0.0.1 Question 0

Question 0A What is the granularity of the data (i.e. what does each row represent)?

I think this dataset is not very fine in granularity (not smooth). It includes only limited parameters like time and weather, but there are still many more other conditions may affect the bike sharing market.

Question 0B For this assignment, we'll be using this data to study bike usage in Washington D.C. Based on the granularity and the variables present in the data, what might some limitations of using this data be? What are two additional data categories/variables that you can collect to address some of these limitations?

It may not be able to show the relationship between bike sharing and traffic system. I would add the average traffic time and public transportation occupancy to it.

0.0.2 Question 2

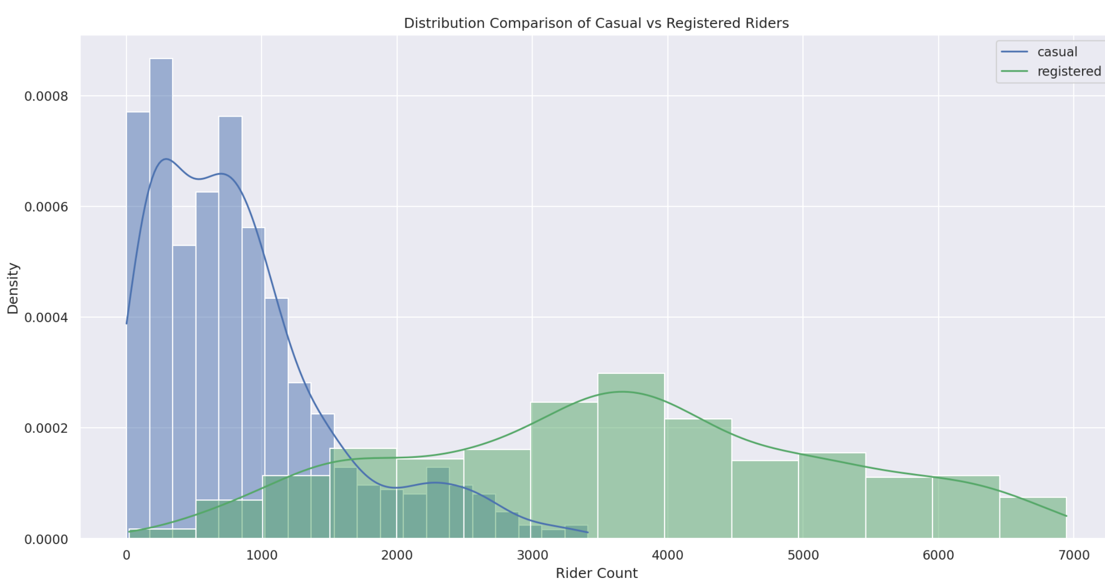
Question 2a Use the `sns.histplot` function to create a plot that overlays the distribution of the daily counts of bike users, using blue to represent `casual` riders, and green to represent `registered` riders. The temporal granularity of the records should be daily counts, which you should have after completing question 1c.

Hint: You will need to set the `stat` parameter appropriately to match the desired plot.

Include a legend, xlabel, ylabel, and title. Read the seaborn plotting tutorial if you're not sure how to add these. After creating the plot, look at it and make sure you understand what the plot is actually telling us, e.g on a given day, the most likely number of registered riders we expect is ~4000, but it could be anywhere from nearly 0 to 7000.

```
In [16]: sns.histplot(daily_counts, x ='casual', color = 'b', kde = True, stat = 'density')
         sns.histplot(daily_counts, x ='registered', color = 'g', kde = True, stat = 'density')


         plt.xlabel('Rider Count')
         plt.ylabel('Density')
         plt.title('Distribution Comparison of Casual vs Registered Riders')
         plt.legend(['casual','registered'])
```

```
Out[16]: <matplotlib.legend.Legend at 0x7f0ca9d8dcd0>
```



0.0.3 Question 2b

In the cell below, descibe the differences you notice between the density curves for casual and registered riders. Consider concepts such as modes, symmetry, skewness, tails, gaps and outliers. Include a comment on the spread of the distributions.

Casual: 1)modes around 300. 2)Asymmetry 3)skew to right 4)long tail 5)small gap 6)few outliers Registered: 1)modes around 3600 2)Symmerty 3)No skewness 4)tail both sides 5)large gap(spread) 6)few outliers Registered is about normally distributed, while casual riders are Obviously concentrated on left.

0.0.4 Question 2c

The density plots do not show us how the counts for registered and casual riders vary together. Use `sns.lmplot` to make a scatter plot to investigate the relationship between casual and registered counts. This time, let's use the `bike` DataFrame to plot hourly counts instead of daily counts.

The `lmplot` function will also try to draw a linear regression line (just as you saw in Data 8). Color the points in the scatterplot according to whether or not the day is a working day (your colors do not have to match ours exactly, but they should be different based on whether the day is a working day).

There are many points in the scatter plot, so make them small to help reduce overplotting. Also make sure to set `fit_reg=True` to generate the linear regression line. You can set the `height` parameter if you want to adjust the size of the `lmplot`.

Hints: * Checkout this helpful tutorial on `lmplot`.

- You will need to set `x`, `y`, and `hue` and the `scatter_kws` in the `sns.lmplot` call.
- You will need to call `plt.title` to add a title for the graph.

```
In [17]: # Make the font size a bit bigger
         sns.set(font_scale=1)
         sns.lmplot(x = 'casual', y = 'registered', data = bike, hue = 'workingday', scatter_kws={"s": 
         plt.xlabel('casual')
         plt.ylabel('registered')
         plt.title('Comparison of Casual vs Registered Riders on Working and Non-working Days')
```

```
Out[17]: Text(0.5, 1.0, 'Comparison of Casual vs Registered Riders on Working and Non-working Days')
```


Comparison of Casual vs Registered Riders on Working and Non-working Days
registered
casual
workingday
no
yes
0
200
400
600
800
1000
0
50
100
150
200
250
300
350

0.0.5 Question 2d

What does this scatterplot seem to reveal about the relationship (if any) between casual and registered riders and whether or not the day is on the weekend? What effect does overplotting have on your ability to describe this relationship?

Registered Riders are more likely to use bike on workingdays, while casual riders are more evenly distributed throughout the whole week. The overplotting in range 0-100 makes it difficult to see what happened here because all plots overlapping each other.

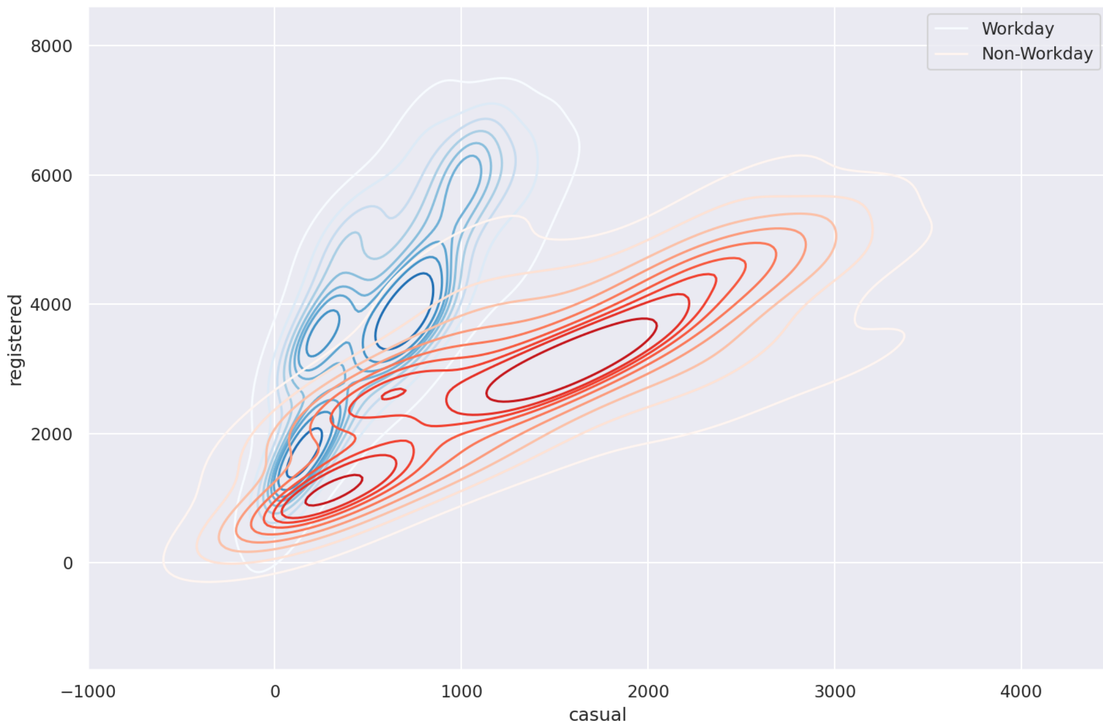
Generating the plot with weekend and weekday separated can be complicated so we will provide a walkthrough below, feel free to use whatever method you wish if you do not want to follow the walkthrough.

Hints: * You can use `loc` with a boolean array and column names at the same time * You will need to call kdeplot twice, each time drawing different data from the `daily_counts` table. * Check out this guide to see an example of how to create a legend. In particular, look at how the example in the guide makes use of the `label` argument in the call to `plt.plot()` and what the `plt.legend()` call does. This is a good exercise to learn how to use examples to get the look you want. * You will want to set the `cmap` parameter of `kdeplot` to `"Reds"` and `"Blues"` (or whatever two contrasting colors you'd like), and also set the `label` parameter to address which type of day you want to plot. You are required for this question to use two sets of contrasting colors for your plots.

After you get your plot working, experiment by setting `shade=True` in `kdeplot` to see the difference between the shaded and unshaded version. Please submit your work with `shade=False`.

In [24]:
```
# Set the figure size for the plot
plt.figure(figsize=(12,8))

# Set 'is_workingday' to a boolean array that is true for all working_days
is_workingday = bike['workingday'] == 'yes'

# Bivariate KDEs require two data inputs.
# In this case, we will need the daily counts for casual and registered riders on workdays
# Hint: consider using the .loc method here.
casual_workday = bike.loc[is_workingday].groupby('dteday').sum()['casual']
registered_workday = bike.loc[is_workingday].groupby('dteday').sum()['registered']

# # Use sns.kdeplot on the two variables above to plot the bivariate KDE for weekday rides
sns.kdeplot(x = casual_workday, y = registered_workday, cmap = 'Blues',label = 'Workday')

not_workingday = bike['workingday'] == 'no'
# # Repeat the same steps above but for rows corresponding to non-workingdays
# # Hint: Again, consider using the .loc method here.
casual_non_workday = bike.loc[not_workingday].groupby('dteday').sum()['casual']
registered_non_workday = bike.loc[not_workingday].groupby('dteday').sum()['registered']

# # Use sns.kdeplot on the two variables above to plot the bivariate KDE for non-workingday ri
sns.kdeplot(x = casual_non_workday, y = registered_non_workday, cmap = 'Reds',label = 'Non-Wor

plt.legend()
```

Out[24]:
```
<matplotlib.legend.Legend at 0x7f0cfe6d8a00>
```


Workday
Non-Workday
8000
6000
4000
2000
0
registered
−1000
0
1000
2000
3000
4000
casual

Question 3bi In your own words, describe what the lines and the color shades of the lines signify about the data.

Lines: Distribution shape; Color shades: Density, darker more data

Question 3bii What additional details can you identify from this contour plot that were difficult to determine from the scatter plot?

We can understand the overlapping part better than before. and the shape shows the correlation between labels.

0.1 4: Joint Plot

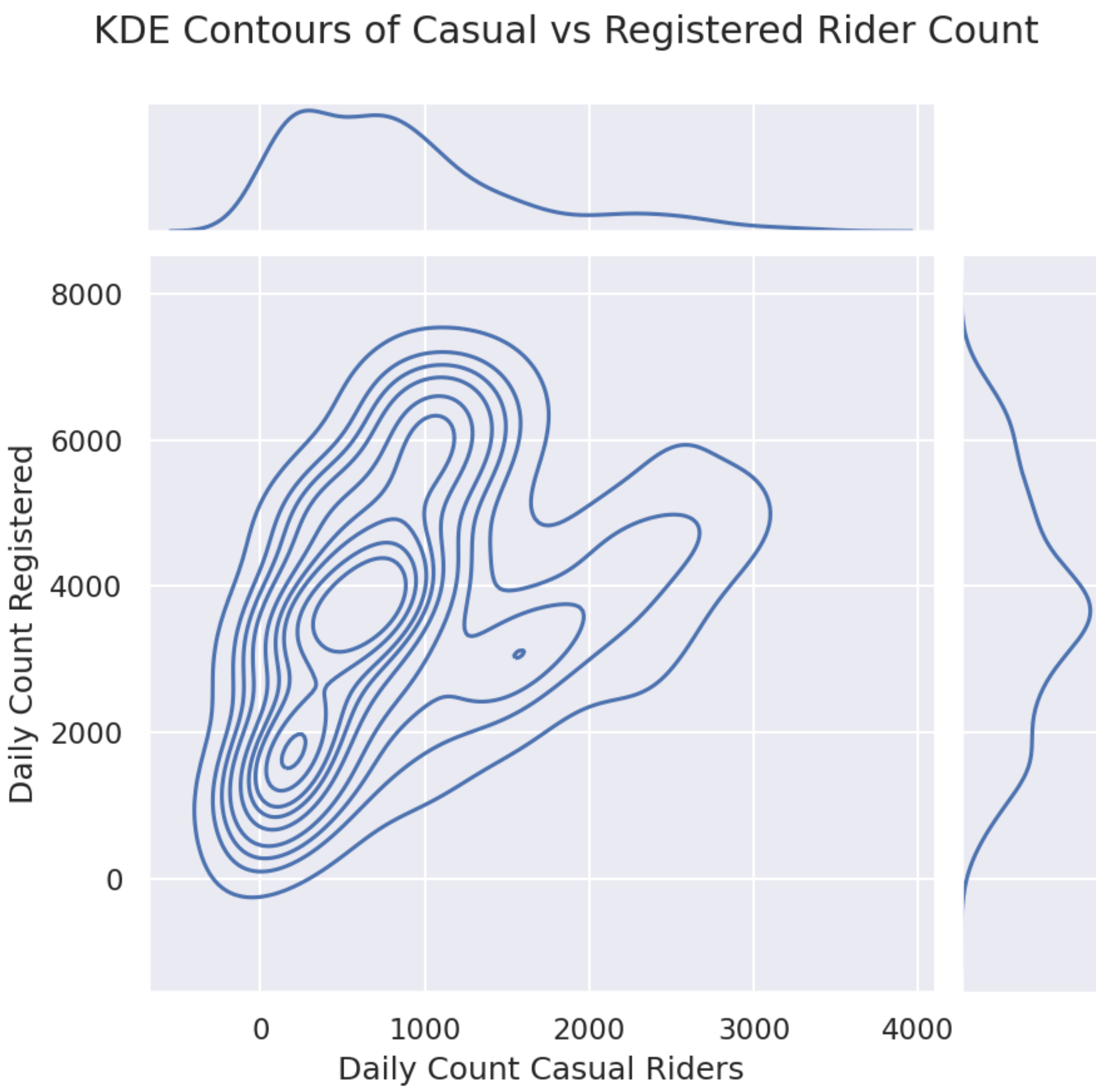
As an alternative approach to visualizing the data, construct the following set of three plots where the main plot shows the contours of the kernel density estimate of daily counts for registered and casual riders plotted together, and the two "margin" plots (at the top and right of the figure) provide the univariate kernel density estimate of each of these variables. Note that this plot makes it harder see the linear relationships between casual and registered for the two different conditions (weekday vs. weekend).

Hints: * The seaborn plotting tutorial has examples that may be helpful. * Take a look at `sns.jointplot` and its `kind` parameter. * `set_axis_labels` can be used to rename axes on the contour plot.

Note: * At the end of the cell, we called `plt.suptitle` to set a custom location for the title. * We also called `plt.subplots_adjust(top=0.9)` in case your title overlaps with your plot.

In [30]:
```
plt.figure(figsize=(12,8))
sns.jointplot(x=daily_counts['casual'], y=daily_counts['registered'], kind = 'kde').set_axis_la
plt.suptitle("KDE Contours of Casual vs Registered Rider Count")
plt.subplots_adjust(top=0.9);
```

```
<Figure size 1800x1200 with 0 Axes>
```


KDE Contours of Casual vs Registered Rider Count
Daily Count Registered
8000
6000
4000
2000
0
0
1000
2000
3000
4000
Daily Count Casual Riders

0.2 5: Understanding Daily Patterns

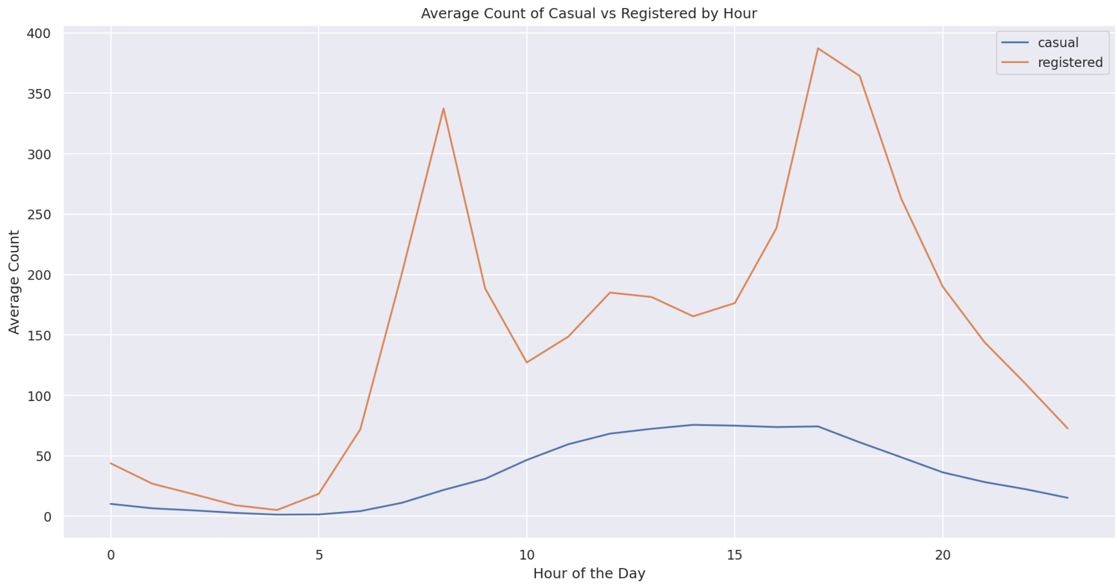
0.2.1 Question 5

Question 5a Let's examine the behavior of riders by plotting the average number of riders for each hour of the day over the **entire dataset**, stratified by rider type.

Your plot should look like the plot below. While we don't expect your plot's colors to match ours exactly, your plot should have different colored lines for different kinds of riders.

```
In [49]: hours_number = bike.loc[:,['hr', 'casual', 'registered']].groupby('hr').agg('mean')
         hours_number
         sns.lineplot(data = hours_number, x = 'hr', y = 'casual')
         sns.lineplot(data = hours_number, x = 'hr', y = 'registered')
         plt.xlabel('Hour of the Day')
         plt.ylabel('Average Count')
         plt.title('Average Count of Casual vs Registered by Hour')
         plt.legend(['casual','registered'])
```

```
Out[49]: <matplotlib.legend.Legend at 0x7f0c907bbc70>
```



Question 5b What can you observe from the plot? Hypothesize about the meaning of the peaks in the registered riders' distribution.

Registered users have two obvious peaks, which could be the peak hours of "going to work" and "end working back to home"

In our case with the bike ridership data, we want 7 curves, one for each day of the week. The x-axis will be the temperature and the y-axis will be a smoothed version of the proportion of casual riders.

You should use `statsmodels.nonparametric.smoothers_lowess.lowess` just like the example above. Unlike the example above, plot ONLY the lowess curve. Do not plot the actual data, which would result in overplotting. For this problem, the simplest way is to use a loop.

You do not need to match the colors on our sample plot as long as the colors in your plot make it easy to distinguish which day they represent.

Hints: * Start by just plotting only one day of the week to make sure you can do that first.

- The `lowess` function expects y coordinate first, then x coordinate. You should also set the `return_sorted` field to `False`.
- Look at the top of this homework notebook for a description of the temperature field to know how to convert to Fahrenheit. By default, the temperature field ranges from 0.0 to 1.0. In case you need it, Fahrenheit $= \text{Celsius} * \frac{9}{5} + 32$.

Note: If you prefer plotting temperatures in Celsius, that's fine as well!

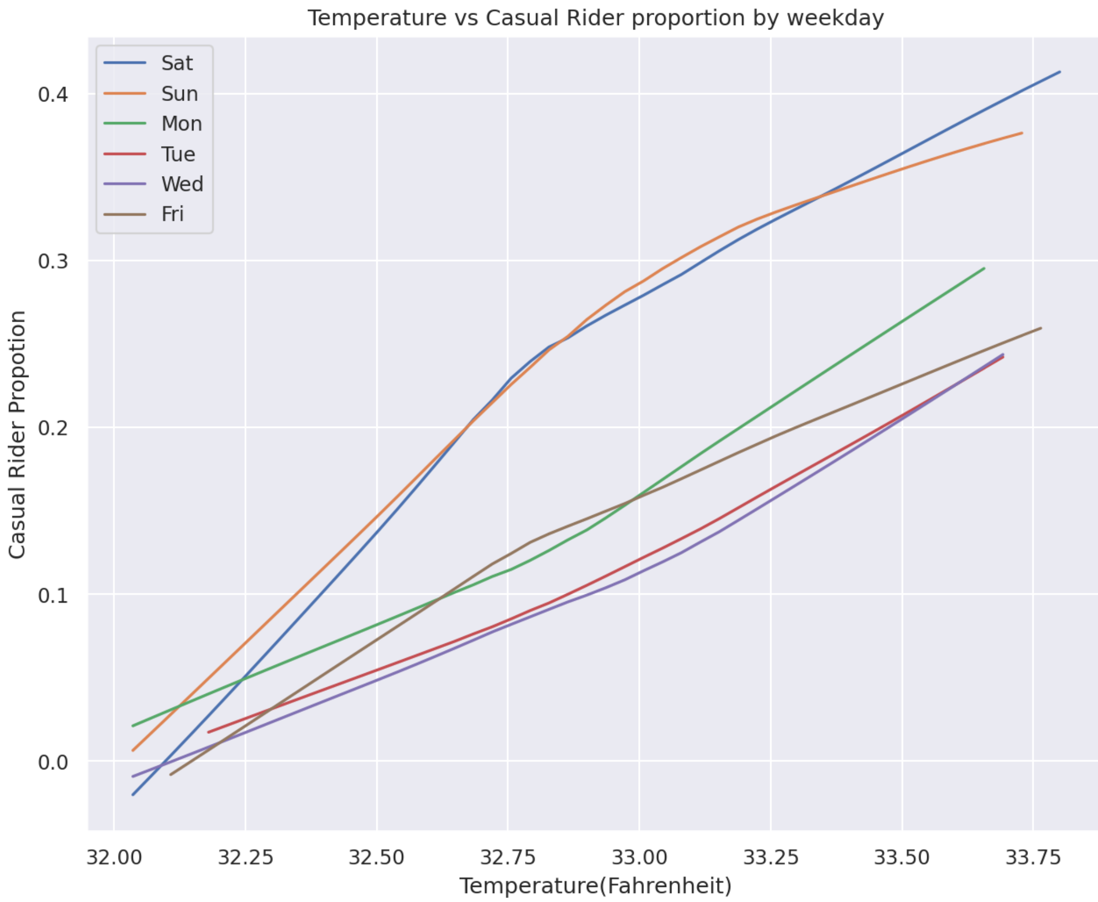
```
In [60]: bike.head()
```

```
Out[60]:    instant      dteday  season  yr  mnth  hr holiday weekday workingday  \
         0        1  2011-01-01       1   0     1   0      no     Sat         no
         1        2  2011-01-01       1   0     1   1      no     Sat         no
         2        3  2011-01-01       1   0     1   2      no     Sat         no
         3        4  2011-01-01       1   0     1   3      no     Sat         no
         4        5  2011-01-01       1   0     1   4      no     Sat         no

           weathersit  temp   atemp   hum  windspeed  casual  registered  cnt  \
         0      Clear  0.24  0.2879  0.81        0.0       3          13   16
         1      Clear  0.22  0.2727  0.80        0.0       8          32   40
         2      Clear  0.22  0.2727  0.80        0.0       5          27   32
         3      Clear  0.24  0.2879  0.75        0.0       3          10   13
         4      Clear  0.24  0.2879  0.75        0.0       0           1    1

            prop_casual
         0     0.187500
         1     0.200000
         2     0.156250
         3     0.230769
         4     0.000000
```

```
In [70]: #I have to code like this, stupidly, since i rmb we are not allowed to use for loop
```

```
from statsmodels.nonparametric.smoothers_lowess import lowess

plt.figure(figsize=(10,8))

Sat_ysmooth = lowess(bike[bike['weekday'] == 'Sat']['prop_casual'], bike[bike['weekday'] == 'Sa
Sat_xobs = bike[bike['weekday'] == 'Sat']['temp']*9/5 + 32
sns.lineplot(Sat_xobs, Sat_ysmooth, label = 'Sat')

Sun_ysmooth = lowess(bike[bike['weekday'] == 'Sun']['prop_casual'], bike[bike['weekday'] == 'Su
Sun_xobs = bike[bike['weekday'] == 'Sun']['temp']*9/5 + 32
sns.lineplot(Sun_xobs, Sun_ysmooth, label = 'Sun')

Mon_ysmooth = lowess(bike[bike['weekday'] == 'Mon']['prop_casual'], bike[bike['weekday'] == 'Mo
Mon_xobs = bike[bike['weekday'] == 'Mon']['temp']*9/5 + 32
sns.lineplot(Mon_xobs, Mon_ysmooth, label = 'Mon')

Tue_ysmooth = lowess(bike[bike['weekday'] == 'Tue']['prop_casual'], bike[bike['weekday'] == 'Tu
Tue_xobs = bike[bike['weekday'] == 'Tue']['temp']*9/5 + 32
sns.lineplot(Tue_xobs, Tue_ysmooth, label = 'Tue')

Wed_ysmooth = lowess(bike[bike['weekday'] == 'Wed']['prop_casual'], bike[bike['weekday'] == 'We
Wed_xobs = bike[bike['weekday'] == 'Wed']['temp']*9/5 + 32
sns.lineplot(Wed_xobs, Wed_ysmooth, label = 'Wed')

Thu_ysmooth = lowess(bike[bike['weekday'] == 'Thu']['prop_casual'], bike[bike['weekday'] == 'Th
Thu_xobs = bike[bike['weekday'] == 'Thu']['temp']*9/5 + 32
sns.lineplot(Thu_xobs, Thu_ysmooth, label = 'Thu')

Fri_ysmooth = lowess(bike[bike['weekday'] == 'Fri']['prop_casual'], bike[bike['weekday'] == 'Fr
Fri_xobs = bike[bike['weekday'] == 'Fri']['temp']*9/5 + 32
sns.lineplot(Fri_xobs, Fri_ysmooth, label = 'Fri')

plt.xlabel('Temperature(Fahrenheit)')
plt.ylabel('Casual Rider Propotion')
plt.title('Temperature vs Casual Rider proportion by weekday')
```

```
Out[70]: Text(0.5, 1.0, 'Temperature vs Casual Rider proportion by weekday')
```



Temperature vs Casual Rider proportion by weekday
Sat
Sun
Mon
Tue
Wed
Fri
0.4
0.3
0.2
0.1
0.0
Casual Rider Propotion
32.00
32.25
32.50
32.75
33.00
33.25
33.50
33.75
Temperature(Fahrenheit)

Question 6c What do you see from the curve plot? How is `prop_casual` changing as a function of temperature? Do you notice anything else interesting?

In everyday, as temp goes higher, the propotion of casual rider goes up as well. And for weekends, these two days are less sensitive to temp change when temp>32.75.

0.2.2 Question 7

Question 7A Imagine you are working for a Bike Sharing Company that collaborates with city planners, transportation agencies, and policy makers in order to implement bike sharing in a city. These stakeholders would like to reduce congestion and lower transportation costs. They also want to ensure the bike sharing program is implemented equitably. In this sense, equity is a social value that is informing the deployment and assessment of your bike sharing technology.

Equity in transportation includes: improving the ability of people of different socio-economic classes, genders, races, and neighborhoods to access and afford the transportation services, and assessing how inclusive transportation systems are over time.

Do you think the `bike` data as it is can help you assess equity? If so, please explain. If not, how would you change the dataset? You may discuss how you would change the granularity, what other kinds of variables you'd introduce to it, or anything else that might help you answer this question.

The current dataset is not enough for assessing equity. Other data like customer info(gender, age, occupation, etc…) and geological distribution of users in city are needed to better assess this problem.

Question 7B Bike sharing is growing in popularity and new cities and regions are making efforts to implement bike sharing systems that complement their other transportation offerings. The goals of these efforts are to have bike sharing serve as an alternate form of transportation in order to alleviate congestion, provide geographic connectivity, reduce carbon emissions, and promote inclusion among communities.

Bike sharing systems have spread to many cities across the country. The company you work for asks you to determine the feasibility of expanding bike sharing to additional cities of the U.S.

Based on your plots in this assignment, what would you recommend and why? Please list at least two reasons why, and mention which plot(s) you drew you analysis from.

Note: There isn't a set right or wrong answer for this question, feel free to come up with your own conclusions based on evidence from your plots!

There is an obvious peak for during peak hours, which users mainly use bikes to commute. So expanding to cities with bad public transportation / severe traffic congestion and with large commuting population should be a good idea.