Data Analysis Exercise 1

February 19, 2023

1 Data Analysis and Manipulation Exercise For Machine Learning Module Series

2 Getting Started

Before we get started with data analysis, we need to ensure that we have Python and the necessary libraries installed. Python can be downloaded from the official website (https://www.python.org/downloads/). We will also be using the following Python libraries:

- NumPy: Used for numerical computations and operations on arrays.
- Pandas: Used for data manipulation and analysis.
- Matplotlib: Used for data visualization.
- Seaborn: A library built on top of Matplotlib for enhanced data visualization.

These libraries can be installed using pip. Open the command prompt and type the following commands:

[ ]:
```
pip install numpy
pip install pandas
pip install matplotlib
pip install seaborn
```

Once we have installed the necessary libraries, we can begin with the data analysis.

2.1 Loading Data

The first step in data analysis is loading the data. We will be using a dataset from Kaggle that contains information about the Titanic passengers. The dataset can be downloaded from the following link: https://www.kaggle.com/c/titanic/data

Save the downloaded file in a folder on your computer. In this tutorial, we assume that the file is saved in a folder named `data` and the file name is `train.csv`.

We will be using the Pandas library to load the data. Open a new Python file and type the following code:

[1]:
```
import pandas as pd

# Write code to load the data
...
```

```
# Write code to print the first 5 rows of the data
...
```

[1]: Ellipsis

The code above imports the Pandas library and loads the data from the CSV file using the `read_csv()` method. The `head()` method is then used to print the first 5 rows of the data.

2.2 Cleaning Data

Once we have loaded the data, we need to clean it to remove any missing or incorrect data. In this tutorial, we will be removing the `Cabin` column and any rows with missing data.

[2]:
```
# Write code to remove the 'Cabin' column
...

# Write code to remove any rows with missing data
...
```

The code above removes the `Cabin` column using the `drop()` method and the `axis` parameter set to 1 to indicate that we want to remove a column. The `dropna()` method is then used to remove any rows with missing data.

2.3 Exploratory Data Analysis

Now that we have cleaned the data, we can perform exploratory data analysis (EDA) to gain insights into the data. In this module, we will be using the Matplotlib and Seaborn libraries to create visualizations.

2.3.1 Visualizing Numeric Data

To visualize the distribution of the numeric data, we can use a histogram.

Write code to create a histogram of the `Age` column:

[3]:
```
import matplotlib.pyplot as plt

# Write code to create a histogram of the 'Age' column
...
```

[3]: Ellipsis

The `hist()` method is used to create the histogram, and the 'bins' parameter is set to 20 to create 20 bins. The `xlabel()` and `ylabel()` methods are used to set the labels for the x-axis and y-axis, respectively.

2.3.2 Visualizing Categorical Data

To visualize the distribution of the categorical data, we can use a bar chart.

Write code to create a bar chart of the `Sex` column:

[4]:
```
import seaborn as sns

# Write code to create a bar chart of the 'Sex' column
...
```

[4]:
```
Ellipsis
```

The `countplot()` method from the Seaborn library is used to create the bar chart, and the `x` parameter is set to `Sex` to indicate that we want to create a bar chart of the `Sex` column.

2.3.3 Visualizing Relationships Between Variables

To visualize the relationships between variables, we can use scatter plots.

Write code to create a scatter plot of the `Age` column and the `Fare` column:

[5]:
```
# Write code to create a scatter plot of the 'Age' and 'Fare' columns
...
```

The `scatter()` method is used to create the scatter plot, and the `x` parameter is set to `Age` and the `y` parameter is set to `Fare` to indicate that we want to create a scatter plot of these two columns.

2.4 Data Preprocessing

Before we can build a machine learning model, we need to preprocess the data. In this module, we will be converting the categorical data into numerical data and splitting the data into training and testing sets.

2.4.1 Converting Categorical Data into Numerical Data

To convert the categorical data into numerical data, we can use the `get_dummies()` method from the Pandas library.

Write code to convert the `Sex` column into numerical data:

[6]:
```
# Write code to convert the 'Sex' column into numerical data
...
```

The `get_dummies()` method is used to convert the `Sex` column into numerical data, and the `columns` parameter is set to `Sex` to indicate that we want to convert the `Sex` column.

2.4.2 Splitting the Data into Training and Testing Sets

To build a machine learning model, we need to split the data into training and testing sets.

Write code to split the data into a training set and a testing set:

[ ]:
```
from sklearn.model_selection import train_test_split

# Split the data into a training set and a testing set
```

```
X = data.drop(['Survived', 'Name', 'Ticket', 'Embarked'], axis=1)
y = data['Survived']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

The `train_test_split()` method from the scikit-learn library is used to split the data into a training set and a testing set. The `test_size` parameter is set to 0.2 to indicate that we want to use 20% of the data for testing.

The `X` variable contains the features, and the `y` variable contains the target variable. The `drop()` method is used to remove the `Survived`, `Name`, `Ticket`, and `Embarked` columns from the `X` variable.

2.5 Building a Machine Learning Model

Now that we have preprocessed the data, we can build a machine learning model. In this module, we will be using the scikit-learn library to build a logistic regression model.

[ ]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score

# Build a logistic regression model
model = LogisticRegression()
model.fit(X_train, y_train)

# Make predictions on the testing set
y_pred = model.predict(X_test)

# Print the accuracy score
print('Accuracy:', accuracy_score(y_test, y_pred))
```

The `LogisticRegression()` method is used to build the logistic regression model, and the `fit()` method is used to train the model on the training set. The `predict()` method is then used to make predictions on the testing set.

The `accuracy_score` method from the scikit-learn library is used to calculate the accuracy of the model.

2.6 Conclusion

In this module, we have learned how to perform data analysis using Python. We started by loading the data into a Pandas DataFrame, and then we explored the data using various data visualization techniques. We then preprocessed the data by converting the categorical data into numerical data and splitting the data into training and testing sets. Finally, we built a machine learning model using logistic regression and evaluated the model's accuracy.

This module covers only the basics of data analysis and machine learning using Python. There are many more techniques and libraries available in Python for data analysis, and you can further explore them to perform more advanced analyses.

[ ]:
```

```