(http://www.pieriandata.com)

Categorical Plots - Statistical Estimation within Categories

Often we have **categorical** data, meaning the data is in distinct groupings, such as Countries or Companies. There is no country value "between" USA and France and there is no company value "between" Google and Apple, unlike continuous data where we know values can exist between data points, such as age or price.

To begin with categorical plots, we'll focus on statistical estimation within categories. Basically this means we will visually report back some statistic (such as mean or count) in a plot. We already know how to get this data with pandas, but often its easier to understand the data if we plot this.

Imports

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

The Data

In [2]:
```
df = pd.read_csv("dm_office_sales.csv")
```

In [3]:
```
df.head()
```

Out[3]:

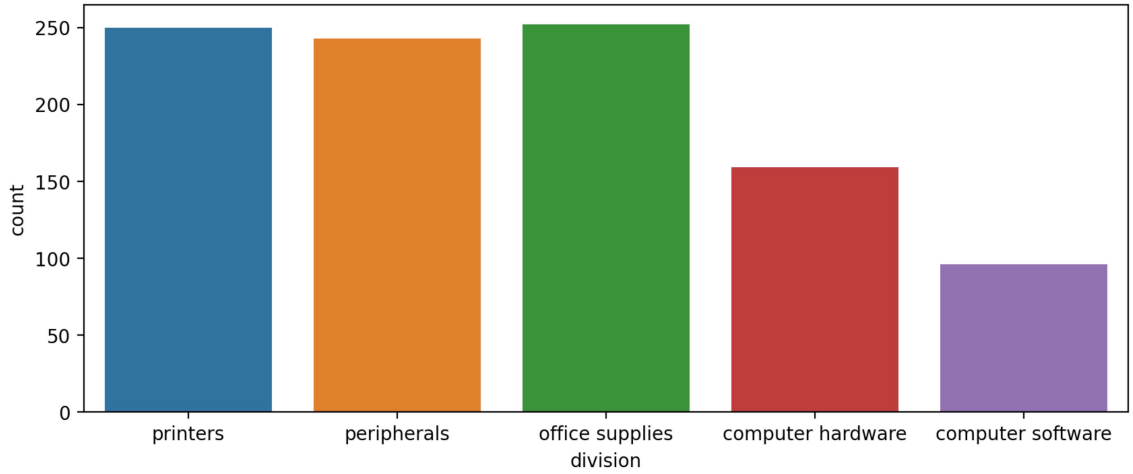
	division	level of education	training level	work experience	salary	sales
0	printers	some college	2	6	91684	372302
1	printers	associate's degree	2	10	119679	495660
2	peripherals	high school	0	9	82045	320453
3	office supplies	associate's degree	2	5	92949	377148
4	office supplies	high school	1	5	71280	312802

Countplot()

A simple plot, it merely shows the total count of rows per category.

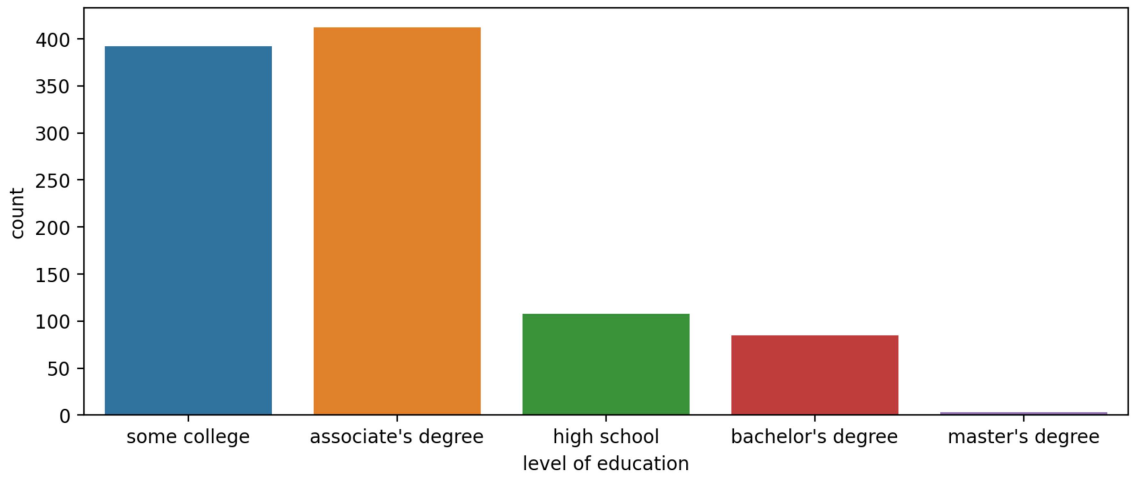
In [4]:
```
plt.figure(figsize=(10,4),dpi=200)
sns.countplot(x='division',data=df)
```

Out[4]:
```
<AxesSubplot:xlabel='division', ylabel='count'>
```



In [5]:
```
plt.figure(figsize=(10,4),dpi=200)
sns.countplot(x='level of education',data=df)
```

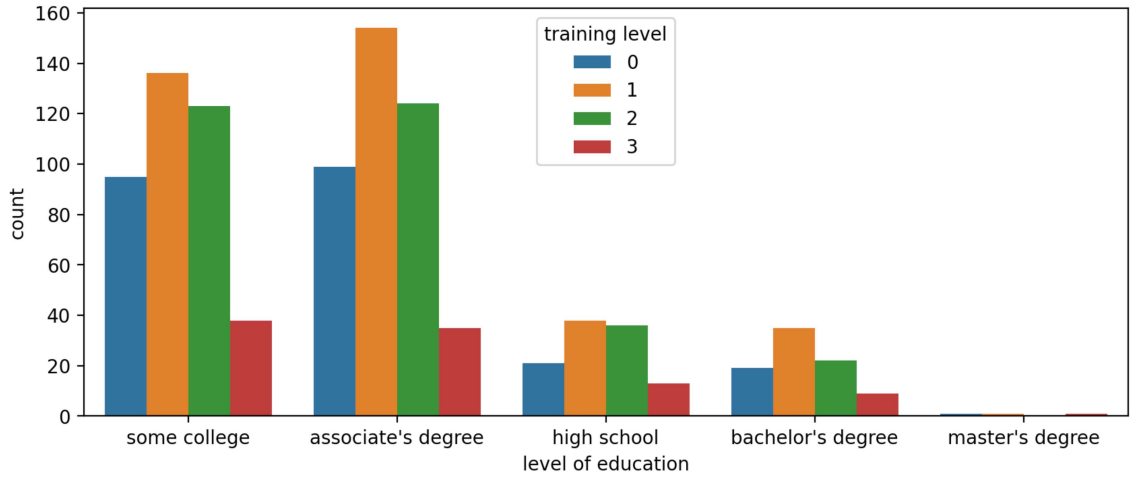
Out[5]:
```
<AxesSubplot:xlabel='level of education', ylabel='count'>
```



Breakdown within another category with 'hue'

In [6]:
```
plt.figure(figsize=(10,4),dpi=200)
sns.countplot(x='level of education',data=df,hue='training level')
```

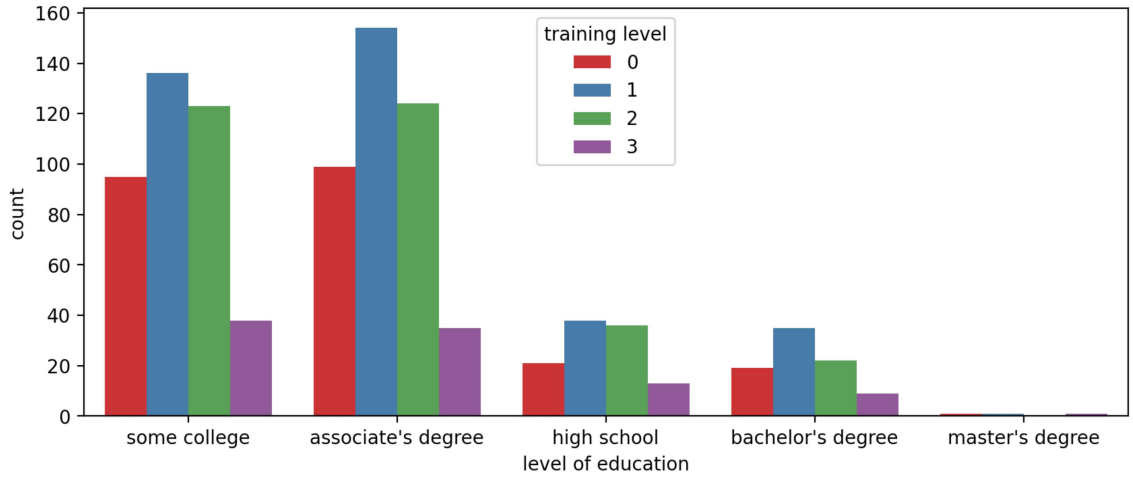
Out[6]:
```
<AxesSubplot:xlabel='level of education', ylabel='count'>
```



NOTE: You can always edit the palette to your liking to any matplotlib [colormap (https://matplotlib.org/3.1.1/gallery/color/colormap_reference.html)](https://matplotlib.org/3.1.1/gallery/color/colormap_reference.html)

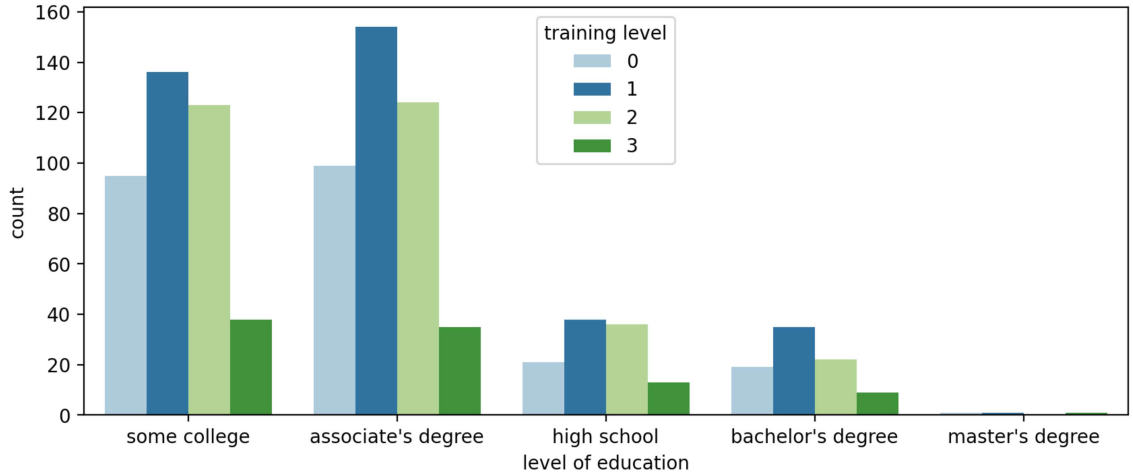
In [7]:
```
plt.figure(figsize=(10,4),dpi=200)
sns.countplot(x='level of education',data=df,hue='training level',palette='
```

Out[7]:
```
<AxesSubplot:xlabel='level of education', ylabel='count'>
```



In [8]:
```
plt.figure(figsize=(10,4),dpi=200)
# Paired would be a good choice if there was a distinct jump from 0 and 1 t
sns.countplot(x='level of education',data=df,hue='training level',palette='
```

Out[8]:
```
<AxesSubplot:xlabel='level of education', ylabel='count'>
```

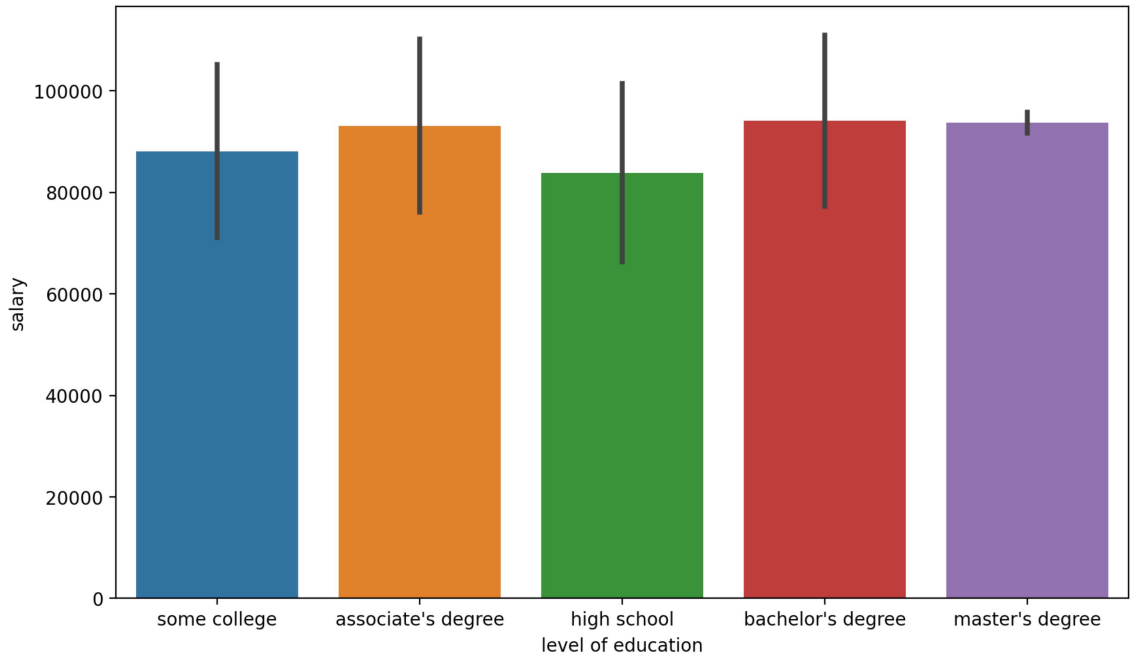


barplot()

So far we've seen the y axis default to a count (similar to a .groupby(x_axis).count() call in pandas). We can expand our visualizations by specifying a specific continuous feature for the y-axis. Keep in mind, you should be careful with these plots, as they may imply a relationship continuity along the y axis where there is none.

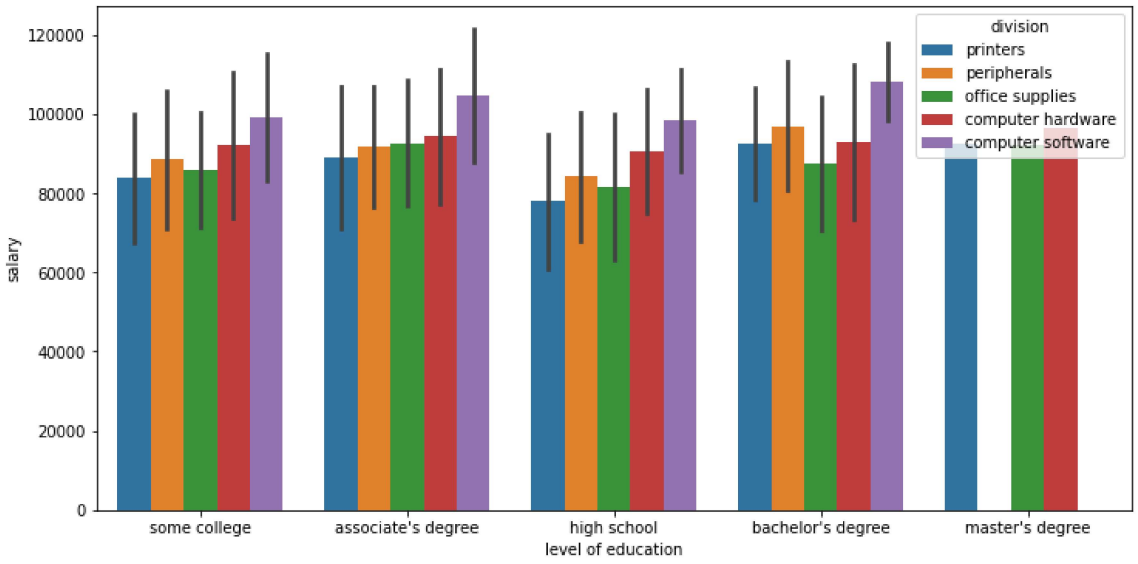
In [9]:
```
plt.figure(figsize=(10,6),dpi=200)
# By default barplot() will show the mean
# Information on the black bar: https://stackoverflow.com/questions/5836247
sns.barplot(x='level of education',y='salary',data=df,estimator=np.mean,ci=
```

Out[9]:
```
<AxesSubplot:xlabel='level of education', ylabel='salary'>
```



In [10]:
```
plt.figure(figsize=(12,6))
sns.barplot(x='level of education',y='salary',data=df,estimator=np.mean,ci=
```

Out[10]:
```
<AxesSubplot:xlabel='level of education', ylabel='salary'>
```



In [11]:
```
plt.figure(figsize=(12,6),dpi=100)

# https://stackoverflow.com/questions/30490740/move-legend-outside-figure-i
sns.barplot(x='level of education',y='salary',data=df,estimator=np.mean,ci=

plt.legend(bbox_to_anchor=(1.05, 1))
```

Out[11]:
```
<matplotlib.legend.Legend at 0x21fa0b503c8>
```