Chris Iyer, 1/27/2023

SUMMARY MEMO:

After testing each race prediction method in this document on the Georgia voter list file (which contains race information), two methods seem the most robust.

If county identification is available for each entry (and/or tract, block group, or block identification), then WRU is the most straightforward and robust choice. WRU is a Bayesian-Improved Surname Geocoding method that is flexible to different levels of geocoding specificity and is evolving to better incorporate other demographic information as well.

If geographic data is not available, I received the best results from a custom merging of the first- and last-name predictions of the Predictrace library (because WRU does not incorporate first-name information). Specifically, the predict_race() function generates a probability that a given entry belongs to each racial group; I generated these probabilities based on first and last names separately, averaged the two, and selected the most likely race for each entry. If only surnames are available, predictrace is comparable to WRU surname-only prediction as well.

However, it is critical to make key decisions about desired false negative and false positive rates. For a given analysis, if we want to maximize our chance of getting *all* the AAPI voters in a specific place, we want a method with high sensitivity (coverage). In this case, consider counting entries as AAPI if at least one of several prediction methods predicts them as AAPI. If instead, we want to be most sure that everyone we predict to be AAPI is in fact AAPI, then we want a method with a high positive predictive value. In this case, consider counting entries as AAPI only if multiple methods all predict AAPI. These decisions will change project-to-project.

I have provided in this folder a file (GAlist_merged_nonan.csv) of 9M+ Georgia voters' names, races, and other information on which I tested these methods. I have also included a file (GA_predictions.csv) with the complete predictions of each method for further combinations and testing (column names correspond to parentheticals below).

Lastly, my code files can be found in that folder as well as this github repo.

Goal: predict race for entries in Texas voter list file. To do so, we want to test a variety of race prediction methods on Georgia voters (who we have race information on to assess accuracy) and select the best.

Links to code files:

- Github repo
- Main google folder
- R notebook (predictrace & wru libraries)
- Python notebook (ethnicolr library)

Results:

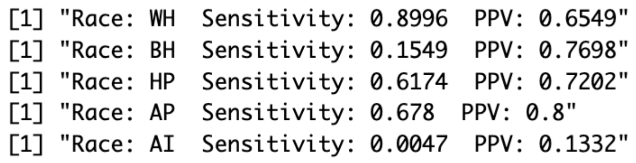
Tl;dr WRU with geocoding reaches pretty solid and balanced performance for AAPIs as well as other racial groups. Predictrace full-name averaging comes in second. These options, as well as numerous different combination schemes of different predictions, can be titrated to minimize false positives vs. false negatives. But, as a first pass, I recommend using WRU with geocoding if you have access to county-, tract-, or block-level information, and using predictrace with an averaging scheme like the one I use in the absence of geographic data.

I report sensitivity (% of people of a category that correctly get the label / coverage) labeled in that category) and positive predictive value (% of people of a label who are indeed of that category)

1 - sensitivity = false negative rate

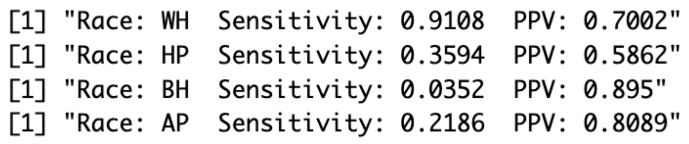
1 - PPV = false positive rate

- Predictrace (R)
    - Surname only (pr_surname)



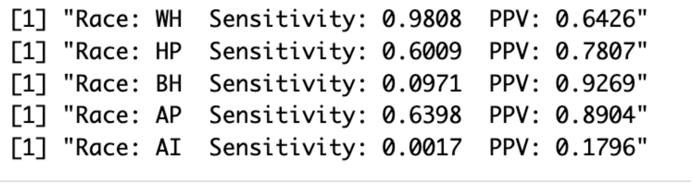
```
[1] "Race: WH  Sensitivity: 0.8996  PPV: 0.6549"
[1] "Race: BH  Sensitivity: 0.1549  PPV: 0.7698"
[1] "Race: HP  Sensitivity: 0.6174  PPV: 0.7202"
[1] "Race: AP  Sensitivity: 0.678  PPV: 0.8"
[1] "Race: AI  Sensitivity: 0.0047  PPV: 0.1332"
```

    - First name only (uses this dataset) (pr_firstname)



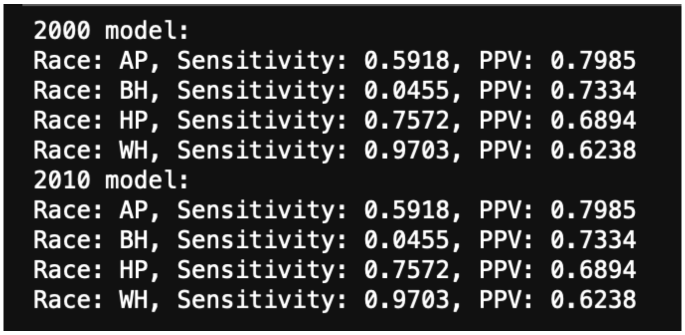
```
[1] "Race: WH  Sensitivity: 0.9108  PPV: 0.7002"
[1] "Race: HP  Sensitivity: 0.3594  PPV: 0.5862"
[1] "Race: BH  Sensitivity: 0.0352  PPV: 0.895"
[1] "Race: AP  Sensitivity: 0.2186  PPV: 0.8089"
```

    - Average probabilities from first and last name predictions; select max (pr_avg)



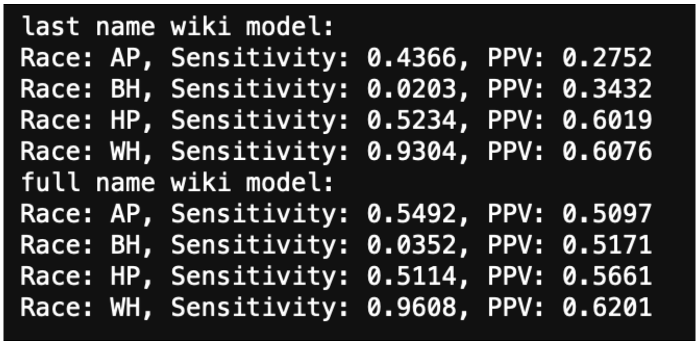
```
[1] "Race: WH  Sensitivity: 0.9808  PPV: 0.6426"
[1] "Race: HP  Sensitivity: 0.6009  PPV: 0.7807"
[1] "Race: BH  Sensitivity: 0.0971  PPV: 0.9269"
[1] "Race: AP  Sensitivity: 0.6398  PPV: 0.8904"
[1] "Race: AI  Sensitivity: 0.0017  PPV: 0.1796"
```

- Ethnicolr (Python)
    - Census models (surname only) (et_census_2000 and et_census_2010)



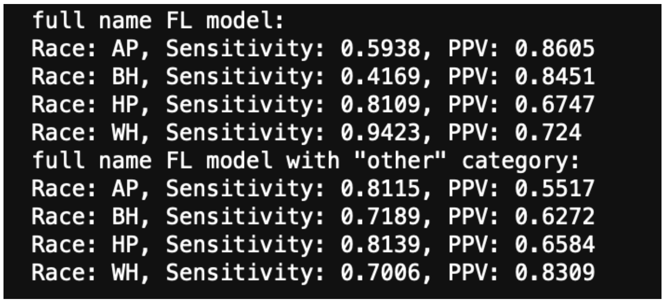
```
2000 model:
Race: AP, Sensitivity: 0.5918, PPV: 0.7985
Race: BH, Sensitivity: 0.0455, PPV: 0.7334
Race: HP, Sensitivity: 0.7572, PPV: 0.6894
Race: WH, Sensitivity: 0.9703, PPV: 0.6238
2010 model:
Race: AP, Sensitivity: 0.5918, PPV: 0.7985
Race: BH, Sensitivity: 0.0455, PPV: 0.7334
Race: HP, Sensitivity: 0.7572, PPV: 0.6894
Race: WH, Sensitivity: 0.9703, PPV: 0.6238
```

    - Wiki models (surname and full name) (et_wiki_surname & et_wiki_full)



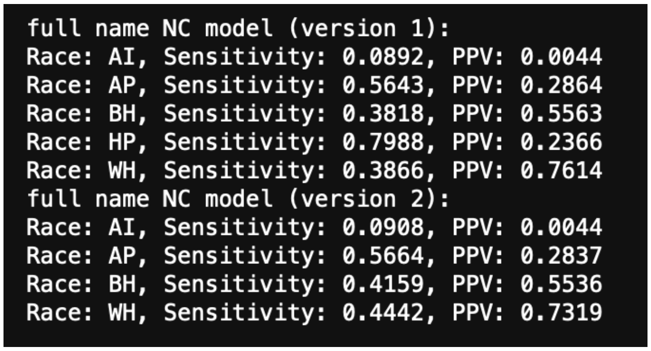
```
last name wiki model:
Race: AP, Sensitivity: 0.4366, PPV: 0.2752
Race: BH, Sensitivity: 0.0203, PPV: 0.3432
Race: HP, Sensitivity: 0.5234, PPV: 0.6019
Race: WH, Sensitivity: 0.9304, PPV: 0.6076
full name wiki model:
Race: AP, Sensitivity: 0.5492, PPV: 0.5097
Race: BH, Sensitivity: 0.0352, PPV: 0.5171
Race: HP, Sensitivity: 0.5114, PPV: 0.5661
Race: WH, Sensitivity: 0.9608, PPV: 0.6201
```

    - FL registered voter models (full name) (et_fl & et_fl_other)



```
full name FL model:
Race: AP, Sensitivity: 0.5938, PPV: 0.8605
Race: BH, Sensitivity: 0.4169, PPV: 0.8451
Race: HP, Sensitivity: 0.8109, PPV: 0.6747
Race: WH, Sensitivity: 0.9423, PPV: 0.724
full name FL model with "other" category:
Race: AP, Sensitivity: 0.8115, PPV: 0.5517
Race: BH, Sensitivity: 0.7189, PPV: 0.6272
Race: HP, Sensitivity: 0.8139, PPV: 0.6584
Race: WH, Sensitivity: 0.7006, PPV: 0.8309
```

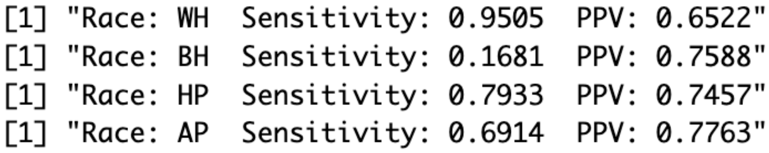
    - NC registered voter model (full name; gives Hispanic Y/N + Race, so I used 2 mappings of preidctions onto the voter list race categories) (et_nc_1 & et_nc_2)



```
full name NC model (version 1):
Race: AI, Sensitivity: 0.0892, PPV: 0.0044
Race: AP, Sensitivity: 0.5643, PPV: 0.2864
Race: BH, Sensitivity: 0.3818, PPV: 0.5563
Race: HP, Sensitivity: 0.7988, PPV: 0.2366
Race: WH, Sensitivity: 0.3866, PPV: 0.7614
full name NC model (version 2):
Race: AI, Sensitivity: 0.0908, PPV: 0.0044
Race: AP, Sensitivity: 0.5664, PPV: 0.2837
Race: BH, Sensitivity: 0.4159, PPV: 0.5536
Race: WH, Sensitivity: 0.4442, PPV: 0.7319
```

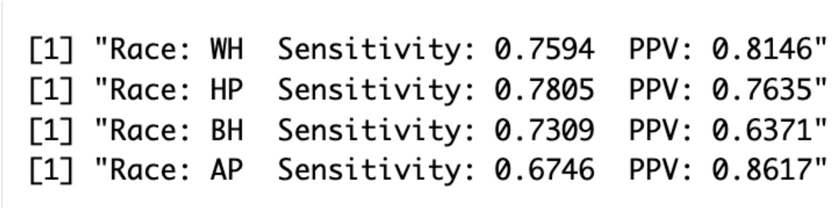
- Wru (R)

  - Surname only (no geocoding) (wru_surname)



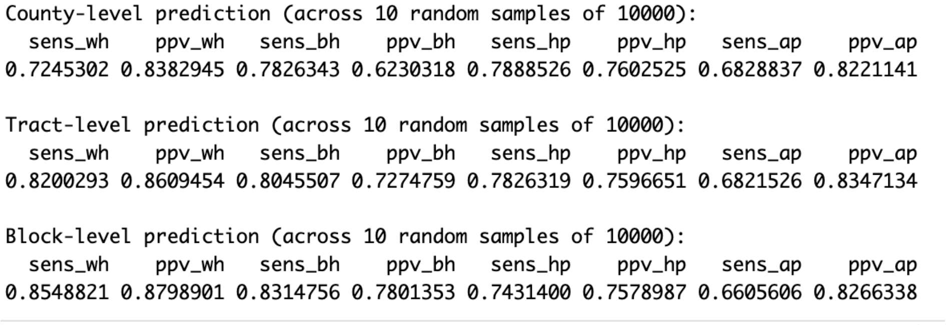
```
[1] "Race: WH  Sensitivity: 0.9505  PPV: 0.6522"
[1] "Race: BH  Sensitivity: 0.1681  PPV: 0.7588"
[1] "Race: HP  Sensitivity: 0.7933  PPV: 0.7457"
[1] "Race: AP  Sensitivity: 0.6914  PPV: 0.7763"
```

  - County-level census geocoding (wru_geo)



```
[1] "Race: WH  Sensitivity: 0.7594  PPV: 0.8146"
[1] "Race: HP  Sensitivity: 0.7805  PPV: 0.7635"
[1] "Race: BH  Sensitivity: 0.7309  PPV: 0.6371"
[1] "Race: AP  Sensitivity: 0.6746  PPV: 0.8617"
```

  - Tract- and block-level census geocoding (using library censusxy to get tract/block ID, and then conditioning WRU predictions on it)
    - Note: censusxy geocoding takes so long that instead of running on the whole dataset, I ran on 10 random samples of 10,000 entries. See results below:



```
County-level prediction (across 10 random samples of 10000):
  sens_wh    ppv_wh   sens_bh    ppv_bh   sens_hp    ppv_hp   sens_ap    ppv_ap
0.7245302 0.8382945 0.7826343 0.6230318 0.7888526 0.7602525 0.6828837 0.8221141

Tract-level prediction (across 10 random samples of 10000):
  sens_wh    ppv_wh   sens_bh    ppv_bh   sens_hp    ppv_hp   sens_ap    ppv_ap
0.8200293 0.8609454 0.8045507 0.7274759 0.7826319 0.7596651 0.6821526 0.8347134

Block-level prediction (across 10 random samples of 10000):
  sens_wh    ppv_wh   sens_bh    ppv_bh   sens_hp    ppv_hp   sens_ap    ppv_ap
0.8548821 0.8798901 0.8314756 0.7801353 0.7431400 0.7578987 0.6605606 0.8266338
```

- Mixing and matching:
  - Combination of (1) and (2): count as AP if wru_geo OR pr_avg == ‘AP’
    - AP: sens = 71%, PPV = 83%
  - Combination of (1) and (2): wru_geo AND pr_avg
    - AP: sens = 60%, PPV = 93%
  - Most balanced: wru_geo OR pr_avg OR (et_fl AND et_fl_other)
  - Max PPV: something like wru_geo AND pr_avg AND et_fl
    - Sens = 50% PPV = 95%
  - Max sens: something like wru_geo OR pr_avg OR et_fl_other
    - Sens = 98%, PPV = 60%

Conclusion: ultimately, there are a view important considerations:

1) Balance of sensitivity and PPV (false negative/false positive). For example, maximizing combinations with OR will get a very high sensitivity, and combinations with AND will get a very high PPV.
2) Simplicity of the model: it’s much nicer to tell someone “I used WRU, it’s a BISG algorithm” than “I used a strange bespoke combination of 4 different models”

For that reason, we opt for WRU with geocoding as the best option analyzed here.

Untested methods:

- List of Asian last names from this paper (contact lauderdale@health.bsd.uchicago.edu to get these lists)
    - This would *only* match Asian names
    - Likely high PPV (low false positive rate) and low sensitivity (high false negative rate)
    - Similar list available from Dan Ichinose at AAAJ - LA
- This thing: https://rpubs.com/jwcb1025/est_ethnicity