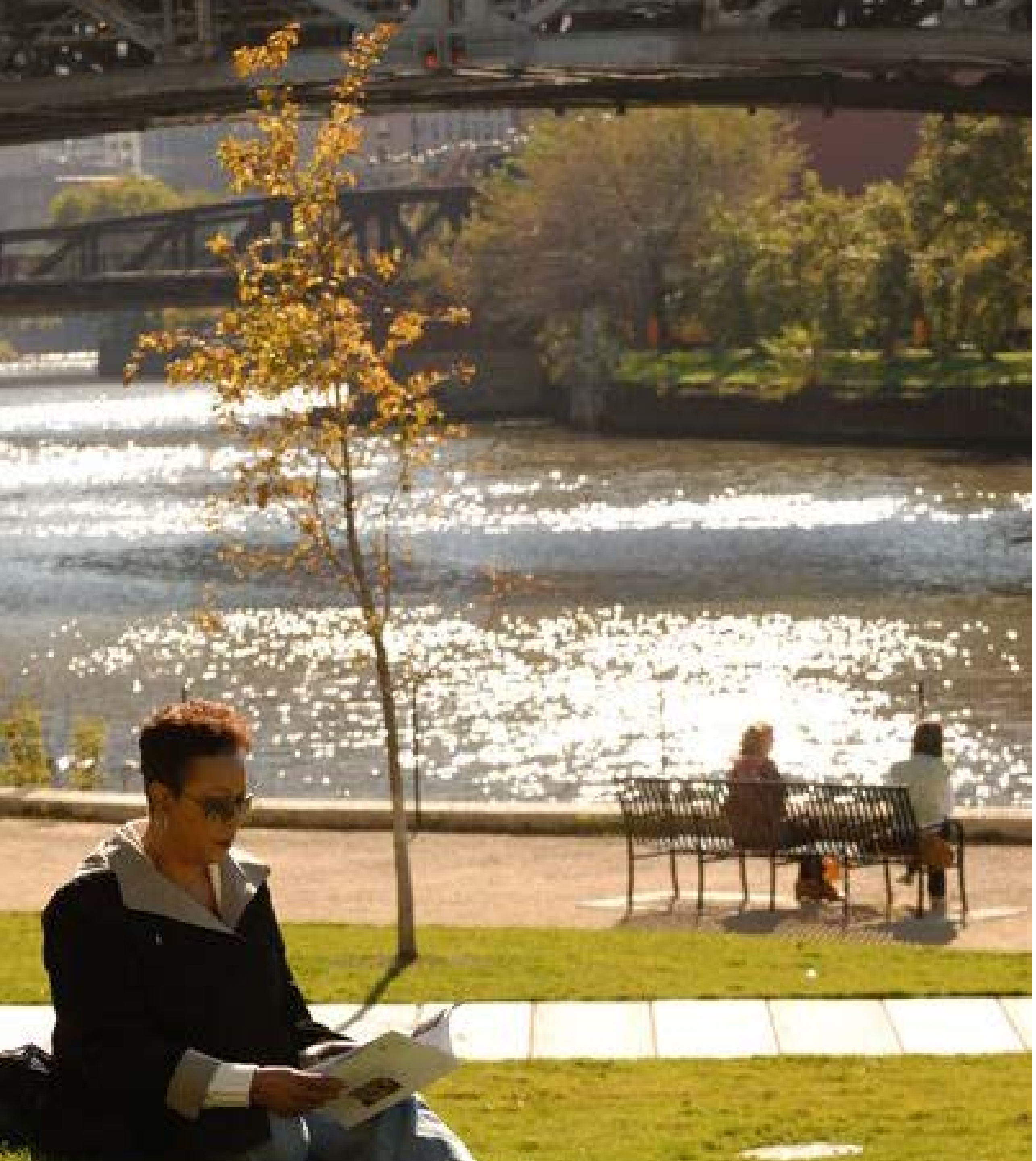
What brings people to parks?

Using social media, amenities and park accessibility to predict park visitation rates

THE TRUST FOR PUBLIC LAND



What brings people to parks?

Why do park users choose one over the other?

- **Hypothesis:** Park usage (i.e. the number of geolocated tweets within a park boundary) can be predicted through park size, the number and types of amenities in the park, and the number of people living with a 10-minute walk of the park.

What brings people to parks?

Who cares and why?

- City officials
- Park professionals
- Community members
- Conservation organizations
- Landscape architects
- Social equity organizations

What brings people to parks?

Data Sources

- NYC Parks and Recreation Department
    - Park locations
    - General park attributes (e.g. park size, park type)
    - Park amenities
- The Trust for Public Land's ParkServe® database
    - 10-minute walk park statistics
- Twitter Streaming API

NYC Parks

Image source: *Friends of the High Line*

THE TRUST FOR PUBLIC LAND

What brings people to parks?

Data Wrangling

- Get amenity count for each park (total and for each amenity)
- Join 10-minute walk statistics

```
# group by park ID and get the count of amenities of interest
athletics_count_df = athletics_df.groupby('gispropnum')[['handball','tennis','basketball','adult_soft', 't
rack_and']].apply(lambda x: x[x=='Yes'].count())
athletics_count_df.head()
```

	handball	tennis	basketball	adult_soft	track_and
gispropnum					
B001	2	0	2	0	0
B007	6	8	4	3	0
B008	12	0	6	2	0
B012	1	0	2	0	0
B016	4	0	3	0	0

```
parks_features_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1705 entries, 0 to 1704
Data columns (total 33 columns):
GISPROPNUM          1705 non-null object
BOROUGH             1705 non-null object
ACRES               1705 non-null float64
SIGNNAME            1705 non-null object
TYPECATEGO          1705 non-null object
handball            1705 non-null float64
tennis              1705 non-null float64
basketball          1705 non-null float64
adult_soft          1705 non-null float64
track_and           1705 non-null float64
play_area_count     1705 non-null float64
preserve_count      1705 non-null float64
spray_count         1705 non-null float64
OBJECTID            1705 non-null int64
TPL_P_NAME          1705 non-null object
FREQUENCY           1705 non-null int64
SUM_SVCAREA         1705 non-null float64
SUM_TOTPOPSVCA      1705 non-null int64
SUM_KIDSVCA         1705 non-null int64
SUM_YOUNGPROSVCA    1705 non-null int64
SUM_SENIORSVCA      1705 non-null int64
SUM_HHILOWSVCA      1705 non-null int64
SUM_HHIMEDSVCA      1705 non-null int64
SUM_HHIHIGHSVCA     1705 non-null int64
SUM_TOTHHSVCA       1705 non-null int64
SUM_WHITE_SVC       1705 non-null int64
SUM_BLACK_SVC       1705 non-null int64
SUM_AMERINDSVC      1705 non-null int64
SUM_ASIAN_SVC       1705 non-null int64
SUM_PACIFICSVC      1705 non-null int64
SUM_OTHRACESVC      1705 non-null int64
SUM_RACE2UPSVC      1705 non-null int64
SUM_HISP_SVC        1705 non-null int64
```

THE TRUST FOR PUBLIC LAND

What brings people to parks?

Data Wrangling

- Choose random sample of 1,705 NYC parks
- Collect geolocated tweets in JSON and parse to CSV

```
In [12]: #take random sample of parks in parks csv
         testy_test = np.random.choice(parks_df.index, 100)
         print(testy_test)
         parks_rand_s = pd.Series(testy_test)

['M007' 'M280' 'Q444' 'M358' 'X336' 'R028' 'Q009' 'X289' 'M278' 'B429'
 'B357' 'M256' 'R098' 'M249' 'B058' 'B085' 'B337' 'B357' 'M017' 'B042'
 'R165' 'R020' 'X536' 'Q561' 'R505' 'B590' 'B261' 'B324' 'M517' 'M025'
 'R117' 'Q266' 'R126' 'B582' 'M079' 'B334' 'B219' 'Q523' 'X292' 'B302'
 'M365' 'Q267' 'M238' 'QT20' 'B121' 'Q354' 'Q544' 'M241' 'X137' 'X291'
 'B359' 'X201' 'X131' 'X365' 'B251' 'Q557' 'Q471' 'R524' 'B432' 'Q051'
 'X508' 'B209' 'Q102' 'X126' 'X266A' 'Q536' 'B565' 'X302' 'B210W' 'Q563'
 'M231' 'M082' 'Q293' 'Q294' 'B154' 'X515' 'M402' 'X180' 'M323' 'B223NA'
 'M127' 'M194' 'BT10' 'B019' 'Q362' 'R059' 'B265' 'X259' 'BT03' 'X040'
 'X080' 'Q165' 'M130' 'X182' 'B210P' 'Q475' 'M198' 'R079' 'Q011' 'X365']

In [13]: parks_rand_s.to_csv(r'Y:/Springboard/Parks_RandSample.csv')
```

```
out_file.write(str(tweet['id_str'])+ ',' + str(tweet['user']['screen_name']) + ',' +
               str(tweet['text']).replace(',',' ').replace("\n",'').replace("\r","").replace('"','').lstrip().lstrip('-') +
               ','+ d.strftime('%Y%m%d') + ',' + str(tweet['geo']['coordinates'][0]) + ',' + str(tweet['geo']['coordinates'][1]) + "\n")
```

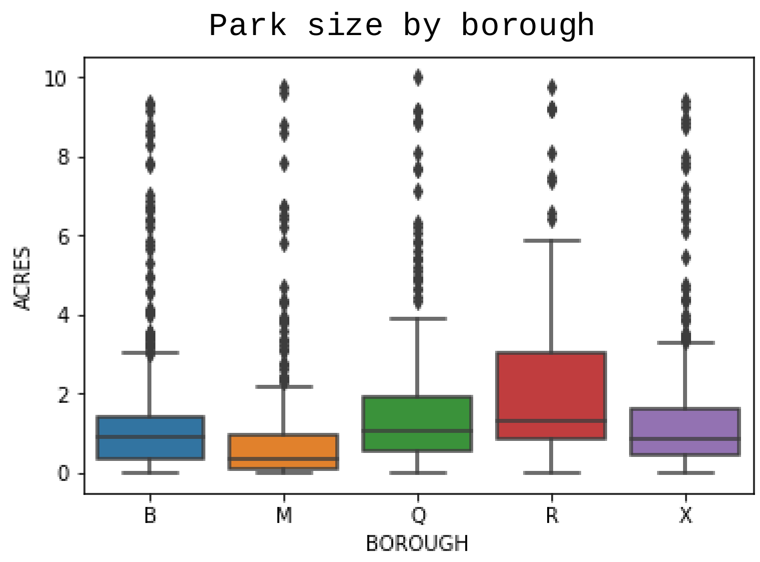
Above code snippet originally obtained from Natural Capital Project and modified for use in this project

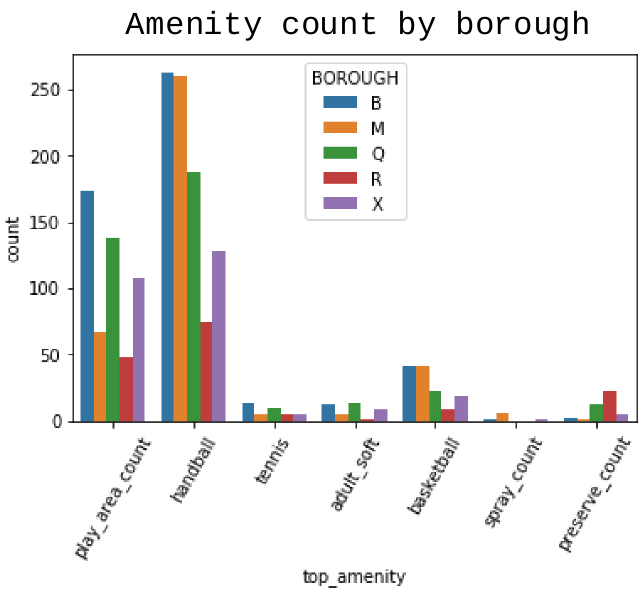
THE TRUST FOR PUBLIC LAND

What brings people to parks?

Initial Findings – Park attributes

- Amenity count does not appear to be strongly correlated with park size
- Manhattan has the lowest median # of park amenities and Queens has the highest.
- Staten Island has the largest parks on average
- No strong correlation between # of people within a 10-minute walk and park size, or # of people within a 10-minute walk and amenity count.
- Handball courts are the most popular amenities in NYC parks.



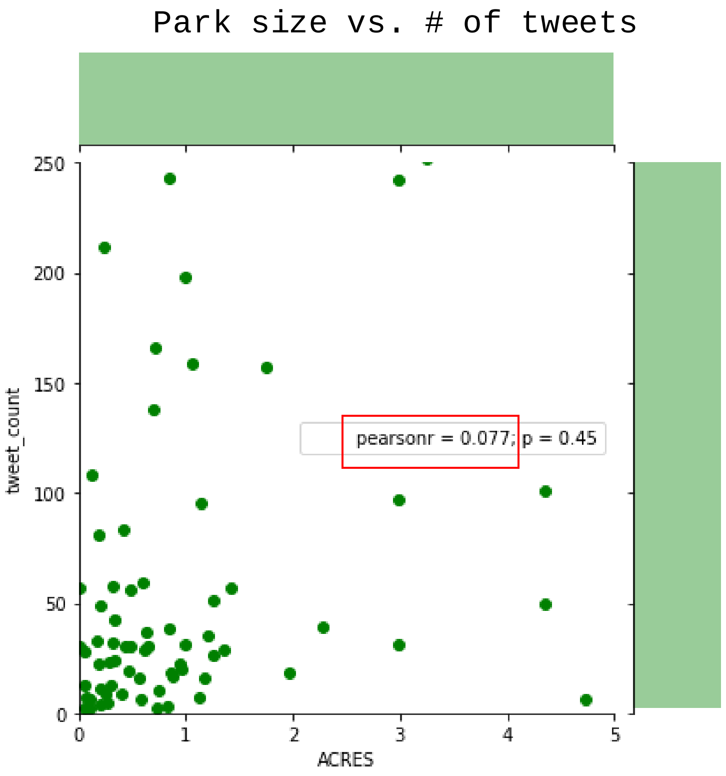


THE TRUST FOR PUBLIC LAND

What brings people to parks?

Initial Findings – Park attributes and tweets

- The number of tweets within a park boundary is **not linearly correlated** with park size, total number of park amenities, or number of people living within a 10-minute walk of a park.
- I **did not find any significant difference** in means for small vs. large parks and # of tweets, parks with many amenities vs parks with few amenities and # of tweets, or parks with many people living within a 10-minute walk vs. parks with few people living within a 10-minute walk and # of tweets.




```
# run significance test for mean park usage per year and total # of park amenities

fewAmenities = parksInfo.loc[parksInfo.total_amenities<1]['tweet_count']
manyAmenities= parksInfo.loc[parksInfo.total_amenities>5]['tweet_count']

# null = available amenities does not impact the # of tweets
# alt = parks with more amenities have more tweets

sample_diff_means = diff_of_means(fewAmenities,manyAmenities)
print(sample_diff_means)

# Acquire permutation samples
perm_replicates = draw_perm_reps(fewAmenities, manyAmenities, diff_of_means, 1000)
print(np.percentile(perm_replicates, [0.5,99.5]))

# Compute and print p-value: p
p = np.sum(perm_replicates <= sample_diff_means) / float(len(perm_replicates))
print('p-value = {}'.format(p))
```

```
165.98255814
[-1210.96947674   966.0840298 ]
p-value = 0.504
```

THE TRUST FOR PUBLIC LAND

Predicting park visitation

Machine Learning – Random Forest Regressor

```
from sklearn.preprocessing import normalize

y = np.array(parksInfo.tweet_count)
X = parksInfo_dummied
#X_norm = normalize(X)

from sklearn.model_selection import train_test_split

train_X, test_X, train_y, test_y = train_test_split(X, y,
```

```
from sklearn.model_selection import GridSearchCV

# Create the parameter grid using list of set values
param_grid = {'n_estimators': [100, 200, 300, 1000, 5000]}

rf = RandomForestRegressor()

grid_search = GridSearchCV(estimator = rf, param_grid = param_grid)

grid_search.fit(train_X, train_y)
grid_search.best_params_
```

```
{'n_estimators': 100}
```

```
rf = RandomForestRegressor(n_estimators = 100)
# Train the model on training data
rf.fit(train_X, train_y)
predict_test = rf.predict(test_X)

print "Train R-squared :: ", rf.score(train_X, train_y)
print "Test R-squared  :: ", rf.score(test_X, test_y)
```

```
Train R-squared ::  0.8653115296513108
Test R-squared  ::  0.25302542372881365
```

- Results are not too great.

  Let’s try to do better than this!

THE TRUST FOR PUBLIC LAND

Predicting park visitation

Machine Learning – Random Forest Classifier

```
# look at distribution of tweets counts to determine class thresholds
parksInfo_dummied.tweet_count.describe()
```

```
count       98.00000
mean       568.55102
std       2435.58974
min          2.00000
25%         22.00000
50%         53.50000
75%        242.75000
max      22373.00000
Name: tweet_count, dtype: float64
```

```
# in the future, use pandas.cut
def label_tweets (row):
   if row['tweet_count'] <= 30:
     return 0
   if row['tweet_count'] > 30 and row['tweet_count'] <= 200:
     return 1
   else:
     return 2

parksInfo_dummied['tweet_class'] = parksInfo_dummied.apply(lambda row: label_tweets (row),axis=1)
parksInfo_dummied.head()
```

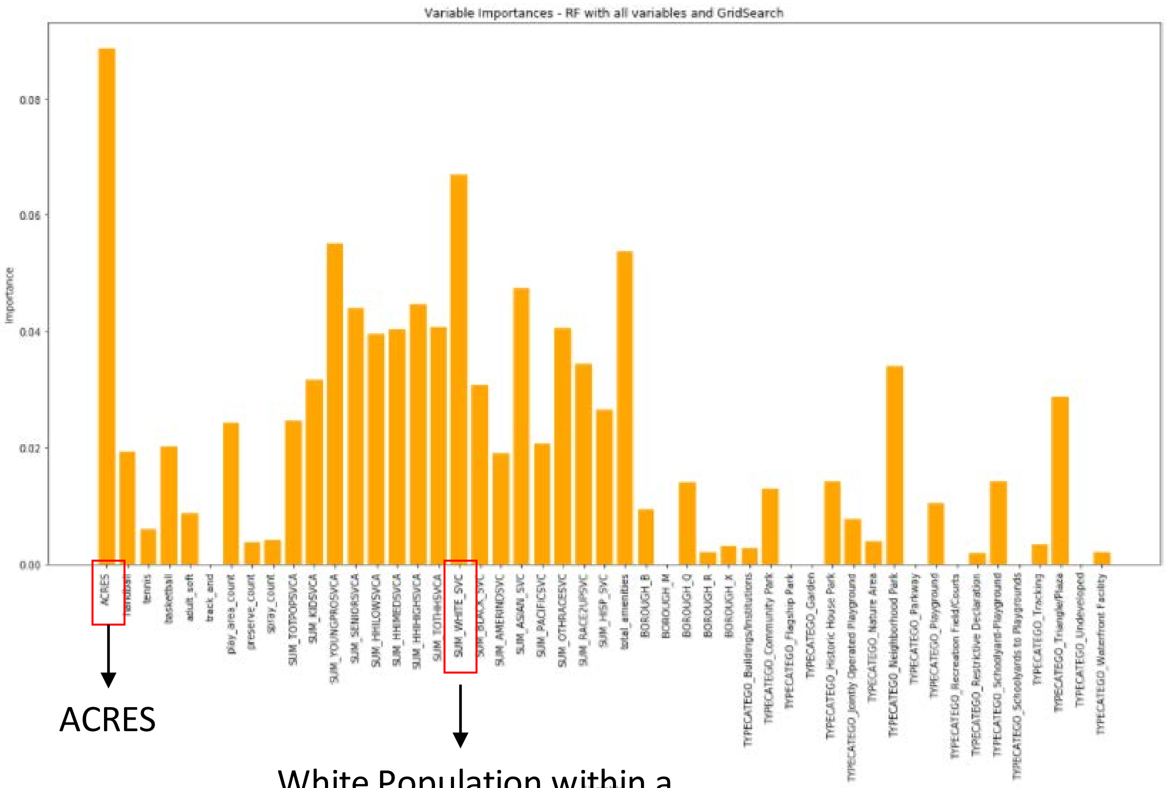
```
# Create the parameter grid based on the results of range
param_grid = {'n_estimators': range(1,50)}
# Create a based model
rf = RandomForestClassifier()
# Instantiate the grid search model
grid_search = GridSearchCV(estimator = rf, param_grid =

grid_search.fit(train_X, train_y)
grid_search.best_params_
```

```
{'n_estimators': 17}
```

```
rf = RandomForestClassifier(n_estimators=17)
# Train the model on training data
rf.fit(train_X, train_y)

predict_test = rf.predict(test_X)

print "Train Accuracy :: ", accuracy_score(train_y, rf.p
print "Test Accuracy  :: ", accuracy_score(test_y, predi
```

```
Train Accuracy ::  0.9871794871794872
Test Accuracy  ::  0.65
```

OK, but not good enough!



ACRES

White Population within a 10-minute walk

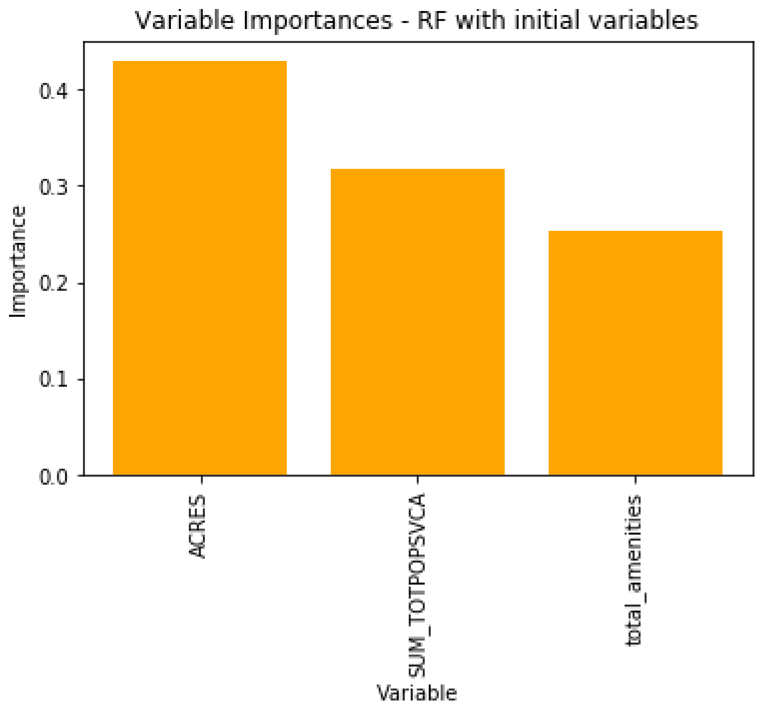
THE TRUST FOR PUBLIC LAND

Predicting park visitation

Machine Learning – Random Forest Classifier

Let's look at the accuracy with the 3 initial variables of interest...

```
: parksInfo_sub = parksInfo_dummied[['ACRES','SUM_TOTPOPSVCA','total_amenities',]]

  y = np.array(parksInfo_dummied.tweet_class)
  X = parksInfo_sub

  train_X, test_X, train_y, test_y = train_test_split(X, y, test_size = 0.2, random_state = 42)

  # Create the parameter grid based on the results of range
  param_grid = {'n_estimators': range(1,50)}
  # Create a based model
  rf = RandomForestClassifier()
  # Instantiate the grid search model
  grid_search = GridSearchCV(estimator = rf, param_grid = param_grid)

  grid_search.fit(train_X, train_y)
  grid_search.best_params_

: {'n_estimators': 43}

: rf = RandomForestClassifier(n_estimators=43)
  rf.fit(train_X, train_y)

  predict_test = rf.predict(test_X)


  print "Train Accuracy :: ", accuracy_score(train_y, rf.predict(train_X))
  print "Test Accuracy  :: ", accuracy_score(test_y, predict_test)

  Train Accuracy ::  0.9871794871794872
  Test Accuracy  ::  0.45
```



Definitely not good enough!

THE TRUST FOR PUBLIC LAND

Predicting park visitation

Conclusions and Next Steps

Datasets for further investigation:

- **Public transportation**
- **More amenities**
- **Crime statistics**
- **Park programming/rental facilities**

Using the data at hand, park size, the number and type of amenities, and the number of people living within a 10-minute walk of the park **are not reliable predictors of park usage**, yet bringing more variables into the analysis strengthens the model.

Questions for further consideration:

- **Is the number of tweets within a park truly representative of park usage/visitation?**
- **Define local perception of a high-quality park. Are high quality parks more likely to have a higher number of tweets?**
- **Can the number of tweets predict park attributes, such as park type (e.g. community, pocket, regional) and park size?**

THE TRUST FOR PUBLIC LAND

Thank you.

Emmalee Dolfi

GIS Sr. Project Manager

emmalee.dolfi@tpl.org

THE TRUST FOR PUBLIC LAND