PharmaSUG China 2021 - Paper AD-061

Leveraging Python Streamlit to do project management

Zhiwei Luo, Pfizer (China) Research and Development Co.,Ltd. China

ABSTRACT

In our daily work,we often need to manage multiple clinical trial projects simultaneously and build up our teamwork and collaboration. During the project processing period, timely tracking and presenting the generation and QC situation of corresponding TLFs and datasets per colleague can effectively help us in project management and further work scheduling. We can use python to get the information of generation files and QC files respectively, then use LoT (List of Table) as a bridge to connect them by file name with “fuzzywuzzy” package for fuzzy match. A Python open-source lightweight web development framework package “Streamlit” is introduced in this paper to build an app for the users (who don’t know Python) as a tool to present the project process in graphs and summary tables timely.

INTRODUCTION

When a clinical trial project starts, the project lead usually assigns tasks to the programmers to produce and QC the corresponding datasets and TLFs based on the approved LoT. In the course of the project, if the project lead wants to know the project progress, there are several tedious places existed, firstly, it involves switching back and forth between multiple folders, secondly, when categorizing the production files and the qc files to the lot, due to the different naming style it is very hard to connect them quickly and accurately, besides the file itself always exist minor differences, in addition, the work is real-time and dynamic therefore, to timely accurately track the progress of the project, is a very time consuming work.

Despite a tedious process, but meaningful for both project lead and SAS programmer.

For project leader, it can provide below merits:

- Keep track of the status of each programming delivery.
- Assign programming resource effectively.
- Assess potential risk timely and update programming status to project team.

For Programmer, it also can provide below merits:

- For Production Programmer, they can know the status of their assignments and address backlog timely, they can know whether QC has been performed or not.
- For QC Programmer, they can know whether specific Production work has been ready for QC. They can know their backlogs to arrange time to clean them.

This paper is come to provide a solution ,to overcome the challenges mentioned above.

PROCESS FOR BUILD THE PROJECT MANAGENMENT STREAMLIT WEB APPLICATION

To track our project progress, we always follow the below process and our application development follow this process as well, the difference is that we automate the processes.

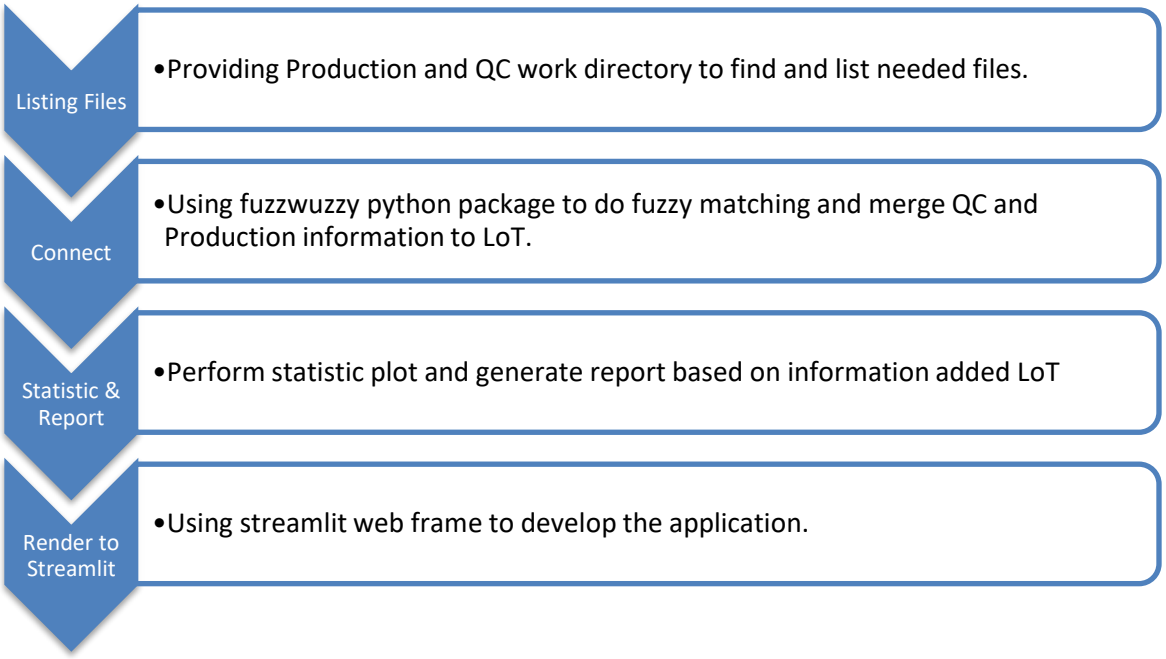

Figure 1: Process for build the streamlit web application

In the following sections I will expand on how to automate these processes.

LISTING FILES

As with other programming languages, Python can get information from the corresponding folders, with the help of the module OS, it becomes easier and more convenient.

```
#Load module
import os

#Paths for QC files and Production files
qcfilepath='xxxxxx'
productionfilepath='xxxxxxxxxxxxxxx'

#List files in the path.
qcfiles= os.listdir(qcfilepath)
productionfiles=os.listdir(productionfilepath)
```

The result of the os.listdir() method was a list that contents the name of all the files under the corresponding folders ,you can do further filter by simply using list comprehension to get what you want.In addition, as far as I know, many companies stored their files in a remote server, if so, package 'pysftp' can help, below is the sample code.

```
#Load module
import pysftp

#User information
user='xxxxxxxxx'
pwd='xxxxxxxxx'

#FTP information
cnopts=pysftp.CnOpts()
cnopts.hostkeys=None
```

```
hostname='xxxxxxxxx'

#Paths for QC files or Production files
qcfilepath='xxxxxx'
productionfilepath='xxxxxxxxxxxxxxx'

#List files in the remote path.
with pysftp.Connection(hostname,username=user,password=pwd,cnopts=cnopts)
as ftp:
   ftp.cwd(qcfilepath) #Set current work folder
   qcfiles=ftp.listdir() #List all the file in the remote path
```

CONNECT

It is always a difficult problem to match production or qc file's information with the LoT. As mentioned above, each SAS programmer has a different naming style for their files and it is difficult to achieve uniformity. In addition, when there are a large number of files with similar names, it is even not easy to distinguish themself with naked eye. In this section I'll introduce a python package named Fuzzywuzzy, to solve this problem which can do fuzzy match efficiently.

Introduction of Fuzzywuzzy

Fuzzywuzzy basically using Levenshtein Distance to measure similarities and providing multiple methods to calculate the difference between two strings, Measuring similarities by scoring from 0 to 100 makes it easy to distinguish the difference between two strings. The following table describe four commonly used methods.

Method	Describe
fuzz.ratio	Simply calculate similarity ratio using the Levenshtein distance
fuzz.partial_ratio	Regarding the short string as a substring against long string and matching it with all substrings that are of the same length
fuzz.token_sort_ratio	Before calculating similarities, several processes will execute first including Changing capitals to lowercase, Eliminating all non-alpha, non-numeric characters and sorting the strings alphabetically.
fuzz.token_set_ratio	The only difference with fuzz.token_sort_ratio,fuzz. Is token_set_ratio will take out the intersection of two strings before calculating similarities.

Table 1:Four commonly used methods provided by Fuzzywuzzy

A simple example with sample code to demonstrate fuzzy matching

Assuming 'adsl_s001_1' is the required file name ,QC programmer may name their corresponding QC file as qc_tlf_adsl_s001_1 or others. I created some possible naming conventions as choices, below sample code will show you how it works.

```
#Load module
#Import modules
from fuzzywuzzy import fuzz
from fuzzywuzzy import process

#Test example
query='adsl_s001_1'
choices=["Adsl_s001_1_xxxx","Qc_tlf_Adsl_s001_1_xxxx","ADSL_S001_1_xxxx",
"ADSL_S002_1_xxxx","ADSL_S002_1_xxxx","ADSI_S002_1_xxxx","adsl_s001_1",
"adsl_s001_1_1"]

#Firstly using fuzz.partial_ratio() to find "adsl_s001_1' is the substring
of compared one.
```

```
bests=process.extract(query,choices,scorer=fuzz.partial_ratio,limit=10)

#output of bests
"""[('Adsl_s001_1_xxxx', 100),
 ('Qc_tlf_Adsl_s001_1_xxxx', 100),
 ('ADSL_S001_1_xxxx', 100),
 ('adsl_s001_1', 100),
 ('adsl_s001_1_1', 100),
 ('ADSL_S002_1_xxxx', 91),
 ('ADSL_S002_1_xxxx', 91),
 ('ADSI_S002_1_xxxx', 82)]"""

#From the output of bests, we can see each compared string was scored, and
if scored 100,we can make sure 'adsl_s001_1' is the substring of it.

bests=[i[0] for i in bests if i[1]==100]

#Deal with similarities after previous matching
#Using simple ratio method to find the most similar one.
best=process.extractOne(query,bests,scorer=fuzz.ratio)

#output of best
# ('adsl_s001_1', 100)
```

We can easily find the most similar one from a large number of choices by using Fuzzywuzzy, Wrapping above steps into a function then apply to LoT dataframe, the 'connect' step was completed.

STATISTIC & REPORT AND RENDER TO STREAMLIT

Statistics and reporting are not the focus of this paper, and the project progress report objectives varies from company to company, so we will not focus on them in this paper. As a substitute,in this section I attempt to build a simple app as a demo to demonstrate the commonly used steps when developing a streamlit app.

Assuming you want to simulate a dynamic normal distribution by python. Firstly, we should figure out what we want to take as input parameters, obviously in this case, input parameters are Mean and SD, then find corresponding stramlit input widgets to implement parameter input. Secondly, we perform intermediate processing by using input data. Finally, we determine what output we want to show with corresponding streamlit display widgets.

In short, a streamlit app=corresponding streamlit widgets + corresponding data source. Below I'll show how this simple web app to be created.

```
#Import modules
import streamlit as st
import pandas as pd
import numpy as np
import plotly.express as px
st.title('Normal Distrubution')

#Firstly Input Parameters
sigma=st.slider('Sigma', 0,20,10)
mean=st.slider('Mean',0,100,20)
```

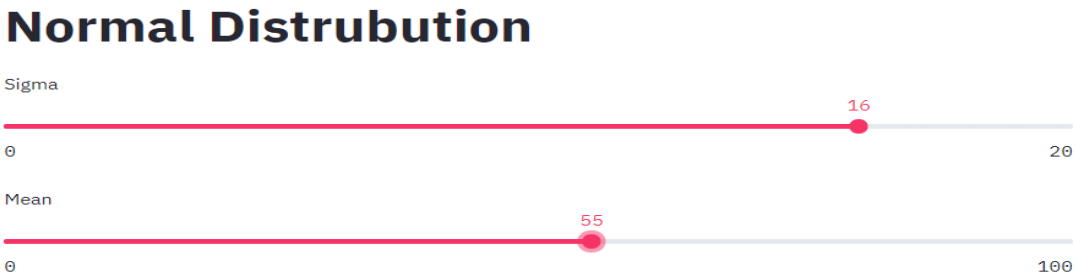

Figure 2:Sample streamlit input widgets

```
#Secondly intermediate caculate processing
data=pd.DataFrame(np.random.normal(mean, sigma,size=(10,10)))
fig = px.histogram(data)


#Finally Choose output and render with correspomding streamlit widget
st.dataframe(data)
st.plotly_chart(fig)
```

	0	1	2	3	4	5	6	7
0	75.2135	45.5016	76.8051	70.9475	71.4090	41.7607	70.5431	44.7522
1	57.1632	68.8762	70.3447	59.8215	83.6908	31.7721	51.3466	64.9765
2	30.8748	47.7179	47.4482	41.2397	43.0869	67.3219	38.7308	58.8666
3	55.1214	21.7286	44.8665	77.7865	43.0295	44.3103	51.4409	54.9013
4	20.2097	75.4442	60.5487	49.9822	60.2586	52.4390	70.3162	59.2536
5	40.9663	71.5194	60.6737	37.2874	59.0455	44.5740	29.7426	49.2182
6	47.7983	67.1775	69.8030	104.0475	44.2184	46.7480	45.0698	54.7709
7	50.5820	81.8222	77.4527	77.2524	73.2370	56.7366	49.5100	58.8119
8	60.5638	49.0541	20.1007	35.7335	9.6432	30.3216	31.1897	70.1651
9	39.7028	61.6309	53.9019	58.3033	91.8197	78.0284	31.4513	76.1920

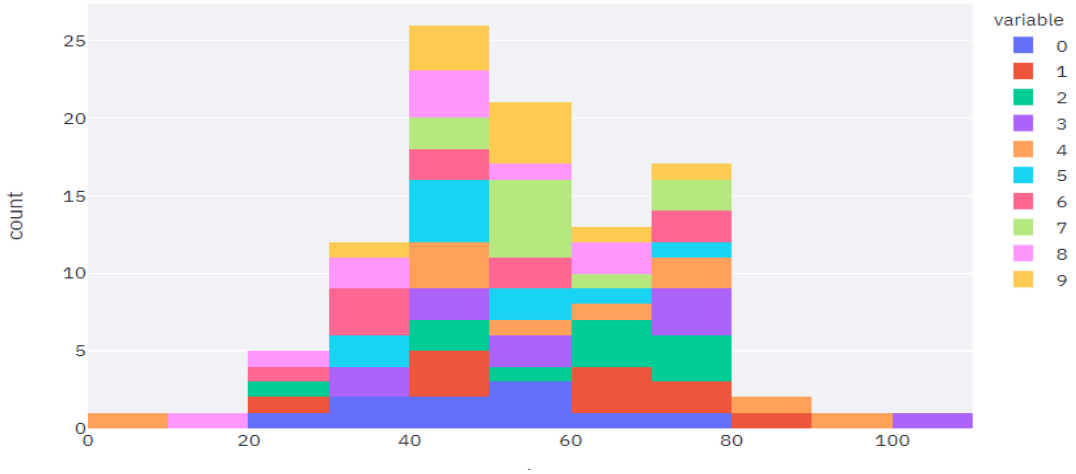

Figure 3:Sample streamlit display widgets

Just 11 lines of pure codes, a normal distribution simulator app was created, even more concise than R shiny.If we put project progress status statistical results into corresponding streamlit display widgets,a project progress tracking tool was created.

CONCLUSION

In the process of clinical trials, there are many complex and repetitive manual work, although many companies have started to promote the use of Python or R to automate some tasks, but not all SAS programmers understand Python, and ultimately these tools are difficult to be used, Streamlit can quickly turn python scripts into easy-to-use app, overcoming the last mile of being applied, hope this paper inspire you to develop more automate tools for our daily work.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Zhiwei Luo
Pfizer (China) Research and Development Co.,Ltd. China
17607086677
luozhiwei@live.cn