



IBM Developer SKILLS NETWORK

Introduction Notebook

Estimated time needed: **10** minutes

Objectives

After completing this lab you will be able to:

- Acquire data in various ways
- Obtain insights from data with Pandas library

Table of Contents

1. [Data Acquisition \(https://#data_acquisition\)](https://#data_acquisition)
2. [Basic Insight of Dataset \(https://#basic_insight\)](https://#basic_insight)

</div>

Data Acquisition

There are various formats for a dataset: .csv, .json, .xlsx etc. The dataset can be stored in different places, on your local machine or sometimes online.

In this section, you will learn how to load a dataset into our Jupyter Notebook.

In our case, the Automobile Dataset is an online source, and it is in a CSV (comma separated value) format. Let's use this dataset as an example to practice data reading.

- Data source: <https://archive.ics.uci.edu/ml/machine-learning-databases/autos/imports-85.data>
(https://archive.ics.uci.edu/ml/machine-learning-databases/autos/imports-85.data?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_campaign=SkillsNetwork-Channel-SkillsNetworkCoursesIBMDDeveloperSkillsNetworkDA0101ENSkillsNetwork20235326-2021-01-01)
- Data type: csv

The Pandas Library is a useful tool that enables us to read various datasets into a dataframe; our Jupyter notebook platforms have a built-in **Pandas Library** so that all we need to do is import Pandas without installing.

In []:

Read Data

We use `pandas.read_csv()` function to read the csv file. In the brackets, we put the file path along with a quotation mark so that pandas will read the file into a dataframe from that address. The file path can be either an URL or your local file address.

Because the data does not include headers, we can add an argument `headers = None` inside the `read_csv()` method so that pandas will not automatically set the first row as a header.

You can also assign the dataset to any variable you create.

This dataset was hosted on IBM Cloud object. Click [HERE \(https://cocl.us/DA101EN_object_storage?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkDA0101ENSkillsNetwork20235326-2021-01-01\)](https://cocl.us/DA101EN_object_storage?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkDA0101ENSkillsNetwork20235326-2021-01-01) for free storage.

In []:

After reading the dataset, we can use the `dataframe.head(n)` method to check the top `n` rows of the dataframe, where `n` is an integer. Contrary to `dataframe.head(n)`, `dataframe.tail(n)` will show you the bottom `n` rows of the dataframe.

In []:

Question #1:

Check the bottom 10 rows of data frame "df".

In []:

Click here for the solution

Add Headers

Take a look at our dataset. Pandas automatically set the header with an integer starting from 0.

To better describe our data, we can introduce a header. This information is available at:

<https://archive.ics.uci.edu/ml/datasets/Automobile> (https://archive.ics.uci.edu/ml/datasets/Automobile?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkDA0101ENSkillsNetwork20235326-2021-01-01).

Thus, we have to add headers manually.

First, we create a list "headers" that include all column names in order. Then, we use `dataframe.columns = headers` to replace the headers with the list we created.

In []:

We replace headers and recheck our dataframe:

In []:

We need to replace the "?" symbol with NaN so the `dropna()` can remove the missing values:

In []:

We can drop missing values along the column "price" as follows:

In []:

Now, we have successfully read the raw dataset and added the correct headers into the dataframe.

Question #2:

Find the name of the columns of the dataframe.

In []:

[Click here for the solution](#)

Save Dataset

Correspondingly, Pandas enables us to save the dataset to csv. By using the `dataframe.to_csv()` method, you can add the file path and name along with quotation marks in the brackets.

For example, if you would save the dataframe **df** as **automobile.csv** to your local machine, you may use the syntax below, where `index = False` means the row names will not be written.

```
df.to_csv("automobile.csv", index=False)
```

We can also read and save other file formats. We can use similar functions like `pd.read_csv()` and `df.to_csv()` for other data formats. The functions are listed in the following table:

Read/Save Other Data Formats

Data Formate	Read	Save
csv	<code>pd.read_csv()</code>	<code>df.to_csv()</code>
json	<code>pd.read_json()</code>	<code>df.to_json()</code>
excel	<code>pd.read_excel()</code>	<code>df.to_excel()</code>
hdf	<code>pd.read_hdf()</code>	<code>df.to_hdf()</code>
sql	<code>pd.read_sql()</code>	<code>df.to_sql()</code>
...

Basic Insight of Dataset

After reading data into Pandas dataframe, it is time for us to explore the dataset.

There are several ways to obtain essential insights of the data to help us better understand our dataset.

Data Types

Data has a variety of types.

The main types stored in Pandas dataframes are **object**, **float**, **int**, **bool** and **datetime64**. In order to better learn about each attribute, it is always good for us to know the data type of each column. In Pandas:

```
In [ ]:
```

A series with the data type of each column is returned.

```
In [ ]:
```

As shown above, it is clear to see that the data type of "symboling" and "curb-weight" are `int64`, "normalized-losses" is `object`, and "wheel-base" is `float64`, etc.

These data types can be changed; we will learn how to accomplish this in a later module.

Describe

If we would like to get a statistical summary of each column e.g. count, column mean value, column standard deviation, etc., we use the describe method:

```
dataframe.describe()
```

This method will provide various summary statistics, excluding `NaN` (Not a Number) values.

```
In [ ]:
```

This shows the statistical summary of all numeric-typed (`int`, `float`) columns.

For example, the attribute "symboling" has 205 counts, the mean value of this column is 0.83, the standard deviation is 1.25, the minimum value is -2, 25th percentile is 0, 50th percentile is 1, 75th percentile is 2, and the maximum value is 3.

However, what if we would also like to check all the columns including those that are of type `object`?

You can add an argument `include = "all"` inside the bracket. Let's try it again.

```
In [ ]:
```

Now it provides the statistical summary of all the columns, including object-typed attributes.

We can now see how many unique values there, which one is the top value and the frequency of top value in the object-typed columns.

Some values in the table above show as "NaN". This is because those numbers are not available regarding a particular column type.

Question #3:

You can select the columns of a dataframe by indicating the name of each column. For example, you can select the three columns as follows:

```
dataframe[[' column 1 ',column 2', 'column 3']]
```

Where "column" is the name of the column, you can apply the method ".describe()" to get the statistics of those columns as follows:

```
dataframe[[' column 1 ',column 2', 'column 3'] ].describe()
```

Apply the method to ".describe()" to the columns 'length' and 'compression-ratio'.

In []:

[Click here for the solution](#)

Info

Another method you can use to check your dataset is:

```
dataframe.info()
```

It provides a concise summary of your DataFrame.

This method prints information about a DataFrame including the index dtype and columns, non-null values and memory usage.

In []:

Excellent! You have just completed the Introduction Notebook!

Thank you for completing this lab!

Author

[Joseph Santarcangelo \(https://www.linkedin.com/in/joseph-s-50398b136/?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkDA0101ENSkillsNetwork20235326-2021-01-01\)](https://www.linkedin.com/in/joseph-s-50398b136/?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkDA0101ENSkillsNetwork20235326-2021-01-01)

Other Contributors

[Mahdi Noorian PhD \(https://www.linkedin.com/in/mahdi-noorian-58219234/?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkDA0101ENSkillsNetwork20235326-2021-01-01\)](https://www.linkedin.com/in/mahdi-noorian-58219234/?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkDA0101ENSkillsNetwork20235326-2021-01-01)

Bahare Talayian

Eric Xiao

Steven Dong

Parizad

Hima Vasudevan

[Fiorella Wenver \(https://www.linkedin.com/in/fiorellawever/?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkDA0101ENSkillsNetwork20235326-2021-01-01\)](https://www.linkedin.com/in/fiorellawever/?utm_medium=Exinfluencer&utm_source=Exinfluencer&utm_content=000026UJ&utm_term=10006555&utm_SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkDA0101ENSkillsNetwork20235326-2021-01-01)

[Yi Yao \(https://www.linkedin.com/in/yi-leng-yao-84451275/\)](https://www.linkedin.com/in/yi-leng-yao-84451275/)

Change Log

Date (YYYY-MM-DD)	Version	Changed By	Change Description
2020-10-30	2.3	Lakshmi	Changed URL of the csv
2020-09-22	2.2	Nayef	Added replace() method to remove '?'
2020-09-09	2.1	Lakshmi	Made changes in info method of dataframe
2020-08-27	2.0	Lavanya	Moved lab to course repo in GitLab

© IBM Corporation 2020. All rights reserved.