```
{
 "cells": [
  {
   "cell_type": "markdown",
   "metadata": {
    "nbgrader": {
     "grade": false,
     "locked": true,
     "solution": false
    }
   },
   "source": [
    "d# Homework 2 (Due: 2021.04.01 11:59 PM)"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {
    "pycharm": {
     "name": "#%% md\n"
    }
   },
   "source": [
    "Name: □□□"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {},
   "source": [
    "Student ID: 2017312172"
   ]
  },
  {
   "cell_type": "markdown",
   "metadata": {
    "nbgrader": {
     "grade": false,
     "locked": true,
     "solution": false
    }
   },
   "source": [
    "For this homework, you are provided with an
```

```
44 input file.\n",
45     "Submit the code that you used for analysis as an
    ipython notebook and an exported PDF file."
46    ]
47   },
48   {
49    "cell_type": "markdown",
50    "metadata": {
51     "nbgrader": {
52      "grade": false,
53      "locked": true,
54      "solution": false
55     }
56    },
57    "source": [
58     "# Q1 Analyzing the Election Data (3 x 8 = 24 pts
   )"
59    ]
60   },
61   {
62    "cell_type": "markdown",
63    "metadata": {
64     "nbgrader": {
65      "grade": false,
66      "locked": true,
67      "solution": false
68     }
69    },
70    "source": [
71     "Run the following cell to set the filename."
72    ]
73   },
74   {
75    "cell_type": "code",
76    "execution_count": 92,
77    "metadata": {
78     "collapsed": true,
79     "nbgrader": {
80      "grade": false,
81      "locked": true,
82      "solution": false
83     }
84    },
85    "outputs": [],
```

```
 86    "source": [
 87     "filename = 'bush-gore-results-fl_demo.csv'"
 88    ]
 89   },
 90   {
 91    "cell_type": "markdown",
 92    "metadata": {
 93     "collapsed": true,
 94     "nbgrader": {
 95      "grade": false,
 96      "locked": true,
 97      "solution": false
 98     }
 99    },
100    "source": [
101     "(a) Determine how many counties Bush won. You
    can assume that each row corresponds to a unique
    county.\n",
102     "    Demo file: 'bush-gore-results-fl_demo.csv
    ' "
103    ]
104   },
105   {
106    "cell_type": "code",
107    "execution_count": 93,
108    "metadata": {
109     "collapsed": true
110    },
111    "outputs": [
112     {
113      "name": "stdout",
114      "output_type": "stream",
115      "text": [
116       "51\n"
117      ]
118     }
119    ],
120    "source": [
121     "import pandas as pd\n",
122     "import numpy as np\n",
123     "df = pd.read_csv(filename)\n",
124     "#All possible candidates are 'bush', 'gore', '
    brow', 'nade', 'harr', 'hage', 'buch', 'mcre', 'phil
    ', 'moor'. You can find a person from the list.\n",
```

```
125     "df_resize = df.loc[:, ['bush', 'gore', 'brow
    ', 'nade', 'harr', 'hage', 'buch', 'mcre', 'phil', '
    moor']]\n",
126     "\n",
127     "bush_won = 0\n",
128     "for length in range(df_resize.shape[0]):\n",
129     "    if df_resize.loc[length,:].max() ==
    df_resize.loc[length,'bush']:\n",
130     "        bush_won += 1\n",
131     "print(bush_won)"
132    ]
133   },
134   {
135    "cell_type": "markdown",
136    "metadata": {
137     "nbgrader": {
138      "grade": false,
139      "locked": true,
140      "solution": false
141     }
142    },
143    "source": [
144     "(b) Determine who won the largest county. The
    column named \"npop\" records the size of each
    county.\n",
145     "    Demo file: 'bush-gore-results-fl_demo.csv
    ' "
146    ]
147   },
148   {
149    "cell_type": "code",
150    "execution_count": 94,
151    "metadata": {
152     "collapsed": true
153    },
154    "outputs": [
155     {
156      "name": "stdout",
157      "output_type": "stream",
158      "text": [
159       "Gore\n"
160      ]
161     }
162    ],
```

```
163    "source": [
164     "import pandas as pd\n",
165     "import numpy as np\n",
166     "#default\n",
167     "filename = 'bush-gore-results-fl_demo.csv'\n",
168     "df = pd.read_csv(filename)\n",
169     "\n",
170     "max_index = df['npop'].argmax()\n",
171     "\n",
172     "\n",
173     "if (df.loc[max_index,'bush']<df.loc[max_index
    , 'gore']):\n",
174     "    print(\"Gore\")\n",
175     "else:\n",
176     "    print(\"Bush\")\n"
177    ]
178   },
179   {
180    "cell_type": "markdown",
181    "metadata": {
182     "nbgrader": {
183      "grade": false,
184      "locked": true,
185      "solution": false
186     }
187    },
188    "source": [
189     "(c) Determine the average number of votes per
    county that Buchanan obtained. You can assume that
    each row corresponds to a unique county.\n",
190     "    Demo file: 'bush-gore-results-fl_demo.csv
    ' "
191    ]
192   },
193   {
194    "cell_type": "code",
195    "execution_count": 95,
196    "metadata": {
197     "collapsed": true
198    },
199    "outputs": [
200     {
201      "name": "stdout",
202      "output_type": "stream",
```

```
203       "text": [
204        "360.67164179104475\n"
205       ]
206      }
207     ],
208     "source": [
209      "import pandas as pd\n",
210      "import numpy as np\n",
211      "#default\n",
212      "filename = 'bush-gore-results-fl_demo.csv'\n",
213      "df = pd.read_csv(filename)\n",
214      "\n",
215      "df_resize = df.loc[:, 'buch']\n",
216      "#print(df_resize)\n",
217      "print(df_resize.mean())"
218     ]
219    },
220    {
221     "cell_type": "markdown",
222     "metadata": {
223      "nbgrader": {
224       "grade": false,
225       "locked": true,
226       "solution": false
227      }
228     },
229     "source": [
230      "(d) Determine, for the number of votes Buchanan
      obtained in Palm Beach, how many standard
     deviations it is away from Buchanan's overall mean,
     in absolute value. The row with coounty number 50
      (\"co\"=50) records the results for Palm Beach
     County. (You can assumed that such a row will exist
     in the test case.)\n",
231      "    Demo file: 'bush-gore-results-fl_demo.csv
     ' "
232     ]
233    },
234    {
235     "cell_type": "code",
236     "execution_count": 96,
237     "metadata": {
238      "collapsed": true
239     },
```

```
240    "outputs": [
241     {
242      "name": "stdout",
243      "output_type": "stream",
244      "text": [
245       "6.993018491215867\n"
246      ]
247     }
248    ],
249    "source": [
250     "#It asks to compute (the absolute difference / std).\n",
251     "\n",
252     "import pandas as pd\n",
253     "import numpy as np\n",
254     "#default\n",
255     "filename = 'bush-gore-results-fl_demo.csv'\n",
256     "df = pd.read_csv(filename)\n",
257     "co = 50\n",
258     "\n",
259     "df_resize = df.loc[:, 'buch']\n",
260     "#print(df_resize)\n",
261     "buch_mean = df_resize.mean()\n",
262     "buch_std = df_resize.std()\n",
263     "print((df_resize.iloc[co-1]-buch_mean)/buch_std)\n",
264     "#print(np.square(df_resize.iloc[co-1]-buch_mean))\n",
265     "\n"
266    ]
267   },
268   {
269    "cell_type": "markdown",
270    "metadata": {},
271    "source": [
272     "(e) Now calculate the above statistic (same as in part f) for all the counties and report them in decreasing order.\n",
273     "    Demo file: 'bush-gore-results-fl_demo.csv ' \n",
274     "    Example output: \n",
275     "    county_50 6.993018  \n",
276     "    county_52 ...\n",
277     "    ...\n",
```

```
278       "    ..."
279      ]
280     },
281     {
282      "cell_type": "code",
283      "execution_count": 97,
284      "metadata": {
285       "collapsed": true
286      },
287      "outputs": [
288       {
289        "name": "stdout",
290        "output_type": "stream",
291        "text": [
292         "county_ 50 \t 6.993018491215867\n",
293         "county_ 52 \t 1.6721223983806142\n",
294         "county_ 28 \t 1.3031713743995565\n",
295         "county_ 6 \t 1.1720381791291807\n",
296         "county_ 15 \t 0.8697650510483141\n",
297         "county_ 5 \t 0.6875121355877918\n",
298         "county_ 51 \t 0.6875121355877918\n",
299         "county_ 41 \t 0.6719539598777472\n",
300         "county_ 43 \t 0.6652861702877281\n",
301         "county_ 53 \t 0.6030534674475496\n",
302         "county_ 16 \t 0.5363755715473585\n",
303         "county_ 64 \t 0.5230399923673202\n",
304         "county_ 48 \t 0.41191016586700174\n",
305         "county_ 55 \t 0.11185963431614161\n",
306         "county_ 35 \t 0.09852405513610338\n",
307         "county_ 56 \t 0.09852405513610338\n",
308         "county_ 34 \t 0.06296251065600145\n",
309         "county_ 36 \t 0.04740433494595685\n",
310         "county_ 40 \t 0.02295577311588676\n",
311         "county_ 9 \t 0.02073317658588039\n",
312         "county_ 46 \t 0.01406538699586127 5\n",
313         "county_ 1 \t 0.00517500087583579\n",
314         "county_ 3 \t -0.028163947074259777\n",
315         "county_ 26 \t -0.04149952625429801\n",
316         "county_ 58 \t -0.07039328114438083\n",
317         "county_ 57 \t -0.14818415969460383\n",
318         "county_ 10 \t -0.16596493193465478\n",
319         "county_ 8 \t -0.17485531805468027\n",
320         "county_ 54 \t -0.2504236000748969\n",
321         "county_ 49 \t -0.257091389664916\n",
```

```
322       "county_ 27 \t -0.2970981272050307\n",
323       "county_ 59 \t -0.30376591679504983\n",
324       "county_ 11 \t -0.30821110985506256\n",
325       "county_ 66 \t -0.3126563029150753\n",
326       "county_ 60 \t -0.325991882095113 54\n",
327       "county_ 42 \t -0.33043707515512627\n",
328       "county_ 61 \t -0.33932746127515173\n",
329       "county_ 30 \t -0.34599525086517086\n",
330       "county_ 31 \t -0.35266304045519\n",
331       "county_ 7 \t -0.3793341988152664\n",
332       "county_ 45 \t -0.3793341988152664\n",
333       "county_ 12 \t -0.3815567953452728\n",
334       "county_ 67 \t -0.38377939187527915\n",
335       "county_ 17 \t -0.394892374525311\n",
336       "county_ 29 \t -0.41045055023535565\n",
337       "county_ 2 \t -0.4171183398253747\n",
338       "county_ 22 \t -0.4215635328853875\n",
339       "county_ 37 \t -0.43045391900541297\n",
340       "county_ 4 \t -0.4348991120654257\n",
341       "county_ 44 \t -0.4749058496055404\n",
342       "county_ 65 \t -0.4771284461355468\n",
343       "county_ 47 \t -0.4837962357255659\n",
344       "county_ 38 \t -0.49268662184559137\n",
345       "county_ 19 \t -0.49490921837559776\n",
346       "county_ 63 \t -0.4971318149056041\n",
347       "county_ 13 \t -0.4993544114356105\n",
348       "county_ 18 \t -0.5060222010256296\n",
349       "county_ 24 \t -0.5126899906156487\n",
350       "county_ 14 \t -0.514912587145655\n",
351       "county_ 20 \t -0.514912587145655\n",
352       "county_ 32 \t -0.514912587145655\n",
353       "county_ 39 \t -0.514912587145655\n",
354       "county_ 62 \t -0.5193577802056678\n",
355       "county_ 23 \t -0.5282481663256933\n",
356       "county_ 25 \t -0.5304707628556997\n",
357       "county_ 33 \t -0.5571419212157761\n",
358       "county_ 21 \t -0.5593645177457826\n"
359      ]
360     }
361    ],
362    "source": [
363     "import pandas as pd\n",
364     "import numpy as np\n",
365     "#default\n",
```

```
366     "filename = 'bush-gore-results-fl_demo.csv'\n",
367     "df = pd.read_csv(filename)\n",
368     "\n",
369     "\n",
370     "dict_var = {}\n",
371     "df_resize = df.loc[:, 'buch']\n",
372     "\n",
373     "#print(df_resize)\n",
374     "buch_mean = df_resize.mean()\n",
375     "buch_std = df_resize.std()\n",
376     "for i in range(df.shape[0]):\n",
377     "    buch_value = df_resize.iloc[i]\n",
378     "    dict_var[i] = ((buch_value-buch_mean)/
    buch_std)\n",
379     "sorted_var = sorted(dict_var.items(), reverse
     = True, key = lambda x: x[1])\n",
380     "for index, values in enumerate(sorted_var):\n",
381     "    print(\"county_\", values[0]+1, \"\\t\",
    values[1])"
382    ]
383   },
384   {
385    "cell_type": "markdown",
386    "metadata": {
387     "nbgrader": {
388      "grade": false,
389      "locked": true,
390      "solution": false
391     }
392    },
393    "source": [
394     "(f) Assuming that the votes were distributed
    across the white, black, and hispanic population
    uniformly, determine which candidate obtained the
    largest number of votes for each subpopulation.\n",
395     "    Demo file: 'bush-gore-results-fl_demo.csv
    ' \n",
396     "    Example output:\n",
397     "    white: bush\n",
398     "    black: ..\n",
399     "    hispanic: .."
400    ]
401   },
402   {
```

```
403    "cell_type": "code",
404    "execution_count": 98,
405    "metadata": {
406     "collapsed": true
407    },
408    "outputs": [
409     {
410      "name": "stdout",
411      "output_type": "stream",
412      "text": [
413       "whit   bush\n",
414       "blac   gore\n",
415       "hisp   gore\n"
416      ]
417     }
418    ],
419    "source": [
420     "import pandas as pd\n",
421     "import numpy as np\n",
422     "#default\n",
423     "filename = 'bush-gore-results-fl_demo.csv'\n",
424     "df = pd.read_csv(filename)\n",
425     "\n",
426     "df_feature = df.loc[:, ['whit','blac','hisp']]\
    n",
427     "candidate = ['bush', 'gore']\n",
428     "df_cand = df.loc[:, candidate]\n",
429     "cand_feat = [[0,0,0], [0,0,0]]\n",
430     "\n",
431     "feature = ['whit', 'blac', 'hisp']\n",
432     "for i in range(df_cand.shape[0]):\n",
433     "    for j in range(3):\n",
434     "        cand_feat[0][j] += df_feature.iloc[i, j
    ] / 100 * df_cand.iloc[i, 0]\n",
435     "        cand_feat[1][j] += df_feature.iloc[i, j
    ] / 100 * df_cand.iloc[i, 1]\n",
436     "\n",
437     "cand_feat = np.asarray(cand_feat)\n",
438     "for i in range(3):\n",
439     "    print(feature[i], \" \", candidate[
    cand_feat.argmax(axis=0)[i]])"
440    ]
441   },
442   {
```

```
443    "cell_type": "markdown",
444    "metadata": {
445     "nbgrader": {
446      "grade": false,
447      "locked": true,
448      "solution": false
449     }
450    },
451    "source": [
452     "(g) Calculate the correlation between the
    difference in votes between Bush and Gore, and the
    votes obtained by Nader.\n",
453     "(FYI: Pearson's correlation coefficient) https
    ://en.wikipedia.org/wiki/
    Pearson_correlation_coefficient\n",
454     "    Demo file: 'bush-gore-results-fl_demo.csv
    ' "
455    ]
456   },
457   {
458    "cell_type": "code",
459    "execution_count": 99,
460    "metadata": {
461     "collapsed": true
462    },
463    "outputs": [
464     {
465      "name": "stdout",
466      "output_type": "stream",
467      "text": [
468       "-0.42364341310857706\n"
469      ]
470     }
471    ],
472    "source": [
473     "#(g) Calculate the correlation between the
    difference in votes between Bush and Gore, and the
    votes obtained by Nader.\n",
474     "#\n",
475     "import pandas as pd\n",
476     "import numpy as np\n",
477     "#default\n",
478     "filename = 'bush-gore-results-fl_demo.csv'\n",
479     "df = pd.read_csv(filename)\n",
```

```
480     "data = np.zeros(shape=(df.shape[0],2))\n",
481     "df_corr = pd.DataFrame(data, columns = ['diff
    ', 'nade'])\n",
482     "df_corr['diff'] = df.loc[:, 'bush']-df.loc[:, '
    gore']\n",
483     "df_corr['nade'] = df.loc[:, 'nade']\n",
484     "\n",
485     "print(df_corr.corr().iloc[0,1])"
486    ]
487   },
488   {
489    "cell_type": "markdown",
490    "metadata": {
491     "nbgrader": {
492      "grade": false,
493      "locked": true,
494      "solution": false
495     },
496     "pycharm": {
497      "name": "#%% md\n"
498     }
499    },
500    "source": [
501     "(h) Find the distance between the county that
    Bush won by the largest margin and the county that
    Gore won by the largest margin. (Just use basic
    Euclidean distance between the latitude (lat) and
    longitude (lon) values for the counties, no need to
    compute spherical distance.) (FYI: Eucliden distane
    is decribed in https://en.wikipedia.org/wiki/
    Euclidean_distance#:~:text=In%20mathematics%2C%20the
    %20Euclidean%20distance,metric%20as%20the%
    20Pythagorean%20metric.)\n",
502     "    Demo file: 'bush-gore-results-fl_demo.csv
    ' "
503    ]
504   },
505   {
506    "cell_type": "code",
507    "execution_count": 100,
508    "metadata": {
509     "collapsed": true
510    },
511    "outputs": [
```

```
512     {
513      "name": "stdout",
514      "output_type": "stream",
515      "text": [
516       "4.341658669218476\n"
517      ]
518     }
519    ],
520    "source": [
521     "#(h) Find the distance between the county that
    Bush won by the largest margin and the county that
    Gore won by the largest margin. (Just use basic
    Euclidean distance between the latitude (lat) and
    longitude (lon) values for the counties, no need to
    compute spherical distance.) (FYI: Eucliden distane
    is decribed in https://en.wikipedia.org/wiki/
    Euclidean_distance#:~:text=In%20mathematics%2C%20the
    %20Euclidean%20distance,metric%20as%20the%
    20Pythagorean%20metric.)\n",
522     "\n",
523     "import pandas as pd\n",
524     "import numpy as np\n",
525     "#default\n",
526     "filename = 'bush-gore-results-fl_demo.csv'\n",
527     "df = pd.read_csv(filename)\n",
528     "df_resize = df.loc[:, ['bush', 'gore', 'brow
    ', 'nade', 'harr', 'hage', 'buch', 'mcre', 'phil', '
    moor']]\n",
529     "df_resize['bush-2']=df_resize.loc[:, 'bush']-
    df_resize.apply(lambda row : row.nlargest(2).values
    [-1], axis=1)\n",
530     "df_resize['gore-2']=df_resize.loc[:, 'gore']-
    df_resize.apply(lambda row : row.nlargest(2).values
    [-1], axis=1)\n",
531     "\n",
532     "bushidx = df_resize['bush-2'].idxmax()\n",
533     "goreidx = df_resize['gore-2'].idxmax()\n",
534     "\n",
535     "latdiff = (np.square(df.loc[bushidx, 'lat']-df.
    loc[goreidx,'lat']))\n",
536     "londiff = (np.square(df.loc[bushidx, 'lon']-df.
    loc[goreidx,'lon']))\n",
537     "print(np.sqrt(latdiff+londiff))\n"
538    ]
```

```
   }
  ],
  "metadata": {
   "kernelspec": {
    "name": "python3",
    "language": "python",
    "display_name": "Python 3"
   },
   "language_info": {
    "codemirror_mode": {
     "name": "ipython",
     "version": 3
    },
    "file_extension": ".py",
    "mimetype": "text/x-python",
    "name": "python",
    "nbconvert_exporter": "python",
    "pygments_lexer": "ipython3",
    "version": "3.7.4"
   }
  },
  "nbformat": 4,
  "nbformat_minor": 2
 }
```