AIR Question 3

April 9, 2020

1 Question 3

Before taking on this question, I will import pandas to help me work with the csv dataset.

[1]:
```
import pandas as pd
```

[2]:
```
# I kept the csv in the same directory as this file
patents = pd.read_csv('patent_drawing.csv')
```

[3]:
```
# I take a quick peak at how the information came in
patents.head()
```

[3]:
```
                         uuid patent_id  \
0  urfl2ulyjgez01g5selfflnz7   4491930
1  nqdxwthotlcted3d961ao373x   4490979
2  9mwinm7as0p0j3245tdxhfuiz   4491969
3  l1n6w0ofqic6yow2t7qwmvqry   4490948
4  86bndneq4omf3mfxi60dzr5mx   4491426

                                                text
0  A better understanding of the invention may be...
1  A better understanding of the invention will b...
2  A better understanding of the invention will b...
3  A better understanding of the present inventio...
4  A better understanding of the present inventio...
```

1.1 Part A

Hopefully I am understanding this question correctly. It seems like we want to see how many rows have the word(s) “view” or “perspective” in them while not including those rows which also have “bottom”, “top”, “front” or “rear” in the text field.

[4]:
```
# To make this more dynamic, I make a list of what we want to find and what we␣
↪want to avoid
target_words = ['view', 'perspective']
avoid_words = ['bottom','top','front','rear']
```

[5]:
```
count = 0

for s in patents['text']: # parsing through each description
    split_text = s.split() # splitting each text by a whitespace, because words␣
 ↪like 'ontop' should not flag
    flag = 0
    for a in avoid_words: # comparing each of the words we want to avoid with␣
 ↪the text
        if a in split_text:
            flag += 1 # if a word we should avoid is found in the text, we flag␣
 ↪it to move to a new text
            break
    if flag < 1: # testing to see if our first search flagged
        for t in target_words: # following the same setup as in our first test
            if t in split_text:
                count+=1
    else:
        pass # if the flag is raised, we move on to the next text line

print('The number of text entries that include \"view\" or \"perspective\" but '
      + str('not \"bottom\", \"top\", \"front\", or \"rear\" is: ')+ str(count))
```

```
The number of text entries that include "view" or "perspective" but not
"bottom", "top", "front", or "rear" is: 3955
```

1.2 Part B

Because some of the patent_ids are strings, it is harder to groupby patent_id. So we need to understand how the patent_id data looks like.

[6]:
```
patents['patent_id'].describe()
```

[6]:
```
count         8156
unique        1096
top        4491287
freq            59
Name: patent_id, dtype: object
```

The simple description actually allows us to understand how to calculate the average number of pictures per patent. We will take the total number of entries and divide by the number of uniquie entries.

[7]:
```
avg = len(patents.index)/patents['patent_id'].nunique()

print('The average number of pictures per patent is: '+ str(avg))
```

```
The average number of pictures per patent is: 7.4416058394160585
```