Page No. :
Date. : / /

Group A

Experiment -1

Piyusha Supe. (TE B)
23C0315 (Batch C)

- Title- Data Wrangling I.

- Problem Statement -
Perform the following operations using Python on any open source dataset (eg. data.csv) Import all required python libraries

1. Locate open source data from web (eg. https://www.kaggle.com)
2. Provide a clear description of the data and its source (ie. URL of website).
3. Load the dataset to pandas dataframe.
4. Data pre-processing: Check for missing values in data using pandas info(), describe() function to get initial statistics. Provide variable description. Types of variables Check dimensions of data frame.
5. Data formatting and normalization. - Summarize types of variables Check dimensions of data frame. data types
6. (i.e. character, numeric, integer, factor and logical) If variables are not in correct datatype apply proper type conversion.
6. Turn catagorical variable to quantitative variable in python.

- Pre-requisite: · Basic of python programming
· Concept of data preprocessing, data formatting Normalization and cleaning.

- Objective - Students should be able to perform the data wrangling operation using python on any open source dataset.

- THEORY :

1] Introduction to dataset -

- A dataset is a collection of records, similar to a relational database table.

1. Instance - A single row of data is called an instance It is an observation from the domain.

2. Feature - A single column of data is called a feature. It is a component of an observation and is also called attribute of data instance.

3. Data Type - Features have a data type. They may be real or integer-valued or may have a catagorical or ordinal value. You can have strings, dates, times, and more complex types - but typically they are reduced to real or catagorical values.

4. Datasets - A collection of instances is a dataset

5. Training dataset - A dataset that we feed into our machine learning algorithm to train our model.

6. Testing dataset - A dataset that we use to validate the accuracy of our model.

7. Data represented in a Table -
Data should be arranged in a 2-D space made up of rows and columns. This type of data structure makes it easy to understand the data and pinpoint any problems.

Eg of raw data in csv -

1., Avatar, 18-12-2009, 7.8
2., Titanic, 18-11-1997, Na
3., Avengers, 27-04-2018, 8.5

The same in a table -

S.No.	Movie.	Release date.	Ratings.
1	Avatar	18-12-2009	7.8
2	Titanic.	18-11-1997	Na.
3	Avengers.	27-04-2018	8.5.

2] Python Libraries for Data Science -

a. Pandas - It is an open source python package that provides high performance, easy to use data structure, data analysis for labeled data.

- Indexing, manipulating, renaming, sorting, merging data frame.
- Update, Add, Delete columns from a data frame.
- Impute missing files, handle missing data or NaNs.
- Plot data with histogram or boxplot.

b. Numpy - • Basic array operations : add, multiply, slice, flatten, reshape, index arrays.
• Advanced array operation, stack arrays, split into sections, broadcast arrays.

c. Mat plot lib - Used for visualization -
- Scatter plots.
- Area plots.
- Bar charts Histograms.
- Pie charts.
- Stem plots.
- Contour plots
- Quiver plots.
- Spectograms.

It also facilitates labels, grids, legends, and formatting entities.

d). Seaborn - Data visualization library -
- Determine relationship between variables (correlation)
- Observe categorical variables for aggregate statistics.
- Analyze univariate or bi-variate distributions and compare them between different data subsets.
- Provide high level abstraction, multiplot grids.

e. Scikit learn - Robust machine learning library for python.
- Classification - Spam detection, image recognition.
- Clustering - Drug response, Stock price.
- Regression - Customer segmentation grouping outcomes.
- Dimensionality reduction. - Visualization increased efficiency.

3] Description of Dataset - The iris dataset is a traditional dataset that contains 50 samples of three iris species.

The columns in this data set are -

1. Id.
2. Sepal length cm.
3. Sepal Width cm.
4. Petal length cm.
5. Petal width cm.
6. Species.

4] Dataframe functions for preprocessing -

Sr.	Dataframe function.	Description.
1.	ds.head (n = 5)	Return first n rows.
2.	ds.tail (n=5)	Return last n rows.
3.	ds.index.	The index (rows lbl) of dataset.
4.	ds.columns.	The column labels of dataset.
5.	ds.shape.	Return a tuple representing dimensionality of dataset.
6.	ds.dtypes	Return the dtypes in dataset.
7.	ds.columns.values.	Returns column values in dataset in array format.
8.	ds.describe (include=all)	Generate descriptive analysis.
9.	ds.['Column Name']	Read data column wise.
10.	dataset.sort_index (axis=1, ascending = False)	Sort object by labels.
11.	ds.sort_values (by = "Column name")	Sort values by column name.

12.	ds.iloc [5]	Purely integer location based indexing.
13.	ds[0:3]	Selecting via [] which slices the rows.
14.	ds.loc [:, ["col1", "col2"]]	Selection by label.
15.	ds.iloc [:n, :]	a subset of first n 'rows' of the original data.
16.	ds.iloc [:, :n]	Subset of first n columns of original data.
17	ds.iloc [:m, :n]	a subset of first m rows and first n columns.
18.	.isnull() isna()	Looks for null values.

5] Algorithm -

Step 1: Import pandas and sklearn library for preprocessing
from sklearn import preprocessing.

Step 2: Load the iris dataset in dataframe object df.

Step 3: Print iris dataset.
df.head().

Step 4: Create a minimum and maximum processor object
min-max-scaler = preprocessing.minMaxScaler().

Step 5 : Separate the feature from class label.

```
x = df.iloc[:, :4].
```

Step 6: Create an object to transform the data to fit minmax processor.

```
x_scaled = min_max_scaler.fit_transform(x)
```

Step 7: Run the normalizer on dataframe -

```
df_normalized = pd.Dataframe(x_scaled)
```

Step 8: View the dataframe - df_normalized.

6] One hot Encoding algorithm -

Steps are as follows -

1. Step 1: Import pandas and sklearn library -

```
from sklearn import preprocessing.
```

Step 2: Load the iris dataset in dataframe object df.

Step 3: Observe the unique values for the Species column.

```
df['Species'].unique()
```

Step 4: Apply label_encoder object for label encoding the observe the unique values for the species column.

Step 5: Remove the target variable from dataset.

```
features_df = df.drop(columns = ['Species'])
```

Step 6: Apply one hot encoding

```
enc = preprocessing.OneHotEncoder()
```

```
enc_df = pd.DataFrame(enc.fit_transform(df[['Species']])).
        toarray().
```

Step 7: Join the encoded values with features variable.

```
df_encode = features_df.join(enc_df).
```

Step 8: Observe the merge dataframe.

```
df_encode.
```

Step 9: Rename the newly encoded columns -

```
df_encode.rename(columns = {0: 'Iris-Setosa',
        1: 'Iris-Versicolor', 2: 'Iris-virginica'},
        inplace = True)
```

Step 10: Observe the merged dataframe.

```
df_encode.
```

7) Dummy variable encoding -

Algorithm -

Step 1: Import pandas and sklearn library for preprocessing.

```
from sklearn import preprocessing.
```

Step 2: Load the iris dataset in dataframe object df.

Step 3: Observe the unique values for the species column

```
df['Species'].unique()
```

Step 4: Apply one-hot encoder with dummy variables for species column.

One_hot_df = pd.get_dummies(df, prefix = "Species", columns = ['Species'], drop_first = True)

Step 5 : Observe the merged dataframe - one_hot_df.

Color	Dummy Encoding →	d1	d2
RED		1	0
GREEN		0	1
BLUE		0	0

- **Conclusion :** In this way we have explored the functions of the python library for data pre-processing, Data wrangling techniques and how to handle missing values on Iris dataset.

- Assignment Question and answers -

(1) Explain what is dataframe ? give suitable example.

→ A dataframe is a table like structure in pandas that stores data in rows and columns.

Eg -

```
data = {'Name': ['Alice', 'Bob', 'Charlie'], 'Age': [25, 30, 35]}
df = pd.DataFrame(data)
print(df)
```

O/P -

```
      Name  Age
0    Alice   25
1      Bob   30
2  Charlie   35
```

(2). What is the limitation of Label Encoding ?

→ Label Encoding assigns numbers to categories (eg. "Red" = 0, "Blue" = 1), but it has a major problem :
- Creates false ranking : Model thinks that "Red" (2) is greater than "Blue" (0), which is incorrect for unordered catagories like colors.

(3). What is need of data Normalization?

→
- Normalization scales data into a smaller range (eg 0 to 1) so that all features contribute equally.
- Prevents large values (like salary) from dominating smaller ones (like age)
- Helps machine learning models work better, especially those using distance-based calculations (eg. KNN, K-Means).

(4). How to Handle Missing data ?

→ 1. Remove missing value row or column or assign it to a mean
2. Use previous (ffill) or next (bfill) values.
3. Use KNN (nearest neighbour) to estimate missing values.
4. Machine learning models like XGBoost can handle missing data.