Exploring Seaborn's Built-in Dataset

February 5, 2020

0.1 EXPLORE CATEGORICAL DATA USING SEABORN

PREPARED BY: Gary-Gregoire Coquillo

Let's import some libraries

[70]:
```
import numpy as np
```

[71]:
```
import matplotlib.pyplot as plt
```

[72]:
```
import seaborn as sns
%matplotlib inline
```

Let's import one of Seaborn's built-in dataset called Tips

It contains data about people tipping at restaurants by day

[73]:
```
tips_dataset = sns.load_dataset('tips')
```

Let's explore the top rows of this dataset

[74]:
```
tips_dataset.head()
```

	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

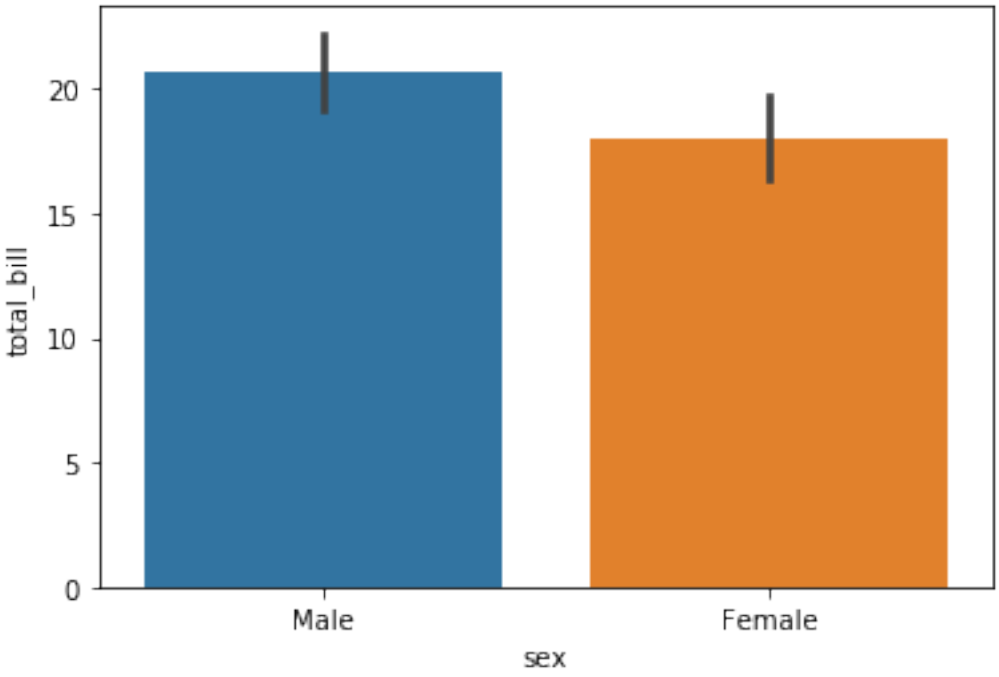
In the dataset above, notice the numbers and categorical columns

Now let's try a Bar Plot where x is the categorical data and y the numbers data By default the total_bill with represented as a Mean. But we can use the estimator to change this. See the next line of codes.

[75]:
```
sns.barplot(x='sex', y='total_bill', data= tips_dataset)

# Here you can see that the mean total_bill is higher for male than female
```

[75]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c2a28ca90>
```

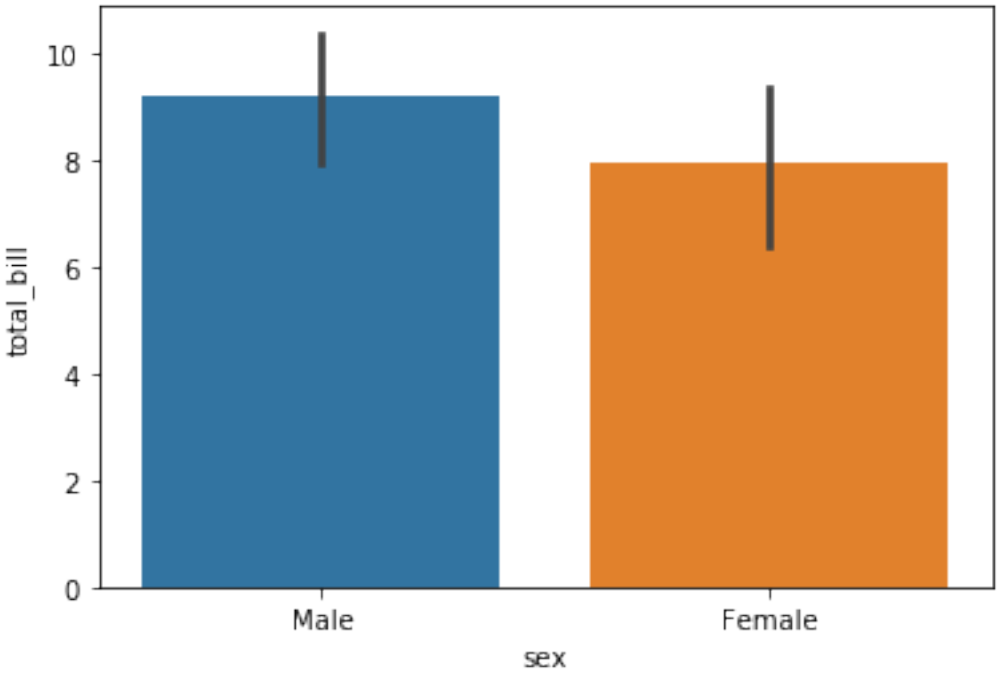


Instead of the mean total_bill in the barplot above, let's report the standard deviation using "estimator" argument. This is the reason why we imported Numpy in the beginning.

[76]:
```
sns.barplot(x= 'sex', y= 'total_bill', data= tips_dataset, estimator= np.std)

# Here you can see that total_bill variations are greater for Male than Female
```

[76]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c2a41b518>
```

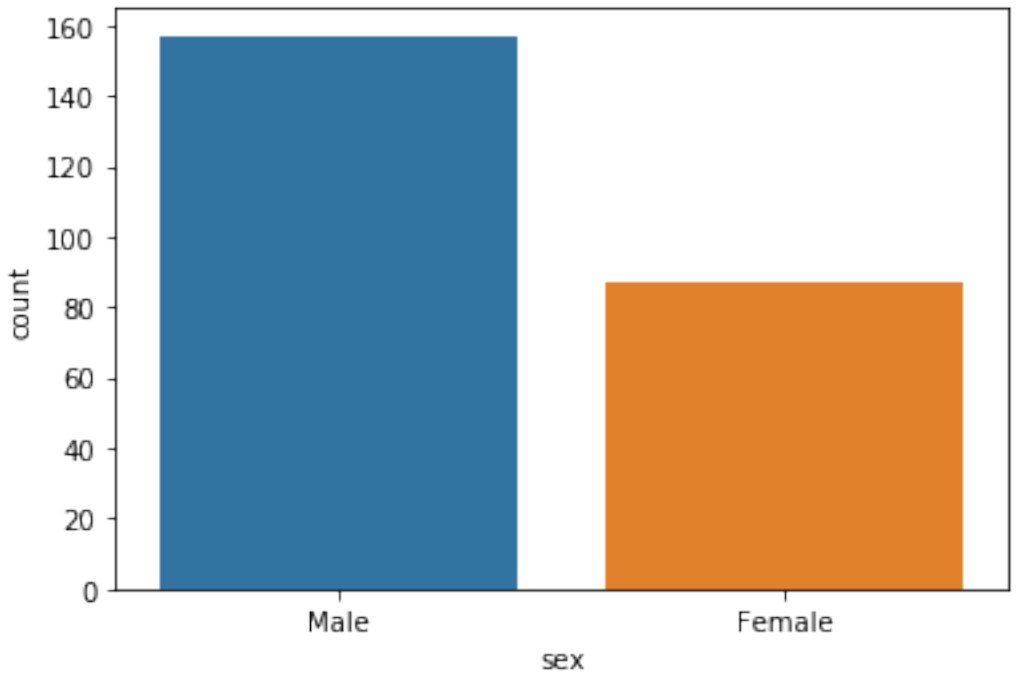


Now let's try a Count Plot. In this case you only need to pass on 1 categorical argument

[77]:
```
sns.countplot(x= 'sex', data= tips_dataset)

# Here we report the total count for each the of sex category, male and female
```

[77]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c2a4d80b8>
```

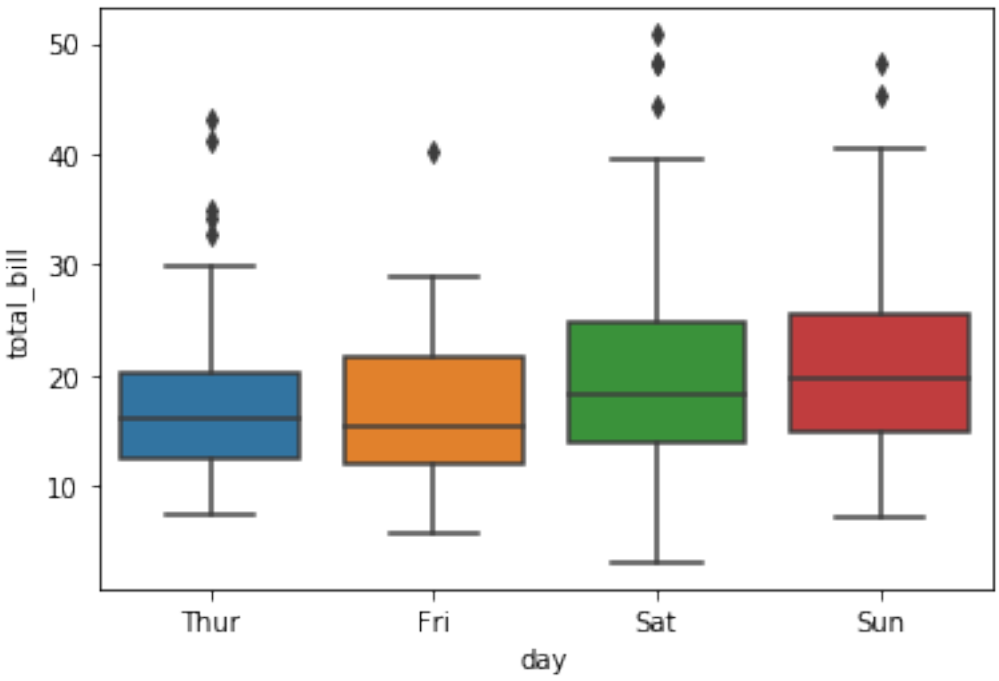


Moving on to a Box Plot example to explore total_bill distribution by day

[78]:
```
sns.boxplot(x= 'day', y= 'total_bill', data= tips_dataset)

# Median total_bill is close to $20 on any given day from the dataset
# The dots on top are outliers
# Biggest variation is on Saturday, including the lowest total_bill of all days
```

[78]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c2a57ba90>
```

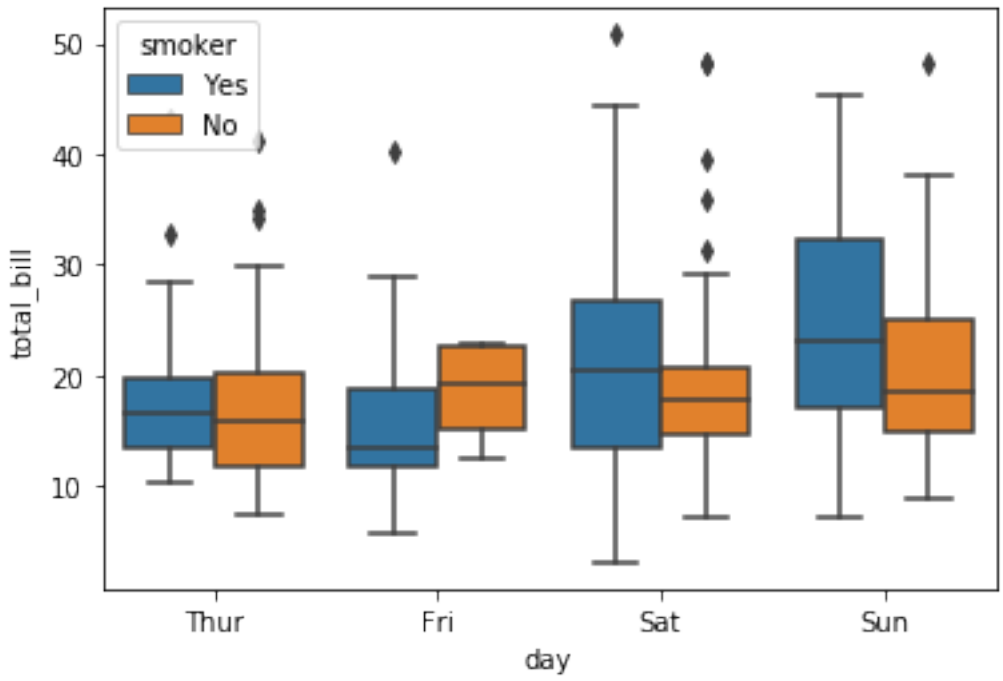


Let's separate each day by smoking customer and non-smoking customer using the "hue" argument

[79]:
```
sns.boxplot(x= 'day', y= 'total_bill', data= tips_dataset, hue= 'smoker' )

# Here non-smokers pay more on average than smokers on Fridays
# Variation (Max-Min) is higher for smokers on Fridays
```

[79]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c2a6a66a0>
```

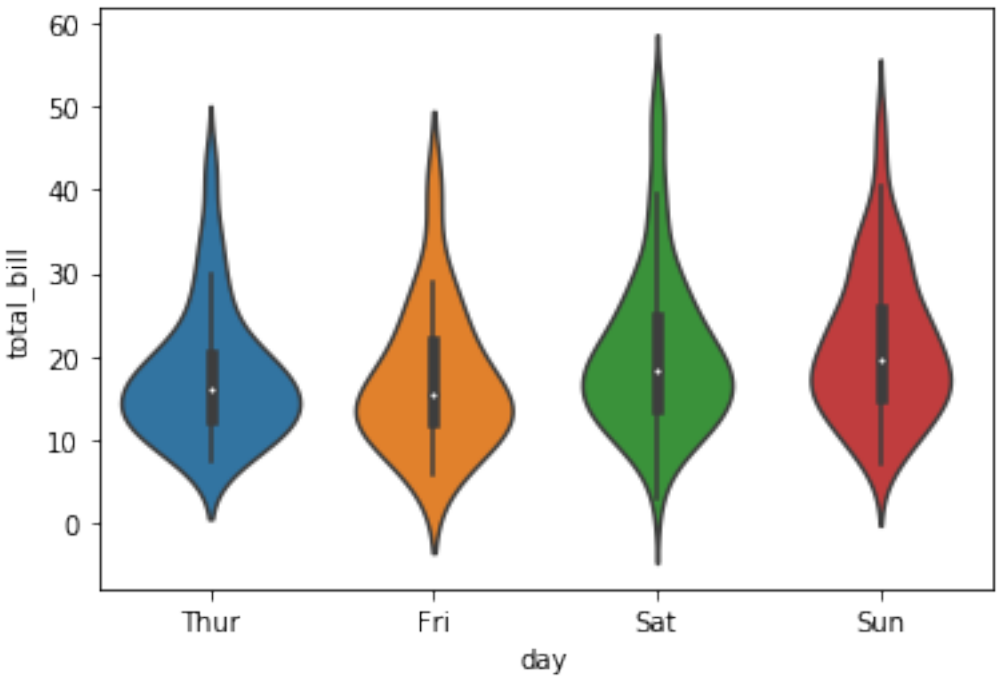


Let's try a more complicated dataset: the Violin Plot

[80]:
```
sns.violinplot(x= 'day', y= 'total_bill', data= tips_dataset)

# Violin plots provide a much clearer view of the density of the data
# You can see where most people pay about $20 for their meal
```

[80]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c2a8b5630>
```

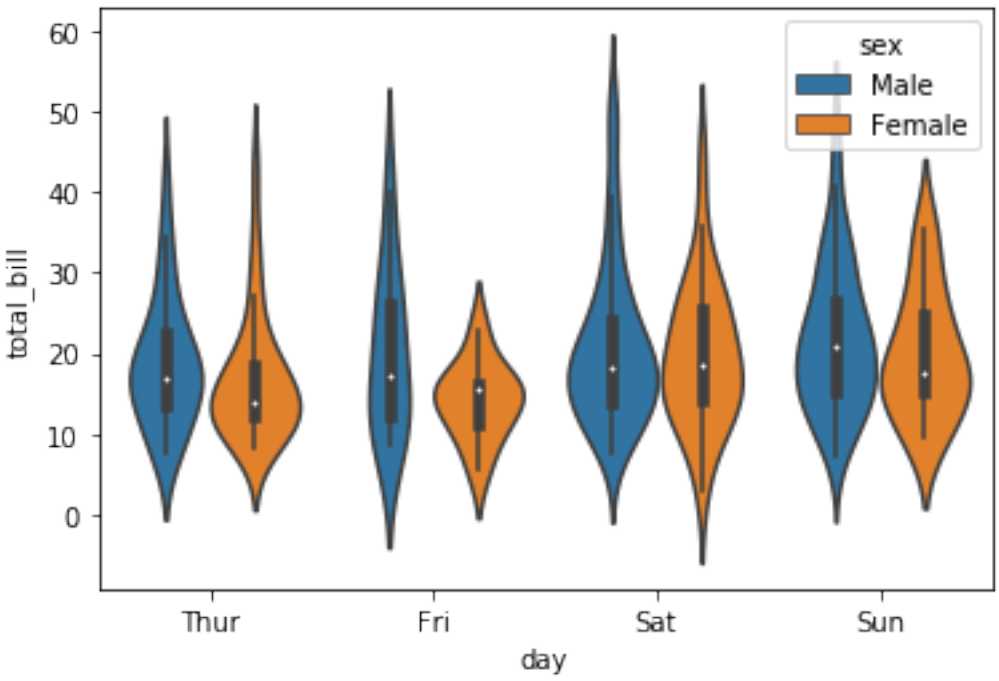


Now let's separate each violin plot with the Male/Female category

[81]:
```
sns.violinplot(x= 'day', y= 'total_bill', data= tips_dataset, hue= 'sex')

# Each category has its own Violin plot. If you want them combine, see the next␣
 ↪line of code
```

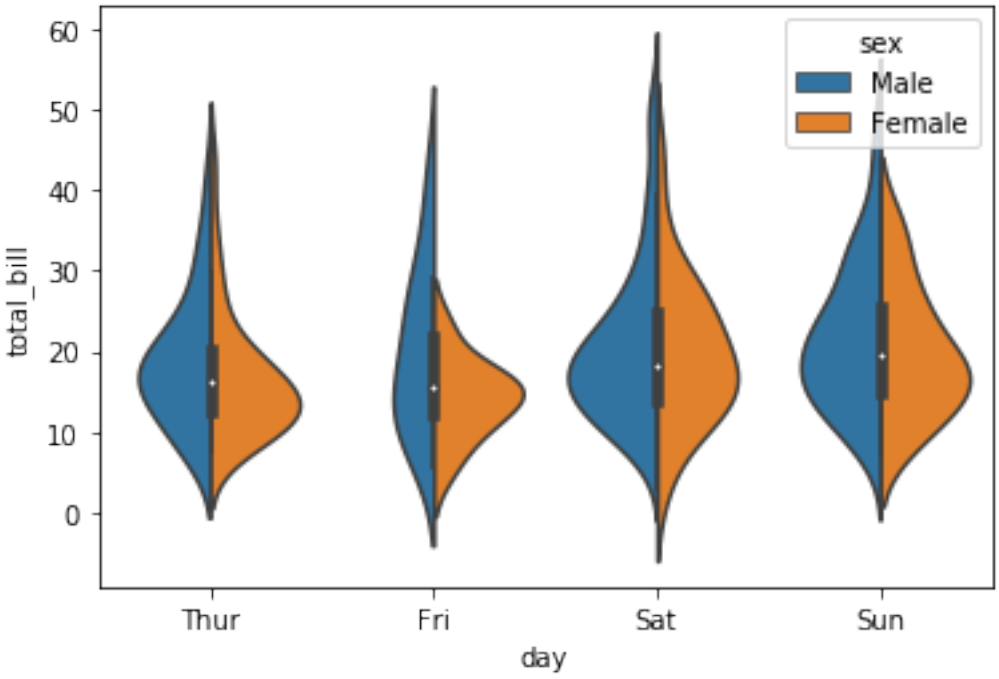
[81]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c2a9da2e8>
```



[82]:
```
sns.violinplot(x= 'day', y= 'total_bill', data= tips_dataset, hue= 'sex',␣
 ↪split= True)

# Now we are combining Male vs Female into one Violin plot for each day
# This gives you a better view of the total_bill distribution for Male vs␣
 ↪Female
```

[82]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c2ab2d978>
```

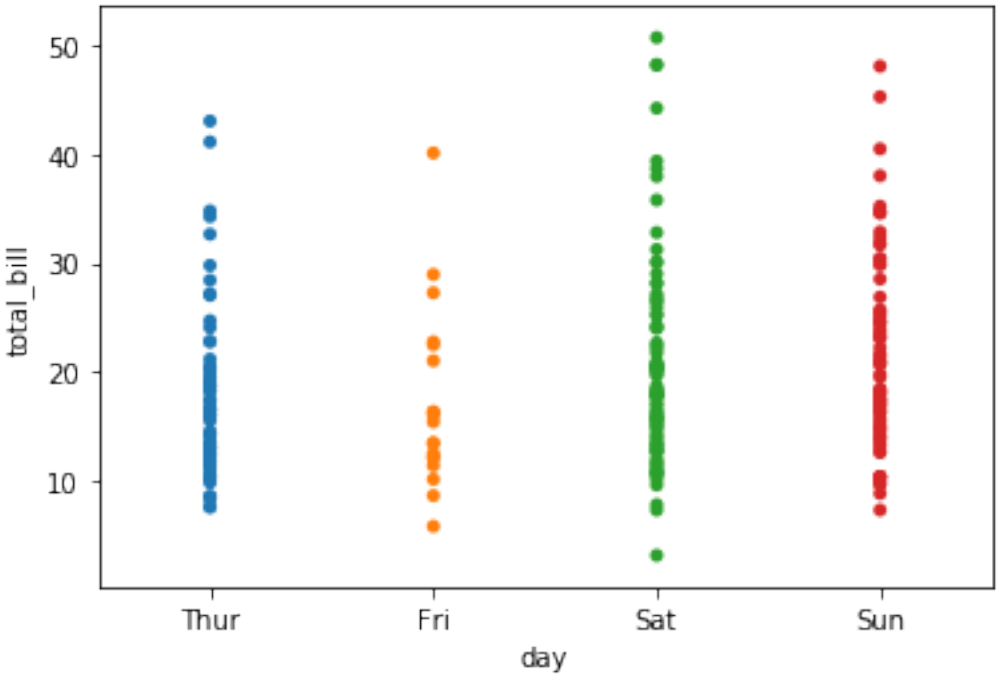


Let's explore the Strip Plot

[83]:
```
sns.stripplot(x= 'day', y= 'total_bill', data= tips_dataset, jitter= False)

# When jitter is set to False, it is hard to look at the density of the data
```

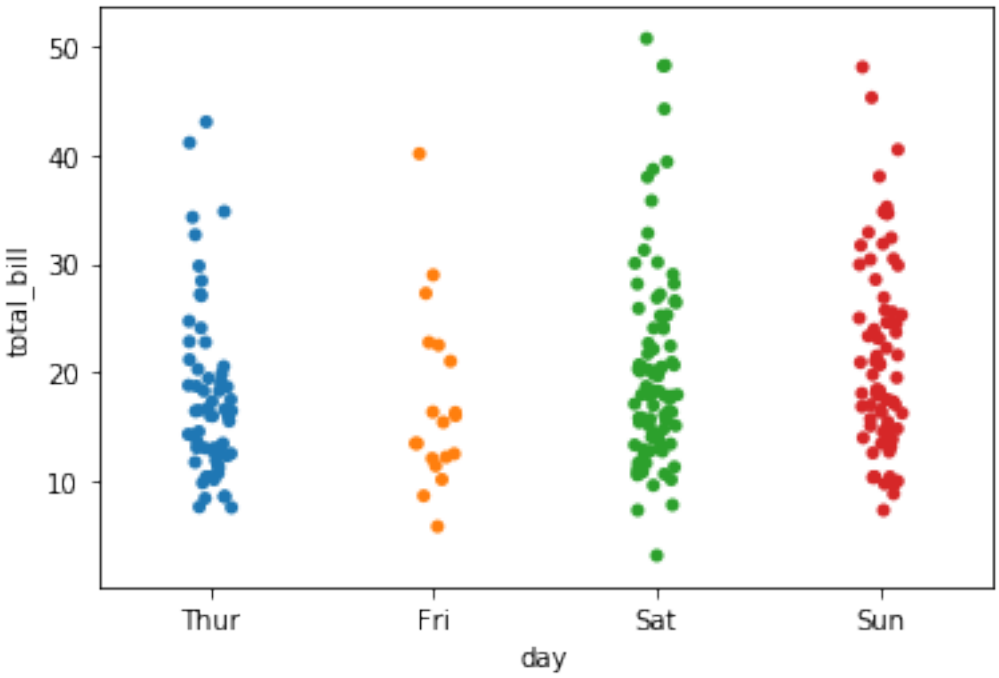
[83]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c2abef400>
```



[84]:
```
sns.stripplot(x= 'day', y= 'total_bill', data= tips_dataset, jitter= True)

# Now that's much better
# Let's split the data into category Male and Female
```

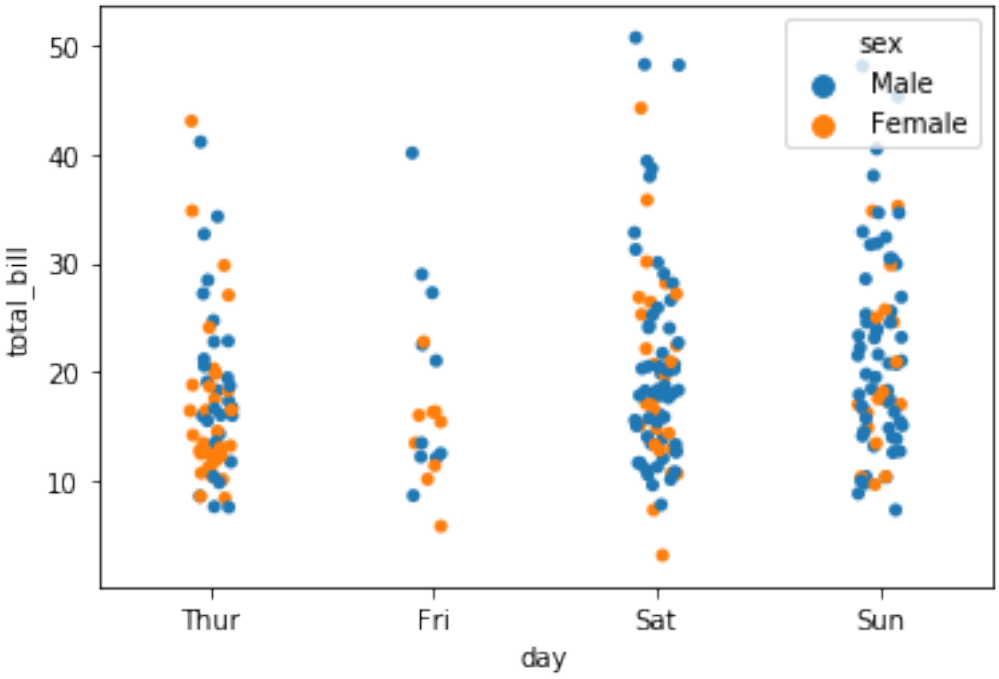
[84]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c2acb3588>
```



[85]:
```
sns.stripplot(x= 'day', y= 'total_bill', data= tips_dataset, jitter= True, hue=␣
↪'sex')

# Male and Female combined for each day
```

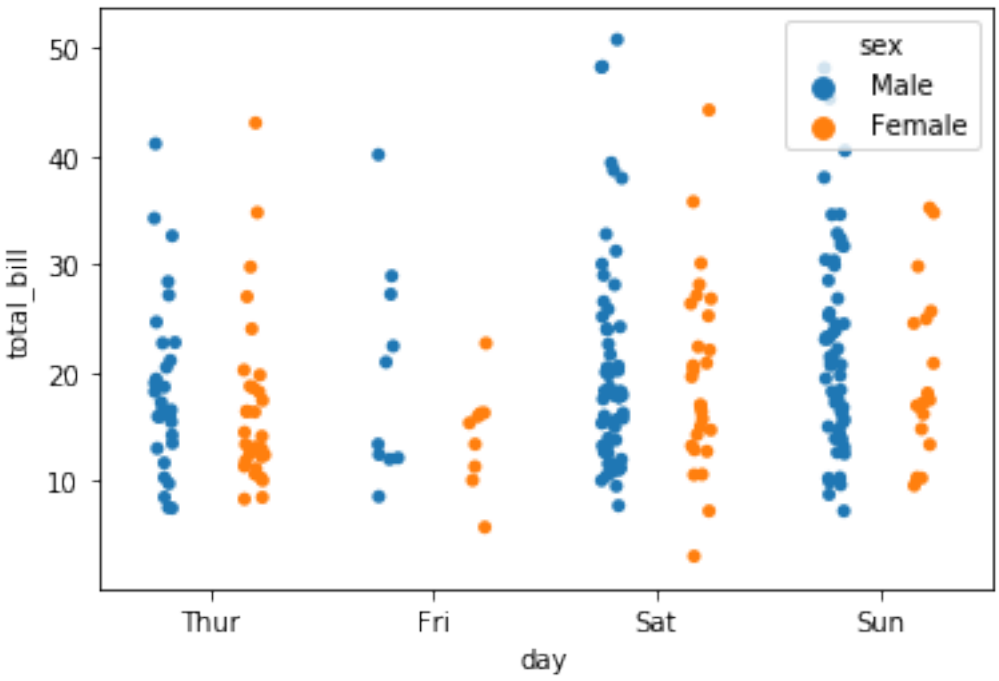
[85]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c2ac29ac8>
```



[86]:
```
sns.stripplot(x= 'day', y= 'total_bill', data= tips_dataset, jitter= True, hue=␣
↪'sex', split= True)

# Male and Female separated for each day
```

[86]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c29aee780>
```

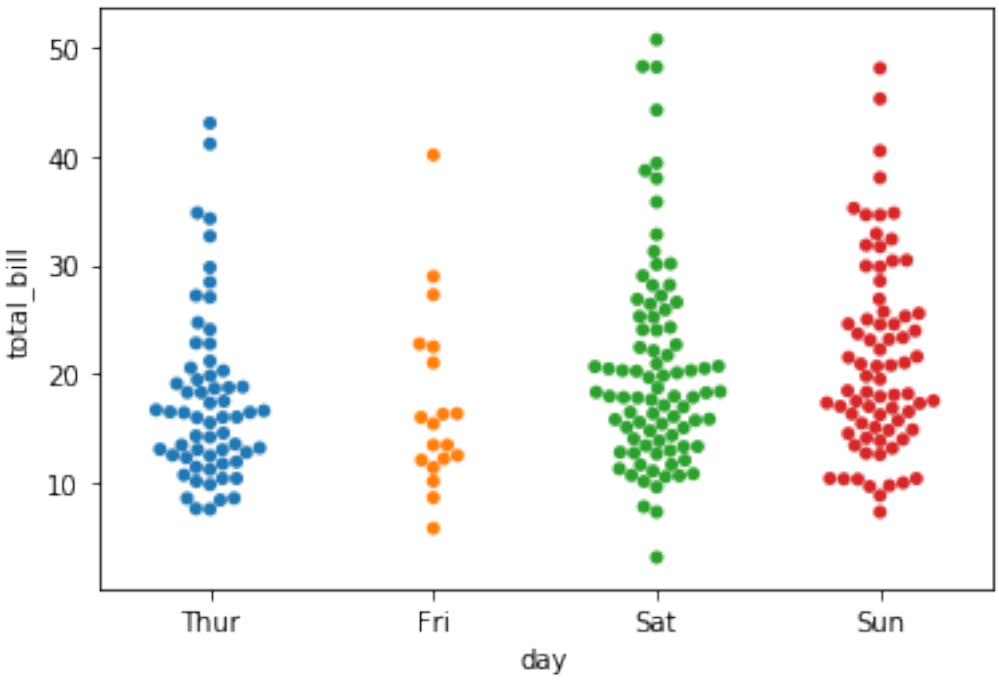


Now time to try a swarmplot

[87]:
```
sns.swarmplot(x= 'day', y= 'total_bill', data= tips_dataset)

# Similar to Strip Plot
# Can run into computational and visual issues with very large datasets
```

[87]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c2ac1f5c0>
```

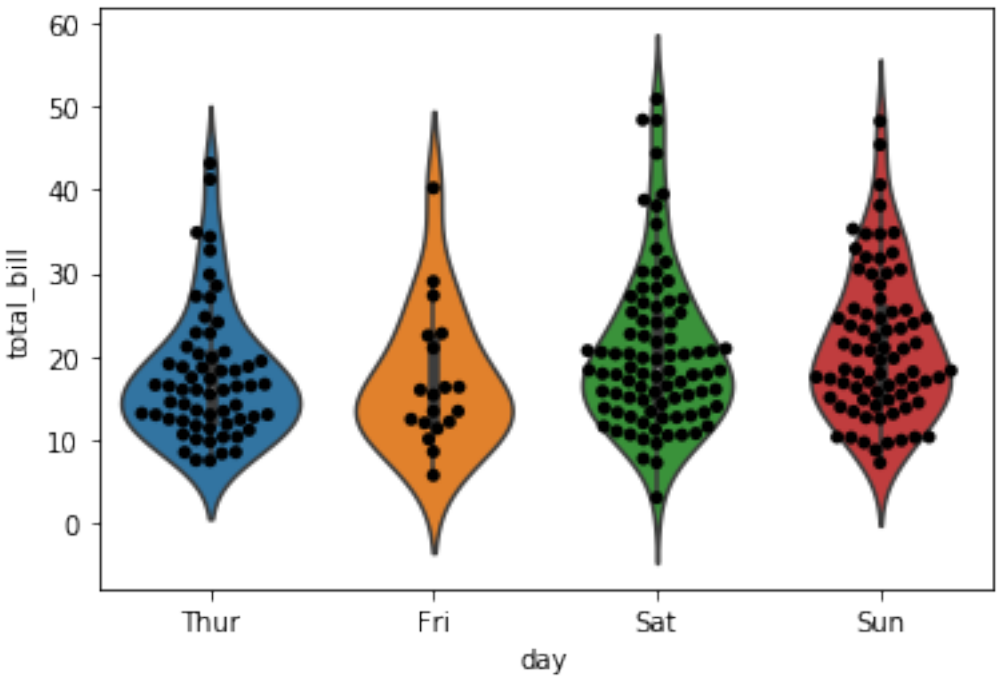


We can combine a Violin Plot and a Swarm Plot in the same graph. Let's try it here by putting 2 lines of code in the same cell

[88]:
```
sns.violinplot(x= 'day', y= 'total_bill', data= tips_dataset)
sns.swarmplot(x= 'day', y= 'total_bill', data= tips_dataset, color= 'black')

# Here we changed the color of the Swarm plot for better visibility
# Both Violin and Swarm plots give a great view of data distribution and␣
 ↪density
```

[88]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x1c2ad2cc88>
```

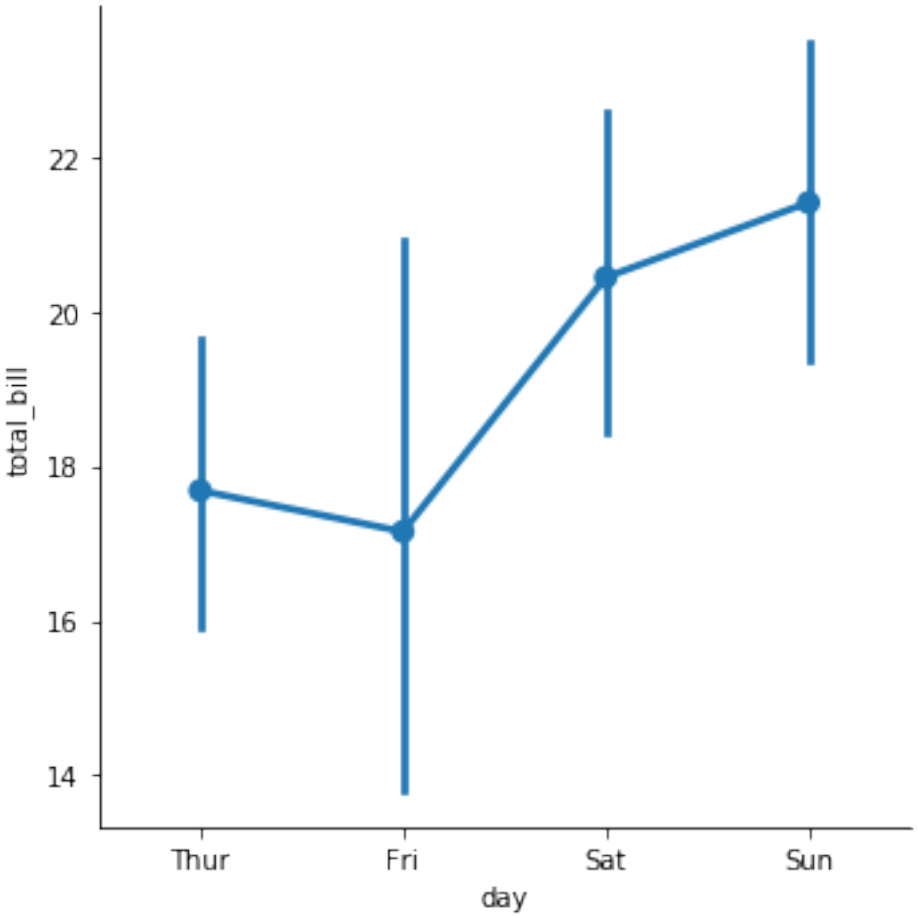


Now let's try the Factor Plot

[89]:
```
sns.factorplot(x= 'day', y= 'total_bill', data= tips_dataset)

# This is the default visual if you don't specify the visual 'kind'
```

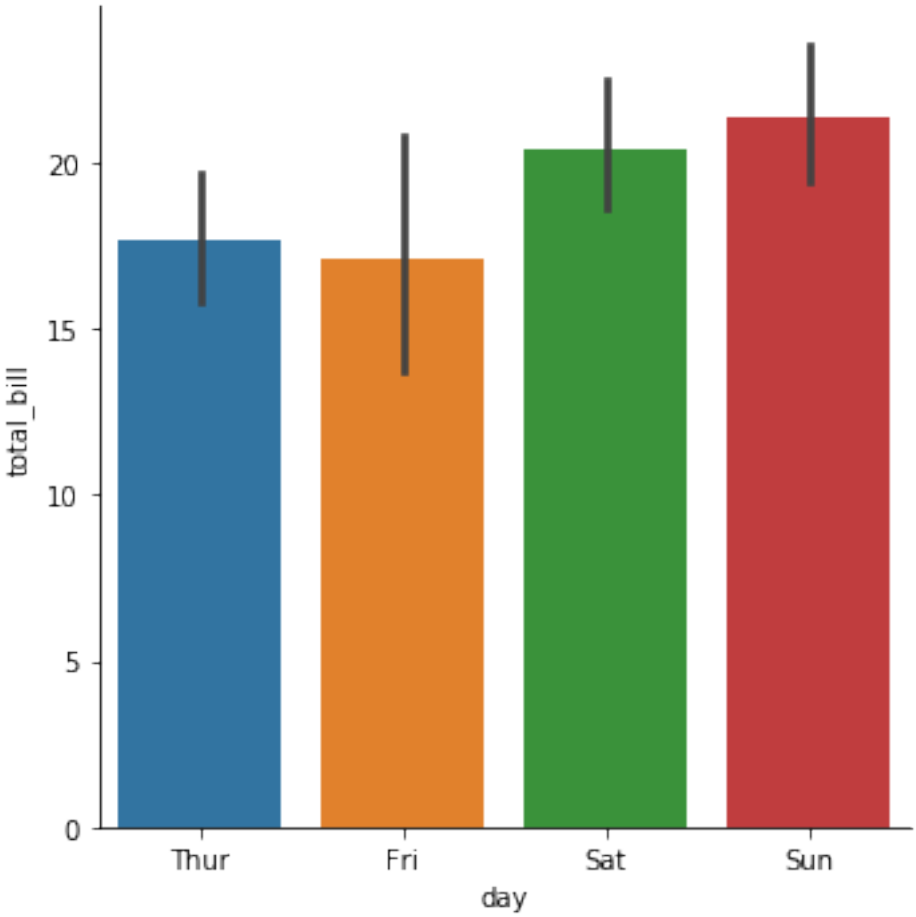
[89]:
```
<seaborn.axisgrid.FacetGrid at 0x1c29ba74e0>
```



[90]:
```
sns.factorplot(x= 'day', y= 'total_bill', data= tips_dataset, kind= 'bar')

# Here we call out the
```

[90]:
```
<seaborn.axisgrid.FacetGrid at 0x1c29a054e0>
```

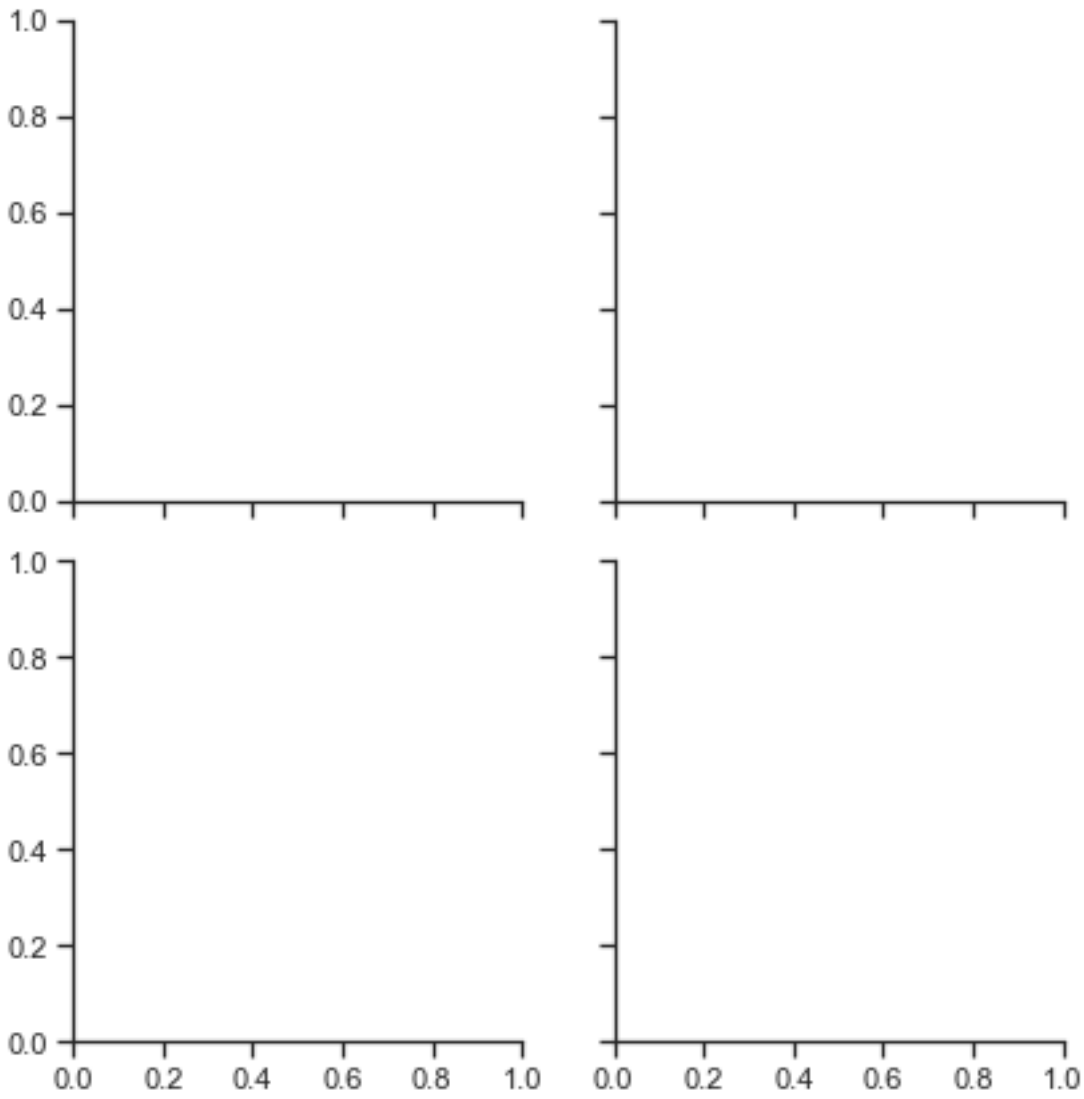


Finally, let's explore Facet Grid

[96]:
```
sns.FacetGrid(data= tips_dataset, row= 'time', col= 'smoker')

# This initializes the grid. Now let's add some  histogram visuals
```

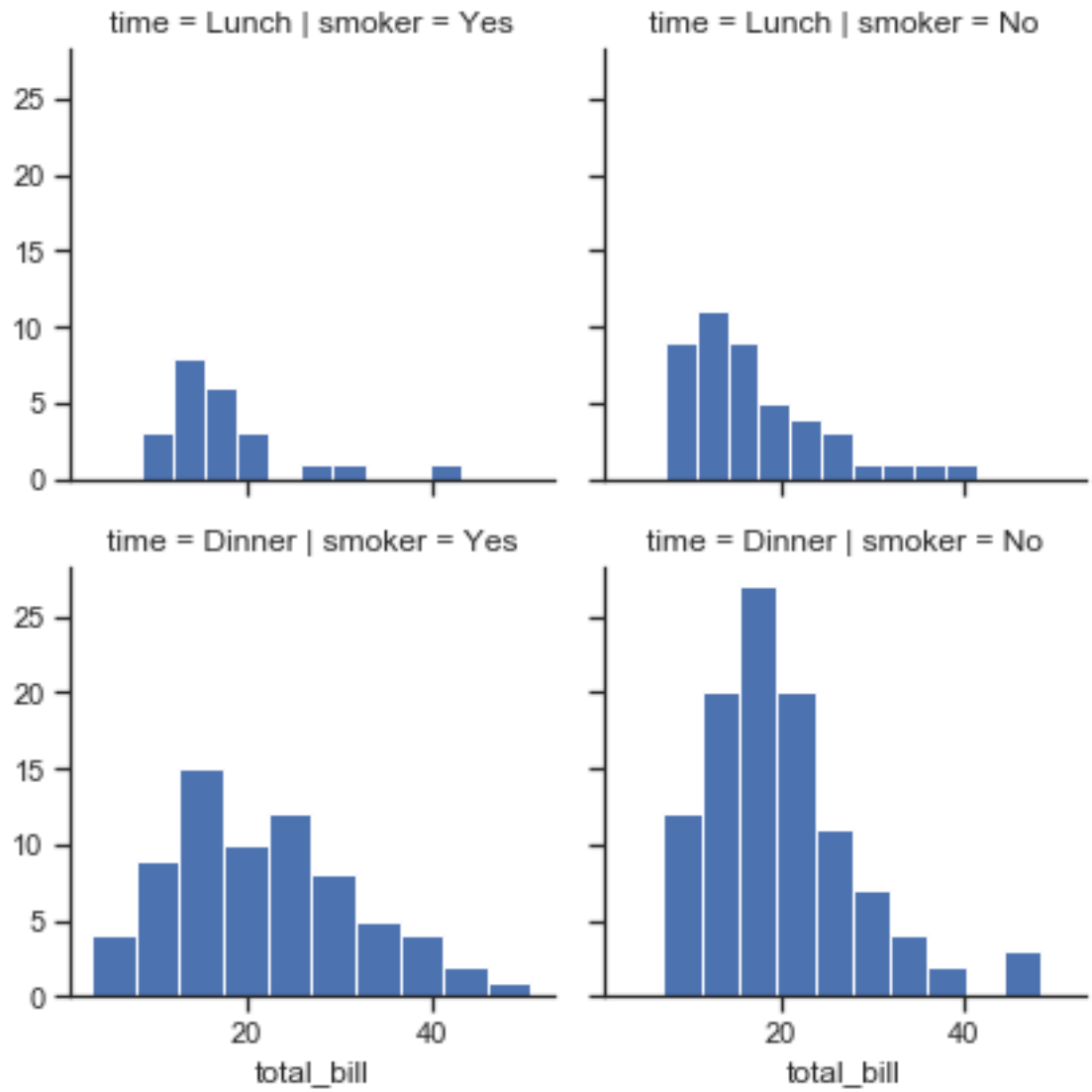
[96]:
```
<seaborn.axisgrid.FacetGrid at 0x1c2b5be710>
```



[98]:
```
(sns.FacetGrid(data= tips_dataset, row= 'time', col= 'smoker')).map(plt.hist,␣
 ↪'total_bill')

# Here we are exploring a histogram of smokers vs non-smokers during lunch and␣
 ↪dinner time
```

[98]: <seaborn.axisgrid.FacetGrid at 0x1c2b98b978>

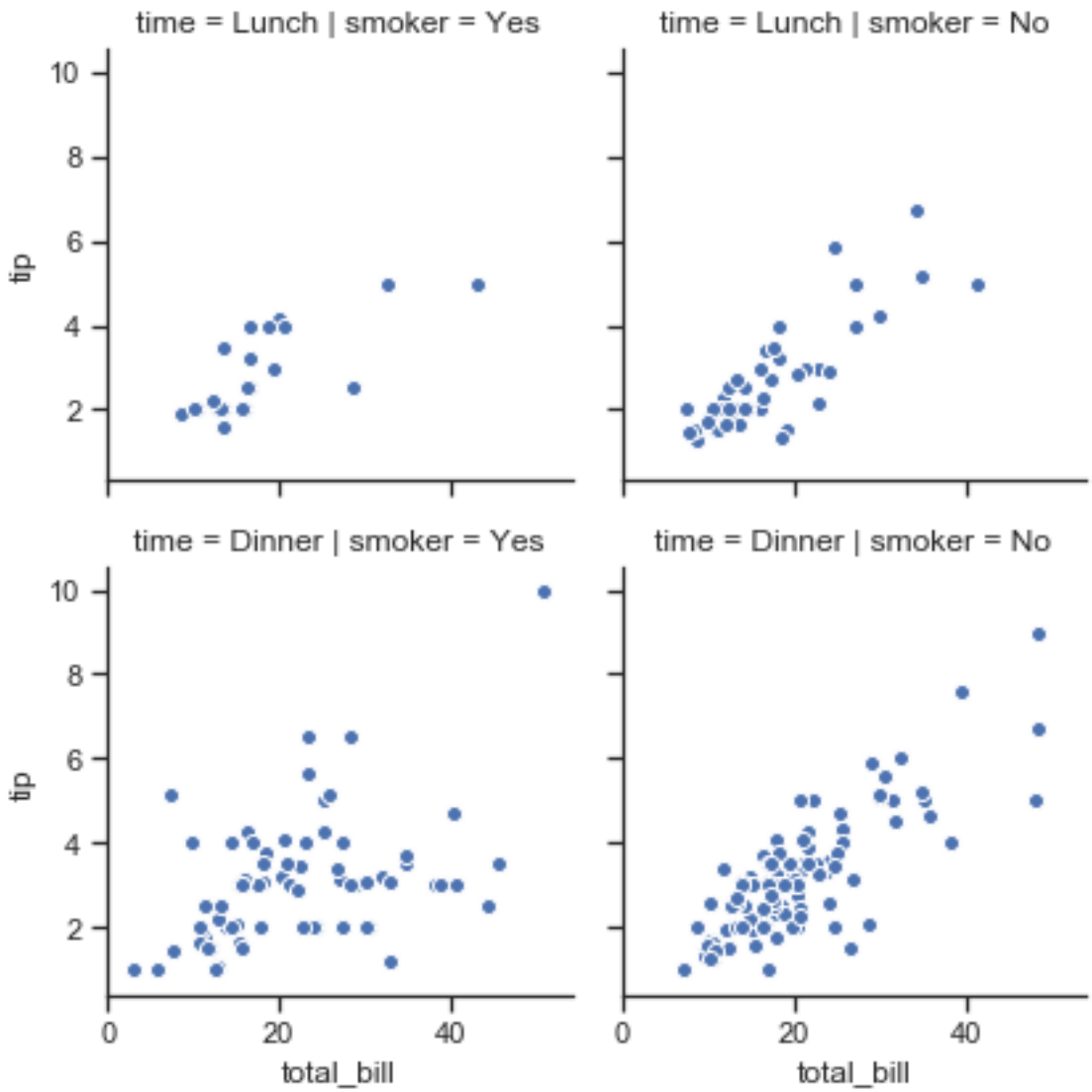


Let's do a Scatter Plot style for the Facet Grid

[103]:
```
graph= sns.FacetGrid(data= tips_dataset, row= 'time', col= 'smoker')
graph.map(plt.scatter, 'total_bill', 'tip', edgecolor= 'w')

# Here we explore smokers vs non-smokers paying tips during lunch vs dinner
```

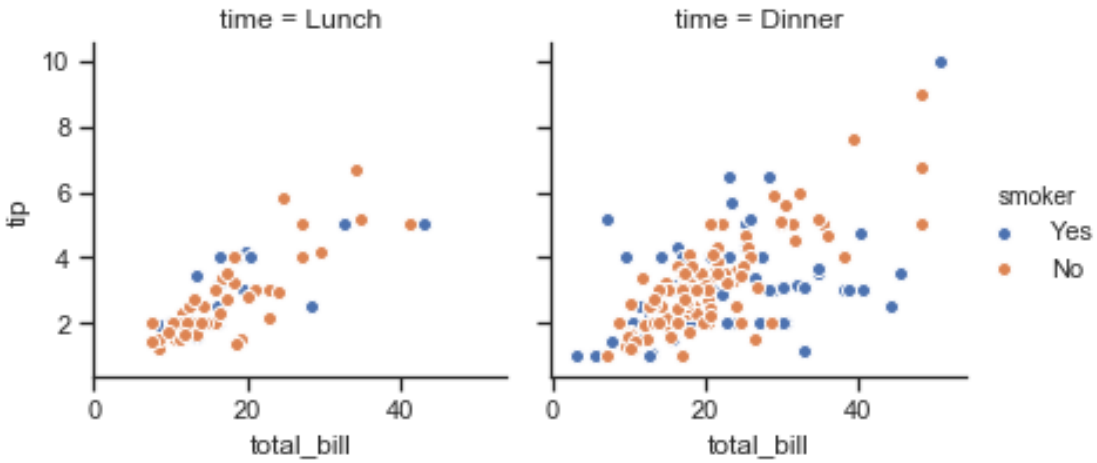
[103]:
```
<seaborn.axisgrid.FacetGrid at 0x1c2c796748>
```



[110]:
```
graph= sns.FacetGrid(data= tips_dataset, col= 'time', hue= 'smoker')
(graph.map(plt.scatter, 'total_bill', 'tip', edgecolor= 'w')).add_legend()

# Here we are assigning one of the variables (smoker) to the color of plot data
```

[110]: <seaborn.axisgrid.FacetGrid at 0x1c2d2a4e80>



1 Hope you enjoyed this Data Exploration with Seaborn!