

**ANL252**

**Python for Data Analytics**

# **END OF COURSE ASSESSMENT**

**July 2023 Presentation**

**Submitted by:**

|  |  |
| --- | --- |
| **Name** | **PI No.** |
| **Hamreesh S/O Arivalagan** | **B2210812** |

**Tutorial Group: ­­­­­­­­­­­­ TV 03**

**Instructor’s Name: Prof. Munish**

**Submission Date: 03/11/2023**

# special IPython command to prepare the notebook for mathplotlib and other libraries

%matplotlib inline

SEED = 42

# import necessary libraries

import pandas as pd

import numpy as np

import matplotlib as mpl

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.impute import SimpleImputer

from sklearn.tree import DecisionTreeClassifier

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

from sklearn.preprocessing import OneHotEncoder, StandardScaler

from sklearn.preprocessing import LabelEncoder

from sklearn.compose import ColumnTransformer

from sklearn.tree import plot\_tree

df = pd.read\_csv("ECA.csv")

df.head()

df.shape



Before preprocessing the dataset, I imported all the necessary libraries for addressing various questions. I will explain the significance of these libraries and their relevant methods as we use them in the questions. First, I stored the 'ECA.csv' file in a Pandas DataFrame variable named ‘df’ using the `read\_csv()` method. Next, I took a glimpse of the dataset by using the Pandas’ library’s `head()` and shape methods. The dataset contains **1340 rows** and **8 columns**.

Question 1

* Handle missing or null values:

# check for missing values in each column

df.isna().sum()

A screenshot of a computer

Description automatically generated

imputer\_mean = SimpleImputer(missing\_values = np.nan,\ strategy='mean')

# replace the missing values for the "age" column with the mean value of the column

df[["age"]] = imputer\_mean.fit\_transform(df[["age"]])

To handle missing or null values, I first used the Pandas library’s methods `isna()` and `sum()` to find the number of missing values for each column in the dataset. The output shows that the ‘age’ column has 123 missing or null values. Since dropping the rows with missing or null values could be costly, given the dataset’s relatively small size of 1340 rows, I chose to implement the SckitLearn’s `SimpleImputer()` class to replace the missing or null values in the ‘age’ column with the mean value of the ‘age’ column.

* Replace values in the “sex” column:

# check data type of columns

df.info()

A screenshot of a computer code

Description automatically generated

NUMERICAL\_VARIABLES = ['PersonID', 'age', 'bmi', \ 'children', 'charges']

df.drop(NUMERICAL\_VARIABLES,axis=1).describe(include=object)

A screenshot of a cell phone

Description automatically generated

df["sex"].unique()



df["sex"] = df["sex"].replace("F", "female")

df["sex"] = df["sex"].replace("M", "male")

To handle the invalid values in the ‘sex’ column, I used the Pandas’`info()` method to check the data types of the columns. Next, I stored all the numerical column names in a variable named NUMERICAL\_VARIABLES. Using the Pandas’ `drop()` method, I removed the numerical columns from the dataset, leaving only the categorical ones. When I applied the `describe()` method to the categorical column, the ‘sex’ column had four unique values. Since the ‘sex’ column should only contain two unique values, I replaced the “F” with “female” and “M” with “male”.

* Convert categorical data into numerical data:

le = LabelEncoder()

df['sex'] = le.fit\_transform(df['sex'])

df['smoker'] = le.fit\_transform(df['smoker'])

df = pd.get\_dummies(df, columns=['region'], \ prefix=["region"])

A screenshot of a computer

Description automatically generated

Machine learning models require all input variables to be numerical. To convert the categorical data into numerical data, I implemented ScikitLearn’s LabelEncoder to transform the ‘sex’ and ‘smoker’ columns and I used one-hot encoding to the ‘region’ column. I used LabelEncoder for columns with binary values and applied one-hot encoding for columns with more than two unique values. As seen in the table above, the one-hot encoding.

I created a LabelEncoder object and assigned it to a variable named ‘le’. Next, I individually fit and transformed the ‘sex’ and ‘smoker’ columns and assigned the output to the DataFrame.

One-hot encoding creates new columns as much as the number of unique samples for each categorical column in the data frame. The columns ‘region’ contains a total of four unique variables. This means that four new columns will be created and filled with 0s and 1s. The column that corresponds to the label will be encoded with a 1.

Question 2

* Histogram for the Age of Primary beneficiary

plt.hist(df.age, edgecolor='black')

plt.title('Distribution of Age')

plt.xlabel('Age')

plt.ylabel('Frequency')

plt.show()

A graph of a number of age

Description automatically generated

The plot above is implemented using the Mathplotlib library, which was imported as ‘plt’. It represents a histogram depicting the distribution of ages of the primary beneficiaries. The x axis displays the age and the y axis shows the frequency, with the age range in between 18 and 64. The average age of the primary beneficiaries is approximately 39. The highest frequency of falls within two age groups: 20 and 40 years old. This suggests that most of the policyholders are either in their early twenties or early forties.

* Scatter plot for relationship between the Charges and BMI

plt.figure(figsize=(12,8))

plt.title("Relationship between Charges and BMI", fontsize=20)

sns.regplot(y="charges", x="bmi", data=df, fit\_reg=True)

A diagram of a graph

Description automatically generated with medium confidence

The plot above is implemented using Seaborn library, which was imported as ‘sns’. It represents a scatter plot that depicts the relationship between charges (medical cost billed by the health insurance) and the BMI of the primary beneficiary. The x axis displays the charges, and the y axis shows the BMI. There is a positive correlation between the Charges and the BMI. This indicates that as the BMI of the primary beneficiary increases, the charges issued by the health insurance also increases. This could be because a higher BMI is associated with a potentially higher medical expenses, which the insurance company has to bear.

Question 3

features = df2.drop(["smoker", “PersonID”], axis=1)

target = df2["smoker"]

X\_train, X\_test, y\_train, y\_test = train\_test\_split(features, target, test\_size=0.2, random\_state=SEED)

# Create and fit the decision tree classifier

clf = DecisionTreeClassifier()

clf.fit(X\_train, y\_train)

To implement the Decision Tree Classifier, I first imported SckitLearn’s ‘train\_test\_split()’ and ‘DecisionTreeClassifier’ methods. I allocated all the dataset’s features (except for the ‘smoker’ column) in the variable named ‘features’ and stored the ‘smoker’ column in the variable named ‘target’. Next, I implemented the ‘train\_test\_split()’ method to split the dataset into training and test sets. In the split, 80% of the data was used for training (‘X\_train’, ’y\_train’) and the remaining 20% was used for testing (‘X\_test’, ’y\_test’). I set the seed value to 42 to ensure that the results are reproducible. I created an instance of the DecisionTreeClassifer, called ‘clf’ and trained it using ‘X\_train’ and y\_train. This will enable the model to make predictions based on the features in ‘X\_train’ and the corresponding target labels in ‘y\_train’.

Question 4

feature\_column\_names = ['age', 'sex', 'bmi', 'children','charges', 'region\_northeast', 'region\_northwest','region\_southeast', 'region\_southwest']

# Plot the Decision Tree

plt.figure(figsize=(24, 16))

plot\_tree(clf, filled=True, feature\_names=feature\_column\_names,\

class\_names=['Non-Smoker', 'Smoker'])

plt.show()

I have plotted the decision tree (as seen below) from the previous question and the following are the key takeaways based on the root node:

* The decision tree splits the data based on the feature ‘charges’. For the value of ‘charges’ > 15264.587, the tree follows the right branch. Otherwise, it follows the left branch.
* The ‘value = [857, 215]’ represents that distribution of samples in the node. Here, there are 857 samples in the ‘Non-Smoker’ class and 215 samples in the ‘Smoker’ class.
* Since there are more ‘Non-Smoker’ samples than ‘Smoker’ samples, the model predicts ‘N on-Smoker’ for the root node.

A diagram of a network

Description automatically generated

Question 5

“Decision Trees Decision tree output is very easy to understand even for people from non-analytical background. It does not require any statistical knowledge to read and interpret them.**[1]**”. Decision tree provides simpler and more accessible values and colourful illustrations to summarize a dataset compared to Exploratory Data Analysis (EDA), which can be complex for users to work through the mathematical concepts of percentiles and standard deviations, etc.

Moreover, “Decision tree is one of the fastest way to identify most significant variables and relation between two or more variables.**[1]**”. Decision trees provide a structured output layout for easy identification of relationships between two or more variables compared to EDA, which requires additional work, such as plotting a correlation matrix, to understand relationships between two or more variables.

It also must be noted that there are downsides to using decision tree compared to EDA. “Decision tree often involves higher time to train the model.**[2]**” Decision trees require more time for training compared to EDA and this could result in in longer time to complete bigger projects.

In conclusion, despite the downside of requiring a longer training time, I feel that the pros of decision tree outweighs its cons compared to EDA.

[https://www.analyticsvidhya.com/blog/2016/04/tree-based-algorithms-complete-tutorial-scratch-in-python/#:~:text=Useful%20in%20Data%20exploration%3A%20Decision,power%20to%20predict%20target%20variable.](https://www.analyticsvidhya.com/blog/2016/04/tree-based-algorithms-complete-tutorial-scratch-in-python/%23:~:text=Useful%20in%20Data%20exploration%3A%20Decision,power%20to%20predict%20target%20variable.)[1]

<https://dhirajkumarblog.medium.com/top-5-advantages-and-disadvantages-of-decision-tree-algorithm-428ebd199d9a>[2]