**Data Science Practicum I**

Python is a high-level programming language that includes data structures, dynamic typing and binding, and can be used for Rapid Application Development (RAD), scripting, or connecting existing components together. It’s easy to learn, making it perfect for beginners into Data Science and Data Analytics, and its readability and syntax reduces program maintenance cost. It first appeared in 1991 as Python 0.9.0, and has had continued stable releases since then. Python 3 was released in 2008, encompassing a complete revision of the programming language. Among several of the new features is a more effective output with mathematical calculations, and a new method of printing outputs completely. In fact, Python 3 was not backwards compatible with Python 2 coding, and Python 2 itself was phased out in 2020. One of the greatest strengths of Python is its large standard library, which provides many tools that can be suited to various tasks. Some of these libraries have been showcased in this Practicum, but the Python Package Index has currently over 290,000 packages within the official repository. More information on the Python software can be found at (<https://python.org>).

Because of its many packages, Python can be used to collect and analyze data, as well as visually output it into many different varieties and types of graphs to showcase your projects. It is also because it is so accessible that I chose to use it for my own Practicum.

**Project Overview**

This project is based on my current job role as a Quality Assurance Tech in crop testing. Each test we conduct has a start date and an end date. Ideally, the sample should only have a day average of seven before we send the results to the customer. I set up this project to replicate the way a data analyst would take the number of results that stay within the seven-day average and determine if our department was staying within the quota or whether there might be issues that need to be addressed. These issues could be whether overtime should be allowed, whether we need to hire more techs or pull techs from other departments to help with the workflow, or if there is a problem with the testing process that needs to be investigated.

This project works in three sections. First, I collect the data and obtain the export file, which had just over 100 samples listed. While I could use this data for private projects, I do not have permission to release it publicly. To compensate for this, I created a secondary output file with similar data, and ran the algorithm through the secondary output as well. Additionally, I created a third file that had 110 samples. This third file had some samples with where start or end dates were purposely left out to mimic either a program or human error. This third file was also run through the algorithm.

Next, I clean and organize the data so that it will be more efficient and visually appealing to graph. This starts with calculating the difference between the start and end dates, phasing out the data of the start and end dates and leaving only the difference, and then organizing the data calculated from the difference to be able to be visualized through a graphical format.

Finally, I graph the data into a format that is legible to both people who have experience in data analytics and those without experience. After some experimentation I determined that a bar chart would best show the comparison between the samples that pass the seven day quota versus the samples that run over eight days and fail.

**Data**

Source: The original data is property of the company that produced it, and as a result can be publicly displayed but not re-distributed. Data of a similar year and month will be re-created via Microsoft Excel for the purpose of being publicly distributed and tested. It is that data that will be further discussed. A third output file of data will also be displayed. This third file has missing or corrupted data in order to ensure the algorithm works with non-perfect data.

Scope: The data consists of just over 100 tests ran between August 1, 2019 and August 31, 2019. Only test dates that had both a start date between August 1 and August 31 are used. The third data files are also run through the analysis with missing start and end dates to showcase events of possible human error. The third data file also ran from August 1, 2019 to August 31, 2019, with 110 test samples in the file.

**Process**

This process was completed on a Windows 10 system, with Jupyter Notebook and Python 3.9.0.

1. Gather the data.
   1. The original data was entered into a macro through Microsoft Excel, the output file being an xlsx.
   2. The data fitting the criteria is saved into an output file that should then be converted into a csv or other readable format for Python. The original data file was converted from an xlsx to a csv using the Microsoft Excel program.
   3. Once the data is compiled, it can be imported into either Python, Python Shell, or Jupyter Notebook. The data file included was written in Jupyter Notebook.
   4. I have included a scraper that can be used to scrape data from social media sites like Twitter to gather data for analysis if an output is not readily available.
   5. Loading the data into Jupyter Notebook can be done with

```df=pd.read\_csv(‘C:/Users/Student/FileFolder/data.csv’, parse\_dates=[0,1])```

1. Clean and organize the data.
   1. Formatting the two columns of data into three columns with the start date and end date remaining, in addition to the diff column which now shows the calculated difference between the start date and end date.

```df[‘diff’]=(df[‘End date’] – df[‘Start date’]).dt.days ```

* 1. Drops all blank or N/A data boxes.

``` df.dropna(inplace=True) ```

* 1. Places the cleaned data into organized columns labeled as diff, under, and over. This also removes the start and end date headers and data. The data is renamed from df to category.

``` category=df.groupby(‘diff’)\

.agg(under=pd.NamedAgg(column='diff',

aggfunc=lambda x: (x < 8).sum()),

over=pd.NamedAgg(column='diff',

aggfunc=lambda x: (x > 7).sum()))\ .reset\_index() ```

* 1. Remove the diff label from the category dataset and display the result.

``` category\_p=category[[‘under’,’over’]]

category\_p.head() ```

* 1. The column heads are changed from under and over to pass and fail.

``` category\_p=columns=[‘pass’,’fail’] ```

* 1. The categories of pass and fail both have their sums calculated and then displayed.

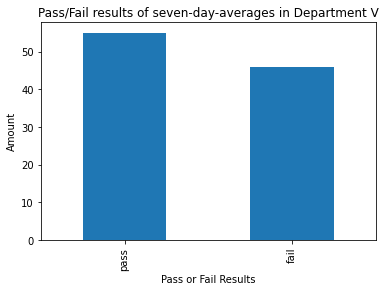
``` category\_pf=category\_p.sum()

category\_pf.head() ```

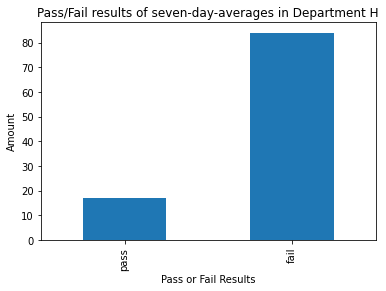
1. Visualizing the results.
   1. Plots the results of the pass and fail sums into a bar graph.

``` my\_plot = category\_pf.plot(kind='bar',title="Pass/Fail results of seven-day-averages in Department") my\_plot.set\_xlabel("Pass or Fail Results") my\_plot.set\_ylabel("Amount")

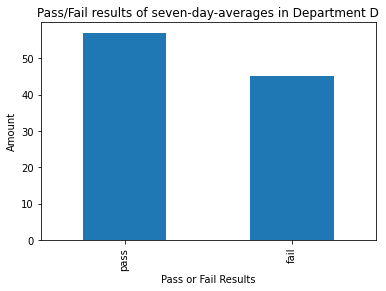
```



*The results of the original data.*



*The results of the mimicry data.*

**

*The results of a third department’s data – some start/finish dates were missing.*

**Tools**

1. Python, the primary coding language for this project.
2. Jupyter Notebook, the program, used to code with Python for this project.
3. Microsoft Excel, the program that created the randomized data and housed the original, protected data.
4. Numpy, for the data structures and multi-dimensional arrays. (https://numpy.org/)
5. Pandas, to manipulate and analyze the data in multiple formats. (https://pandas.pydata.org)
6. Pandas\_profiling, allows exploratory data analysis to be completed quickly (https://pandas-profiling.github.io/pandas-profiling/docs/master/index.html)
7. Matplotlib (specifically .pyplot, .image, and %matplotlib inline), visualization tools included with python that allows the creation of the used bar graphs. (https://matplotlib.org/)
8. Seaborn, a second visualization library that allows us to further manipulate our graphs and labels associated with them. (https://seaborn.pydata.org/#:~:text=Seaborn%20is%20a%20Python%20data,can%20read%20the%20introductory%20notes.)
9. Datetime, a module that allows the manipulation of dates and times in different ways. (https://docs.python.org/3/library/datetime.html)

**Credits**

A lot of gratitude is owed to:

Stackoverflow.com for helping me discover solutions to error messages I’ve never seen before by combing its forums.

Regis University for teaching me the skills in Python and analytical thinking to finish this project and find confidence in my abilities.

My supervisor B., who gave me the inspiration for this project and helped me relate it to our real-world job.