Data generation

Why Fake Data Generator?

- Data is king
- All ML projects need data
- Hard to get real data with increase in privacy and security concerns
- Companies are restricting customer data usage
- Load test – need large volume of data

Objective

The objective of this project is to build a package to generate random sample data set defined by user

- **Flexibility/Configurability:** Simple option to provide configuration:, so users need flexibility to generate data based on their conditions.
- **Extensibility:** Support custom data input: Each field needs different data, for example, medical data is different from sales data. So, it is helpful if the package supports bringing their own data.
- Manage relation between data columns
- Generate pandas data frame: Simple to output or re structure in the required format

Targeted Users

- Student
- Developer/Programmer
- Data scientists
- Analyst/Researcher

Implementation

Create a package to generate a dataset using configuration. Provide flexibility to users to bring their own data or define rules to generate data. My primary objective is to create a project using some of the concepts I learned from Data 515 class.

- Design concepts that are considered in implementation:

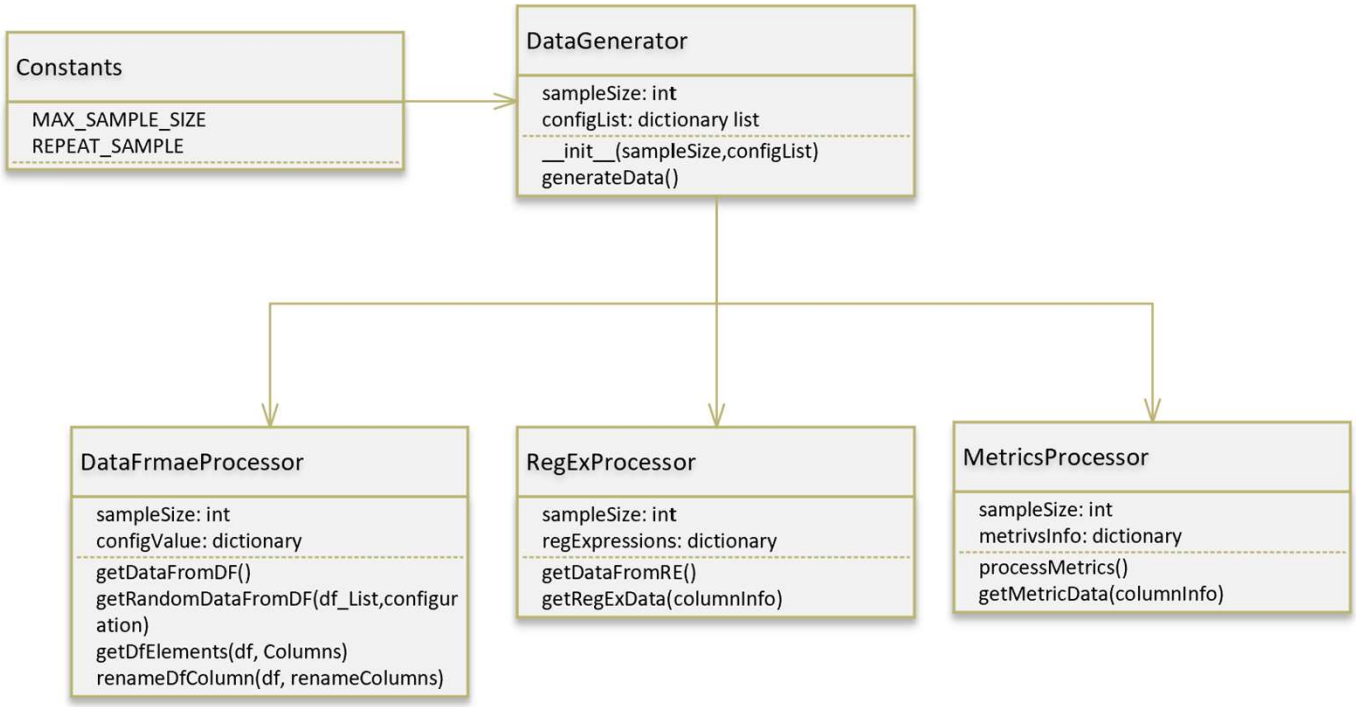
- Object oriented design: Created Data Generator class and defined various methods in it.
- General purpose deep modules: Defined helper class as general-purpose deep module. Depending on user input, the data generator performs various tasks and returns requested data.
- Separation of concern: Created modules or functions to perform each separate/independent tasks, for example defined independent functions to process data from data frames and process regular expressions
- Information hiding: Most of the complex processing logic is hidden from the user and defined in helper class.
- Error handling: Included error handling using try and except and raised error to end user if there is any failures
- Testing: Included a few unit test cases and examples to understand how to prepare the configuration

Libraries

Following libraries/modules are used in this project.

- pandas: To use pandas data frame and operations on it
- random: To generate random value or sample
- rsts: To generate random string using simple regular expression

Class diagram



How to use

- Install myfaker package using setup
- In your script file, Import myfaker package
- Define data schema and prepare configuration dictionary list
- Create an object and call generate Data function with parameters (number of rows to generate, configuration dictionary list)
- Capture return data frame

Example#2

Generate random data using regular expression. In this scenario no input data is required.

```
from MyFaker.code.helper import DataGenerator

configList = [{'sourceType':'RegularExpression', 'values' : [ {'name' :'Featurs'
               ,'columns': [{'colName' : 'FirstName', 'prefix' : 'FirstName',  'sufix' : '', 'regExpression' : '[0-9]{2}'  }
                           ,{'colName' : 'UserEmail', 'regExpression' : '[A-Za-z]{6,10}[0-9]{2}@[a-z]{5}\d\.com' } ]}]}
             ,{'sourceType':'Metrics', 'values' : [ {'name' :'Featurs'
               ,'columns': [{'colName' : 'SalesQuantity', 'dataType' : 'int', 'startValue' : '10', 'endValue' : '40'  }
                           ,{'colName' : 'SalesAmount', 'dataType' : 'float', 'startValue' : '10', 'endValue' : '40'  } ]}]}
            ]

dg = DataGenerator(10,configList)
dfr = dg.processInput()
print(dfr)
```

	FirstName	UserEmail	SalesQuantity	SalesAmount
0	FirstName77	EdazLv72@yhopz7.com	40	15.711852
1	FirstName61	AuueXfjS62@foxfx2.com	12	22.351783
2	FirstName22	EuzYtEDx98@golia4.com	31	37.666205
3	FirstName47	tiJapfiSG18@dzkix6.com	25	11.626575
4	FirstName44	nzHOYMmrnt64@wxzzm0.com	10	14.879003
5	FirstName43	BacxPA97@hgftb6.com	17	27.401190
6	FirstName38	iUcXaW58@duzvu8.com	33	28.602415
7	FirstName95	RMGamxKL56@cguxl0.com	31	33.526898
8	FirstName63	FdzygrzbF49@fpueb5.com	26	28.541427
9	FirstName94	YHDtLYXs56@ibgbb3.com	19	18.988063

Example#2

Generate random data using data frame. In this scenario user can get data from a file or create a list and convert it to data frame.

```
import pandas as pd
from MyFaker.code.helper import DataGenerator

df_Color = pd.DataFrame(['Green','Red','Yellow','Blue'], columns = ['Color'])

configList = [{'sourceType':'dataframe', 'values' : [ {'name':'df_Color', 'df': df_Color
                    ,'columns': [{'colName':'Color', 'colRename':'TagColor', 'prefix':'' }
                                ]}]}
              ,{'sourceType':'Metrics', 'values':[{'name':'Featurs'
                ,'columns': [{'colName':'SalesQuantity', 'dataType':'int', 'startValue':'10', 'endValue':'40'}]}]}
             ]

dg = DataGenerator(10,configList)
dfr = dg.processInput()
print(dfr)
```

	TagColor	SalesQuantity
0	Yellow	15
1	Blue	37
2	Red	32
3	Yellow	24
4	Green	18
5	Yellow	13
6	Yellow	33
7	Red	27
8	Red	26
9	Blue	10

Comparison

Following libraries/modules are used in this project.

Limitations & Future Work

Limitations:

- rstr module doesn't support all regular expressions, any unsupported complex expression required own implementation
- Not using distributes design, so result data set size & volume depends on user system capacity

Future Extensions:

- Simplify how user can pass configuration, something like user pass (categorical, categorical, categorical, int, int, float)
- Extend to generate how many distinct values should generate for each categorical feature
- For Metrics – right now we generate random number, extend it to support any specific distribution