PS 2 - George Wang

1. This submission is my work alone and complies with the 30538 integrity policy. **GW**
2. I have uploaded the names of anyone I worked with on the problem set here **GW** (2 point)
3. Late coins used this pset: ***0*** Late coins left after submission: ***4***

Q1.1

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

start = time.time()
df = pd.read_csv('parking_tickets_one_percent.csv')
end = time.time()

duration = end - start

print("It take", duration, "seconds to read.")

assert len(df) == 287458, f"Expected 287458, but found {len(df)}"
```

```
It take 0.7287547588348389 seconds to read.
```

```
/var/folders/k2/prgbv7z97knbd104r93pncfc0000gp/T/ipykernel_26932/3173250861.py:7: DtypeWarnin

Columns (7) have mixed types. Specify dtype option on import or set low_memory=False.
```

Q1.2

```
def count_na(df):
    n = df.isna().sum().reset_index()
    n.columns = ['Variable', 'Num of NA']
    return n

na_table = count_na(df)
# na_table
```

There are many NA values in the following variables: license__plate__state, license__plate__type, zipcode, unit, notice__level, and hearing__disposition. Variables (hearing__disposition, notice__level, and zipcode) are missing much more frequently.

- hearing__disposition: Many missing values of hearing disposition means that many minor violations (e.g. prohibited parking, exprired registration, stop sign) do not result in hearing. People usually just pay tickets without disputation, so hearing is not required, thus NA value in the data set.
- notice__level: Many violation might not go trhough multiple notice levels. Peple may just resolve the case by paying tickets or being dismissed without further notice. Thus, some missing values occur in the data set.
- -zipcode: If a person's zipcode is not required for certain cases or there's an oversight when collecting it, missing values could occur. People may also unwilling to reveal zip code due to privacy concern.

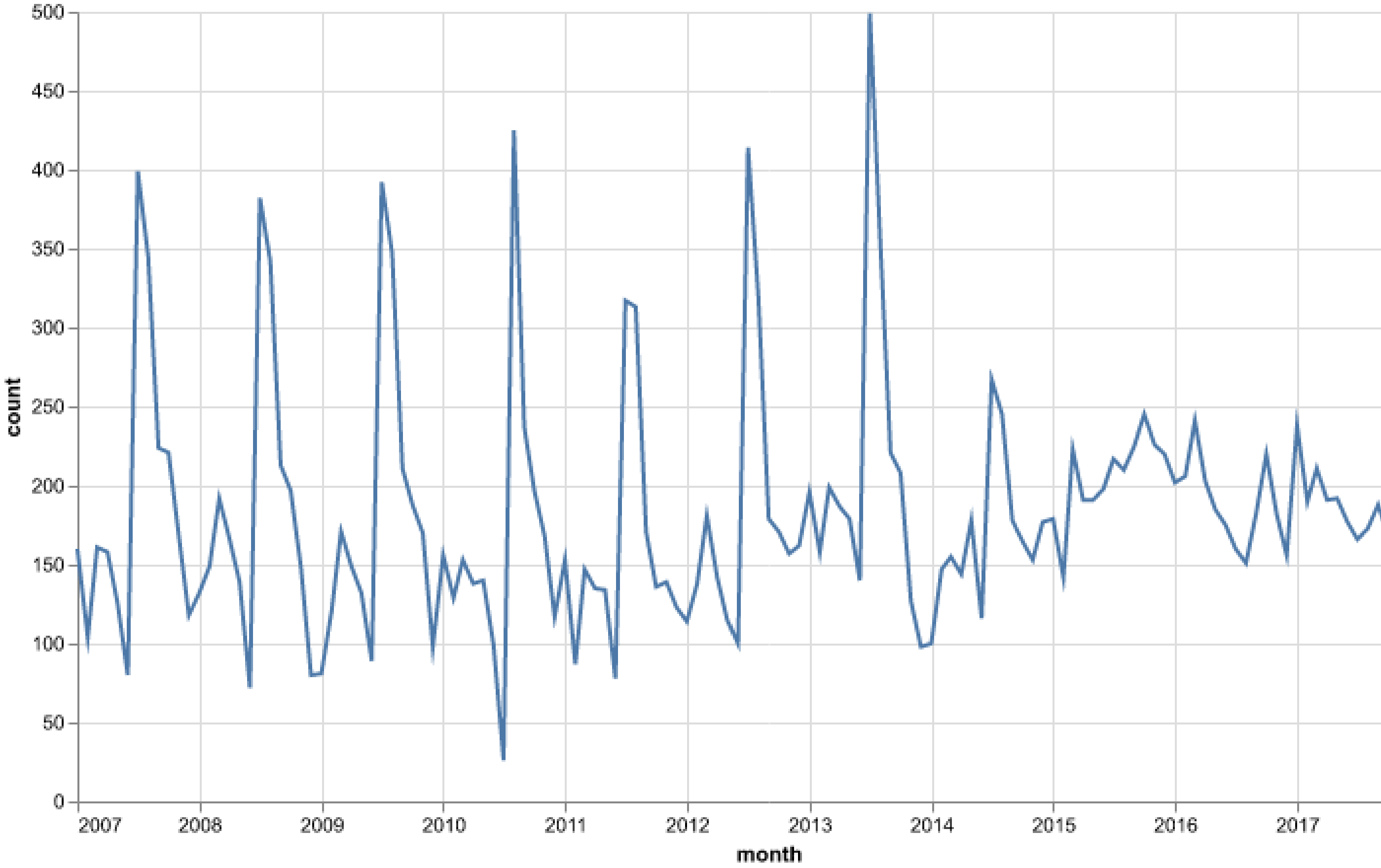
Q1.3 Original code: 0964125 New: 0964125B

Q1.4 Original fine level: $120 New fine level: $200

Q2.1

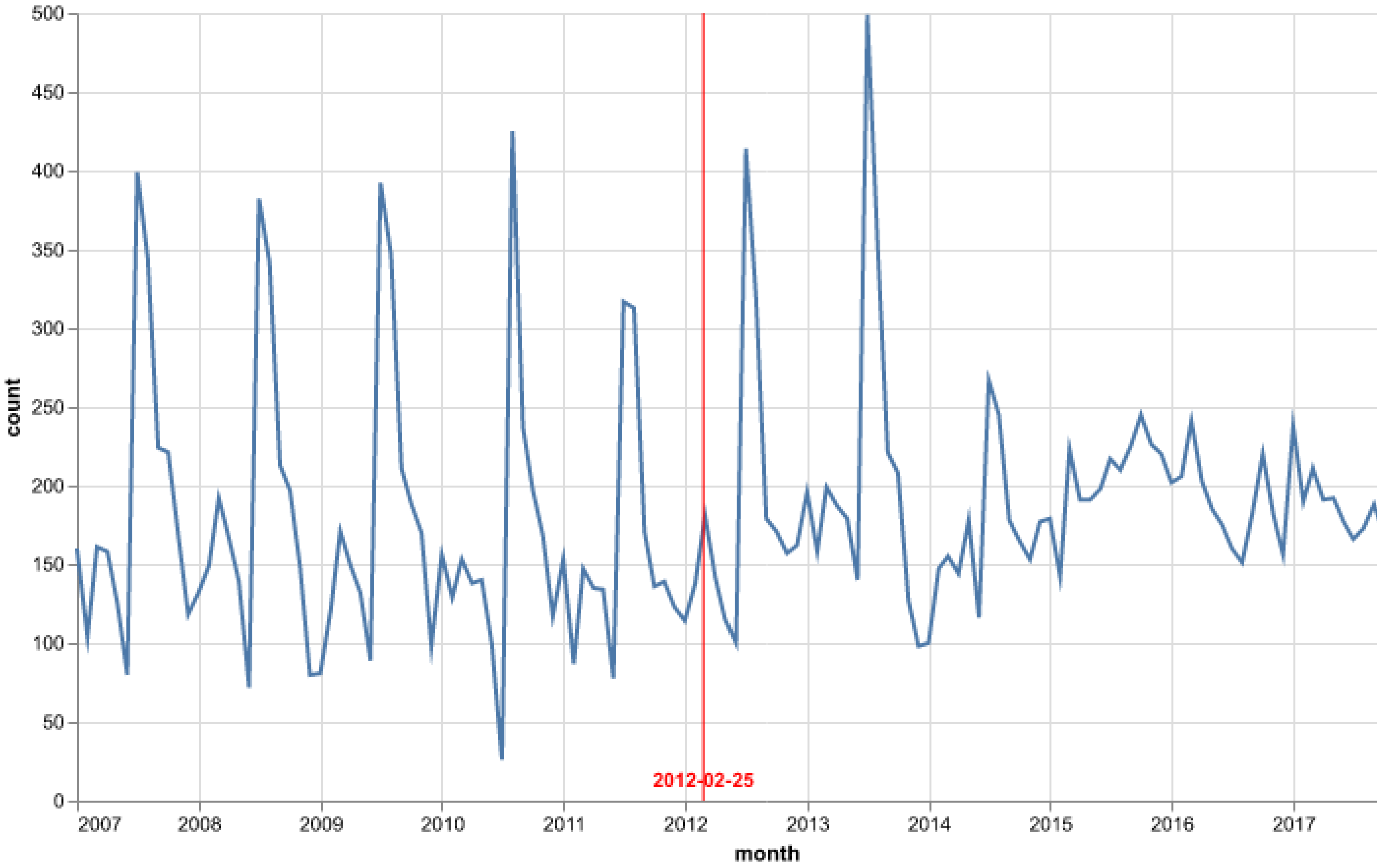
```
df['violation_code'] = df['violation_code'].replace({'0964125': '0964125ALL', '0964125B': '09

df['issue_date'] = pd.to_datetime(df['issue_date'])

df_missing_sticker = df[df['violation_code'] == '0964125ALL']

df_missing_sticker['month'] = df_missing_sticker['issue_date'].dt.to_period('M').astype(str)

df_missing_sticker_by_month = df_missing_sticker.groupby('month').size().reset_index(name='co

line = alt.Chart(df_missing_sticker_by_month).mark_line().encode(
    x='month:T',
    y='count:Q'
).properties(
    width=700,
    height=400
)
line
```

```
/var/folders/k2/prgbv7z97knbd104r93pncfc0000gp/T/ipykernel_26932/1297832246.py:7: SettingWit


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guid
```



Q2.2

```
label = alt.Chart(pd.DataFrame({'month': ['2012-02-25'], 'label': ['2012-02-25']})).mark_tex
    align='center',
    dy=190,
    color='red',
    fontSize=10,
    fontWeight='bold'
).encode(
    x='month:T',
    text='label:N'
)

# Update 'x' encoding to match the 'month' format used in the line chart
vline = alt.Chart(pd.DataFrame({'month': ['2012-02-25']})).mark_rule(color='red').encode(
    x='month:T'
)

chart = line + label + vline

chart
```



I used this page on labels: https://altair-viz.github.io/user_guide/marks/text.html.

Q2.3

```
sample_ratio = 0.01
fine_level1_prior = 120
fine_level1_post = 200

df_missing_sticker_2011 = df_missing_sticker.loc[df['issue_date'].dt.year == 2011]
df_missing_sticker_2012 = df_missing_sticker.loc[df['issue_date'].dt.year == 2012]

total_fine_count_2011 = df_missing_sticker_2011.shape[0]

prior_annual_revenue = total_fine_count_2011 / sample_ratio * fine_level1_prior
prior_annual_revenue

post_annual_revenue = total_fine_count_2011 / sample_ratio * fine_level1_post
post_annual_revenue
```

```
print('Total ticket num for missing sticker:', total_fine_count_2011)
print("Revenue increase =", post_annual_revenue-prior_annual_revenue)
```

```
Total ticket num for missing sticker: 1933
Revenue increase = 15464000.0
```

Q2.4

```
#2011
paid_tickets_2011 = df_missing_sticker_2011[df_missing_sticker_2011['ticket_queue'] == 'Paid
total_tickets_2011 = df_missing_sticker_2011.shape[0]

repayment_rate_2011 = paid_tickets_2011 / total_tickets_2011
print('2011 Payment rate:', repayment_rate_2011)

#2012
paid_tickets_2012 = df_missing_sticker_2012[df_missing_sticker_2012['ticket_queue'] == 'Paid
total_tickets_2012 = df_missing_sticker_2012.shape[0]

repayment_rate_2012 = paid_tickets_2012 / total_tickets_2012
print('2012 Payment rate:', repayment_rate_2012)

revenue_2012 = total_fine_count_2011 * repayment_rate_2012 * fine_level1_post / sample_ratio
revenue_2012

revenue_2011 = total_fine_count_2011 * repayment_rate_2011 * fine_level1_prior / sample_rati
revenue_2011

print('Considering payment rate, the revenue increase is', revenue_2012-revenue_2011)
```

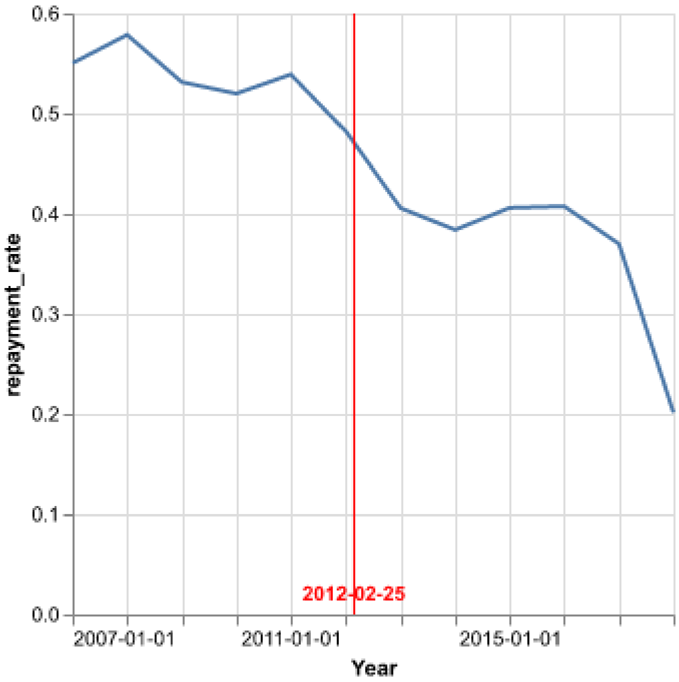
```
2011 Payment rate: 0.5390584583548887
2012 Payment rate: 0.4822080291970803
Considering payment rate, the revenue increase is 6138162.408759121
```

The repayment rate decrease from 0.54 to 0.48 (6% decrease). The revenue increase from to 12,504,000 to 18,642,162 ($6,138,162 increase).

Q2.5

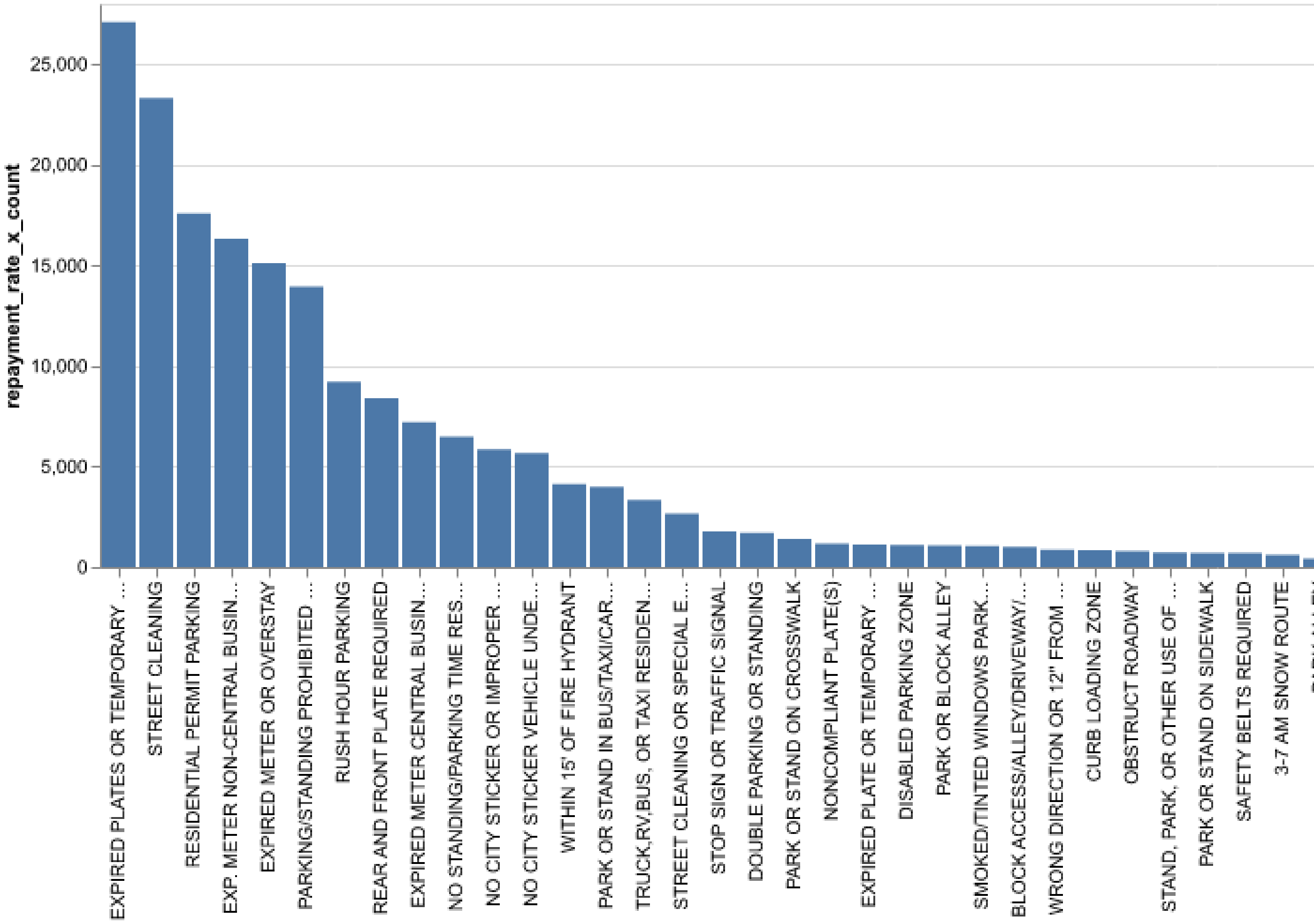
```
df_missing_sticker['year'] = df_missing_sticker['issue_date'].dt.year

repayment_rates = df_missing_sticker.groupby('year').apply(
    lambda x: (x['ticket_queue'] == 'Paid').sum() / len(x)
).reset_index(name='repayment_rate')

repayment_rates['date'] = pd.to_datetime(repayment_rates['year'], format='%Y')

vline = alt.Chart(pd.DataFrame({'date': ['2012-02-25']})).mark_rule(color='red').encode(
    x=alt.X('date:T', axis=alt.Axis(format='%Y-%m-%d'))
)

repayment_rates_line_chart = alt.Chart(repayment_rates).mark_line().encode(
    x=alt.X('date:T', title='Year'),
    y='repayment_rate:Q'
)

label = alt.Chart(pd.DataFrame({'month': ['2012-02-25'], 'label': ['2012-02-25']})).mark_tex
    align='center',
    dy=140,
    color='red',
    fontSize=10,
    fontWeight='bold'
).encode(
    x='month:T',
    text='label:N'
)

repayment_rates_chart = repayment_rates_line_chart + vline + label

repayment_rates_chart
```

```
/var/folders/k2/prgbv7z97knbd104r93pncfc0000gp/T/ipykernel_26932/678825342.py:1: SettingWith


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guid
```



Q2.6

```
ticket_by_type = df.groupby('violation_description').size().reset_index(name='count').sort_va


repayment_rate_by_type = df.groupby('violation_description').apply(
    lambda x: ((x['ticket_queue'] == 'Paid').sum() / len(x))
).reset_index(name='repayment_rate').sort_values(by='repayment_rate', ascending=False)

merged_df_by_type = pd.merge(ticket_by_type, repayment_rate_by_type, on='violation_descriptio

merged_df_by_type['repayment_rate_x_count'] = merged_df_by_type['repayment_rate'] * merged_di

merged_df_by_type.sort_values(by='repayment_rate_x_count', ascending=False)

chart_top_3_violations = alt.Chart(merged_df_by_type).mark_bar().encode(
    x=alt.X('violation_description:O', sort='-y'),
    y='repayment_rate_x_count:Q'
)
chart_top_3_violations
```



Although some violations have 100% repayments rate, the number of ticket of those types might not be high enough. Thus, I only compare the repayment rate * count (or the number of paid tickets). After sorting values, we see 'EXPIRED PLATES OR TEMPORARY REGISTRATION', 'STREET CLEANING', and 'RESIDENTIAL PERMIT PARKING' are the top 3 violations with highest number of paid tickets.

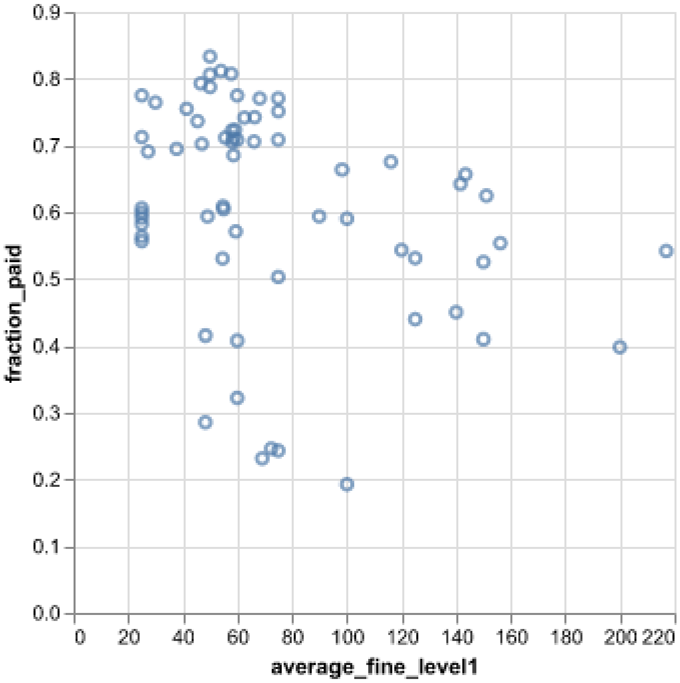
Q3.1

```
df_grouped = df.groupby('violation_description').apply(
    lambda x: pd.Series({
        'fraction_paid': (x['ticket_queue'] == 'Paid').mean(),
        'average_fine_level1': x['fine_level1_amount'].mean(),
        'count': len(x)
    })
).reset_index()

# Sort the DataFrame by the total number of tickets issued (count)
df_grouped_sorted = df_grouped.sort_values(by='count', ascending=False)

# Display the rows for the 5 most common violation descriptions
df_top_5_common = df_grouped_sorted.head(5)
print(df_top_5_common)
```

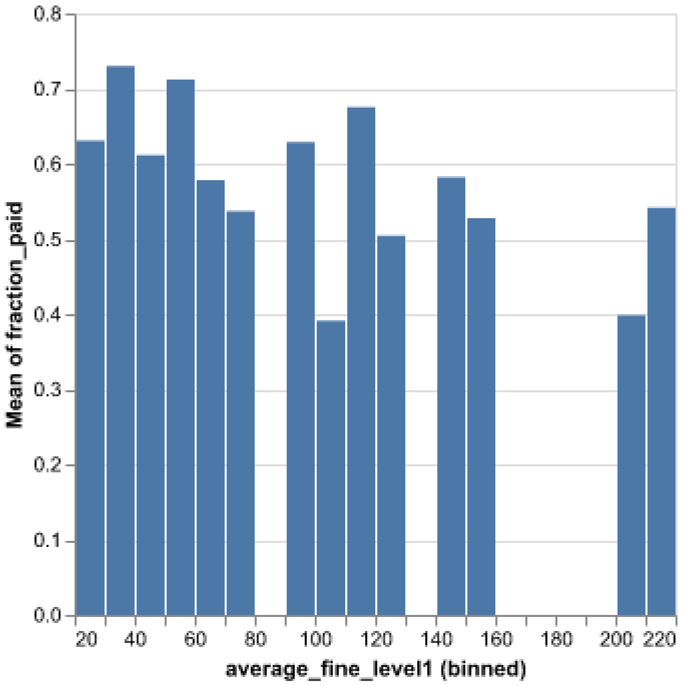
```
                        violation_description  fraction_paid  \
23   EXPIRED PLATES OR TEMPORARY REGISTRATION       0.604361
101                           STREET CLEANING       0.811612
90                   RESIDENTIAL PERMIT PARKING       0.742262
19   EXP. METER NON-CENTRAL BUSINESS DISTRICT       0.792913
81        PARKING/STANDING PROHIBITED ANYTIME       0.705817

     average_fine_level1    count
23             54.968869  44811.0
101            54.004249  28712.0
90             66.338302  23683.0
19             46.598058  20600.0
81             66.142864  19753.0
```

Q3.2

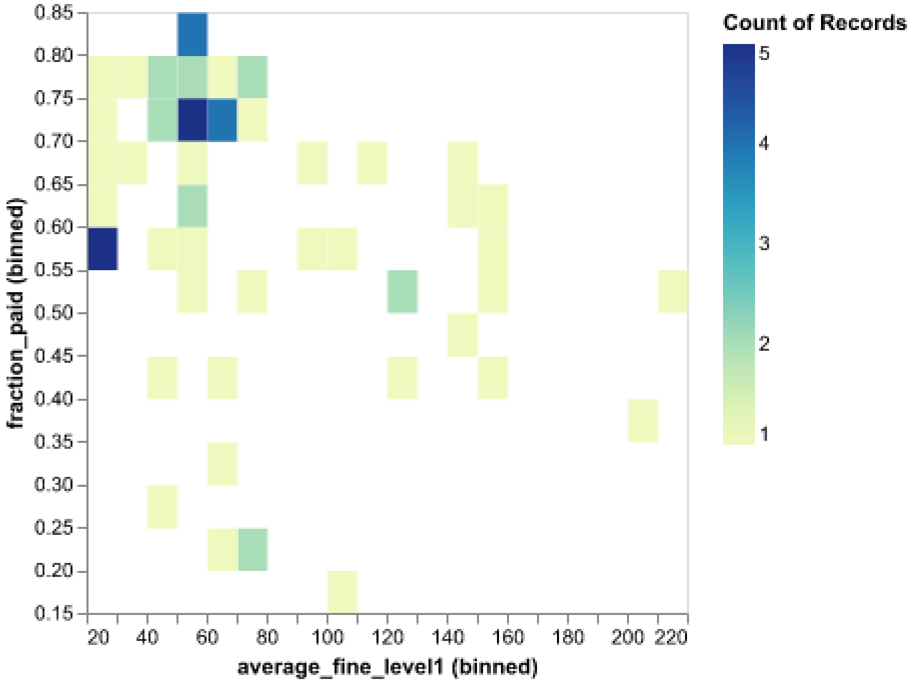
```
violation_counts = df.groupby('violation_description').size()
df_grouped_sorted = df_grouped_sorted[df_grouped_sorted['violation_description'].isin(violat

df_grouped_sorted = df_grouped_sorted[df_grouped_sorted['average_fine_level1'] != 500]

scatter_fine_x_fraction = alt.Chart(df_grouped_sorted).mark_point().encode(
    x=alt.X('average_fine_level1:Q'),
    y='fraction_paid:Q'
)

bar_fine_x_fraction = alt.Chart(df_grouped_sorted).mark_bar().encode(
    x=alt.X('average_fine_level1:Q', bin=alt.Bin(maxbins=20)),  # Binned average fine
    y='mean(fraction_paid):Q'  # Mean fraction paid per bin
)

heatmap_fine_x_fraction = alt.Chart(df_grouped_sorted).mark_rect().encode(
    x=alt.X('average_fine_level1:Q', bin=alt.Bin(maxbins=20)),
    y=alt.Y('fraction_paid:Q', bin=alt.Bin(maxbins=20)),
    color='count()'
)
scatter_fine_x_fraction
```



```
bar_fine_x_fraction
```



```
heatmap_fine_x_fraction
```



Headline: The scatter plot shows that when the average fine amount increases, the fraction of tickets paid tends to decrease. Sub-message: Most data points clustered between fine levels of 20 and 60, where most people pay the tickets.

Bar Chart Headline: Moderate Fines Show Higher Repayment Fraction Sub-message: There are some gaps (when fine levels are 80-90, 130-140, 160-200), probably due to data collection issues.

Heatmap: Headline: Most tickets have high fraction of repayment and low fine level. Sub-message: There seems to be a negative relationship between fraction of repayment rate and fine level.

Q3.3 I will choose the thir chart (heatmap) because it contain another layer of information through color. Compared to other charts, the heatmap is more intuitive for people to see where data entreies are clustered. It also shows the seemingly negative correlation between fraction of payment rate and fine levels.

Q4.1

```
df_grouped = df.groupby('violation_description').apply(
    lambda x: pd.Series({
        'original_fine': x['fine_level1_amount'].mean(),
        'unpaid_fine': x['fine_level2_amount'].mean(),  #
        'count': len(x)
    })
).reset_index()

df_grouped['fine_increase_ratio'] = df_grouped['unpaid_fine'] / df_grouped['original_fine']

df_not_double = df_grouped[(df_grouped['fine_increase_ratio'] != 2) & (df_grouped['count'] >=

df_not_double['fine_increase_amount'] = df_not_double['unpaid_fine'] - df_not_double['origina

# Display the results
print(df_not_double)
```

```
                       violation_description  original_fine  unpaid_fine  \
5       BLOCK ACCESS/ALLEY/DRIVEWAY/FIRELANE     141.592780   266.751108
15                     DISABLED PARKING ZONE     216.986234   358.308751
42  NO CITY STICKER VEHICLE OVER 16,000 LBS.     500.000000   955.343511
54   OBSTRUCTED OR IMPROPERLY TINTED WINDOWS     156.180812   225.645756
62                       PARK OR BLOCK ALLEY     150.000000   259.926829
79                  PARK/STAND ON BICYCLE PATH     143.432203   278.601695
95     SMOKED/TINTED WINDOWS PARKED/STANDING     151.090159   209.516794
```

```
    count  fine_increase_ratio  fine_increase_amount
5   1579.0             1.883932            125.158328
15  2034.0             1.651297            141.322517
42   131.0             1.910687            455.343511
54   271.0             1.444773             69.464945
62  2050.0             1.732846            109.926829
79   236.0             1.942393            135.169492
95  1697.0             1.386700             58.426635
```

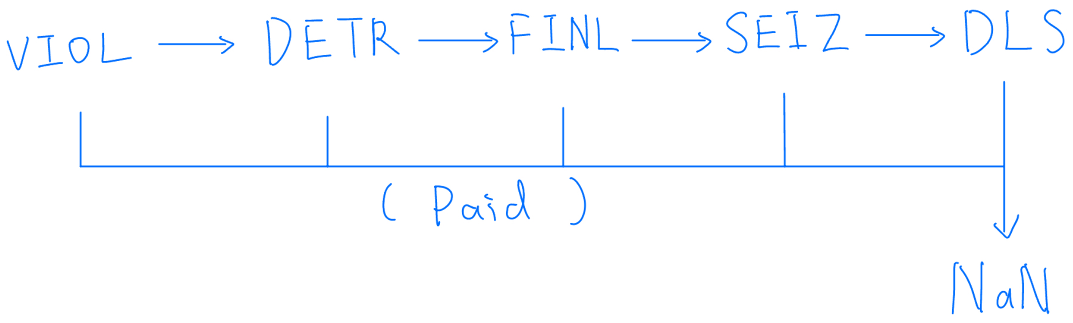
```
/var/folders/k2/prgbv7z97knbd104r93pncfc0000gp/T/ipykernel_26932/2039520114.py:13: SettingWi


A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide
```

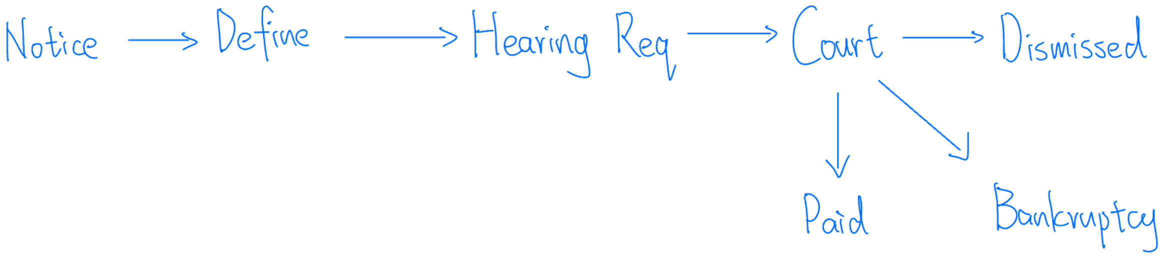
No, it does not hold for all violations. Blocking Access to an Alley, Driveway, or Firelane: The fine increases by $125.16 when unpaid. Parking in a Disabled Parking Zone: The fine increases by $141.32 when unpaid. Failure to Display City Sticker for Vehicles Over 16,000 lbs: The fine increases by $455.34 when unpaid. Obstructed or Improperly Tinted Windows: The fine increases by $69.46 when unpaid. Parking or Blocking an Alley: The fine increases by $109.93 when unpaid. Parking or Standing on a Bicycle Path: The fine increases by $135.17 when unpaid.

Q4.2

```python
print(df['notice_level'].unique())
print(df[df['ticket_queue'] == 'Paid']['notice_level'].unique())

print(df['ticket_queue'].unique())
```

```
['DETR' 'VIOL' 'SEIZ' nan 'FINL' 'DLS']
['DETR' 'VIOL' nan 'FINL' 'SEIZ' 'DLS']
['Paid' 'Notice' 'Define' 'Dismissed' 'Bankruptcy' 'Court' 'Hearing Req']
```

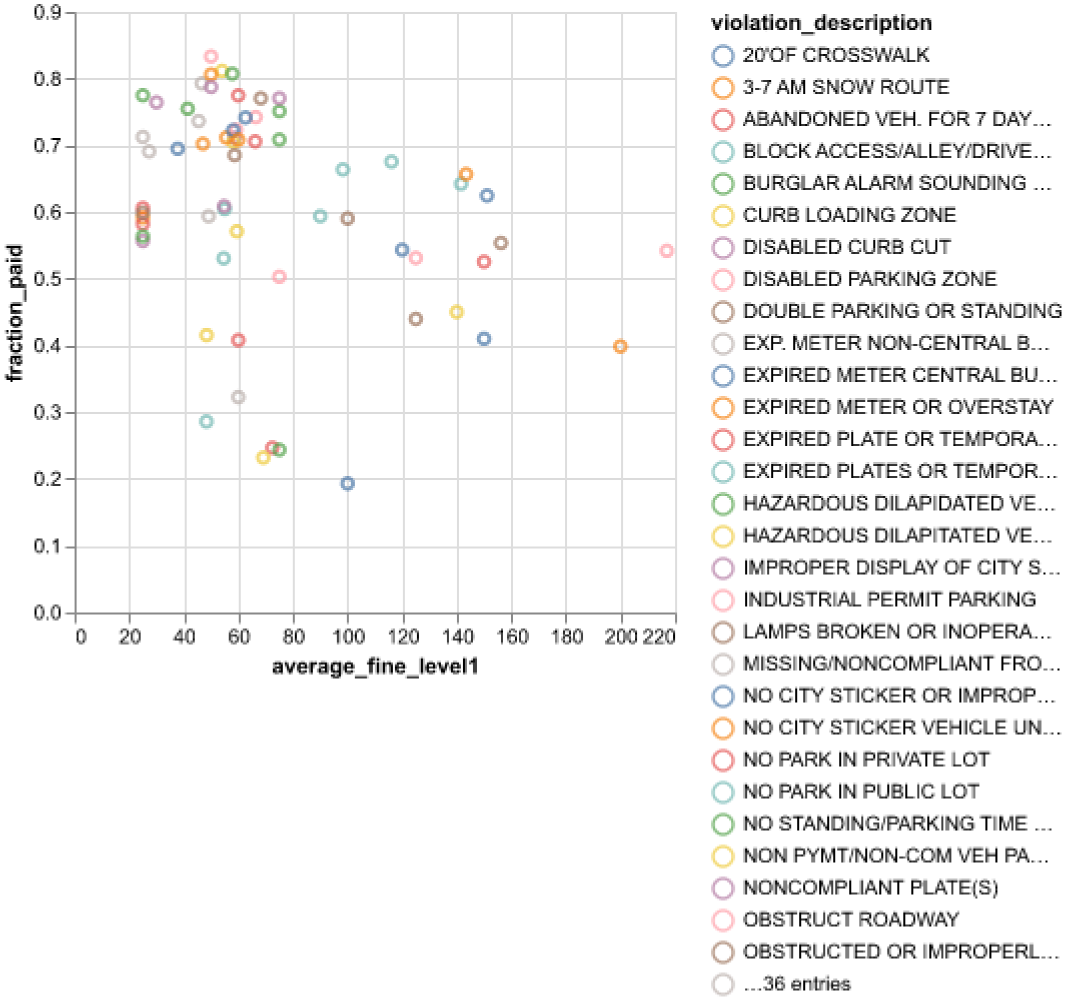




If someone contests their ticket and is found not liable, then notice_level will be NaN, and ticket_queue will be dismissed.

Q4.3

```
scatter_fine_x_fraction

## Chart A
text_labels_1 = scatter_fine_x_fraction.mark_text(
    align='left',
    dx=5,
    dy=-5
).encode(
    text='violation_description:N'  # Label each point with the violation description
)

chart_with_labels = scatter_fine_x_fraction + text_labels_1
chart_with_labels

# Chart B
scatter_fine_with_legend = alt.Chart(df_grouped_sorted).mark_point().encode(
    x=alt.X('average_fine_level1:Q'),
    y='fraction_paid:Q',
    color='violation_description:N'  # Use violation description as the legend
)
scatter_fine_with_legend
```

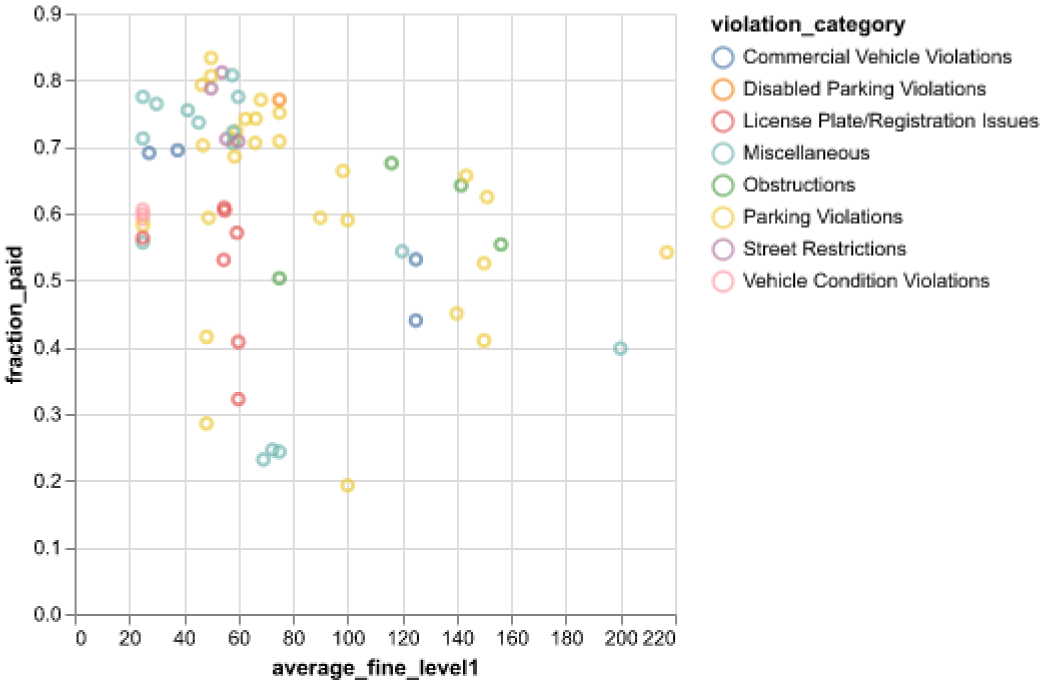


Revising

```
# Option: choosing top 10
top_10_violations = df_grouped_sorted.nlargest(10, 'count')['violation_description']

df_grouped_sorted['violation_label'] = df_grouped_sorted['violation_description'].apply(
    lambda x: x if x in top_10_violations.values else 'Other'
```

```
)

scatter_fine_x_fraction_top10 = alt.Chart(df_grouped_sorted).mark_point().encode(
    x=alt.X('average_fine_level1:Q'),
    y='fraction_paid:Q',
    color='violation_label:N'
)

scatter_fine_x_fraction_top10

# Option: relabel all types
print(print(df['violation_description'].unique()))

def categorize_violation(description):
    if any(keyword in description for keyword in ['PARK', 'STAND', 'METER']):
        return 'Parking Violations'
    elif any(keyword in description for keyword in ['PLATE', 'REGISTRATION']):
        return 'License Plate/Registration Issues'
    elif any(keyword in description for keyword in ['SNOW ROUTE', 'RUSH HOUR', 'CLEANING']):
        return 'Street Restrictions'
    elif 'DISABLED' in description:
        return 'Disabled Parking Violations'
    elif any(keyword in description for keyword in ['BLOCK', 'OBSTRUCT', 'HYDRANT']):
        return 'Obstructions'
    elif any(keyword in description for keyword in ['WINDOWS', 'LAMPS', 'CRACKED']):
        return 'Vehicle Condition Violations'
    elif any(keyword in description for keyword in ['TRUCK', 'BUS', 'TRAILER']):
        return 'Commercial Vehicle Violations'
    else:
        return 'Miscellaneous'

# Apply the categorization to the violation descriptions
df_grouped_sorted['violation_category'] = df_grouped_sorted['violation_description'].apply(ca

# Create a scatter plot with these meaningful categories
scatter_fine_x_fraction_category = alt.Chart(df_grouped_sorted).mark_point().encode(
    x=alt.X('average_fine_level1:Q'),
    y='fraction_paid:Q',
    color='violation_category:N'  # Use the new violation category for color
)

scatter_fine_x_fraction_category
```

```
['RESIDENTIAL PERMIT PARKING' 'PARKING/STANDING PROHIBITED ANYTIME'
 'EXPIRED PLATES OR TEMPORARY REGISTRATION' "WITHIN 15' OF FIRE HYDRANT"
 '3-7 AM SNOW ROUTE' 'REAR AND FRONT PLATE REQUIRED'
 'PARK OR STAND IN BUS/TAXI/CARRIAGE STAND' 'DISABLED PARKING ZONE'
 'NO CITY STICKER OR IMPROPER DISPLAY' 'OUTSIDE METERED SPACE'
 'OBSTRUCT ROADWAY' 'DOUBLE PARKING OR STANDING'
 'TRUCK,RV,BUS, OR TAXI RESIDENTIAL STREET'
 'SMOKED/TINTED WINDOWS PARKED/STANDING' 'PARK OR BLOCK ALLEY'
 'RUSH HOUR PARKING' 'STREET CLEANING OR SPECIAL EVENT'
 'EXP. METER NON-CENTRAL BUSINESS DISTRICT'
 'EXPIRED METER CENTRAL BUSINESS DISTRICT'
 'WINDOWS MISSING OR CRACKED BEYOND 6' 'PARK OR STAND ON SIDEWALK'
 'SAFETY BELTS REQUIRED' 'TRUCK,MOTOR HOME, BUS BUSINESS STREET'
 'NO STANDING/PARKING TIME RESTRICTED'
 'BLOCK ACCESS/ALLEY/DRIVEWAY/FIRELANE' 'PARK OR STAND ON PARKWAY'
 'ABANDONED VEH. FOR 7 DAYS OR INOPERABLE' 'HAZARDOUS DILAPITATED VEHICLE'
 'PARK OR STAND ON CROSSWALK' 'DISABLED CURB CUT'
 'STOP SIGN OR TRAFFIC SIGNAL' "WRONG DIRECTION OR 12'' FROM CURB"
 'CURB LOADING ZONE' 'TRUCK TRAILOR/SEMI/TRAILER PROHIBITED'
 'NONCOMPLIANT PLATE(S)' 'PARK IN FIRE LANE'
 'BURGLAR ALARM SOUNDING OVER 4 MINUTES'
 'PARK VEHICLE SOLE PURPOSE OF DISPLAYING FOR SALE'
 'OBSTRUCTED OR IMPROPERLY TINTED WINDOWS'
 'PARK OR STAND IN VIADUCT/UNDERPASS' 'LAMPS BROKEN OR INOPERABLE'
 'NO PARK IN PRIVATE LOT' 'Special Events' 'NO PARKING IN LOOP'
 'PARK IN CITY LOT WHEN CLOSED' 'FAIL TO PAY OR OUTSIDE SPACE IN CITY LOT'
 'PROPER FRONT AND REAR BUMPERS REQUIRED' 'PARK OR STAND ON CITY PROPERTY'
 'PARK VEHICLE TO GREASE OR REPAIR' 'UNSAFE CONDITION' 'DISABLED PARKING'
 "SNOW ROUTE: 2' OF SNOW OR MORE"
 'PARK OR STAND NEAR FIRE STATION OR RR XX'
 'PARK OR STAND ON CHA PROPERTY' "TWO HEAD LAMPS REQUIRED VISIBLE 1000'"
 'PARK/STAND ON BICYCLE PATH' 'USE OF SIREN/BELL/WHISTLE PROHIBITED'
 'INDUSTRIAL PERMIT PARKING' 'PARKED/STANDING UNATTENDED W/MOTOR RUNNI'
 'NO OR IMPROPER MUFFLER' "REAR PLATE LIT AND LEGIBLE FOR 50'"
 'NO PARK IN PUBLIC LOT' 'FRONT PLATE REQUIRED FOR TRUCK TRACTORS'
 'UNDER FIRE ESCAPE' 'NO OPERATOR SIGNAL' "20'OF CROSSWALK"
 'OUTSIDE DIAGONAL MARKINGS' 'NO DISPLAY OF BACK-IN PERMIT'
 'EXCESS FUMES/SMOKE DURING OPERATION'
 "PARK OR STAND ON OR WITHIN 10' RR TRACKS"
 'SAFETY BELTS REQUIRED ON SCHOOL BUS' 'PARK OR STAND ON BRIDGE'
 'IMPROPER LAMP FOR PARKED VEH ON UNLIT ST'
 "RED REAR LAMP REQUIRED VISIBLE 500'" 'FAIL TO DISPLAY TV NEWS PERMIT'
 "VEH 6' OR HIGHER WITHIN 20' OF CROSSWALK"
 'REAR PLATE REQUIRED MOTORCYCLE/TRAILER'
 'TWO RED REAR TRAILER REFLECTORS REQUIRED'
 'PARK IN CITY LOT OVER 30 DAYS' 'PARK OR STAND WITHIN INTERSECTION'
 'STREET CLEANING' 'PROJECTING LOAD (LEFT OR RIGHT SIDE)'
 'PARK OUTSIDE METERED SPACE' 'COMMERCIAL IDENTIFICATION ETC. REQUIRED'
 'SPECIAL EVENTS RESTRICTION' 'EXPIRED METER OR OVERSTAY'
 'STAND, PARK, OR OTHER USE OF BUS LANE'
 'IMPROPER LAMPS NON-MOTOR VEHICLE' 'PARK VEHICLE TO SELL MERCHANDISE'
 'REAR VIEW MIRROR REQUIRED' 'PROJECTING LOAD (REAR)'
 "SNOW ROUTE: 2'' OF SNOW OR MORE"
 "2 REAR TRAILER LAMPS REQ'D VISIBLE 500'"
 'BACK-UP LAMP LIT DURING OPERATION' "MOTORCYCLE HEAD LAMP VISIBLE 500'"
 'PARK/STAND IN WRIGLEY BUS PERMIT ZONE'
 'PARK MOTORCYCLE/SCOOTER PARK AT 90 DEGREE ANGLE' 'THEATER ENTRANCE/EXIT'
 'INVALID PLACARD' 'NO CITY STICKER VEHICLE UNDER/EQUAL TO 16,000 LBS.'
 'NO CITY STICKER VEHICLE OVER 16,000 LBS.'
 'IMPROPER DISPLAY OF CITY STICKER'
 'EXCESSIVE DIESEL POWERED VEHICLE ENGINE RUNNING' 'PARK ALLEY'
 'BRAKES REQUIRED DURING OPERATION'
 'HAND BRAKES:PROPER STOPPING CAPABILITY'
 'IMPROPER SIDE COWL/FENDER LAMPS' 'SERVICE BRAKES:STOPPING CAPABILITY'
 'TRUCK OR SEMI-TRAILER PROHIBITED'
 'BRAKES REQUIRED IN GOOD WORKING ORDER' 'SUSPENSION MODIFIED BEYOND 3'
 'HORN REQUIRED DURING OPERATION' 'BLOCK ALLEY'
 'HAZARDOUS DILAPIDATED VEHICLE' 'DEPR./DIMMED LAMPS'
 'EXPIRED PLATE OR TEMPORARY REGISTRATION'
 'NON PYMT/NON-COM VEH PARKED IN COM LOADING ZONE'
 'MISSING/NONCOMPLIANT FRONT AND/OR REAR PLATE'
 'MORE THAN FOUR FRONT MOUNTED LAMPS']
None
```



Q5.1

```
df_extra = df.groupby('violation_code')['violation_description'].nunique().reset_index()

df_extra = df_extra[df_extra['violation_description'] > 1]

df['most_common_violation_description'] = df.groupby('violation_code')['violation_descriptio
        lambda x: x.mode()[0]
    )

top_3_codes = df[df['violation_code'].isin(df_extra['violation_code'])].groupby('violation_co
top_3_codes
```

```
violation_code
0964040B       32082
0964125ALL     25004
0976160A       16853
dtype: int64
```

```
df_extra = df.groupby('violation_code')['violation_description'].nunique().reset_index()

df_extra = df_extra[df_extra['violation_description'] > 1]
```

```
df['most_common_violation_description'] = df.groupby('violation_code')['violation_descriptio
        lambda x: x.mode()[0]
    )

top_3_codes = df[df['violation_code'].isin(df_extra['violation_code'])].groupby('violation_co
top_3_codes
```

```
violation_code
0964040B      32082
0964125ALL    25004
0976160A      16853
dtype: int64
```