Tree Count Density vs Average Property Valuations in NYC

Vedant Gannu (gannuv@rpi.edu), **Fred Buchanan** (buchaf@rpi.edu), **Shepard Gordon** (gordos4@rpi.edu), **Alex He** (buchaf@rpi.edu)

Abstract

The goal of our project was to study and analyze data related to one of the United Nations 17 sustainable goals [1]. Our team chose data from NYC's Open Data website to better understand how effective NYC is at meeting UN goal 11 ("Sustainable Cities and Communities"). The datasets chosen were 1) 2015 Street Tree Census [2] and 2) NYC Property Valuation and Assessment Data [3], both of which have been previously collected for NYC. These datasets were selected to analyze correlations that may exist between tree count density and property value.

This poster will mainly cover the dataset acquisition and workflow process, code run for DataFrame creation, procedure for aggregating the DataFrames, and displaying the results of log scale regression and Gini Importance test from using a Random Forest Classifier. Random Forest Regressor is trained on the tree counts and mean property value. High importance indicates greater predictive power.

Motivation

A positive correlation would encourage developers to plant more trees to increase their property values, leading to more trees and thus an overall more sustainable and healthy city.

2015 Street Tree Census - Tree Data | NYC Open Data (cityofnewyork.us)

Poster: MT15A-08

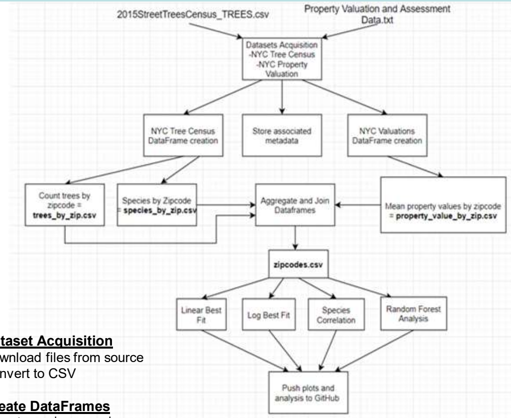
Glossary:

RPI – Rensselaer Polytechnic Institute
TWC – Tetherless World Constellation at Rensselaer Polytechnic Institute

Acknowledgments:

Prof Thilanka and TA Aradhya Kasat from Data Science Class 2020

Dataset Acquisition and Workflow



1. **Dataset Acquisition**
   Download files from source
   Convert to CSV

1. **Create DataFrames**

```
import pandas as pd

def count_trees(inpath="raw/trees.csv", outpath="data/trees_by_zip.csv"):
   df = pd.read_csv(inpath)
   tree_counts = df.groupby('zipcode')['tree_id'].count()
   tree_counts.to_frame(name="number_of_trees").to_csv(outpath)


def species_by_zip(inpath="raw/trees.csv", outpath="data/species_by_zip.csv"):
   df = pd.read_csv(inpath)

   table = pd.pivot_table(df, values='tree_id', index='zipcode', columns = 'spc_common',
      aggfunc='count').fillna(0)

   table.to_csv(outpath)

def average_property_value(inpath="raw/avroll.csv",outpath="data/property_value_by_zip.csv",
drop_less_than = 50):
    df = pd.read_csv(inpath)
    df = df.dropna(subset=['ZIP','FULLVAL'])
    df["ZIP"] = df["ZIP"].astype(int)
    tree_counts = df.groupby('ZIP')['FULLVAL'].agg(['mean','median', 'min', 'max', 'count'])
    tree_counts = tree_counts[tree_counts['count'] >= drop_less_than]
    tree_counts.to_csv(outpath)
```

1. **Calculate values by zipcode**

```
tree_counts = df.groupby('zipcode')['tree_id'].count()

tree_counts = df.groupby('ZIP')['FULLVAL'].agg(['mean','median', 'min', 'max', 'count'])
tree_counts[tree_counts['mean'] < 4.9e6]['mean'].hist(bins=30)
```

1. **Aggregation**

```
trees = pd.read_csv(trees_path)
properties = pd.read_csv(property_path)
species = pd.read_csv(species_path)

merged = trees.merge(properties, left_on='zipcode', right_on='ZIP')
merged = merged.merge(species, left_on='zipcode', right_on='zipcode')
merged = merged[['zipcode', 'number_of_trees', 'mean'] + list(species.columns)]
merged = merged.rename(columns = {'mean':'mean_property_value'})

merged.to_csv(outpath)
```

1. **Analysis**

```
import numpy as np
import statsmodels.formula.api as smf
from sklearn.ensemble import RandomForestRegressor

results = smf.ols('np.log(mean_property_value) ~ np.log(number_of_trees)', data=df).fit()

space = np.linspace(df['number_of_trees'].min(), df['number_of_trees'].max())
predictions = results.predict(exog=dict(number_of_trees=space))

count_columns = [column for column in df.columns if '_count' in column]
results = smf.ols('mean_property_value ~ ' + " + ".join(count_columns), data=df).fit()

X = df[count_columns]
y = df['mean_property_value']
clf = RandomForestRegressor(random_state=0)
clf.fit(X,y)
importances = pd.DataFrame(dict(name=count_columns,importance=clf.feature_importances_))
   .sort_values('importance', ascending=False).head(10)
```

Results

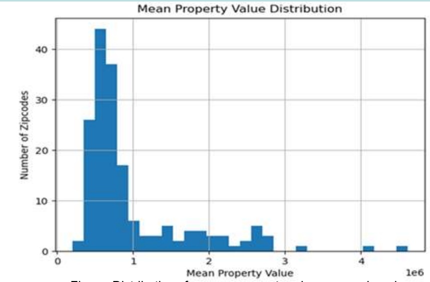


Figure: Distribution of average property value across zipcodes

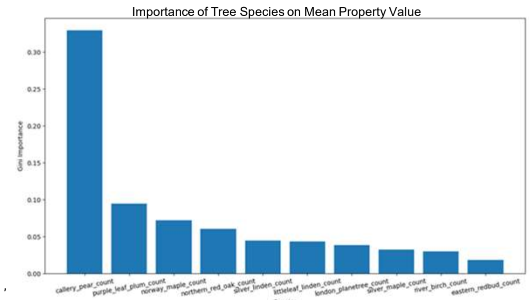
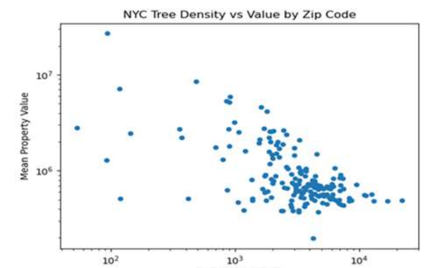


Figure: Gini Importance test outcome for various species



OLS Regression Results

Dep. Variable:	np.log(mean_property_value)	R-squared:	0.365
Model:	OLS	Adj. R-squared:	0.361
Method:	Least Squares	F-statistic:	100.6
Date:	Wed, 16 Dec 2020	Prob (F-statistic):	5.49e-19
Time:	14:58:17	Log-Likelihood:	-149.90
No. Observations:	177	AIC:	303.8
Df Residuals:	175	BIC:	310.2
Df Model:	1		
Covariance Type:	nonrobust		

6. **Visualizations**

```
import matplotlib.pyplot as plt

merged.plot('number_of_trees', 'mean_property_value',
kind='scatter')

plt.plot(space, predictions, 'r-', label="Best Fit")

plt.bar(importances['name'], importances['importance'])
```

Resources:

1. THE 17 GOALS | Sustainable Development. (n.d.). Retrieved December 16, 2020, from https://sdgs.un.org/goals
2. Department of Parks and Recreation (DPR). (2016, June 3). 2015 Street Tree Census - Tree Data. Retrieved December 11, 2020, from https://data.cityofnewyork.us/Environment/2015-Street-Tree-Census-Tree-Data/pi5s-9p35
3. Department of Finance (DOF). (2020, May 26). Property Valuation and Assessment Data. Retrieved December 11, 2020, from https://data.cityofnewyork.us/City-Government/Property-Valuation-and-Assessment-Data/yjxr-fw8i