Fundamentals of Machine Learning
Lab Assignment-8

Name: Hitha Choudhary G
USN: 22BTRAD015
Branch: CSE in AI&DE

Implementing Pandas

- Pandas is a Python library used for working with data sets.
- It has functions for analysing, cleaning, exploring, and manipulating data.
- Pandas allows us to analyse big data and make conclusions based on statistical theories.
- Pandas can clean messy data sets, and make them readable and relevant. Relevant data is very important in data science.
- The data produced by Pandas are often used as input for plotting functions of Matplotlib, statistical analysis in SciPy, and machine learning algorithms in Scikit-learn.
- Here is a list of things that we can do using Pandas.

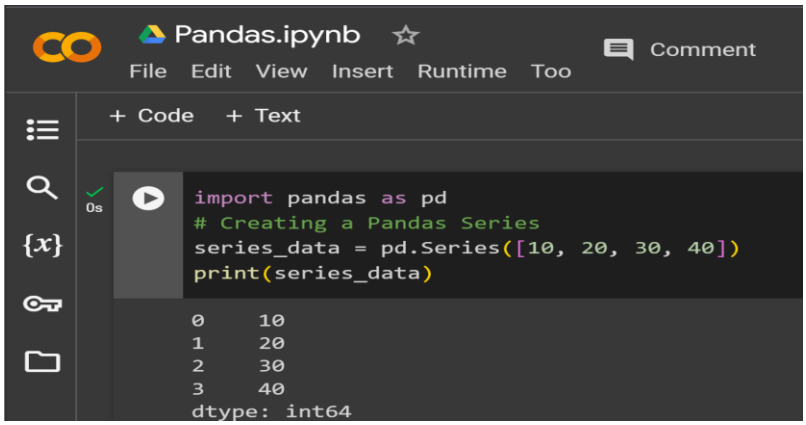
Creating a Series: A one-dimensional array capable of storing a variety of data types is how it is defined. The term "index" refers to the row labels of a series.

In the below code we imported pandas as pd and created a series using "pd.Series()".

Code:

```
import pandas as pd

# Creating a Pandas Series

series_data = pd.Series([10, 20, 30, 40])

print(series_data)
```

Output:



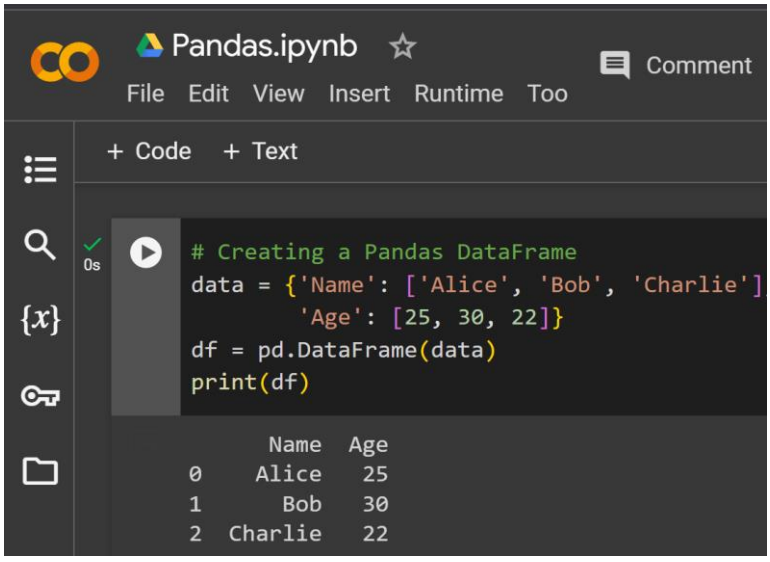
Creating a DataFrame: As a standard method for storing data, DataFrame has two distinct indexes-row index and column index. It has the following characteristics: It can be thought of as a series structure dictionary with indexed rows and columns. It is referred to as "columns" for rows and "index" for columns.

In the below code we are creating a dictionary which is later represented as a DataFrame using “pd.DataFrame()”.

Code:

```
# Creating a Pandas DataFrame

data = {'Name': ['Alice', 'Bob', 'Charlie'],

        'Age': [25, 30, 22]}

df = pd.DataFrame(data)

print(df)
```

Output:



Operations in a DataFrame: Selecting a particular column by label and selecting rows based on a condition.

Code:

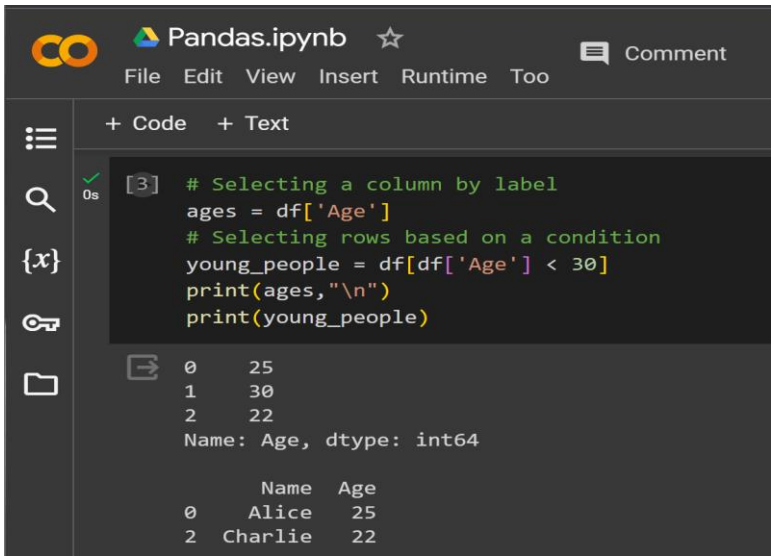
```
# Selecting a column by label

ages = df['Age']

# Selecting rows based on a condition
```

young_people = df[df['Age'] < 30]

print(ages,"\n")

print(young_people)

Output:



In the above code we are accessing the column age from the above mentioned dataframe and we are also filtering the age < 30 and storing it in the variable “young_people”.

Handling Missing Values in a DF: We use “df.dropna()” to find out the missing values if any present in the DataFrame. We also use”df.fillna()” to fill in a missing value.

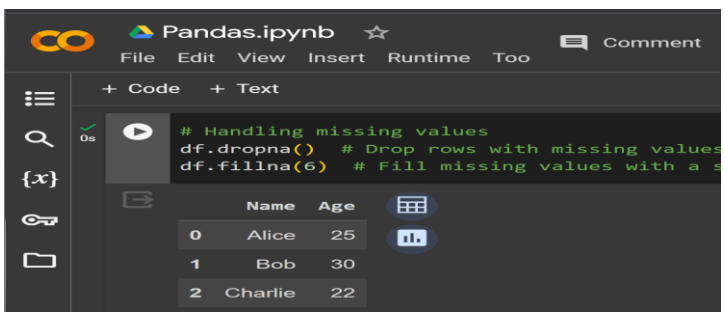
Code:

Handling missing values

df.dropna() # Drop rows with missing values

df.fillna(6) # Fill missing values with a specified value

Output:



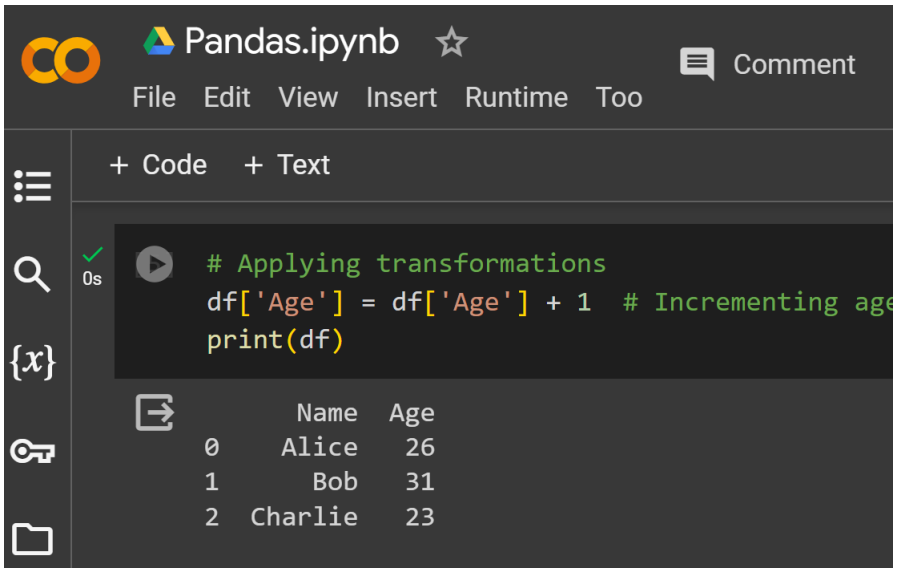
Since the above DataFrame does not contain any missing values there are no changes made.

Transformation: We can transform a particular column in the dataframe. For example in the age column we can transform the column by incrementing all the age values by 1.

Code:

```
# Applying transformations

df['Age'] = df['Age'] + 1  # Incrementing ages by 1

print(df)
```

Output:

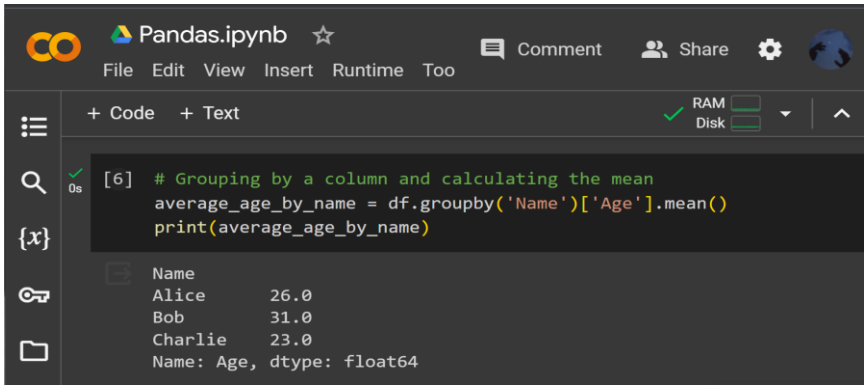


GroupBy: We can group different columns and calculate the mean of it by using “df.groupby()” for grouping and “.mean()” for finding the mean.

Code:

```
# Grouping by a column and calculating the mean

average_age_by_name = df.groupby('Name')['Age'].mean()

print(average_age_by_name)
```

Output:



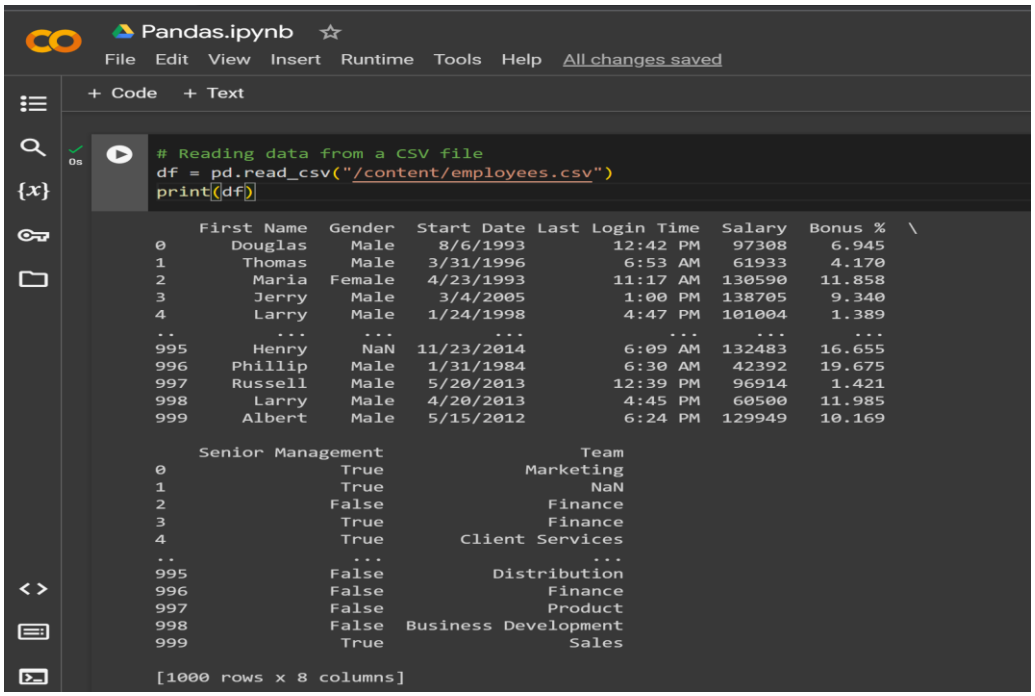
We are grouping the columns “name” and “age” and finding the mean of it. The result is shown above.

Read CSV files using Pandas: A simple way to store big data sets is to use CSV files (comma separated files). CSV files contains plain text and is a well know format that can be read by everyone including Pandas.

Code:

```
# Reading data from a CSV file

df = pd.read_csv("/content/employees.csv")

print(df)
```

Output:

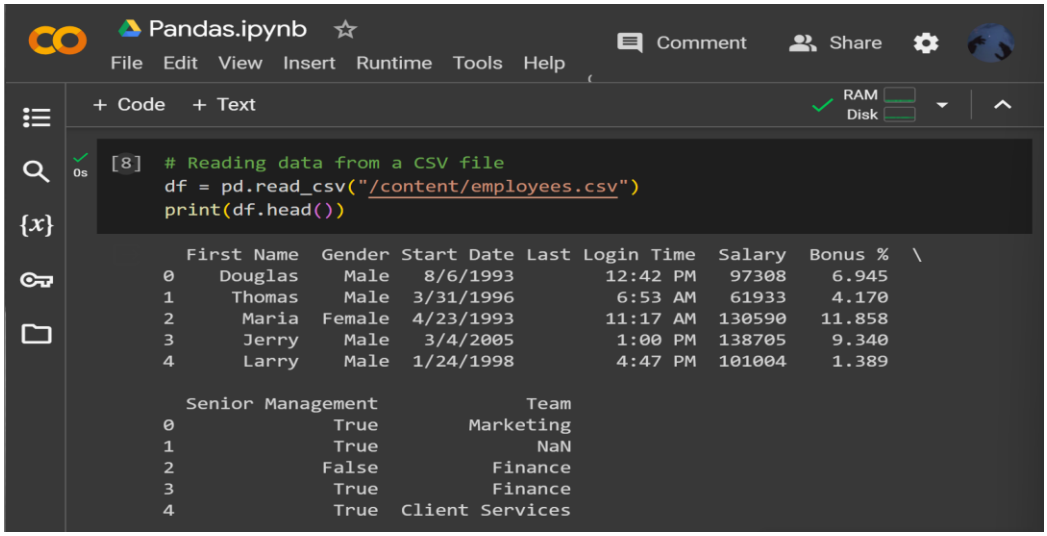


Printing the first five rows from the csv file: Using “df.head()” we can print only the first five rows of data from the CSV file.

Code:

```
# Reading data from a CSV file

df = pd.read_csv("/content/employees.csv")

print(df.head())
```

Output:



Github Link: https://github.com/hithachoudhary/machinelearning