**PyPoll Analysis**

The first step to perform this analysis is to investigate the dataset: election\_data.csv that was provided for purposes of doing this exercise.

By opening this file into our code editor, we can see that is composed by only three columns:

Voter ID, Count and Candidate which hold records of the elections of four candidates.

Once we finished with getting acquainted with our dataset, we can start writing our code to get the information that we want. First, we import that required libraries: os and csv, next we specify the path where our csv of interest is located at:

csvpath = os.path.join('Resources', 'election\_data.csv')

And then we pass this path into our os function “with open (path, ‘r’) as file:”

with open(csvpath, 'r') as csvfile:

csvreader = csv.reader(csvfile)

header = next(csvreader)

By doing this we tell our computer to open and read the file that we specify with the path we pass as the argument. The next step in our code is to use csv.reader function to read the desired file in a csv way and then we skip the first line because we do not want to include the header of the columns in our next step.

Now, we use a for loop to read the csv we load into the csvreader line by line:

vote\_count = 0

candidates = {}

for row in csvreader:

vote\_count += 1

if row[2] not in candidates:

candidates[row[2]] = 1

else:

candidates[row[2]] += 1

The besides reading our csv, the main objective of this for loop is to obtain the total vote count of the csv which we get from placing a counter: vote\_count and get the unique values of candidates who participate in the election in order to that we create an empty dictionary that we can fill according to the if/else statement we place inside the for loop. This if statement checks if the candidate is already in the dictionary and if is not it place its name a key and set with a value of 1, meanwhile the else statement checks if the candidate already exists then it adds 1 more to the value the already has. This if/else statements let us know how the votes were disturbed among all the candidates.

print(f'Election Results \n-------------------------\nTotal Votes: {vote\_count}\n-------------------------')

for key, value in candidates.items():

cand\_props = candidates[key]/vote\_count

cand\_percent = "{:.3%}".format(cand\_props)

candidates[key] = [cand\_percent, value]

for keys, value in candidates.items():

print(f'{keys}: {candidates[keys][0]} ({candidates[keys][1]})')

print(f"-------------------------\n"

f"Winner: Khan\n"

f"-------------------------")

Our final block of code iterates through the dictionary we created and calculate the proportion of vote for each candidate and then convert cand\_props to a % format cand\_percent. Finally, we add this cand\_percent as a value to its corresponding candidate.

The last step is to run another for loop in our final version of our dictionary to display the final results which outputs:

Election Results

-------------------------

Total Votes: 3521001

-------------------------

Khan: 63.000% (2218231)

Correy: 20.000% (704200)

Li: 14.000% (492940)

O'Tooley: 3.000% (105630)

-------------------------

Winner: Khan

-------------------------

We can conclude that Khan totally surpass the other three candidates by obtaining 63% of the votes 2,218,231 out of 3,521,001 which led him to victory. The next candidates who obtained more votes besides him were Correy 20%, Li 14% and O’Tooley 3%.