Experiment 1: a) Creating and loading different datasets in Python.

b) Reshaping, Filtering, Scaling, Merging the data and Handling the missing values in datasets.

Theory:

1. **pandas**

Pandas is one of the most famous Python libraries and used for importing and managing the datasets. It is an open-source data manipulation and analysis library.

2. **numpy**

Numpy Python library is used for including any type of mathematical operation in the code. It is the fundamental package for scientific calculation in Python. It also supports to add large, multidimensional arrays and matrices.

3. **csv file**

To use the dataset in our code, we usually put it into a CSV file. CSV stands for "Comma-Separated Values" files; it is a file format which allows us to save the tabular data, such as spreadsheets.

4. **read_csv() & read_excel()**

Now to import the dataset, we will use read_csv() & read_excel() functions of pandas library, which are used to read a csv & Excel file and performs various operations on it. Using this function, we can read a csv file locally as well as through an URL.

5. **describe(), head(), tail(), shape**

With .describe(), we can get an overview of the values each column contains. You can have a look at the first five rows with .head() whereas you can display the last five rows with .tail(). You also use the .shape attribute of the DataFrame to see its dimensionality.

6. **Missing Value Imputation**

The next step of data preprocessing is to handle missing data in the datasets. If our dataset contains some missing data, then it may create a huge problem for our machine learning model. Hence it is necessary to handle missing values present in the dataset.

Some of the ways to handle missing data are:

By deleting the particular row: In this way, we just delete the specific row or column which consists of null values. But this way is not so efficient and removing data may lead to loss of information which will not give the accurate output.

By calculating the mean: In this way, we will calculate the mean of that column which contains any missing value and will put it on the place of missing value. This strategy is useful for the features which have numeric data such as age, salary, year, etc.

Implementation:

1. **Using sklearn.impute.SimpleImputer:**

Syntax:

```
class sklearn.impute.SimpleImputer(*, missing_values=nan, strategy='mean')
```

Univariate imputer for completing missing values with simple strategies. Replace missing values using a descriptive statistic (e.g. mean, median, or most frequent) along each column, or using a constant value.

2. **sklearn.preprocessing.MinMaxScaler**

Syntax:

```
class sklearn.preprocessing.MinMaxScaler(feature_range=(0, 1), ..)
```

Transform features by scaling each feature to a given range.

3. **sklearn.preprocessing.StandardScaler**

Syntax:

```
class sklearn.preprocessing.StandardScaler()
```

Standardize features by removing the mean and scaling to unit variance.

Conclusion: In this way, we understood the pre-processing and analysis steps required to be carried out before the actual start of the ML tasks.