CS –673 Mid Term Project

Analyzing New York City Data

Instructor: Anthony Escalona

Scalable database

Group - AVENGERS

Team Members

AJAY PODENDLA

DEEPIKA DHANAVEL

PAVANKALYAN SUDAGANI

GNANA PRASUNA

Table of content:

- Problems
- Conclusion
- Future steps

Goals of the project:

We are going to take the data set of trees from the open data New York city from 2005and 2015 and use them to find the top three species in Newyork and we also used another dataset nybb to find the location of the trees. For doing these we mostly use pandas

Later we connect the csv to Postgres and insert the data into Postgres. We will do different operations like join and all other functions for analyzing the data.

We use different charts like bar charts, pie charts and geo charts for data visualization and show different graphs for the overview of the trees

Introduction:

The importance of trees is immense and extends across various aspects of the environment, human well-being, and the planet's ecosystem. Trees play a critical role in producing oxygen and water conservation.

So, In this project we took this key element of trees in newyork and analyzed the data and showed them through charts

In this project we used panda's library for data manipulation and for information about the data.

We also used Geopandas for analyzing and handling the spatial data GeoPandas provides a range of spatial operations that can be performed on GeoDataFrames

GeoPandas builds on the pandas library

Design:

We used Jupiter notebook for doing the coding and Postgres for sql.

In these project we used three different datasets in which two are tree dataset and one dataset is for location.

All these datasets are taken from open newyork dataset which are real world example which will help us in future.

We will explore these data using different liabraies

We will import all the libraries like Pandas, NumPy, matplotlib, seaborn

First import the tree data with the help of path and stored in a data frame we can get the info with the help of info() in these first data there are around 42 columns

Now the Analyzing and manipulation will be done

first we will handle all the missing values with the help of fillna().

Code:
```
df2['zip_city'].fillna('Unknown', inplace=True)
df2['boro_ct'].fillna(0, inplace=True)
```

We will convert all the common species names to title case

Code:
```
df2['spc_common'] = [name.title() if not pd.isna(name) else name for name in df2['spc_common']]
```

Then we Fill 'health' with 'dead' where 'status' is not 'alive'

Code: df2['health'] = ['Dead' if status != 'Alive' else health for status, health in zip(df2['status'], df2['health'])]

Drop rows where 'health' is empty
Code: df2.dropna(subset=['health'], axis=0, inplace=True)

Selecting a subset of columns
Code: selected_columns = ['tree_dbh', 'status', 'health', 'spc_common', 'boroname', 'Latitude', 'longitude']
df_subset = df2[selected_columns]

Result:

```
#   Column      Non-Null Count   Dtype
---  ------      --------------   -----
 0   tree_dbh    683787 non-null  int64
 1   status      683787 non-null  object
 2   health      683787 non-null  object
 3   spc_common  652168 non-null  object
 4   boroname    683787 non-null  object
 5   Latitude    683787 non-null  float64
 6   longitude   683787 non-null  float64
```

Then we normalize the data for tree count

Code: species_counts = df2["spc_common"].value_counts(normalize=True, dropna=True).reset_index()
species_counts.columns = ["species", "proportion"]

	species	proportion
0	London Planetree	0.133423
1	Honeylocust	0.098539
2	Callery Pear	0.090362
3	Pin Oak	0.081551

4	Norway Maple	0.052424
5	Littleleaf Linden	0.045605
6	Cherry	0.044895
7	Japanese Zelkova	0.044863
8	Ginkgo	0.032237
9	Sophora	0.029652

We can find the top 10 species with the help of head(10).

Now we will take the 2005 tree_data and do the conversion like typecasting changing the latitudes to float all of those.

Handling the missing data and modifing the data types.

We will use the nybb dataset to find the number of trees in a region.

```
the_geom  BoroCode      BoroName  \
0  MULTIPOLYGON (((-74.05050806403247 40.56642203...     5  Staten Island
1  MULTIPOLYGON (((-73.89680883223778 40.79580844...     2          Bronx
2  MULTIPOLYGON (((-73.82644661516991 40.59052744...     4         Queens
3  MULTIPOLYGON (((-74.01092841268026 40.68449147...     1      Manhattan
4  MULTIPOLYGON (((-73.86327471071958 40.58387684...     3       Brooklyn
```

We use different colors for different regions like

{'Staten Island': 'red', 'Bronx': 'green', 'Queens': 'orange', 'Manhattan': 'blue', 'Brooklyn': 'pink'}

We will now create the tables in Postgres with the help of python we can change anything in the tables from jupyter notebook and in Postgres it automatically changes

We need to install psycopg2 to connect with the database. Psycopg2 is a PostgreSQL adapter for the Python programming language. It provides a PostgreSQL database API for Python, allowing Python programs to connect to and interact with PostgreSQL databases.

First query is aggregating data from the trees_data table to count the total number of trees for each borough (boroname). The result set includes the borough name and the corresponding total number of trees. The GROUP BY clause is used to group the data by borough.

Steps:

Define the table creation SQL query for "borough"
Execute the table creation query for "borough"
Read data from CSV into a pandas DataFrame
Iterate over rows in the DataFrame and insert into PostgreSQL
Commit the changes and close the connection

These query is used for aggregating data from the trees_data table to calculate the average diameter at breast height (tree_dbh) for each tree species (spc_common). The result set includes the species name and the corresponding average diameter. The GROUP BY clause is used to group the data by tree species.

Steps:
Retrieve data from the "trees" table
Retrieve data from the "trees_2005" table
Syntax:

```
for row in trees_2005_data:
   print(row)
```

In these we will use the join operation for joining the data_tree and borough.

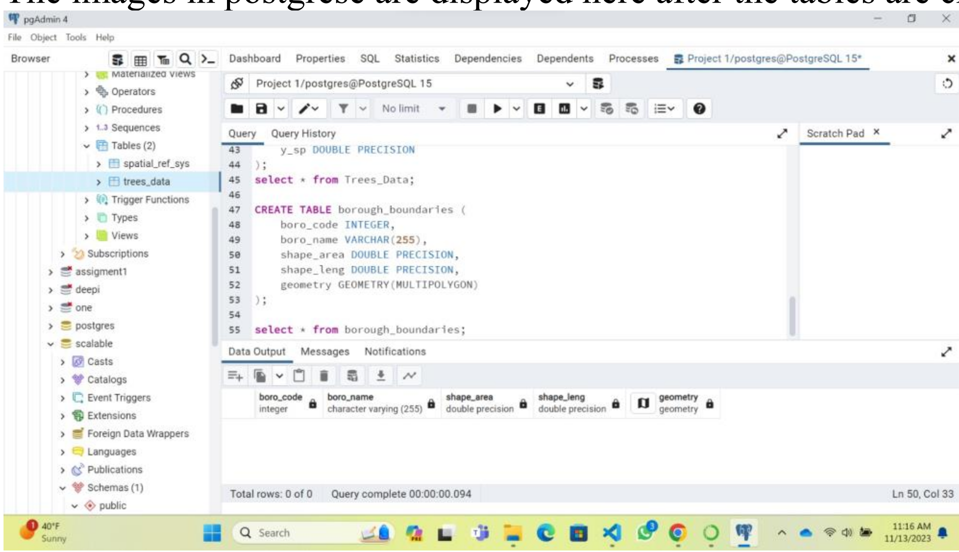
```
join_query = """
   SELECT trees_data.created_at, trees_data.tree_id, trees_data.block_id,
trees_data.the_geom, trees_data.tree_dbh,
         borough.BoroCode, borough.BoroName, borough.Shape_Leng,
borough.Shape_Area
   FROM trees_data
```

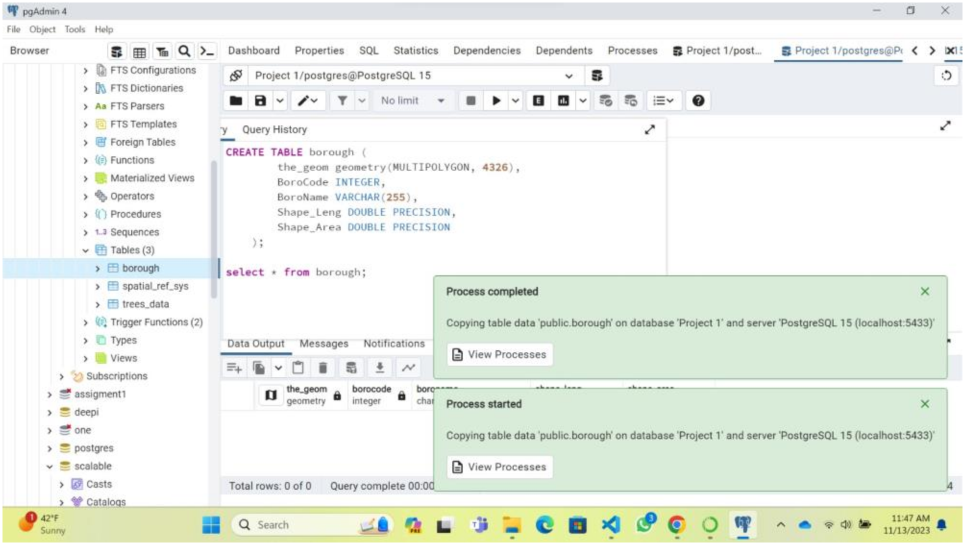
```
JOIN borough ON trees_data.borocode = borough.borocode
LIMIT 2;
```

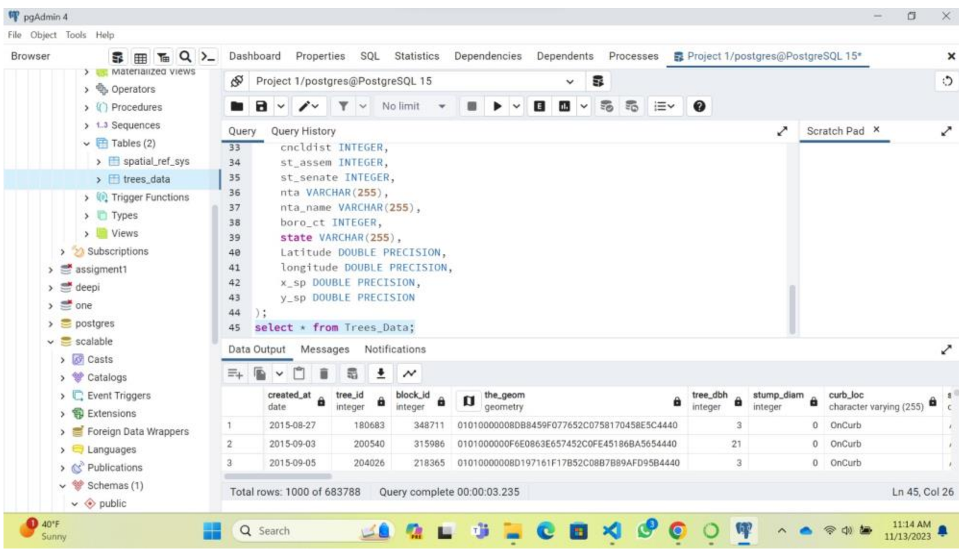
In these query the aggregating data from the trees_2005 table to count the total number of trees for each ZIP code (zipcode). The result set includes the ZIP code and the corresponding total number of trees. The GROUP BY clause is used to group the data by ZIP code.

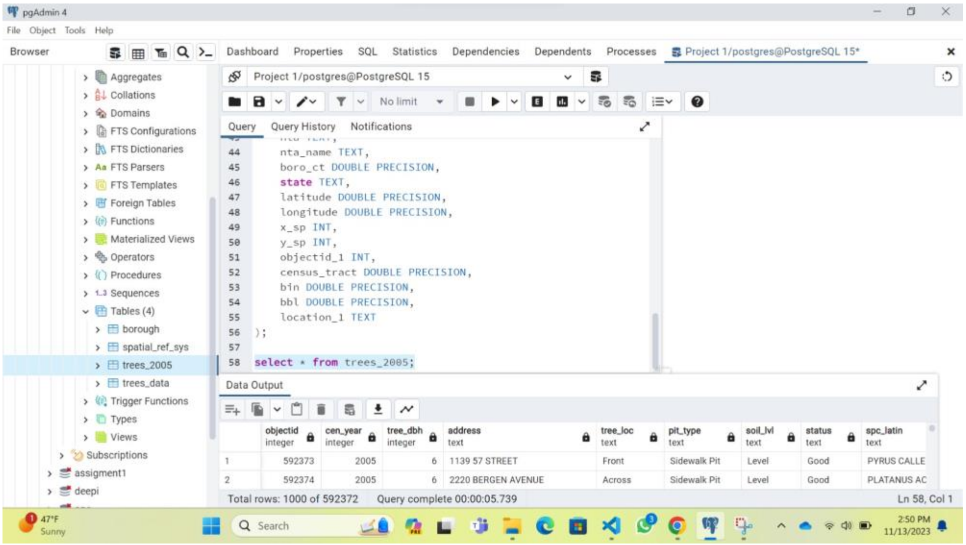
Always make sure to close the connection conn.close()

The images in postgrese are displayed here after the tables are created




pgAdmin 4
File Object Tools Help
Browser
Dashboard Properties SQL Statistics Dependencies Dependents Processes Project 1/post... Project 1/postgres@P
Project 1/postgres@PostgreSQL 15
No limit
Query History
CREATE TABLE borough (
the_geom geometry(MULTIPOLYGON, 4326),
BoroCode INTEGER,
BoroName VARCHAR(255),
Shape_Leng DOUBLE PRECISION,
Shape_Area DOUBLE PRECISION
);
select * from borough;
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (3)
borough
spatial_ref_sys
trees_data
Trigger Functions (2)
Types
Views
Subscriptions
assigment1
deepi
one
postgres
scalable
Casts
Catalogs
Process completed
Copying table data 'public.borough' on database 'Project 1' and server 'PostgreSQL 15 (localhost:5433)'
View Processes
Process started
Copying table data 'public.borough' on database 'Project 1' and server 'PostgreSQL 15 (localhost:5433)'
View Processes
Data Output Messages Notifications
the_geom geometry
borocode integer
Total rows: 0 of 0 Query complete 00:00
42°F Sunny
Search
11:47 AM 11/13/2023


pgAdmin 4
File Object Tools Help
Browser
Dashboard Properties SQL Statistics Dependencies Dependents Processes Project 1/postgres@PostgreSQL 15*
Project 1/postgres@PostgreSQL 15
No limit
Query Query History
Scratch Pad
33 cncldist INTEGER,
34 st_assem INTEGER,
35 st_senate INTEGER,
36 nta VARCHAR(255),
37 nta_name VARCHAR(255),
38 boro_ct INTEGER,
39 state VARCHAR(255),
40 Latitude DOUBLE PRECISION,
41 longitude DOUBLE PRECISION,
42 x_sp DOUBLE PRECISION,
43 y_sp DOUBLE PRECISION
44);
45 select * from Trees_Data;
Materialized Views
Operators
Procedures
Sequences
Tables (2)
spatial_ref_sys
trees_data
Trigger Functions
Types
Views
Subscriptions
assigment1
deepi
one
postgres
scalable
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas (1)
public
Data Output Messages Notifications
created_at date | tree_id integer | block_id integer | the_geom geometry | tree_dbh integer | stump_diam integer | curb_loc character varying (255)
1 | 2015-08-27 | 180683 | 348711 | 01010000008DB8459F077652C0758170458E5C4440 | 3 | 0 | OnCurb
2 | 2015-09-03 | 200540 | 315986 | 0101000000F6E0863E657452C0FE45186BA5654440 | 21 | 0 | OnCurb
3 | 2015-09-05 | 204026 | 218365 | 01010000008D197161F17B52C08B7B89AFD95B4440 | 3 | 0 | OnCurb
Total rows: 1000 of 683788 Query complete 00:00:03.235
Ln 45, Col 26
40°F Sunny
Search
11:14 AM 11/13/2023



Now we will do data visualization in these first we will do the bar chart for that we need to import mat plot and seaborn lib

Matplot lib is basically used for interactive visualization and for drawing the plots. With the help of matplot we can create bar charts, scatter plots and pie charts.

Seaborn is a statistical data visualization library in Python built on top of Matplotlib. Seaborn is particularly well-suited for visualizing complex datasets with multiple variables and is often used in conjunction with Pandas DataFrames.

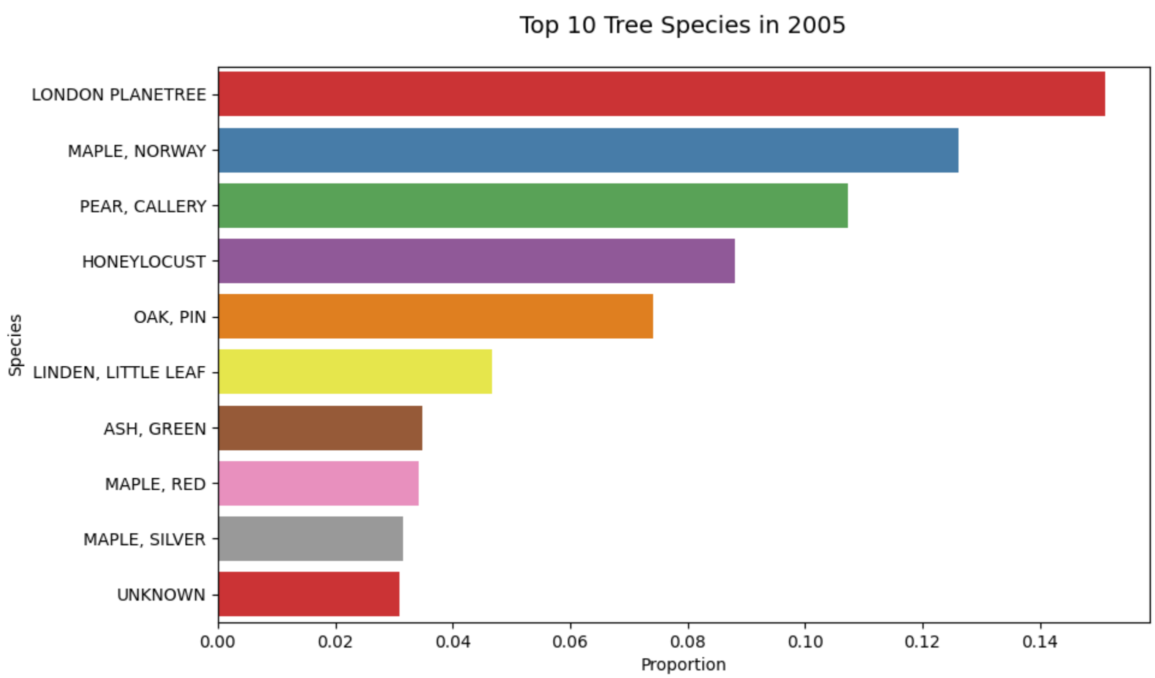
The first bar chart contains species and there proportion.
Normalize tree counts by tree species for df1

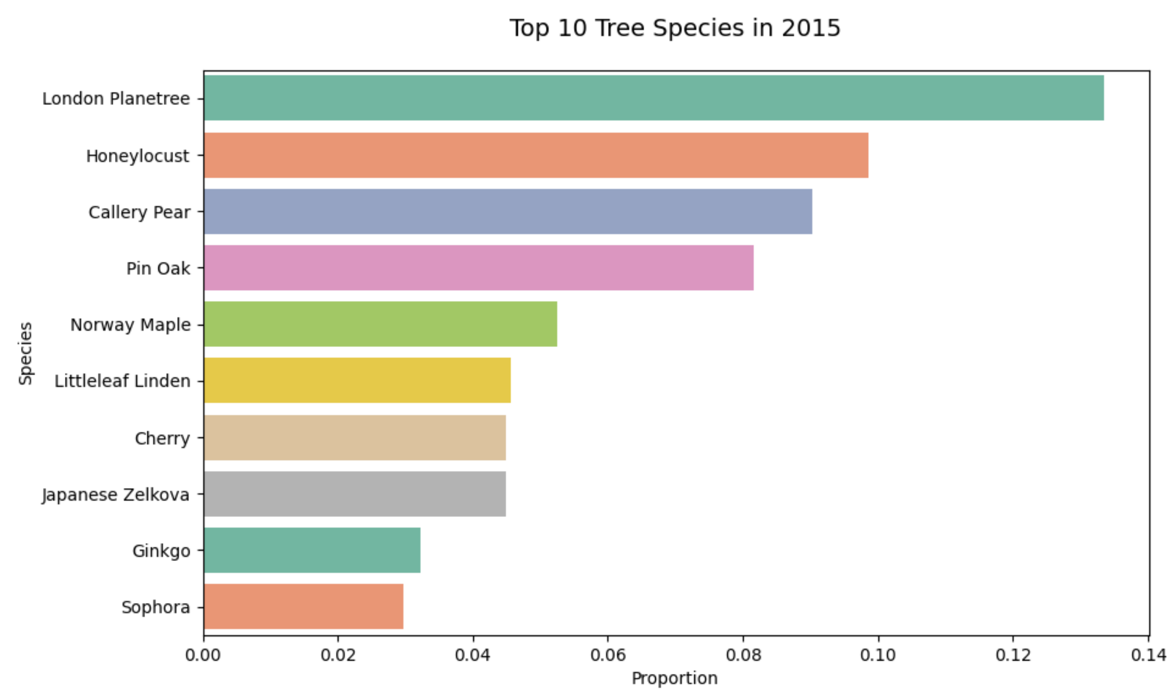
Plotting for df1 with different colors
Code:

```
plt.figure(figsize=(10, 6))
sns.barplot(data=top_species_df1.sort_values(by='proportion', ascending=False), x='proportion', y='species', palette='Set1')
plt.title('Top 10 Tree Species in 2005', fontsize=14, pad=20)
plt.xlabel('Proportion')
plt.ylabel('Species')
plt.show()
```

Plotting for df with different colors
Code:

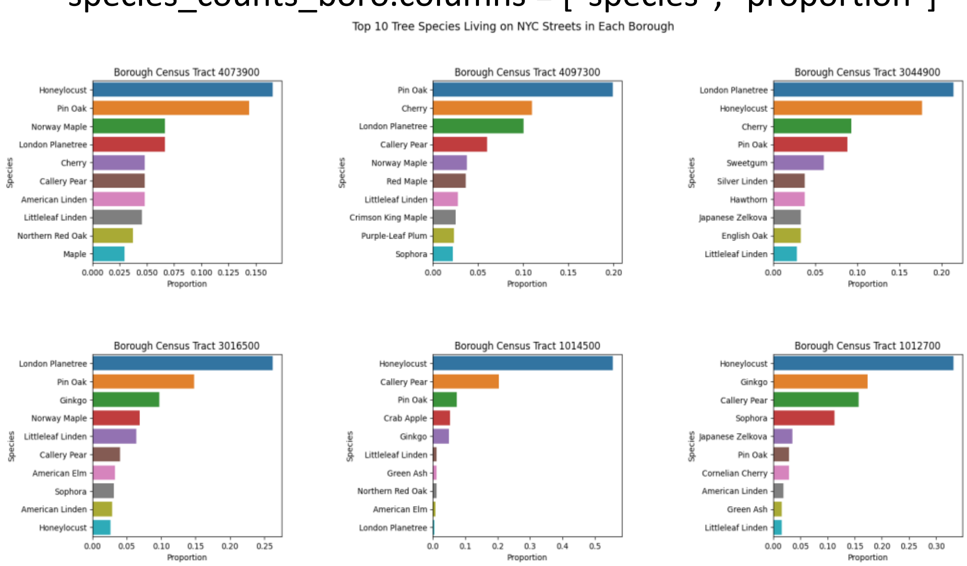
```
plt.figure(figsize=(10, 6))
sns.barplot(data=top_species_df.sort_values(by='proportion', ascending=False),
x='proportion', y='species', palette='Set2')
plt.title('Top 10 Tree Species in 2015', fontsize=14, pad=20)
plt.xlabel('Proportion')
plt.ylabel('Species')
plt.show()
```





Now we will create the subplots for each borough

```
for i, boro_ct in enumerate(unique_boro_cts[:6]):
    ax = plt.subplot(2, 3, i + 1)
    species_counts_boro = df2[df2['boro_ct'] ==
boro_ct]["spc_common"].value_counts(normalize=True,
dropna=True).reset_index()
    species_counts_boro.columns = ["species", "proportion"]
```



The statistics of all the trees are displayed .

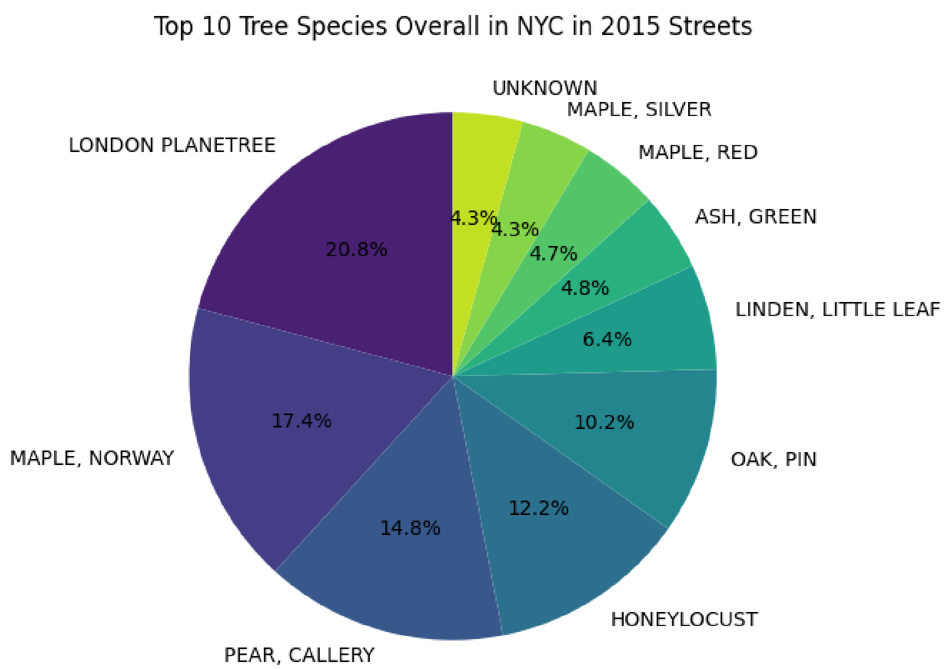
There are 169 different species of tree living on NYC streets.

The top 10 most common tree species represent 72.5% of the total NYC street trees population.
159 species make up the remaining 27.5% of the tree population.

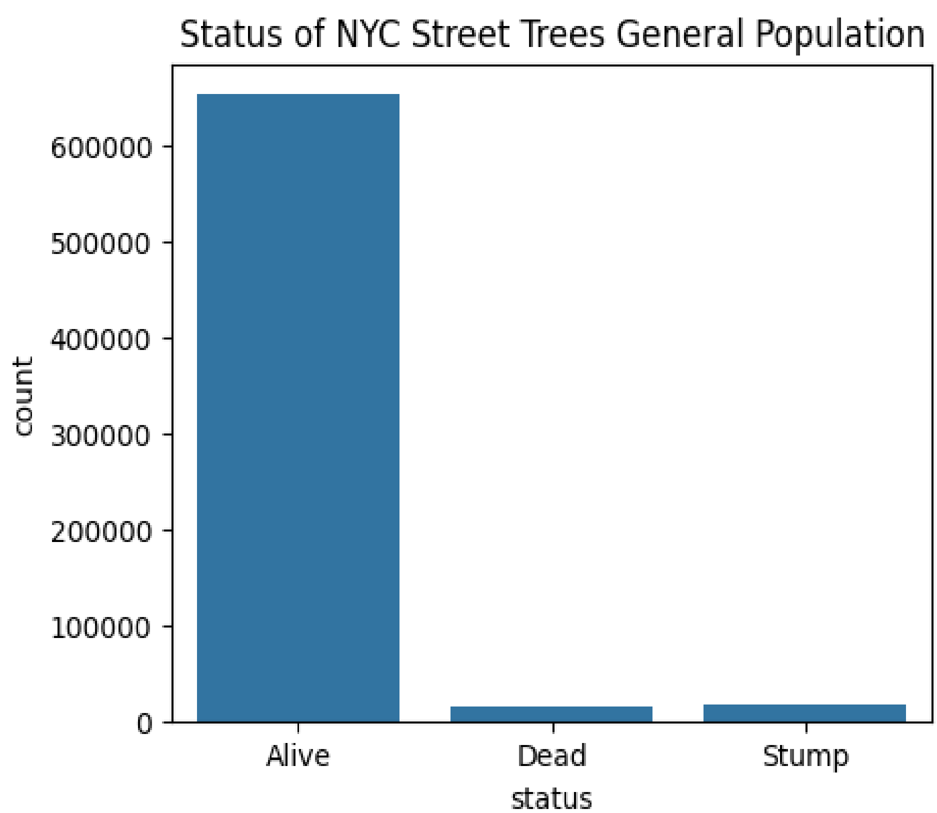
Overall, the most common species of tree is the LONDON PLANETREE.
15.11% of all NYC street trees are LONDON PLANETREE.
12.61% are MAPLE, NORWAY
10.74% are PEAR, CALLERY
8.81% are HONEYLOCUST
7.41% are OAK, PIN
4.67% are LINDEN, LITTLE LEAF
3.48% are ASH, GREEN
3.42% are MAPLE, RED
3.15% are MAPLE, SILVER
3.1% are UNKNOWN

In the borough census tract 4073900, the most common species of tree is the Honeylocust.
16.53% of all NYC street trees in 4073900 are Honeylocust.
14.4% are Pin Oak
6.67% are Norway Maple
6.67% are London Planetree
4.8% are Cherry
4.8% are Callery Pear
4.8% are American Linden
4.53% are Littleleaf Linden
3.73% are Northern Red Oak
2.93% are Maple

Now we have statistics visualization of pie chart



Top 10 Tree Species Overall in NYC in 2015 Streets
UNKNOWN
MAPLE, SILVER
MAPLE, RED
ASH, GREEN
LINDEN, LITTLE LEAF
OAK, PIN
HONEYLOCUST
PEAR, CALLERY
MAPLE, NORWAY
LONDON PLANETREE
20.8%
17.4%
14.8%
12.2%
10.2%
6.4%
4.8%
4.7%
4.3%
4.3%



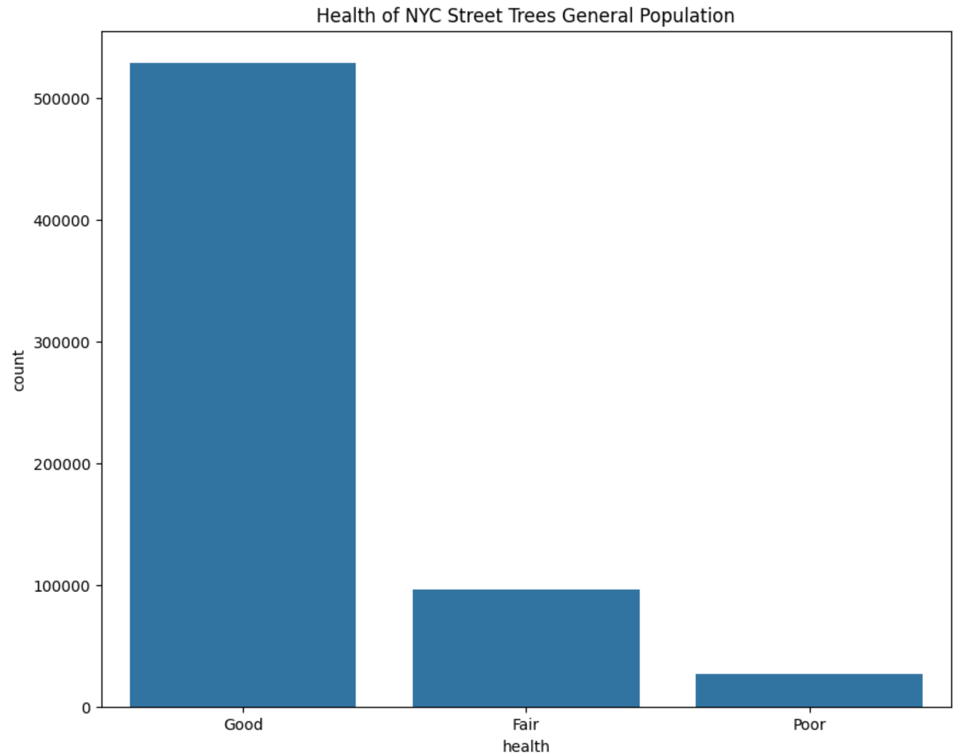
Status of NYC Street Trees General Population
count
0
100000
200000
300000
400000
500000
600000
Alive
Dead
Stump
status

These are the status of the trees using normalizing.
Alive 0.953765
Stump 0.025818
Dead 0.020417

normalized value counts for health of all trees.
health_counts = df2[df2.status=='Alive'].health.value_counts(normalize=True, dropna=True)
health_counts
Good 0.810906
Fair 0.147973
Poor 0.041121

normalized value counts for health of all trees by borough
health_counts_boro =
df2[df2.status=='Alive'].groupby('boro_ct')['health'].value_counts(normalize=True)
.unstack().sort_values(by='Good')
health_counts_boro

health	**Fair**	**Good**	**Poor**
boro_ct			
4091800	0.153846	0.153846	0.692308
3025902	0.714286	0.285714	NaN
1015601	0.548780	0.317073	0.134146
4099802	0.557143	0.342857	0.100000
4097202	0.656250	0.343750	NaN
...	...	...	...
1008602	NaN	1.000000	NaN
2031900	NaN	1.000000	NaN
4071600	NaN	1.000000	NaN
5001800	NaN	1.000000	NaN
4091601	0.333333	NaN	0.66666



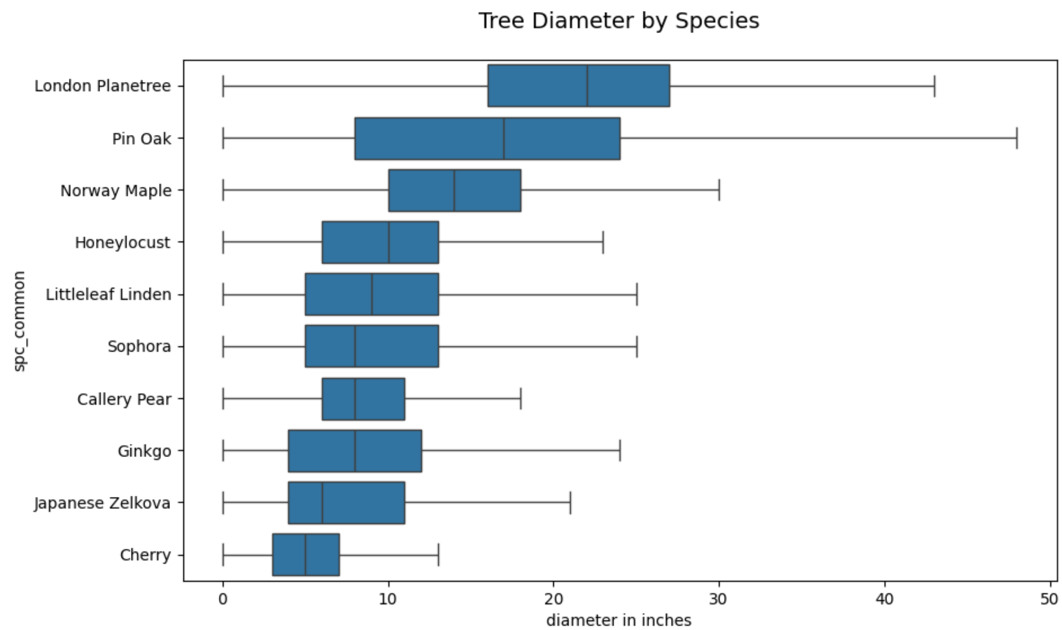
calculated mean tree diameter by species using pandas group by

```
pc_common
London Planetree     21.560657
Pin Oak              16.867707
Norway Maple         14.330516
Honeylocust          10.210958
Littleleaf Linden    10.045827
Sophora               9.254628
Callery Pear          8.958307
Ginkgo                8.625476
Japanese Zelkova      7.863559
Cherry                5.691041
Name: tree_dbh, dtype: float64
```

build a dataframe of statistics that describe the tree diameter by species
top_species_diam_report = pd.DataFrame(index=top_species_diameter.index, columns=['mean', 'var', 'std', '25%', '50%', '75%'])

mean	var	std	25%	50%	75%	
spc_common						
London Planetree	21.560657	81.958918	9.053116	16.0	22.0	27.0
Pin Oak	16.867707	102.227271	10.11075	8.0	17.0	24.0
Norway Maple	14.330516	35.995413	5.999618	10.0	14.0	18.0
Honeylocust	10.210958	27.01634	5.197724	6.0	10.0	13.0
Littleleaf Linden	10.045827	42.755473	6.538767	5.0	9.0	13.0
Sophora	9.254628	31.575953	5.619248	5.0	8.0	13.0
Callery Pear	8.958307	22.631097	4.757215	6.0	8.0	11.0
Ginkgo	8.625476	31.594492	5.620898	4.0	8.0	12.0
Japanese Zelkova	7.863559	28.063757	5.297524	4.0	6.0	11.0
Cherry	5.691041	16.401706	4.049902	3.0	5.0	7.0

Here we use subplot for tree diameter by species with spc_common and diameter in inches.



Open shapefiles with NYC Borough boundaries

	boro_code	boro_name	s shape_area	shape_leng	geometry
0	5.0	Staten Island	1.623621e+09	325917.353950	MULTIPOLYGON (((-74.05051 40.56642, -74.05047 ...
1	2.0	Bronx	1.187175e+09	463179.772813	MULTIPOLYGON (((-73.89681 40.79581, -73.89694 ...
2	4.0	Queens	3.041419e+09	888199.730955	MULTIPOLYGON (((-73.82645 40.59053, -73.82642 ...
3	1.0	Manhattan	6.365205e+08	357713.308660	MULTIPOLYGON (((-74.01093 40.68449, -74.01193 ...
4	3.0	Brooklyn	1.934138e+09	728148.532410	MULTIPOLYGON (((-73.86327 40.58388, -73.86381 ...

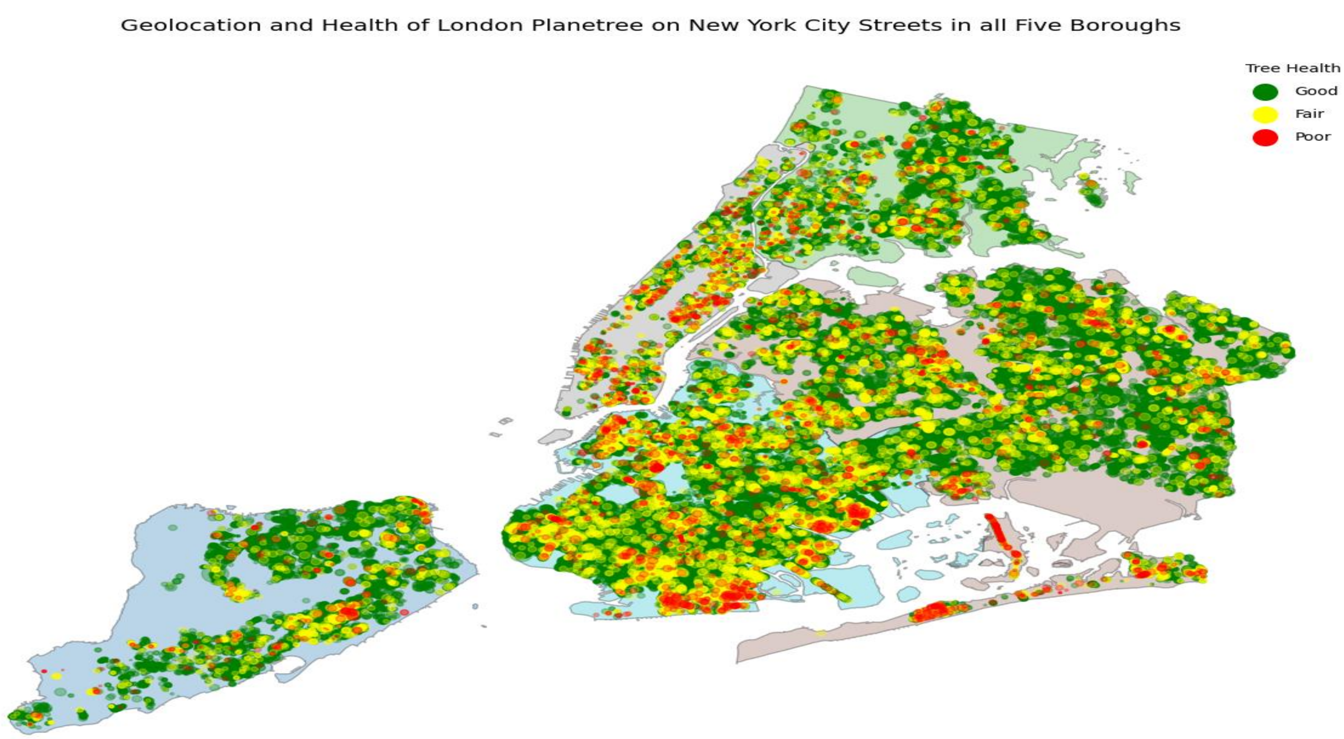
We use different colors for different region to know the difference easily

boro_names = borough_boundaries.boro_name.to_list()

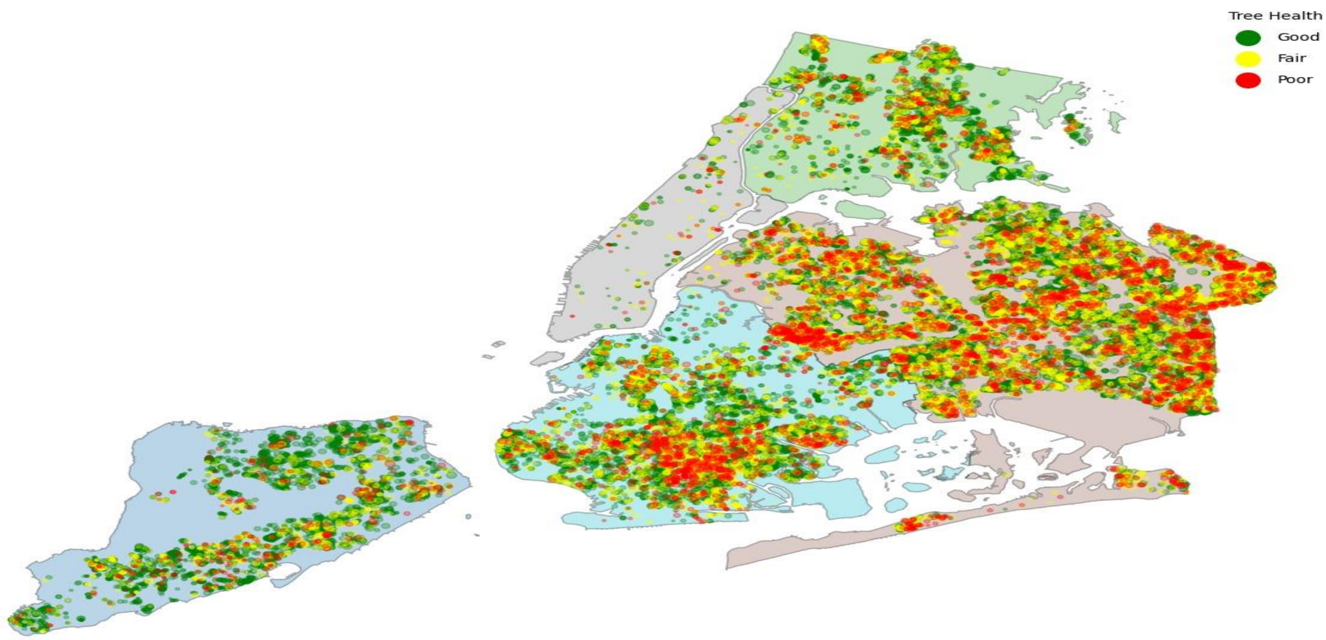
```
boro_colors = ['red', 'green', 'orange', 'blue', 'pink']
dict(list(zip(boro_names, boro_colors)))
```

```
Staten Island': 'red',
 'Bronx': 'green',
 'Queens': 'orange',
 'Manhattan': 'blue',
 'Brooklyn': 'pink'}
```

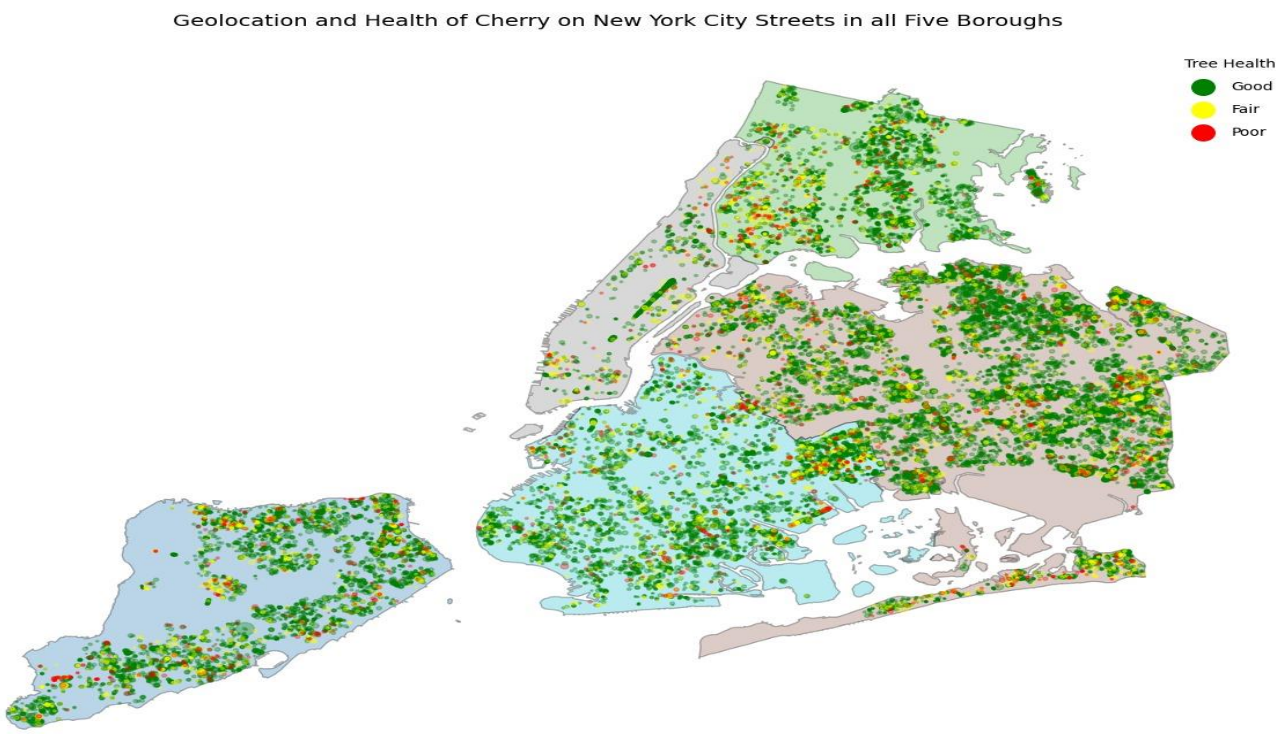
The geolocation of the health condition of planetree on new york street in all five doroughs



Geolocation and Health of Norway Maple on New York City Streets in all Five Boroughs



Geolocation and Health of Pin Oak on New York City Streets in all Five Boroughs



During these project we faced some difficulties like:

For cleaning the data it took a lot of time as the dataset are huge and fixing every column and filtering the data is problematic

During the visualizing subplot we use different techniques like enumerate for displaying the plots

Inserting the data with python to postgres is a bit problematic as some values are missing and some special characters.

The main problem occurred during the project is type casting during the context of latitude and longitude

Conclusion:

During this project we have come across some conclusions that from 2005 to 2015 the proportion of the trees have been gradually decreased but the London Plantree remains the highest proportion species in new York.

We can also see the status of the trees in this
Alive 0.953765
Stump 0.025818
Dead 0.020417

From this we can say that people are looking for the trees as they are mostly Alive and the proportion of decrease in trees from 2005 and 2015 not that much and people are becoming aware of the trees importance.

We also subplot using borough(location) to find the most proportion trees across all areas. The london Plantree and Honey locust are in a good portion in the graph.

There are only 4.3 % are the remaining trees in proportion other than top 10. I think we need to increase the total number of unknown trees and others which will help in the future.

Future aspects:

In this dataset we can see the geographical view of the trees. The health condition and there proration. we can take a weather dataset and see in what places the trees species are growing which will help us to plant those trees which are suitable for weather condition.

We can take the most polluted areas and compare the number of the trees which will help us to decrease the pollution and give people a healthy life.