Exercise Sheet 5 – Data Wrangling

INF161 Autumn Semester 2020

Exercise 1 (Down the rabbit hole on Prosecco) *For this exercise, you will evaluate a hypothesis of mine, which is that manufacturers of poor wines try to compensate for the poor taste of their wines by choosing long and fancy names. In particular, you will evaluate the hypothesis for Prosecco wines, since Prosecco is considered by many to be a cheaper version of Champagne, which could mean that Prosecco manufacturers have more to prove than those of other wines. You will be working with the `winemag-data-130k-v2.csv` dataset.*

Step 1: *Load the `winemag-data-130k-v2.csv` file as a `pandas` dataframe. Create a new dataframe composed only of the rows corresponding to reviews of Prosecco wines.*

Tip: Use `DataFrame.loc` to filter rows, and `DataFrame.reset_index` to re-index the rows.

You should end up with a dataframe that looks like this:

	index	Unnamed: 0	country	...	title	variety	winery
0	315	315	Italy	...	Bellussi NV Extra Dry (Prosecco di Valdobbiad...	Prosecco	Bellussi
1	319	319	Italy	...	Paladin 2007 Millesimato Brut Prosecco (Veneto)	Prosecco	Paladin
2	320	320	Italy	...	Perlage 2008 Col di Manza Extra Dry Millesimat...	Prosecco	Perlage
3	321	321	Italy	...	Sant Eurosia 2007 Brut (Prosecco di Valdobbia...	Prosecco	Sant Eurosia
4	322	322	Italy	...	Sant Eurosia 2007 Millesimato Dry (Prosecco d...	Prosecco	Sant Eurosia
..	...	...	...	...	...	...	...
231	129399	129399	Italy	...	Mionetto NV Brut (Prosecco del Veneto)	Prosecco	Mionetto
232	129400	129400	Italy	...	Moletto NV Frizzante Prosecco (Marca Trevigiana)	Prosecco	Moletto
233	129401	129401	Italy	...	Perlage NV Canah Brut (Prosecco di Valdobbiad...	Prosecco	Perlage
234	129404	129404	Italy	...	Valdo NV Extra Dry (Prosecco del Veneto)	Prosecco	Valdo
235	129929	129929	Italy	...	Col Vetoraz Spumanti NV Prosecco Superiore di...	Prosecco	Col Vetoraz Spumanti

Step 2: *From the Prosecco dataframe you created in the previous step, create two new dataframes that are composed of the rows with more than* 89 *points and less than* 85 *points, respectively. These two new dataframes should only have the columns "title," "price," and "points." Sort both dataframes by price in descending order.*

Tip: You just need two functions for this; `DataFrame.loc` *and* `DataFrame.sort_values`*.*

It should looks like this:

```
Good Prosecco:
                                                 title  price  points
143  Bisol 2007 Cartizze  (Prosecco Superiore di Ca...     48      90
216  Bisol NV Cartizze  (Prosecco Superiore di Cart...     41      90
235  Col Vetoraz Spumanti NV  Prosecco Superiore di...     38      91
123  Ruggeri & C. 2007 Giustino B. Extra Dry  (Pros...     36      91
154  Bortolomiol 2006 Cartizze  (Prosecco Superiore...     35      90
180  Bortolomiol 2006 Cartizze  (Prosecco Superiore...     35      90
144  Adami NV Cartizze Dry  (Prosecco Superiore di ...     32      90
162          Adami NV  Prosecco Superiore di Cartizze     32      90
103  Bortolomiol 2008 Cartizze Dry  (Prosecco Super...     30      90
104  Nino Franco 2007 Rive di San Floriano Brut  (P...     30      90
124    Ruggeri & C. NV  Prosecco Superiore di Cartizze     30      90
215     Bortolomiol NV  Prosecco Superiore di Cartizze     30      90
163        Bisol NV Crede  (Prosecco di Valdobbiadene)     25      90
145  Sorelle Bronca NV Particella 68 Extra Dry  (Pr...     24      90
179  Astoria NV Cartizze  (Prosecco Superiore di Ca...     21      90
178  Astoria 2006 Millesimato Extra Dry  (Prosecco ...     20      90
39   Sorelle Bronca NV Extra Dry Particella 68  (Pr...   <NA>      90

Bad Prosecco:
                                                 title  price  points
133       Varaschin NV  Prosecco Superiore di Cartizze     29      84
217  Viszlay Vineyards 2009 Estate Bottled Prosecco...     28      82
146  Perlage NV Col di Manza Extra Dry  (Prosecco d...     22      84
129  Maschio dei Cavalieri NV Maschio dei Cavalieri...     20      82
147     Varaschin NV Brut  (Prosecco di Valdobbiadene)     19      83
134  Bellenda NV San Fermo Brut  (Prosecco di Coneg...     19      84
148  Toffoli NV Brut  (Prosecco di Conegliano e Val...     18      84
213     Canella NV Extra Dry  (Prosecco di Conegliano)     18      84
95   Mionetto NV Certified Organic Extra Dry Prosec...     16      84
164  Carmina NV Brut  (Prosecco di Conegliano e Val...     16      82
214  Collalbrigo NV Extra Dry  (Prosecco di Conegli...     16      84
132  Col Saliz NV Extra Dry  (Prosecco di Valdobbia...     15      84
93   Cantina San Martino NV Pittaro Extra Dry  (Pro...     15      84
161  Carmina NV Brut  (Prosecco di Conegliano e Val...     15      83
32                          Tosti NV Prosecco (Italy)     15      83
117                Tiamo NV Extra Dry Prosecco (Veneto)    14      84
131  Terra Serena NV Extra Dry  (Prosecco di Conegl...     12      84
94   Lisabella NV Gran Resèe Prosecco (Colli Trevig...     12      84
149                          Zonin NV Prosecco (Italy)     12      84
49   Le Vigne di Alice 2007 Millesimato Doro Brut  ...   <NA>      81
130  Le Vigne di Alice NV Tajad Brut  (Prosecco del...   <NA>      81
211  Villa Granda NV Frizzante Prosecco (Colli Trev...   <NA>      84
212  Tenuta Santomè NV Extra Dry  (Prosecco del Ven...   <NA>      84
```

Step 3: *Add a new column to both dataframes, where each element is the number of characters in the title of the corresponding wine (this is where my theory comes in!). Finally, for both dataframes (coresponding to good and bad Prosecco, respectively), print the average number of characters for both dataframes.*

Tip: Use `DataFrame.map` *to compute title length for each wine.*

Spoiler: It seems my hypothesis was wrong, and that the data indicates that the opposite is true, i.e., better Prosecco, on average, have longer names :)

It should look like this:

```
Good Prosecco:
                                                title  price  points  title_length
143  Bisol 2007 Cartizze  (Prosecco Superiore di Ca...     48      90            53
216  Bisol NV Cartizze  (Prosecco Superiore di Cart...     41      90            51
235  Col Vetoraz Spumanti NV  Prosecco Superiore di...     38      91            55
123  Ruggeri & C. 2007 Giustino B. Extra Dry  (Pros...     36      91            68
154  Bortolomiol 2006 Cartizze  (Prosecco Superiore...     35      90            59
180  Bortolomiol 2006 Cartizze  (Prosecco Superiore...     35      90            59
144  Adami NV Cartizze Dry  (Prosecco Superiore di ...     32      90            55
162           Adami NV  Prosecco Superiore di Cartizze     32      90            40
103  Bortolomiol 2008 Cartizze Dry  (Prosecco Super...     30      90            63
104  Nino Franco 2007 Rive di San Floriano Brut  (P...     30      90            71
124    Ruggeri & C. NV  Prosecco Superiore di Cartizze     30      90            47
215     Bortolomiol NV  Prosecco Superiore di Cartizze     30      90            46
163        Bisol NV Crede  (Prosecco di Valdobbiadene)     25      90            43
145  Sorelle Bronca NV Particella 68 Extra Dry  (Pr...     24      90            70
179  Astoria NV Cartizze  (Prosecco Superiore di Ca...     21      90            53
178  Astoria 2006 Millesimato Extra Dry  (Prosecco ...     20      90            76
39   Sorelle Bronca NV Extra Dry Particella 68  (Pr...   <NA>      90            70
Good Prosecco mean title length 57.588235294117645

Bad Prosecco:
                                                title  price  points  title_length
133       Varaschin NV  Prosecco Superiore di Cartizze     29      84            44
217  Viszlay Vineyards 2009 Estate Bottled Prosecco...     28      82            69
146  Perlage NV Col di Manza Extra Dry  (Prosecco d...     22      84            62
129  Maschio dei Cavalieri NV Maschio dei Cavalieri...     20      82            75
147     Varaschin NV Brut  (Prosecco di Valdobbiadene)     19      83            46
134  Bellenda NV San Fermo Brut  (Prosecco di Coneg...     19      84            68
148  Toffoli NV Brut  (Prosecco di Conegliano e Val...     18      84            57
213     Canella NV Extra Dry  (Prosecco di Conegliano)     18      84            46
95   Mionetto NV Certified Organic Extra Dry Prosec...     16      84            57
164  Carmina NV Brut  (Prosecco di Conegliano e Val...     16      82            57
214  Collalbrigo NV Extra Dry  (Prosecco di Conegli...     16      84            50
132  Col Saliz NV Extra Dry  (Prosecco di Valdobbia...     15      84            51
93   Cantina San Martino NV Pittaro Extra Dry  (Pro...     15      84            63
161  Carmina NV Brut  (Prosecco di Conegliano e Val...     15      83            57
32                         Tosti NV Prosecco (Italy)     15      83            25
117                Tiamo NV Extra Dry Prosecco (Veneto)     14      84            36
131  Terra Serena NV Extra Dry  (Prosecco di Conegl...     12      84            67
94   Lisabella NV Gran Resèe Prosecco (Colli Trevig...     12      84            51
149                         Zonin NV Prosecco (Italy)     12      84            25
49   Le Vigne di Alice 2007 Millesimato Doro Brut  ...   <NA>      81            73
130  Le Vigne di Alice NV Tajad Brut  (Prosecco del...   <NA>      81            54
211  Villa Granda NV Frizzante Prosecco (Colli Trev...   <NA>      84            53
212  Tenuta Santomè NV Extra Dry  (Prosecco del Ven...   <NA>      84            50
Bad Prosecco mean title length 53.73913043478261
```

Exercise 2 (The ramen king) *For this exercise, you will figure out which country has the best ramen, as judged by reviews in that country.*

Step 1: *Load the* `ramen-ratings.csv` *file as a* `pandas` *dataframe. Next, group the rows of the dataframe by country (using* `DataFrame.groupby`*) and compute the average number of stars for each country. Also compute the* 10*-th and* 90*-th quantiles. Save these as columns ine a new dataframe indexed by country. Who has the best ramen? Since Norway is not part of the data set we have not eliminated the possibility that Norway is the ramen king, although I think we can agree on that the chance of that being true is quite slim.*

Tip: Stars is not given as a float; you need to convert it. Kaggle has a great tutorial on how to use `DataFrame.groupby` *at* `https://www.kaggle.com/residentmario/grouping-and-sorting`*.*

It should look like this:

	Mean	q10	q90
Country			
Australia	3.138636	2.025	4.000
Bangladesh	3.714286	3.250	4.000
Brazil	4.350000	4.000	4.800
Cambodia	4.200000	3.500	5.000
Canada	2.243902	0.250	3.500
China	3.421893	1.750	4.500
Colombia	3.291667	2.875	3.625
Dubai	3.583333	3.350	3.750
Estonia	3.500000	3.300	3.700
Fiji	3.875000	3.475	4.175
Finland	3.583333	3.500	3.700
Germany	3.638889	3.000	4.350
Ghana	3.500000	3.500	3.500
Holland	3.562500	3.500	3.675
Hong Kong	3.801825	2.750	5.000
Hungary	3.611111	2.950	4.150
India	3.395161	2.000	4.250
Indonesia	4.067460	3.250	5.000
Japan	3.981605	3.000	5.000
Malaysia	4.154194	3.250	5.000
Mexico	3.730000	3.000	4.000
Myanmar	3.946429	2.975	5.000
Nepal	3.553571	3.325	4.175
Netherlands	2.483333	0.500	3.500
Nigeria	1.500000	1.500	1.500
Pakistan	3.000000	2.400	3.600
Philippines	3.329787	1.900	4.500
Poland	3.625000	3.250	4.000
Sarawak	4.333333	4.000	4.800
Singapore	4.126147	3.200	5.000
South Korea	3.790554	2.750	5.000
Sweden	3.250000	3.050	3.450
Taiwan	3.665402	2.000	5.000
Thailand	3.384817	2.000	4.750
UK	2.997101	1.500	4.050
USA	3.457043	2.000	4.750
United States	3.750000	3.750	3.750
Vietnam	3.187963	2.000	4.000

Step 2: *Since we are anyway looking at this data set we may as well squeeze some more information out of it. Let us next answer the question of which style of ramen is most popular by country. In particular, you should, separately for each country, compute the fraction of reviews of each style of ramen.*

Start by creating a new clean dataframe by loading the `.csv` *file from disk again. Next, group the rows by both country and style. Then, compute the total number of rewviews for each country and style. Finally, normalize each entry by the total number of reviews from that country.*

Tip: You just need `DataFrame.groupby` *and division and only a few lines of code (around* 6*). The challenge is to use groups correctly.*

It should look like this (where the rightmost column is the fraction computed; note how the entries for each columnn sum to 1*):*

```
Country        Style
Australia      Cup      0.772727
               Pack     0.227273
Bangladesh     Pack     1.000000
Brazil         Cup      0.400000
               Pack     0.600000
                          ...
United States  Pack     1.000000
Vietnam        Bowl     0.185185
               Cup      0.074074
               Pack     0.722222
               Tray     0.018519
Name: Stars, Length: 87, dtype: float64
```

Exercise 3 (Restaurants; more items is more good) *For this exercise, you will analyze who orders more items at restaurants, men or women. You will be working with the* `order.csv` *and* `customers.csv` *datasets.*

Step 1: *Start by loading the* `order.csv` *and* `customers.csv` *files into separate dataframes. Now, you need to merge the two dataframes to compute the statistics we want. Do achieve this, note that there is a customer ID column in both of the dataframes. Use the* `DataFrames.join` *method to merge the two dataframes. Next, select the "item_count" and "gender" columns.*

Tip: Kaggle has a good tutorial on how to use join; see `https://www.kaggle.com/residentmario/renaming-and-combining`*.*

You should end up with a dataframe that looks like this:

```
        item_count gender
0                1   Male
1                1   <NA>
2                2   <NA>
3                1   Male
4                4   Male
...            ...    ...
135298           1   <NA>
135299           3   Male
135300           4   Male
135301           3   <NA>
135302           1   Male

[135502 rows x 2 columns]
```

Step 2: *Finally, use compute the average number of items separately for men and women. As it turns out, men and women order about the same number of items.*

Tip: The spelling of male and female is not consistent.

```
Step 2
        item_count
gender
female    2.381009
male      2.414087
```

Exercise 4 (Nostalgia?) *2020 has not been a great year. Hence, for the final assignment, we will go back to the glorious year 2000. In particular, you will be working with billboard data from the year 2000.*

Step 1: *Start by loading the `billboard.csv` file into a dataframe. Notice how there are separate columns for each week, which makes it hard to compute statistics over the entire period. Hence, the first step is to change the format of the dataframe such that*

- *there is a single column "week" that indicates which week a given row corresponds to and*
- *another column "rank" that contains the rank of each song and week.*

You will have one row for each song and week in the resulting dataframe, i.e., there will be 72 rows for each song (one for each week).

Tip: Use the `pandas.melt` function. The entire conversion can be done in a single function call; see the section "Column headers are values, not variable names" of `https://tidyr.tidyverse.org/articles/tidy-data.html` (in the R language, but the idea is the same).

It should look like this:

```
       year      artist.inverted                                track  time genre date.entered date.peaked        week  rank
0      2000       Destiny's Child             Independent Women Part I  3:38  Rock   2000-09-23  2000-11-18   x1st.week  78.0
1      2000               Santana                         Maria, Maria  4:18  Rock   2000-02-12  2000-04-08   x1st.week  15.0
2      2000         Savage Garden                   I Knew I Loved You  4:07  Rock   1999-10-23  2000-01-29   x1st.week  71.0
3      2000               Madonna                                Music  3:45  Rock   2000-08-12  2000-09-16   x1st.week  41.0
4      2000  Aguilera, Christina  Come On Over Baby (All I Want Is You)  3:38  Rock   2000-08-05  2000-10-14   x1st.week  57.0
...     ...                   ...                                  ...   ...   ...          ...         ...         ...   ...
24087  2000       Ghostface Killah                     Cherchez LaGhost  3:04   R&B   2000-08-05  2000-08-05  x76th.week   NaN
24088  2000           Smith, Will                        Freakin' It  3:58   Rap   2000-02-12  2000-02-12  x76th.week   NaN
24089  2000         Zombie Nation                        Kernkraft 400  3:30  Rock   2000-09-02  2000-09-02  x76th.week   NaN
24090  2000        Eastsidaz, The                             Got Beef  3:58   Rap   2000-07-01  2000-07-01  x76th.week   NaN
24091  2000                Fragma                       Toca's Miracle  3:22   R&B   2000-10-28  2000-10-28  x76th.week   NaN

[24092 rows x 9 columns]
```

Step 2: *The weeks are given as strings, which could be a problem. Convert these to integers.*

*Tip: Use the **map** function (as you did with the Prosecco problem) and do something like what is suggested at this link to remove non-digit characters `https://stackoverflow.com/questions/17336943/removing-non-numeric\-characters-from-a-string`.*

```
       year      artist.inverted                                track  time genre date.entered date.peaked  week  rank
0      2000       Destiny's Child             Independent Women Part I  3:38  Rock   2000-09-23  2000-11-18     1  78.0
1      2000               Santana                         Maria, Maria  4:18  Rock   2000-02-12  2000-04-08     1  15.0
2      2000         Savage Garden                   I Knew I Loved You  4:07  Rock   1999-10-23  2000-01-29     1  71.0
3      2000               Madonna                                Music  3:45  Rock   2000-08-12  2000-09-16     1  41.0
4      2000  Aguilera, Christina  Come On Over Baby (All I Want Is You)  3:38  Rock   2000-08-05  2000-10-14     1  57.0
...     ...                   ...                                  ...   ...   ...          ...         ...   ...   ...
24087  2000       Ghostface Killah                     Cherchez LaGhost  3:04   R&B   2000-08-05  2000-08-05    76   NaN
24088  2000           Smith, Will                        Freakin' It  3:58   Rap   2000-02-12  2000-02-12    76   NaN
24089  2000         Zombie Nation                        Kernkraft 400  3:30  Rock   2000-09-02  2000-09-02    76   NaN
24090  2000        Eastsidaz, The                             Got Beef  3:58   Rap   2000-07-01  2000-07-01    76   NaN
24091  2000                Fragma                       Toca's Miracle  3:22   R&B   2000-10-28  2000-10-28    76   NaN

[24092 rows x 9 columns]
```

Step 3: *Finally, group the rows by both "artist.inverted" and "track", and compute the average rank of the resulting group. Select the "rank" column, sort in ascending order, and print the first* 10 *rows, i.e., the* 10 *songs with lowest (best) average rank.*

```
artist.inverted                 track
Santana                         Maria, Maria                 10.500000
Madonna                         Music                        13.458333
N'Sync                          Bye Bye Bye                  14.260870
Elliott, Missy "Misdemeanor"    Hot Boyz                     14.333333
Destiny's Child                 Independent Women Part I     14.821429
Iglesias, Enrique               Be With You                  15.850000
Aaliyah                         Try Again                    16.656250
Savage Garden                   I Knew I Loved You           17.363636
Houston, Whitney                My Love Is Your Love         17.857143
Pink                            There U Go                   18.625000
Name: rank, dtype: float64
```

Now, just kick back with the music video of N'Sync's Bye Bye Bye; you have deserved it `https://www.youtube.com/watch?v=Eo-KmOd3i7s`