PSET 2

For an additional 5% bonus (in aggregate, not for each section), we ask you to please answer the following question for each section.
We would like to hear how you used AI (if at all) on this section. Where did you find it useful/not useful? We are asking for knowledge-sharing purposes; your answer will not impact your grade on the assignment.

1. **PS2:** Due Sat Jan 24th at 5:00PM Central.

"This submission is my work alone and complies with the 30538 integrity policy." Add your initials to indicate your agreement: JK

Before you begin this problem set, please rename `ps2_template.qmd` as `ps2.qmd`.

Please submit correctly and rename ps2_template.qmd as ps2.qmd. (5%)

Read in and characterize data (30 %)

We would like to hear how you used AI (if at all) on this section. Where did you find it useful/not useful? We are asking for knowledge-sharing purposes; your answer will not impact your grade on the assignment.
Answer: I used AI to confirm how to code the time it takes to read files (we had code like that in Machine Learning but never definitely learned) I also used it to sharpen question 6 to specify the issue date range, especially using shape[0] to count the rows, which was syntax I couldn't quite remember.
1.

```
import pandas as pd
import time
import os
import altair as alt
alt.renderers.enable('png')
```

```
start_time = time.time()
park = pd.read_csv("parking_tickets_one_percent.csv")
end_time = time.time()
print(end_time - start_time)

assert len(park) == 287458, "Incorrect Row Count"
```

2.

```
file_size = os.path.getsize("parking_tickets_one_percent.csv") / 1048576
file_size
full_file_size = file_size * 100
full_file_size
```

3.

```
park
park.columns

#Index is Unnamed:0
#Chronological ordering is issue_date

park_500 = park.head(500)
park_500["issue_date"].is_monotonic_increasing
```

4.

```
def na_columns(park):
    park_na = park.isnull().sum().reset_index()
    park_na.columns = ["Variable", "NA Count"]
    return park_na

na_columns(park_500)
```

5.

Zip Code, Notice Level, and Hearing Disposition are missing the most

According to the articles we read, ZIP code was likely not registered because the ticket was not followed up with.

Notice Level was likely na because for many neighborhoods, no notice is sent to the motorist, their vehicle is just seized.

Hearing Dispotion was likely na because for many neighborhoods, compounding tickets were given to the same vehicle, so there was no contestation for the first few tickets.

6.

```
park_2017 = park[
    (park["issue_date"] >= "2017-01-01") &
    (park["issue_date"] < "2018-01-01")
].shape[0]
park_2017

park_2017*100
```

ProPublica offered that about 200,000 to 250,000 tickets are issued a year.

This implies that after the huge ticket fee spike, many more tickets were issued, which makes sense because of the unreasonable price.

Visual Encoding (10%)

We would like to hear how you used AI (if at all) on this section. Where did you find it useful/not useful? We are asking for knowledge-sharing purposes; your answer will not impact your grade on the assignment.
Answer: I had the idea of using a map to categorize the data series, but did not know how to

markdown, so used it to tweak my mapping to help mark it down.
1.

```
park_map = {'Unnamed: 0': 'N', 'ticket_number': 'N', 'issue_date': 'T',
 ↪  'violation_location': 'O',
       'license_plate_number': 'N', 'license_plate_state': 'N',
 ↪  'license_plate_type': 'N',
       'zipcode': 'Q', 'violation_code': 'Q', 'violation_description': 'O',
 ↪  'unit': 'O',
       'unit_description': 'O', 'vehicle_make': 'O', 'fine_level1_amount':
 ↪  'N',
       'fine_level2_amount': 'N', 'current_amount_due': 'N',
 ↪  'total_payments': 'Q',
       'ticket_queue': 'O', 'ticket_queue_date': 'N', 'notice_level': 'O',
       'hearing_disposition': 'O', 'notice_number': 'Q', 'officer': 'Q',
 ↪  'address': 'N'
     }

map_encode = pd.DataFrame(
    park_map.items(),
    columns=["Variable", "Encoding Type"]
)

print(map_encode.to_markdown(index=False))
```

Understanding the structure of the data and summarizing (20%)

We would like to hear how you used AI (if at all) on this section. Where did you find it useful/not useful? We are asking for knowledge-sharing purposes; your answer will not impact your grade on the assignment.
Answer: Did not use AI for this portion!
1.

No notice | V VIOL ———> Contest | | V V SEIZ Not Liable (Dismissed) | V DETR | V FINL | V DLS

Notice | V Hearing Req ————————-> Court | | V V Dismissed ———————— | | | V V V Paid Bankruptcy Define

If someone constests their ticket and is found not liable, then the process halts, both for notice_level and for ticket_queue and the owner pays no ticket.

Aggregating, transforming, visualizing the data (30%)

We would like to hear how you used AI (if at all) on this section. Where did you find it useful/not useful? We are asking for knowledge-sharing purposes; your answer will not impact your grade on the assignment.
Answer: I used AI to help me sort the bars of question 1 by frequency (sort="-y") I also used it for question 3 to fix the MaxRowsError, tweaking the datetime solution to accommodate the data. I also used it when I ran into problems subsetting the data and using the correct type to work with the heatmaps and lasagna maps.
1.

```
top_20 =
 ↪  (park["violation_description"].value_counts().head(20).reset_index())
top_20.columns = ["Violation Type", "Count"]
top_20

alt.Chart(top_20).mark_bar(color="teal").encode(
  alt.X("Violation Type:N", sort="-y", title="Type of Violation"),
  alt.Y("Count:Q", title="Total Violations"),
)
```

2.

```
park["ticket_queue"]
park["paid"] = (park["ticket_queue"] == "Paid")
paid_fraction = (park.groupby("vehicle_make")["paid"].mean().reset_index())
paid_fraction.columns = ["Vehicle Make", "Paid"]

alt.Chart(paid_fraction).mark_bar(color="teal").encode(
  alt.X("Vehicle Make:N", sort="-y", title="Type of Vehicle"),
  alt.Y("Paid:Q", title="Paid Tickets"),
)
```

#Some vehicles makes may be more or less likely to have tickets depending either on how expensive they are, or how prominently they are found in certain neighborhoods. #If a neighborhood is likely to be targeted more often, like minority neighborhoods, according to the article, and that neighborhood has more of a certain vehicle make, then they accrue more tickets, and pay less.

3.

```
park["issue_date"] = pd.to_datetime(park["issue_date"])
tickets_small =
 ↪  (park.groupby(park["issue_date"].dt.date).size().reset_index(name="count"))
tickets_small.columns = ["Issue Date", "Count"]

alt.Chart(park).mark_area(
    color="lightblue",
    interpolate='step-after',
    line=True
).encode(
    alt.X('Issue Date:T', title = "Issue Date"),
    alt.Y('Count:Q', title = "Number of Tickets Issued")
)
```

4.

```
ticket_by_day = (park.groupby([park["issue_date"].dt.month.rename("month"),
 ↪  park["issue_date"].dt.day.rename("day")]).size().reset_index(name="count"))
ticket_by_day["month"] = ticket_by_day["month"].astype(str)
ticket_by_day["day"] = ticket_by_day["day"].astype(str)

alt.Chart(ticket_by_day, title="Number of Tickets Issued by Month and
 ↪  Day").mark_rect().encode(
    alt.X("day:O").title("Day"),
    alt.Y("month:O").title("Month"),
    alt.Color("count:Q", title="Tickets Issued"),
    tooltip=[
        alt.Tooltip("month:O", title="Month"),
        alt.Tooltip("day:O", title="Day"),
        alt.Tooltip("count:Q", title="Tickets")
    ]
)
```

This used magnitude channel as well with color saturation, but also spatial position for the x/y position

5.

```
top_5 = park["violation_description"].value_counts().head(5).index.tolist()
top_5_park = park[park["violation_description"].isin(top_5)]
tickets_over_time = (top_5_park.groupby([top_5_park["issue_date"].dt.date,
 ↪  "violation_description"]).size().reset_index(name="count"))
tickets_over_time.columns = ["date", "violation", "count"]

color_condition = (
    alt.when(alt.expr.month("datum.value") == 1, alt.expr.date("datum.value")
 ↪  == 1)
    .then(alt.value("black"))
    .otherwise(alt.value(None))
)

alt.Chart(tickets_over_time, width=300, height=100).mark_rect().encode(
    alt.X("date:T")
        .title("Time")
        .axis(
            format="%Y",
            labelAngle=0,
            labelOverlap=False,
            labelColor=color_condition,
            tickColor=color_condition,
        ),
    alt.Y("violation:N").title(None),
    alt.Color("count:Q").title("Tickets Issued")
)
```

6.

#The pros of using the step chart is that it is very intuitive and sequential, but a con is difficult to differentiate between car types #The pros of using the heatmap is that it does a good job showing contrast between magnitudes, but a con is that it is difficult to compare between magnitudes if they are not abut. #The pros of using the lasagna map is that it is also fairly intuitive and can compare between models well, but a con is that it is limited in the data it shows.

7.

#In my opinion it is helpful to see the sheer magnitude of violations as a way of emphasizing how unevely they are distributed over time, so I think the filled step chart is best.