Data Intake Report

Name: File ingestion and schema validation
Report date: 13.4.2021
Internship Batch:LISP01
Version:2.0
Data intake by:Ines Perko
Data intake reviewer:<intern who reviewed the report>
Data storage location: https://github.com/inesp93/File-ingestion-and-schema-validation

Tabular data details:

Total number of observations	84897528
Total number of files	2
Total number of features	13
Base format of the file	.csv
Size of the data	6,57 GB

Proposed Approach:

The initial dataset is https://www.kaggle.com/mkechinov/ecommerce-behavior-data-from-multi-category-store?select=2019-Oct.csv, where the file 2019-Oct.csv has 5.28 GB. First I tried to read the .csv file with pandas pd.read_csv( ). Even though it was 5GB, it was successful and it was done in 2minutes.

```
In [2]: import pandas as pd #normal pandas

In [5]: %%time
        pd.read_csv('Desktop/week 6/2019-October/2019-Oct.csv')
        Wall time: 2min
```

In version 1 of my data intake report, I couldn't install Modin, but now I managed to install it. Also, I had to extend my virtual memory because it didn't want to work. Now I have results to compare:

```
%%time
import modin.pandas as pd
x=pd.read_csv('Desktop/week 6/2019-October/2019-Oct.csv')
Wall time: 53.9 s
```

Without modin, wall time was 2 minutes, and with modin it is 53.9. Modin speeded up pandas workflows. Now I meet another problem. What I mean is when it comes to the following point it runs out of the memory.

```
: #read the file using config file
file_type = config_data['file_type']
source_file = "Desktop/week 6/2019-October/" + config_data['file_name'] + f'.{file_type}'
#print("",source_file)
df = pd.read_csv(source_file,config_data['inbound_delimiter'])
df.head()
```

```
(pid=None) [2021-04-13 22:00:08,248 E 13072 1552] create_request_queue.cc:119: Not enough memory to create object d251967856448
cebffffffff0100000001000000 after 5 tries, will return OutOfMemory to the client
```

Therefore, the approach that I had before (i.e., using only part of the dataset), is still my main approach, until I fix this problem.

The new dataset is called New and it has 1,30 GB. It has columns: 'product_id', 'price', 'brand', 'user_id'. I replaced the Nan values in the column brand with the string ‘unknown’. I modified the yaml file with columns: {'product_id', 'price', 'brand', 'user_id'}.
I

```
In [33]: config_data
```

```
Out[33]: {'file_type': 'csv',
          'dataset_name': 'newdata',
          'file_name': 'New',
          'table_name': 'edsurv',
          'inbound_delimiter': ',',
          'outbound_delimiter': '|',
          'skip_leading_rows': 1,
          'columns': {'product_id': None,
           'price': None,
           'brand': None,
           'user_id': None}}
```

```
In [34]: #read the file using config file
file_type = config_data['file_type']
source_file = "Desktop/week 6/" + config_data['file_name'] + f'.{file_type}'
#print("",source_file)
df = pd.read_csv(source_file,config_data['inbound_delimiter'])
df.head()
```

Out[34]:

	product_id	price	brand	user_id
0	44600062	35.79	shiseido	541312140
1	3900821	33.20	aqua	554748717
2	17200506	543.10	unknown	519107250
3	1307067	251.74	lenovo	550050854
4	1004237	1081.98	apple	535871217

I checked if the header of the file is validated.

```
In [35]: #validate the header of the file
util.col_header_val(df,config_data)
```

```
column name and column length validation passed
```

```
Out[35]: 1
```

```
In [36]: print("columns of files are:" ,df.columns)
print("columns of YAML are:" ,config_data['columns'])
```

```
columns of files are: Index(['product_id', 'price', 'brand', 'user_id'], dtype='object')
columns of YAML are: {'product_id': None, 'price': None, 'brand': None, 'user_id': None}
```

Furthermore, I continued with the code, but I am still not finished with the inspection.

```
import os
import math

if util.col_header_val(df,config_data)==0:
    print("validation failed")
    # write code to reject the file
else:
    print("col validation passed")
    count_row = df.shape[0]  # Gives number of rows
    print("total number of rows", count_row)
    count_col = df.shape[1] # Gives number of columns
    print("total number of col", count_col)
    file_size = os.path.getsize(source_file)
    fs=(file_size/1073741824)
    print("Size of the file is %.2f GB" % round(fs, 2))
    # write the code to perform further action
    # in the pipleine
```

```
column name and column length validation passed
col validation passed
total number of rows 42448764
total number of col 4
Size of the file is 1.31 GB
```