data. It can be overcome with pruning through setting the hyperparameters and prevent the model from developing into a complicated one.

Outliers can significantly affect the structure of a decision tree as it can lead to a split and cause the interpretation to be veered and therefore not give a good general picture. Decision trees data can be imbalanced and prevent classification of classes in a more balanced and fair way.

Decision tree remains a useful tool for analysts other than its traditional role of prediction if suitable techniques are deployed to minimise unnecessary errors and misinterpretations.

(256 words)

References

Bailey, J. (2017). *Plagiarism and Programming: How to Code Without Plagiarizing*.Turnitin.

https://www.turnitin.com/blog/plagiarism-and-programming-how-to-code-without-plagiarizing-2

Obesity Medicine Association. (2023). *How Much Does Obesity Cost the U.S?*   
 https://obesitymedicine.org/health-economic-impact-of-obesity/

OpenAI. (2023). ChatGPT (August 3 version) [Large language model].

https://chat.openai.com/chat

Shaw, Z. A. (2017). Learn Python 3 the Hard Way. Pearson Technology Group.

https://suss.vitalsource.com/books/9780134693903

Truth Initiative. (2023). Why smoking remains higher in these Southern and Midwestern states. <https://truthinitiative.org/research-resources/smoking-region/why-smoking->

remains-higher-these-southern-and-midwestern-states