```
Team 10 Dec 17 2020 EDA part 1
```

In [46]:
```
%matplotlib inline

import json
import requests
from bs4 import BeautifulSoup
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import seaborn as sns
import scipy.stats as stats
import shapefile as shp
import geopandas as gpd
import statsmodels.api as sm

import warnings
warnings.filterwarnings('ignore')
```

USDA Dataset

In [47]:
```
# read csv file accidents.csv
df = pd.read_csv('fooddesert.csv')
df.head()
```

Out[47]:

	CensusTract	State	County	Urban	POP2010	OHU2010	GroupQuartersFlag	NUMGQTRS	PO
0	1001020100	Alabama	Autauga	1	1912	693	0	0	
1	1001020200	Alabama	Autauga	1	2170	743	0	181	
2	1001020300	Alabama	Autauga	1	3373	1256	0	0	
3	1001020400	Alabama	Autauga	1	4386	1722	0	0	
4	1001020500	Alabama	Autauga	1	10766	4082	0	181	

5 rows × 147 columns

In [48]:
```
df.shape
```

Out[48]:
```
(72864, 147)
```

```
In [49]: list(df.columns)
```

```
Out[49]: ['CensusTract',
          'State',
          'County',
          'Urban',
          'POP2010',
          'OHU2010',
          'GroupQuartersFlag',
          'NUMGQTRS',
          'PCTGQTRS',
          'LILATracts_1And10',
          'LILATracts_halfAnd10',
          'LILATracts_1And20',
          'LILATracts_Vehicle',
          'HUNVFlag',
          'LowIncomeTracts',
          'PovertyRate',
          'MedianFamilyIncome',
          'LA1and10',
          'LAhalfand10',
In [50]: # filter DC data only
         df_dc = df[df['State'].isin(['District of Columbia'])].set_index('CensusTra

         # check for null vals
         df_dc.isnull().sum()
```

```
Out[50]: State                    0
         County                   0
         Urban                    0
         POP2010                  0
         OHU2010                  0
         GroupQuartersFlag        0
         NUMGQTRS                 0
         PCTGQTRS                 0
         LILATracts_1And10        0
         LILATracts_halfAnd10     0
         LILATracts_1And20        0
         LILATracts_Vehicle       0
         HUNVFlag                 0
         LowIncomeTracts          0
         PovertyRate              0
         MedianFamilyIncome       0
         LA1and10                 0
         LAhalfand10              0
         LA1and20                 0
         LATracts_half            0
         LATracts1                0
         LATracts10               0
         LATracts20               0
         LATractsVehicle_20       0
         LAPOP1_10                0
         LAPOP05_10               0
         LAPOP1_20                0
         LALOWI1_10               0
         LALOWI05_10              0
         LALOWI1_20               0
                                 ..
         lawhite20                0
         lawhite20share           0
         lablack20                0
         lablack20share           0
         laasian20                0
         laasian20share           0
         lanhopi20                0
         lanhopi20share           0
         laaian20                 0
         laaian20share            0
         laomultir20              0
         laomultir20share         0
         lahisp20                 0
         lahisp20share            0
         lahunv20                 0
         lahunv20share            0
         lasnap20                 0
         lasnap20share            0
         TractLOWI                0
         TractKids                0
```

```
TractSeniors            0
TractWhite              0
TractBlack              0
TractAsian              0
TractNHOPI              0
TractAIAN               0
TractOMultir            0
TractHispanic           0
TractHUNV               0
TractSNAP               0
Length: 146, dtype: int64
```

In [51]:
```
# shape of df post filtering
df_dc.shape
```

Out[51]:
```
(179, 146)
```

Replacing 0 vals by 1 for race variables

In [52]:
```
# create instance of df with only race values
df_race = df_dc[['TractWhite', 'TractBlack','TractAsian','TractNHOPI','Trac
                'TractOMultir','TractHispanic']]

# see any columns that have value of 0 for race
df_race.columns[(df_race == 0).any()]
```

Out[52]:
```
Index(['TractAsian', 'TractNHOPI', 'TractAIAN', 'TractOMultir',
       'TractHispanic'],
      dtype='object')
```

In [53]:
```
# replace 0 race vals by 1
df_race = df_race[['TractWhite', 'TractBlack','TractAsian','TractNHOPI','Tr
                'TractOMultir','TractHispanic']].replace(0,1) # replace 0 v

# see any columns that have value of 0 for race
df_race.columns[(df_race == 0).any()] # no cols --good
```

Out[53]:
```
Index([], dtype='object')
```

In [54]:
```
# merging updated race df with original dataframe
df_all = pd.merge(df_dc, df_race, how='inner', left_on=None, right_on=None,
         left_index=True, right_index=True)

df_all.head(2)
```

Out[54]:

	State	County	Urban	POP2010	OHU2010	GroupQuartersFlag	NUMGQTRS	PC
CensusTract								
11001000100	District of Columbia	District of Columbia	1	4890	2686	0	24	(
11001000201	District of Columbia	District of Columbia	1	3916	2	1	3908	(

2 rows × 153 columns

In [55]:
```
# create population variable, because the pop2010 var has discrepancies wit
df_all['totalpop'] = df_all['TractWhite_y'] + df_all['TractBlack_y'] + df_a
df_all.head()
```

Out[55]:

	State	County	Urban	POP2010	OHU2010	GroupQuartersFlag	NUMGQTRS	PC
CensusTract								
11001000100	District of Columbia	District of Columbia	1	4890	2686	0	24	(
11001000201	District of Columbia	District of Columbia	1	3916	2	1	3908	(
11001000202	District of Columbia	District of Columbia	1	5425	1933	0	1135	(
11001000300	District of Columbia	District of Columbia	1	6233	2754	0	0	(
11001000400	District of Columbia	District of Columbia	1	1455	636	0	6	(

5 rows × 154 columns

Calculating proportions by race

In [56]:
```
# create new columns with the proportion of race
df_all['prop_white']  = df_all['TractWhite_y']/df_all['totalpop']
df_all['prop_black']  = df_all['TractBlack_y']/df_all['totalpop']
df_all['prop_asian']  = df_all['TractAsian_y']/df_all['totalpop']
df_all['prop_hisp']   = df_all['TractHispanic_y']/df_all['totalpop']
df_all['prop_nhopi']  = df_all['TractNHOPI_y']/df_all['totalpop']
df_all['prop_aian']   = df_all['TractAIAN_y']/df_all['totalpop']
df_all['prop_multir'] = df_all['TractOMultir_y']/df_all['totalpop']

# check new columns were added
df_all.columns
```

Out[56]:
```
Index(['State', 'County', 'Urban', 'POP2010', 'OHU2010', 'GroupQuartersFl
ag',
       'NUMGQTRS', 'PCTGQTRS', 'LILATracts_1And10', 'LILATracts_halfAnd1
0',
       ...
       'TractOMultir_y', 'TractHispanic_y', 'totalpop', 'prop_white',
       'prop_black', 'prop_asian', 'prop_hisp', 'prop_nhopi', 'prop_aia
n',
       'prop_multir'],
      dtype='object', length=161)
```

In [57]:
```
df_all['totalprop'] = df_all['prop_white'] + df_all['prop_black'] + df_all[
df_all['totalprop'].value_counts()
```

Out[57]:
```
1.0    121
1.0     39
1.0     15
1.0      4
Name: totalprop, dtype: int64
```

Visualizng propportions, option A

Using facetgrid and melting df

In [58]:
```
# transform dataset for visualization
df_ex = df_all[['prop_white',
       'prop_black', 'prop_asian', 'prop_hisp', 'prop_nhopi', 'prop_aian',
       'prop_multir']].reset_index()
```

In [59]:
```
df_exm = pd.melt(df_ex, id_vars=['CensusTract'],
        value_vars=['prop_white',
        'prop_black', 'prop_asian', 'prop_hisp', 'prop_nhopi', 'prop_aian',
        'prop_multir'], var_name='race', value_name='race_proportion')

df_exm.head()
```

Out[59]:

	CensusTract	race	race_proportion
0	11001000100	prop_white	0.856922
1	11001000201	prop_white	0.701503
2	11001000202	prop_white	0.815707
3	11001000300	prop_white	0.828236
4	11001000400	prop_white	0.789740

In [60]:
```
df_exm.groupby(['CensusTract','race'])
df_exm.head(2)
```

Out[60]:

	CensusTract	race	race_proportion
0	11001000100	prop_white	0.856922
1	11001000201	prop_white	0.701503

In [61]:
```
# use FacetGrid to breakdown hour by borough
# g = sns.FacetGrid(df_exm, row="CensusTract")
# g.map_dataframe(sns.barplot, x="race", y='race_proportion')
# g.set_xticklabels(rotation=90)
```

Visualizing proportions, option B

Using stack bar and wide version

In [62]:
```
df_prop = df_all[['prop_white', 'prop_black', 'prop_hisp', 'prop_asian', 'p
df_prop.head(3)
```

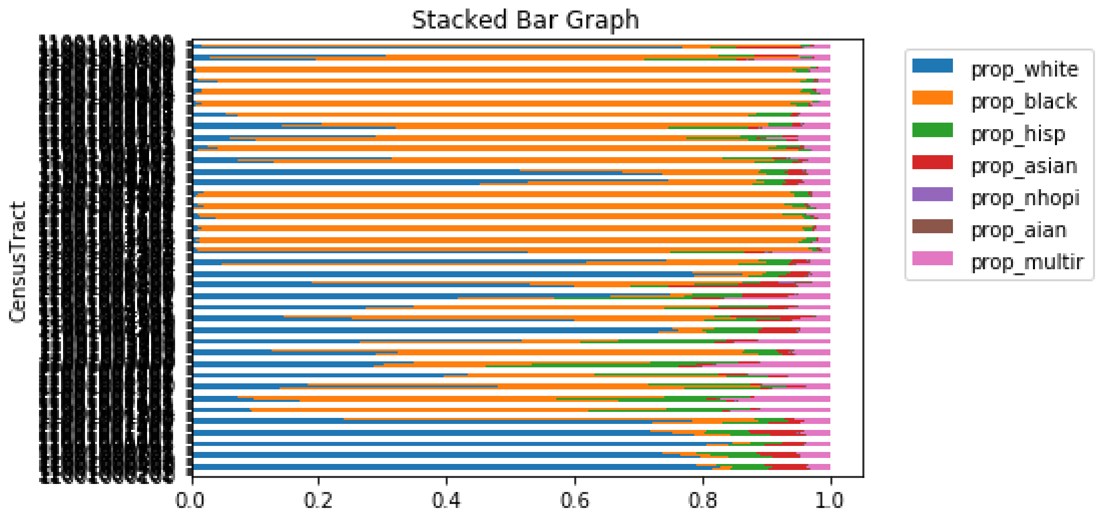
Out[62]:

	CensusTract	prop_white	prop_black	prop_hisp	prop_asian	prop_nhopi	prop_aian	prop_multir
0	11001000100	0.856922	0.020247	0.057077	0.038758	0.000386	0.001735	0.024875
1	11001000201	0.701503	0.061799	0.065617	0.115008	0.001193	0.001670	0.053209
2	11001000202	0.815707	0.027566	0.059466	0.065014	0.000520	0.000867	0.030860

In [63]:
```python
# plot a Stacked Bar Chart using matplotlib
df_prop.plot(
    x = 'CensusTract',
    kind = 'barh',
    stacked = True,
    title = 'Stacked Bar Graph',
    mark_right = True)

plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
```

Out[63]:
```
<matplotlib.legend.Legend at 0x1a22f49da0>
```



Creating diversity index

Some tracts have more diversity than ohters -- quantify issue

```
In [64]: # create entropy index
         # more info here: https://docs.google.com/presentation/d/1gw4hh7QoRWM19Lm1F
         df_all['entropy_index'] = -(df_all['prop_white']*np.log(df_all['prop_white'
                                     df_all['prop_black']*np.log(df_all['prop_black']) +
                                     df_all['prop_hisp']*np.log(df_all['prop_hisp']) +
                                     df_all['prop_asian']*np.log(df_all['prop_asian']) +
                                     df_all['prop_aian']*np.log(df_all['prop_aian']) +
                                     df_all['prop_multir']*np.log(df_all['prop_multir'])
                                     df_all['prop_nhopi']*np.log(df_all['prop_nhopi'])
                                    )
         # max entropy value = ln(# groups) = ln(7) = 1.95
         df_all.head(4)
```

Out[64]:

	State	County	Urban	POP2010	OHU2010	GroupQuartersFlag	NUMGQTRS	PC
CensusTract								
11001000100	District of Columbia	District of Columbia	1	4890	2686	0	24	(
11001000201	District of Columbia	District of Columbia	1	3916	2	1	3908	(
11001000202	District of Columbia	District of Columbia	1	5425	1933	0	1135	(
11001000300	District of Columbia	District of Columbia	1	6233	2754	0	0	(

4 rows × 163 columns

```
In [65]: # sort df by entropy values and check whether any value is null
         df_sorted = df_all.sort_values(by = 'entropy_index', ascending = False)
         df_sorted[df_sorted['entropy_index'].isnull()]
```

Out[65]:

	State	County	Urban	POP2010	OHU2010	GroupQuartersFlag	NUMGQTRS	PCTGQTI
CensusTract								

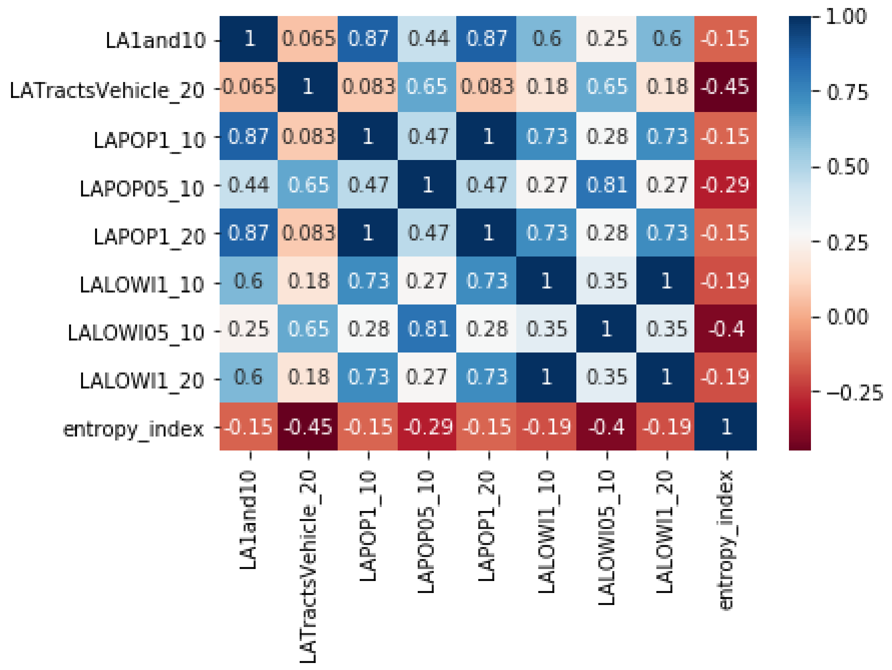
0 rows × 163 columns

```
In [66]: # pbc of first question
         pbc = stats.pointbiserialr(df_sorted['LAPOP1_10'], df_sorted['entropy_index
         pbc
```

```
Out[66]: PointbiserialrResult(correlation=-0.14667770470181346, pvalue=0.050080508
         50270673)
```

```
In [67]: # this looks at the correlation between all the variables of interest in da
         corrMatrix_la = df_sorted[['LA1and10','LATractsVehicle_20','LAPOP1_10', 'LA
                                   'LAPOP1_20','LALOWI1_10','LALOWI05_10', 'LALOWI1_20', 'ent
         #print(round(corrMatrix_la, 2))
```

In [68]:
```
# correlation matrix of entropy an other LA flags
sns.heatmap(corrMatrix_la, annot=True, cmap='RdBu')
plt.show()
```

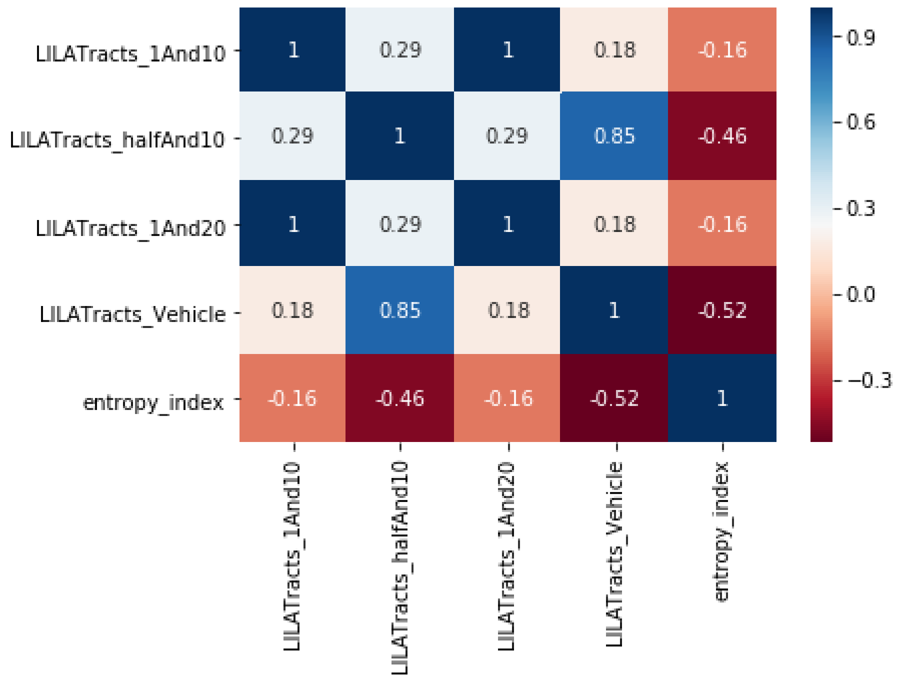


In [69]:
```
# correlation between entropy index and LATractsVehicle 20 miles
stats.pearsonr(df_sorted.entropy_index, df_sorted.LATractsVehicle_20) # ret
```

Out[69]:
```
(-0.45013558901120126, 2.583464373426191e-10)
```

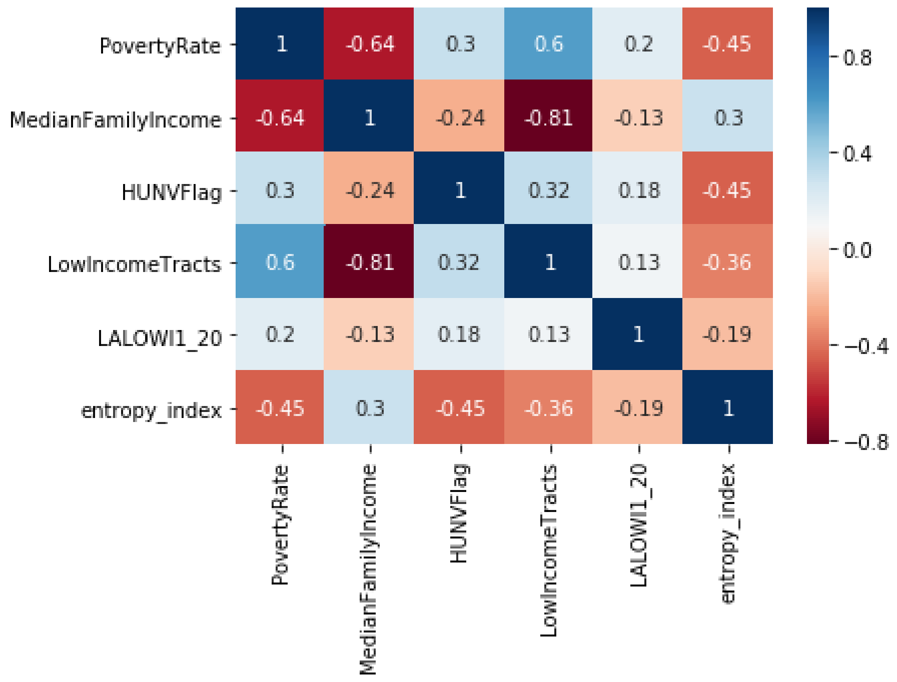
In [70]:
```
# this looks at the correlation between all the variables in dataset
corrMatrix_lila = df_sorted[['LILATracts_1And10','LILATracts_halfAnd10', 'L
#print(round(corrMatrix_lila, 2))
```

In [71]:
```
# correlation matrix of entropy an other LA flags
sns.heatmap(corrMatrix_lila, annot=True, cmap='RdBu')
plt.show()
```



In [72]:
```
# this looks at the correlation between all the variables in dataset
corrMatrix_other = df_sorted[['PovertyRate', 'MedianFamilyIncome', 'HUNVFla
#print(round(corrMatrix_other, 2))
```

```
In [73]: # correlation matrix of entropy an other LA flags
         sns.heatmap(corrMatrix_other, annot=True, cmap='RdBu')
         plt.show()
```



	PovertyRate	MedianFamilyIncome	HUNVFlag	LowIncomeTracts	LALOWI1_20	entropy_index
PovertyRate	1	-0.64	0.3	0.6	0.2	-0.45
MedianFamilyIncome	-0.64	1	-0.24	-0.81	-0.13	0.3
HUNVFlag	0.3	-0.24	1	0.32	0.18	-0.45
LowIncomeTracts	0.6	-0.81	0.32	1	0.13	-0.36
LALOWI1_20	0.2	-0.13	0.18	0.13	1	-0.19
entropy_index	-0.45	0.3	-0.45	-0.36	-0.19	1

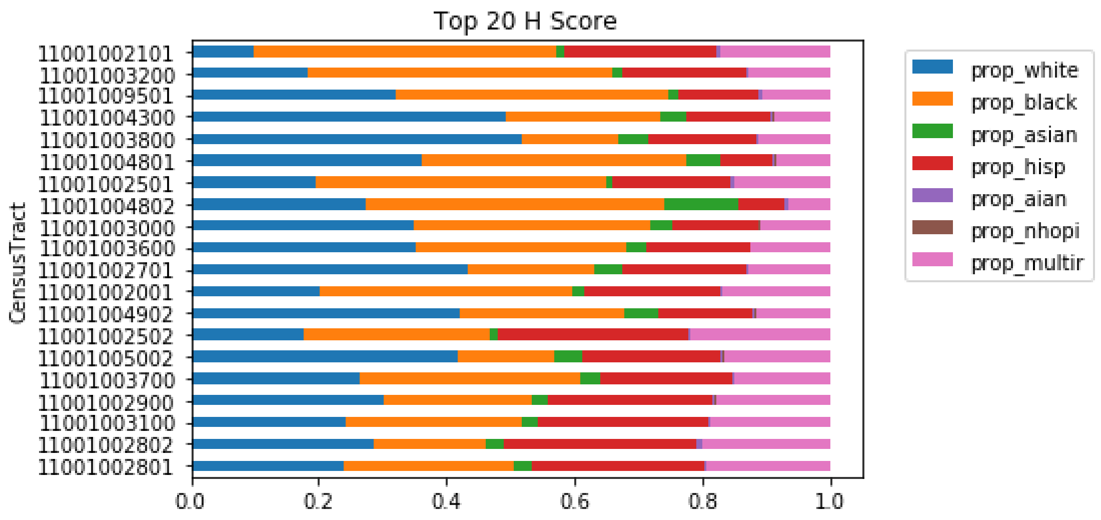
What are the most diverse and least diverse tracts?

```
In [74]: # create top 20 and bottom 20 dfs by entropy
         top_20 = df_sorted[['prop_white', 'prop_black', 'prop_asian', 'prop_hisp',


         # plot a Stacked Bar Chart using matplotlib
         top_20.reset_index().plot(
             x = 'CensusTract',
             kind = 'barh',
             stacked = True,
             title = 'Top 20 H Score',
             mark_right = True)

         plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
```

```
Out[74]: <matplotlib.legend.Legend at 0x1a20dc6a20>
```

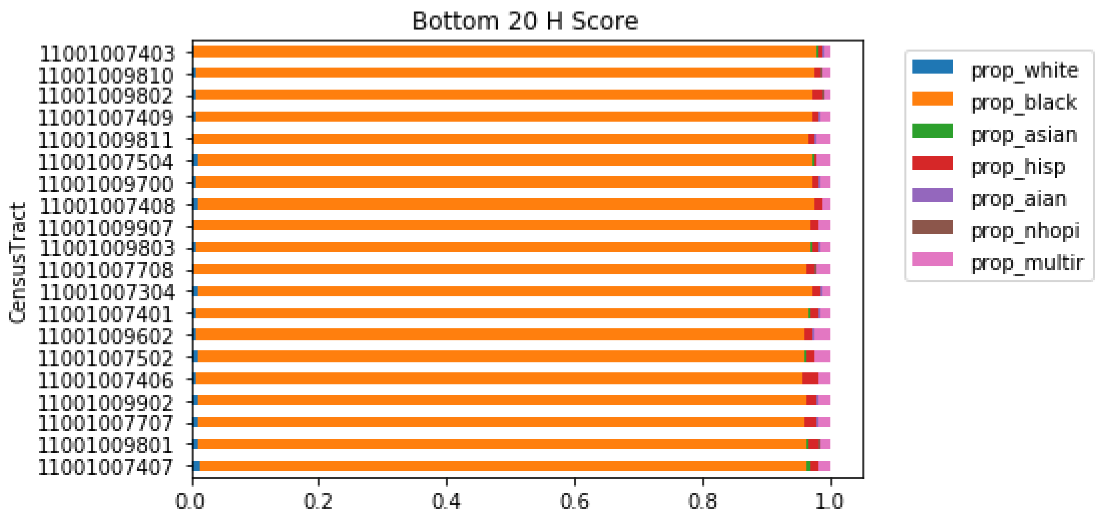


```
In [75]: # create bottom 20 df by entropy index
         bottom_20 = df_sorted[['prop_white', 'prop_black', 'prop_asian', 'prop_hisp

         # plot a Stacked Bar Chart using matplotlib
         bottom_20.reset_index().plot(
             x = 'CensusTract',
             kind = 'barh',
             stacked = True,
             title = 'Bottom 20 H Score',
             mark_right = True)

         plt.legend(bbox_to_anchor=(1.05, 1), loc='upper left')
```

```
Out[75]: <matplotlib.legend.Legend at 0x1a21094518>
```



Part 2: Mapping Data

Refer to other pdf file for mapping data.

```
In [ ]:
```