Victor Hugo Avila Felipe - A01794425

Data Analysis with Python

IBM: DA0101EN

LEARNING OBJECTIVES

In this course you will learn about:

```
Data Acquisition
How to Obtain Basic Insight From a Dataset
Data Wrangling
Exploratory Data Analysis
Model Development
Model Evaluation
```

1 Introduction to Data Analysis with Python

- Problem requiring data analysis
- dataset to analyze in python
- overview of packages
- import and export data
- Basic insights

Can we estimate the price of used cars?

The problem

why data analysis? data everywhere, helps discovery of information.

Tom wants to sell his car, but wants the best price. what affects the price?

Understand the data

There are documentation and the .csv file.

The first attribute, "symboling", corresponds to the insurance risk level of a car. Cars are initially assigned a risk factor symbol associated with their price. Then, if an automobile is more risky, this symbol is adjusted by moving it up the scale. A value of +3 indicates that the auto is risky, -3 that it is probably pretty safe. The second attribute "normalized-losses" is the relative average loss payment per insured vehicle year. This value is normalized for all autos within a particular size classification (two-door small, station wagons, sports/speciality, etc...), and represents the average loss per car per year. The values range from 65 to 256. The other attributes are easy to understand.

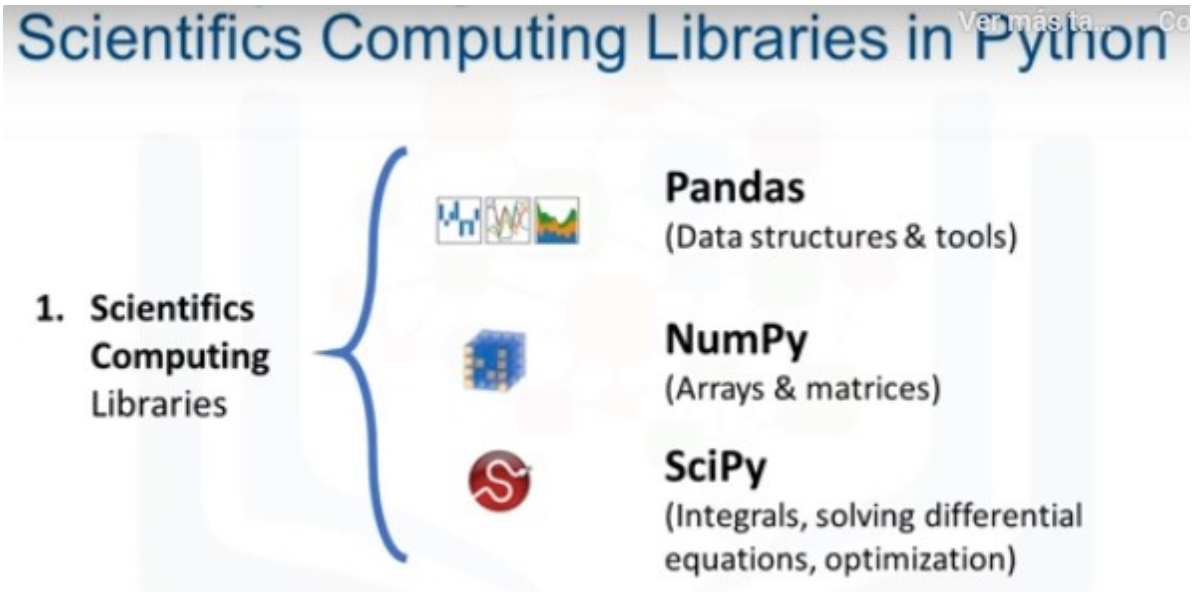
Thus, the goal of this project is to predict "price" in terms of other car features.

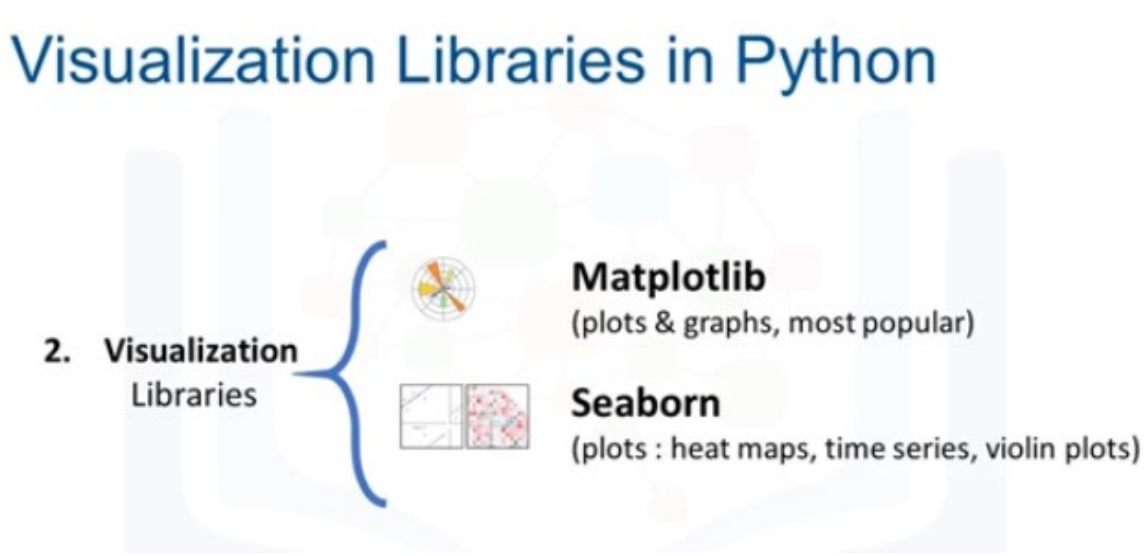
No.	Attribute name	attribute range	No.	Attribute name	attribute range
1	symboling	-3, -2, -1, 0, 1, 2, 3.	14	curb-weight	continuous from 1488 to 4066.
2	normalized-losses	continuous from 65 to 256.	15	engine-type	dohc, dohcv, l, ohc, ohcf, ohcv, rotor.
3	make	audi, bmw, etc.	16	num-of-cylinders	eight, five, four, six, three, twelve, two.
4	fuel-type	diesel, gas.	17	engine-size	continuous from 61 to 326.
5	aspiration	std, turbo.	18	fuel-system	1bbl, 2bbl, 4bbl, idi, mfi, mpfi, spdi, spfi.
6	num-of-doors	four, two.	19	bore	continuous from 2.54 to 3.94.
7	body-style	hardtop, wagon, etc.	20	stroke	continuous from 2.07 to 4.17.
8	drive-wheels	4wd, fwd, rwd.	21	compression-ratio	continuous from 7 to 23.
9	engine-location	front, rear.	22	horsepower	continuous from 48 to 288.
10	wheel-base	continuous from 86.6 120.9.	23	peak-rpm	continuous from 4150 to 6600.
11	length	continuous from 141.1 to 208.1.	24	city-mpg	continuous from 13 to 49.
12	width	continuous from 60.3 to 72.3.	25	highway-mpg	continuous from 16 to 54.
13	height	continuous from 47.8 to 59.8.	26	price	continuous from 5118 to 45400.

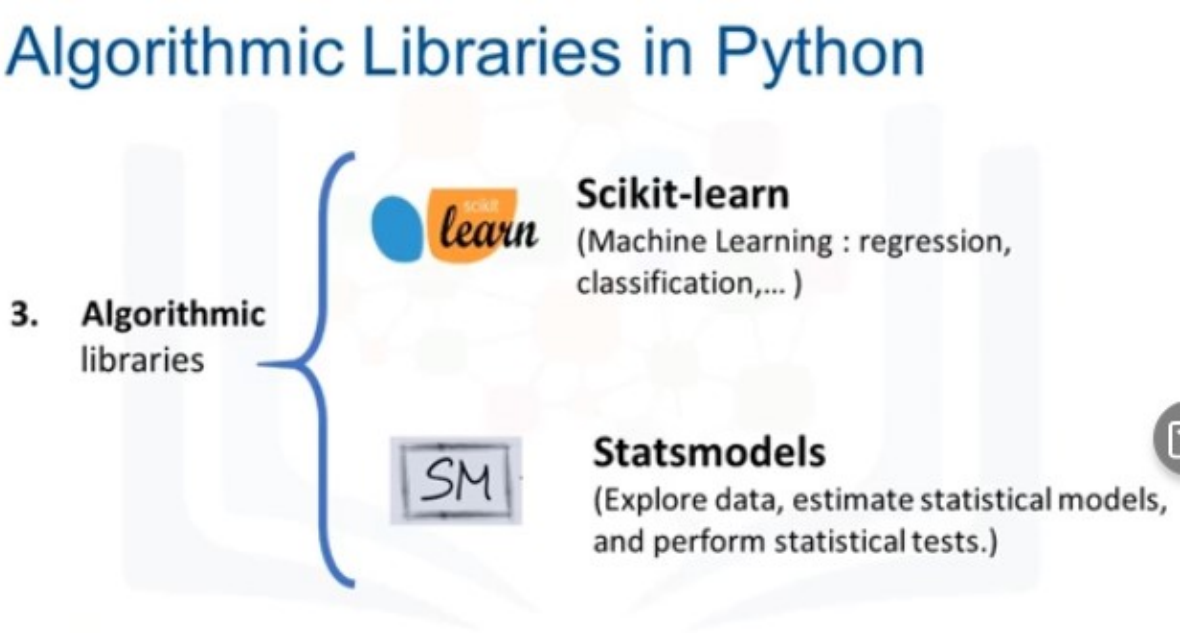
Target (Label)

Python packages

A Python library is a collection of functions and methods that allow you to perform lots of actions without writing any code. The libraries usually contain built-in modules providing different functionalities, which you can use directly. And there are extensive libraries, offering a broad range of facilities.







importing and exporting data

Format and file path. read_csv() df df.head(n) df.tail(n)

Add headers df.columns = headers headers = ["a","b","c"]

export df to csv: df.to_csv(path)

csv, json, excel, sql

Analyzing data

Check data type data distribution

object, float, int y datatime

- potential info and type mismatch
- compatibility with python methods

dataframe.dtypes

df.describe(include="all") -> count, mean, std deviation, min, 25%, 50%, 75%, max all -> UNIQUE, top, freq

Data lab

- 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_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkDA0101ENSkillsNetwork20235326-2021-0
- Data type: csv

In [10]:
```
#install specific version of libraries used in  lab
import sys
!{sys.executable} -m pip install pandas
!{sys.executable} -m pip install numpy
!{sys.executable} -m pip install matplotlib
!{sys.executable} -m pip install scipy
!{sys.executable} -m pip install seaborn
!{sys.executable} -m pip install ipywidgets
```

```
Requirement already satisfied: pandas in /home/flynn/anaconda3/lib/python3.9/
site-packages (1.4.2)
Requirement already satisfied: python-dateutil>=2.8.1 in /home/flynn/anaconda
3/lib/python3.9/site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in /home/flynn/anaconda3/lib/pyth
on3.9/site-packages (from pandas) (2021.3)
Requirement already satisfied: numpy>=1.18.5 in /home/flynn/anaconda3/lib/pyt
hon3.9/site-packages (from pandas) (1.21.5)
Requirement already satisfied: six>=1.5 in /home/flynn/anaconda3/lib/python3.
9/site-packages (from python-dateutil>=2.8.1->pandas) (1.16.0)
Requirement already satisfied: numpy in /home/flynn/anaconda3/lib/python3.9/s
ite-packages (1.21.5)
Requirement already satisfied: matplotlib in /home/flynn/anaconda3/lib/python
3.9/site-packages (3.5.1)
Requirement already satisfied: kiwisolver>=1.0.1 in /home/flynn/anaconda3/lib
/python3.9/site-packages (from matplotlib) (1.3.2)
Requirement already satisfied: numpy>=1.17 in /home/flynn/anaconda3/lib/pytho
n3.9/site-packages (from matplotlib) (1.21.5)
Requirement already satisfied: fonttools>=4.22.0 in /home/flynn/anaconda3/lib
```

In [13]:

```
  Input In [13]
    pip install pandas
        ^
SyntaxError: invalid syntax
```

In [11]:

```
# import pandas library
import pandas as pd
```

In [12]:

```
#This function will download the dataset into your browser

from pyodide.http import pyfetch

async def download(url, filename):
    response = await pyfetch(url)
    if response.status == 200:
        with open(filename, "wb") as f:
```

```
---------------------------------------------------------------------------
ModuleNotFoundError                       Traceback (most recent call last)
Input In [12], in <cell line: 3>()
      1 #This function will download the dataset into your browser
----> 3 from pyodide.http import pyfetch
      5 async def download(url, filename):
      6     response = await pyfetch(url)

ModuleNotFoundError: No module named 'pyodide'
```

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.

In [ ]:

```
path = "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM
```

In [ ]:

```
#you will need to download the dataset; if you are running locally, please com
await download(path, "auto.csv")
```

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_id=NA-SkillsNetwork-Channel-SkillsNetworkCoursesIBMDeveloperSkillsNetworkDA0101ENSkillsNetwork20235326-2021-01-01 for free storage

Import pandas library

import pandas as pd

Read the online file by the URL provides above, and assign it to variable "df"

df = pd.read_csv(path, header=None)

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 [ ]:
```
# show the first 5 rows using dataframe.head() method
print("The first 5 rows of the dataframe")
```

In [ ]:
```
print("The last 10 rows of the dataframe\n")
```

In [ ]:
```
# create headers list
headers = ["symboling","normalized-losses","make","fuel-type","aspiration", "n
           "drive-wheels","engine-location","wheel-base", "length","width","heig
           "num-of-cylinders", "engine-size","fuel-system","bore","stroke","comp
           "peak-rpm","city-mpg","highway-mpg","price"]
```

In [ ]:
```
df.columns = headers
```

In [ ]:
```
df1=df.replace('?',np.NaN)
```

In [ ]:
```
df=df1.dropna(subset=["price"], axis=0)
```

In [ ]:

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.

In [ ]:

Read/Save Other Data Formats

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

In [ ]:

In [ ]:
```
# check the data type of data frame "df" by .dtypes
```

In [ ]:

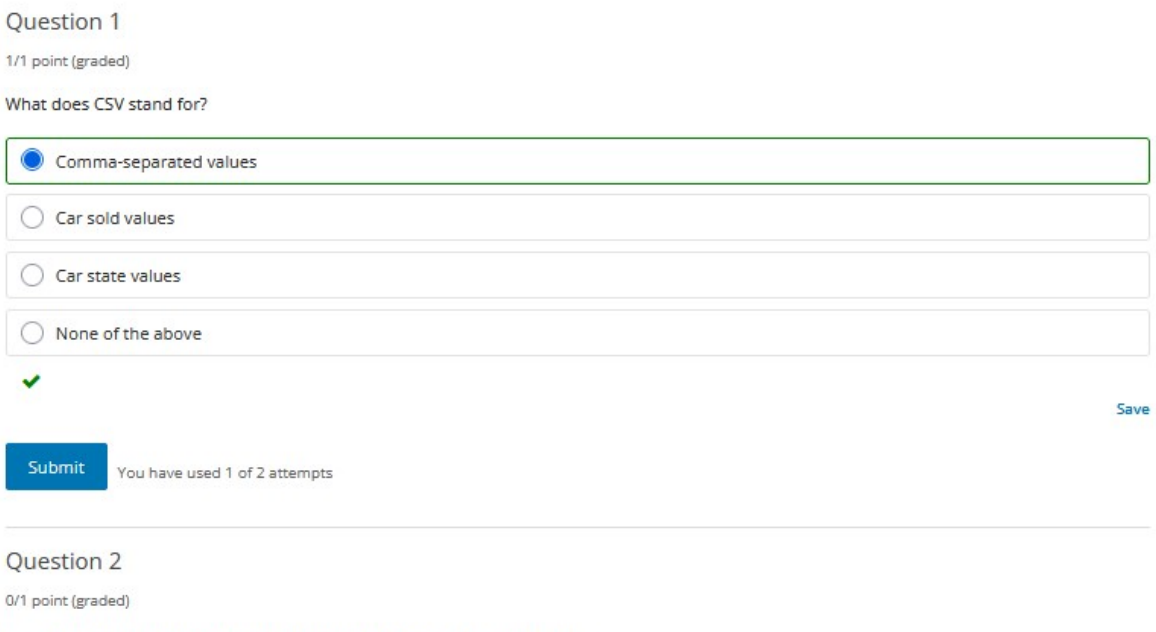
In [ ]:

In [ ]:
```
# describe all the columns in "df"
```

In [ ]:

In [ ]:

In [ ]:
```
# look at the info of "df"
```



In the data set, which of the following represents an attribute or feature?

- ○ Row
- ○ Column
- ◉ Each element in the dataset

✖

Submit You have used 2 of 2 attempts

Question 3

1/1 point (graded)

What is the name of what we want to predict?

- ◉ Target
- ○ Feature
- ○ Dataframe

✔

Save

Submit You have used 1 of 2 attempts

Question 4

1/1 point (graded)

What is the command to display the first five rows of a dataframe `df` ?

- ◉ `df.head()`
- ○ `df.tail()`

✔

Submit You have used 1 of 1 attempt

Question 5

1/1 point (graded)

What command do you use to get the data type of each row of the dataframe `df` ?

- ◉ `df.dtypes`
- ○ `df.head()`
- ○ `df.tail()`

✔

Save

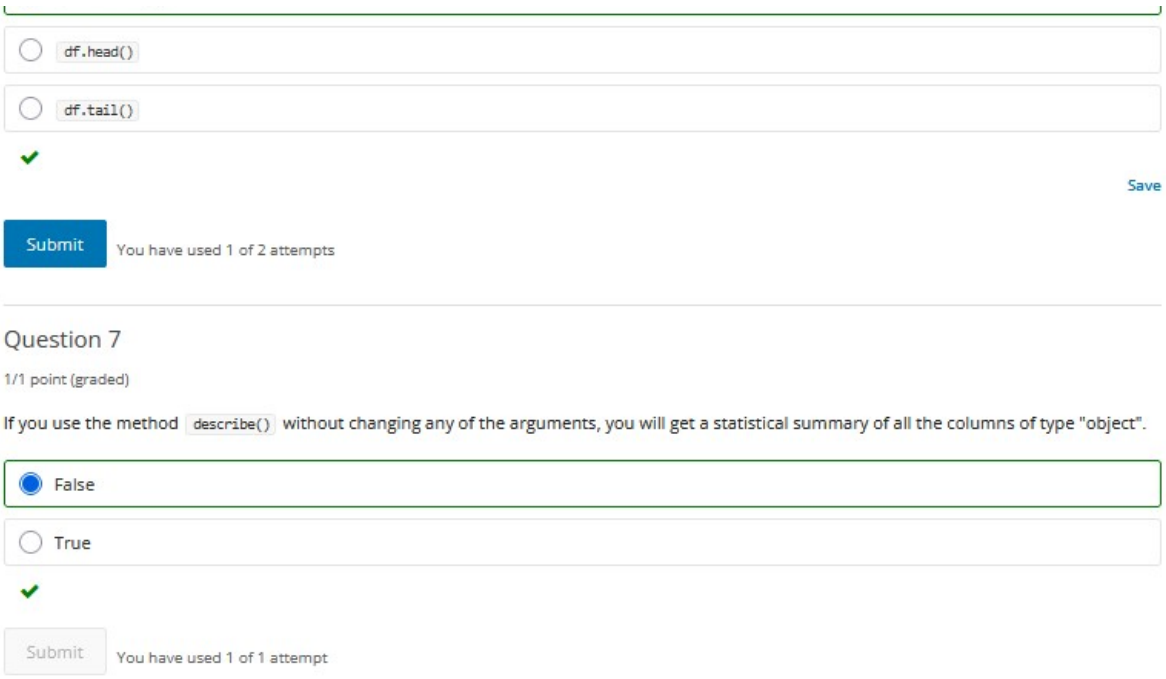
Submit You have used 1 of 2 attempts

Question 6

1/1 point (graded)

How do you get a statistical summary of a dataframe `df` ?

- ◉ `df.describe()`



Module 2 - Cleaning and Preparing the Data

Pre-Processing Data in Python.- convert raw to for another data.

- identify and handling missing value
- data formatting
- Data normalization
- Data Binning
- Turning categorical values to numerical values

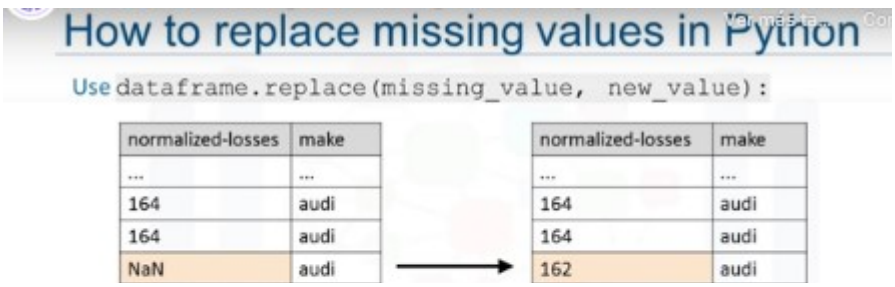
Dealing with a nmissing value

? "N/A", 0

- check with the data collection source
- Drop the missing values: drop variable or data entry
- replace the missing values: with average. by frequency. based on other functions.

dataframes.dropna() axis 0 the entire row, 1 drops the entire column, implace true writes the result back

Replace missing values



158	audi
...	...

158	audi
...	...

```
mean = df["normalized-losses"].mean()
df["normalized-losses"].replace(np.nan, mean)
```

COGNITIVE CLASS.ai

How to deal with missing data?

Check with the data collection source

Drop the missing values
- drop the variable
- drop the data entry

Replace the missing values
- replace it with an average (of similar datapoints)
- replace it by frequency
- replace it based on other functions

Leave it as missing data

COGNITIVE CLASS.ai

Data Formatting

- Data are usually collected from different places and stored in different formats.
- Bringing data into a common standard of expression allows users to make meaningful comparison.

Non-formatted:
- confusing
- hard to aggregate
- hard to compare

City
NY
New York
N.Y
N.Y

City
New York
New York
New York
New York

Formatted:
- more clear
- easy to aggregate
- easy to compare

Applying calculations to an entire column

- Convert "mpg" to "L/100km" in Car dataset.

city-mpg
21
21
19
...

city-L/100km
11.2
11.2
12.4
...

```
df["city-mpg"]= 235/df["city-mpg"]

df.rename(columns={"city_mpg": "city-L/100km"}, inplace=True)
```

Incorrect data types

- Sometimes the wrong data type is assigned to a feature.

```
df["price"].tail(5)

200    16845
201    19045
202    21485
203    22470
204    22625
Name: price, dtype: object
```

Correcting data types

To *identify* data types:

- Use **`dataframe.dtypes()`** to identify data type.

To *convert* data types:

- Use **`dataframe.astype()`** to convert data type.

Example: convert data type to integer in column "price"

```
df["price"] = df["price"].astype("int")
```

Binning

- Binning: Grouping of values into "bins"
- Converts numeric into categorical variables
- Group a set of numerical values into a set of "bins"
- "price" is a feature range from 5,000 to 45,500
  (in order to have a **better representation** of price)

price: 5000, 10000,12000,12000, 30000, 31000, 39000, 44000,44500

bins: low Mid High

Binning in Python pandas

price
13495
16500
18920
41315
5151
6295
...

→

price	price-binned
13495	Low
16500	Low
18920	Medium
41315	High
5151	Low
6295	Low
...	...

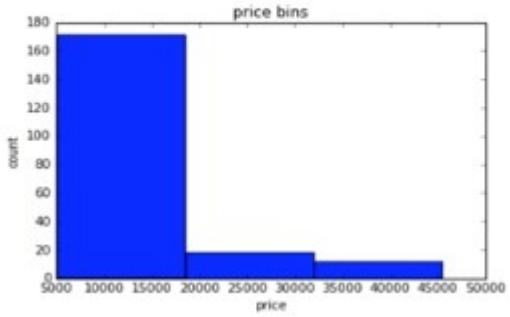
```
bins = np.linspace(min(df["price"]), max(df["price"]), 4)

group_names = ["Low", "Medium", "High"]

df["price-binned"] = pd.cut(df["price"], bins, labels=group_names, include_lowest=True )
```

Visualizing binned data

- E.g., Histograms



Categorical → Numeric

Solution:

- Add dummy variables for each unique category
- Assign 0 or 1 in each category

Car	Fuel	...	gas	diesel
A	gas	...	1	0
B	diesel	...	0	1
C	gas	...	1	0
D	gas	...	1	0

"One-hot encoding"

Dummy variables in Python pandas

- Use pandas.get_dummies() method.
- Convert categorical variables to dummy variables (0 or 1)

fuel
gas
diesel
gas
gas

→

gas	diesel
1	0
0	1
1	0
1	0

```
pd.get_dummies(df['fuel'])
```

LAB

Question #1:

Based on the example above, replace NaN in "stroke" column with the mean value.

avg_stroke = df["stroke"].astype("float").mean(axis = 0) print("Average of stroke:", avg_stroke)

df["stroke"].replace(np.nan, avg_stroke, inplace = True)

Calculate the mean value for the "horsepower" column

Question #2:

According to the example above, transform mpg to L/100km in the column of "highway-mpg" and change the name of column to "highway-L/100km".

df["highway-mpg"] = 235/df["highway-mpg"]

df.rename(columns={'highway-mpg':'highway-L/100km'}, inplace=True)

df.head()

Question #3:

According to the example above, normalize the column "height".

df['height'] = df['height']/df['height'].max()

df[["length","width","height"]].head()

Question #4:

Similar to before, create an indicator variable for the column "aspiration"
dummy_variable_2 = pd.get_dummies(df['aspiration'])

dummy_variable_2.rename(columns={'std':'aspiration-std', 'turbo': 'aspiration-turbo'}, inplace=True)

dummy_variable_2.head()

Question #5:

Merge the new dataframe to the original dataframe, then drop the column 'aspiration'.

df = pd.concat([df, dummy_variable_2], axis=1)

```
df.drop('aspiration', axis = 1, inplace=True)
```

Question 1

1/1 point (graded)

Consider the dataframe `df`. What is the result of the following operation: `df['symbolling'] = df['symbolling'] + 1`?

- (●) Every element in the column "symbolling" will increase by one.
- ( ) Every element in the row "symbolling" will increase by one.
- ( ) Every element in the dataframe will increase by one.

✔

Save | Show answer

Submit You have used 1 of 2 attempts

✔ Correct (1/1 point)

Question 2

1/1 point (graded)

Consider the dataframe `df`. What does the command `df.rename(columns={'a':'b'})` change about the dataframe `df`?

- ( ) Renames column "a" of the dataframe to "b".
- ( ) Renames row "a" to "b".
- (●) Nothing. You must set the parameter "inplace = True".

✔

Save | Show answer

Question 3

1/1 point (graded)

Consider the dataframe "df". What is the result of the following operation `df['price'] = df['price'].astype(int)`?

- ( ) Convert or cast the row 'price' to an integer value.
- (●) Convert or cast the column 'price' to an integer value.
- ( ) Convert or cast the entire dataframe to an integer value.

✔

Save | Show answer

Submit You have used 1 of 2 attempts

✔ Correct (1/1 point)

Question 4

1/1 point (graded)

Consider the column of the dataframe `df['a']`. The column has been standardized. What is the standard deviation of the values as a result of applying the following operation: `df['a'].std()`?

- (●) 1
- ( ) 0

3

Show answer

Submit You have used 2 of 2 attempts

Question 5 a)

1/1 point (graded)

Consider the column of the dataframe, df['Fuel'], with two values: 'gas' and' diesel'. What will be the name of the new columns pd.get_dumies(df['Fuel']) ?

- 1 and 0
- Just 'diesel'
- Just 'gas'
- 'gas' and 'diesel'

Save | Show answer

Submit You have used 1 of 2 attempts

Correct (1/1 point)

Question 5 b)

1/1 point (graded)

What are the values of the new columns from part 5a)?

- 1 and 0
- Just 'diesel'
- Just 'gas'
- 'gas' and 'diesel'

Save | Show answer

Submit You have used 1 of 2 attempts

Correct (1/1 point)

In [ ]: