311 Housing Issues¶

We're hoping to model the relationship between pests and missed garbage collection in NYC. For convenience, we'll assume the 311 Service Requests cover all requests made to the city for both complaints about pests and complaints about missed garbage collection.

Below is a query to find 311 requests that mention "pest" in their descriptor. Its purpose is to give you a template query for the following problems.

- Reference for LIKE
- Reference for functions dealing with strings like LOWER
- Reference for date_extract_y, this is unique to SoQL on this platform
- Reference for using AND in our WHERE statement

In [1]:
```
import requests
import pandas as pd
```

In [2]:
```
# url = "https://data.cityofnewyork.us/resource/erm2-nwe9.json"
# query = """
# SELECT
#     agency, agency_name,
#     complaint_type,
#     descriptor,
#     COUNT(unique_key) as cnt,
#     date_extract_m(created_date) as month
# WHERE
#     lower(descriptor) like '%pest%' AND
#     date_extract_y(created_date) = 2021 AND
#     month > 10
# GROUP BY
#     agency, agency_name, complaint_type, descriptor, month
# LIMIT
#     50000
# """
# params = {"$query": query}

# resp = requests.get(url=url, params=params)
# assert resp.status_code == 200
```

Assignment Project Exam Help
https://powcoder.com
Add WeChat powcoder

In [3]:
```
# df = pd.DataFrame(resp.json())
# df["cnt"] = df.cnt.astype(float)
# df.sort_values("cnt").tail(10)
```

Question 0 - getting the response variable¶

All references to time should be extracted from the `created_date` variable. I **highly** recommend you to limit the time range you're querying when you're still testing out your query, e.g. limit to a single year and month.

Using the **SoQL language similar to the example above**, please create a data frame, called `pest`, where each row corresponds to a different year, month, and borough. The columns should be the year, month, borough, and number of requests made. Please make sure the number of requests made is of type float.

Please make sure all data with the following constraints are included:

- The agency should be `'HPD'`
- The year should be 2021 or earlier (so the data likely remains static during this assignment)

- The lower-cased value from `complaint_type` should be `"unsanitary condition"`
- The lower-cased value from `descriptor` must contain the word `pest`
- You can assume that the number of data are less than 50000.

Do NOT use `pandas` functionalities to perform the aggregation.

In [4]:

```
### TEST FUNCTION: test_311_get_y
# DO NOT REMOVE THE LINE ABOVE
```

Question 1 - getting the independent variable¶

All references to time should be extracted from the `created_date` variable. I **highly** recommend you to limit the time range you're querying when you're still testing out your query, e.g. limit to a single year and month.

Using the **SoQL language similar to the example above**, please create a data frame, called `missed_collect`, where each row corresponds to a different year, month, and borough. The columns should be the year, month, borough, and number of requests made. Please make sure the number of requests made is of type float.

Please make sure all data with the following constraints are included:

- The agency should be `'DSNY'`
- The year should be 2021 or earlier (so the data likely remains static during this assignment)
- The lower-cased value from `complaint_type` must contain the word `miss` (this includes recycles and other variables which we will aggregate over)
- You can assume that the number of data are less than 50000.

Do NOT use `pandas` functionalities to perform the aggregation.

Assignment Project Exam Help

In [5]:

https://powcoder.com

```
### TEST FUNCTION: test_311_get_x
# DO NOT REMOVE THE LINE ABOVE
```

Question 2 - examine the data¶

Add WeChat powcoder

If you cannot solve the two problems above, please use `pest.csv` and `missed_collect.csv` to proceed.

For data in Manhattan, please use `seaborn.lineplot`, to plot the requests about pests and missed collections on the same plot.

- Make sure the two lines have different colors.
- Please make a data frame with both sets of data titled `mdf`
  - Make sure `mdf` has a column titled `"date"` for the x-axis and is a timestamp see `pandas.to_datetime()`
  - Make sure `mdf` only has data from Manhattan
  - Make sure `mdf` has a column titled `"counts"` for the y-axis
- You should set the different colored lines using the `hue` argument. The variable you pass to this from `mdf` should be called `"var"`.
- Your graph should clearly show which dataset has more available data

Hint: you may need to call `reset_index(inplace=True, drop=True)` if you encounter errors about duplicate index.

Side comment: in an exam you would be asked about interpreting this graph.

In [6]:

```
### TEST FUNCTION: test_plot_wrangle
# DO NOT REMOVE THE LINE ABOVE
```

In [7]:

Question 3 - creating a base model dataset¶

Please create a new data frame called `bdf` where we will model the relationship between the number of pest requests vs the number of missed collection requests.
Here are some specifications for `bdf`:

- It should have a column titled `"counts"` which are the number of pest requests
- It should have a column titled `"month"`
- It should have a column titled `"log_counts"` which are `log(1 + the number of pest requests)` where `log` is the natural log.
- It should have a column titled `lag0` which are the number of missed collection requests.
- All the variables above should be aligned by date, i.e. the data should be sourced from the same date. Please create a column named `lag1` which is similar to `lag0` except that it is one month behind. (e.g., if `lag0` corresponds to a record in Januray, 2010, then `lag1` is from February, 2010.)
- You should have an "indicator" column for each borough but one (which one you drop will not matter), i.e. the values in this column should be 1 if the record is from this borough and 0 otherwise. Their titles do not matter.
- You should have a column titled `year` that contains the year of the pest request counts.
- Each row should maintain the same interpretation as a record for a year, month, and borough.

WARNING, this can be very time consuming...it's best to assume some months are missing in 2020 for some boroughs, i.e. "shifting" the data is not advised for obtaining the `lag1`.

Assignment Project Exam Help

In [8]:

```
### TEST FUNCTION: test_model_wrangle
# DO NOT REMOVE THE LINE ABOVE
```

https://powcoder.com

In [9]:

Question 4 - Split the data¶

Add WeChat powcoder

Please use the file `hw4_q3.csv` if you want to move on for the following problems.
Please call `reset_index(inplace=True, drop=True)` on your data frame once more.
Please create a variable called `validate` that is a boolean series with values `True` corresponding to records with `year` being 2020. Please create a variable called `test` that is a boolean series with values `True` corresponding to records with `year` being 2021. Please create a variable called `train` that is a boolean series that is the complement of the two boolean series above.

In [10]:

```
### TEST FUNCTION: test_data_split
# DO NOT REMOVE THE LINE ABOVE
```

In [11]:

In [12]:

Question 5 - Choosing the best model¶

Let's fit several different models (finally) using the data points that correspond to `train` then predict on the `validate` records to calculate the root mean squared error, i.e. $\sqrt{\frac{1}{n}\sum_i|Y_i - \hat{Y}_i|^2}$. All the following model will be a ordinary linear regression that includes an intercept.

1. `counts` regressed on the all the borough columns (recall we dropped one), this is the baseline

2. `counts` regressed on `lag0` with all the borough columns (recall we dropped one)
3. `counts` regressed on `lag1` with all the borough columns (recall we dropped one)
4. `log_counts` regressed on `lag0` with all the borough columns (recall we dropped one)
5. `log_counts` regressed on `lag1` with all the borough columns (recall we dropped one)

For each model, please perform the appropriate transformation so the units for the RMSE are comparable across models. Please store the RMSEs in a list called `rmse` for the different models in the order listed above.

In [13]:

```
### TEST FUNCTION: test_models
# DO NOT REMOVE THE LINE ABOVE
```

In [14]:

In [15]:

Assignment Project Exam Help

https://powcoder.com

Add WeChat powcoder