

# seaborn.barplot

## seaborn.barplot

Bar graphs are useful for displaying relationships between categorical data and at least one numerical variable.

`seaborn.countplot` is a barplot where the dependent variable is the number of instances of each instance of the independent variable.

dataset: IMDB 5000 Movie Dataset (<https://www.kaggle.com/deepmatrix/imdb-5000-movie-dataset>)

```
%matplotlib inline
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
plt.rcParams['figure.figsize'] = (20.0, 10.0)
plt.rcParams['font.family'] = "serif"
```

```
df = pd.read_csv('../../../../datasets/movie_metadata.csv')
```

```
df.head()
```

	color	director_name	num_critic_for_reviews	duration	director_facebook_likes	actor_3_facebook_likes
0	Color	James Cameron	723.0	178.0	0.0	855.0
1	Color	Gore Verbinski	302.0	169.0	563.0	1000.0
2	Color	Sam Mendes	602.0	148.0	0.0	161.0
3	Color	Christopher Nolan	813.0	164.0	22000.0	23000.0
4	NaN	Doug Walker	NaN	NaN	131.0	NaN

5 rows × 7 columns

For the bar plot, let's look at the number of movies in each category, allowing each movie to be counted more than once.

```
# split each movie's genre list, then form a set from the unwrapped list of all genres
categories = set([s for genre_list in df.genres.unique() for s in genre_list.split("|")])

# one-hot encode each movie's classification
for cat in categories:
    df[cat] = df.genres.transform(lambda s: int(cat in s))
# drop other columns
df = df[['director_name', 'genres', 'duration'] + list(categories)]
df.head()
```

director_name	genres	duration	Reality-TV	Family	Biography	Comedy	Action	Cri
---------------	--------	----------	------------	--------	-----------	--------	--------	-----

	director_name	genres	duration	Reality-TV	Family	Biography	Comedy	Action	Cri
0	James Cameron	Action Adventure Fantasy Sci-Fi	178.0	0	0	0	0	1	0
1	Gore Verbinski	Action Adventure Fantasy	169.0	0	0	0	0	1	0
2	Sam Mendes	Action Adventure Thriller	148.0	0	0	0	0	1	0
3	Christopher Nolan	Action Thriller	164.0	0	0	0	0	1	0
4	Doug Walker	Documentary	NaN	0	0	0	0	0	0

5 rows × 29 columns

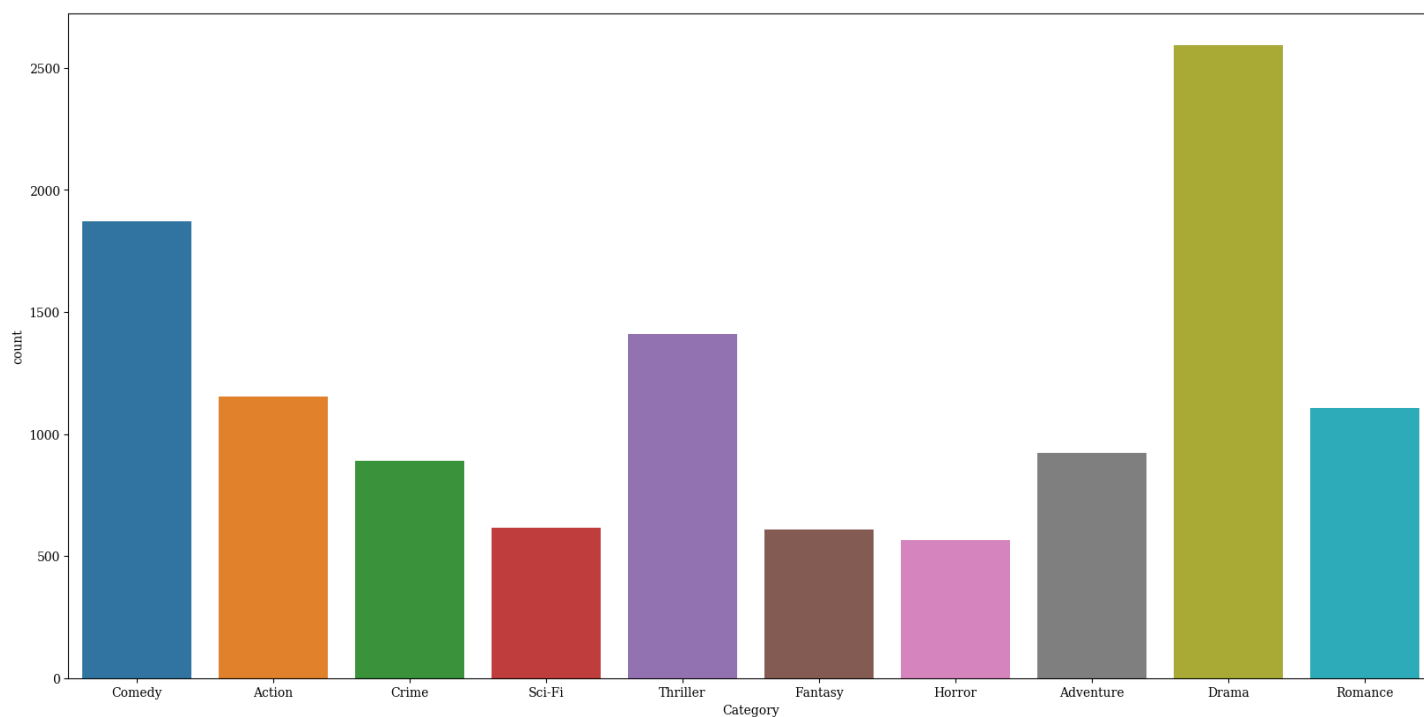
```
# convert from wide to long format and remove null classificaitons
df = pd.melt(df,
             id_vars=['duration'],
             value_vars = list(categories),
             var_name = 'Category',
             value_name = 'Count')
df = df.loc[df.Count>0]
top_categories = df.groupby('Category').aggregate(sum).sort_values('Count', ascending=False).index
howmany=10
# add an indicator whether a movie is short or long, split at 100 minutes runtime
df['islong'] = df.duration.transform(lambda x: int(x > 100))
df = df.loc[df.Category.isin(top_categories[:howmany])]
# sort in descending order
#df = df.loc[df.groupby('Category').transform(sum).sort_values('Count', ascending=False).index]
```

```
df.head()
```

	duration	Category	Count	islong
15136	100.0	Comedy	1	0
15148	106.0	Comedy	1	1
15164	104.0	Comedy	1	1
15170	106.0	Comedy	1	1
15172	103.0	Comedy	1	1

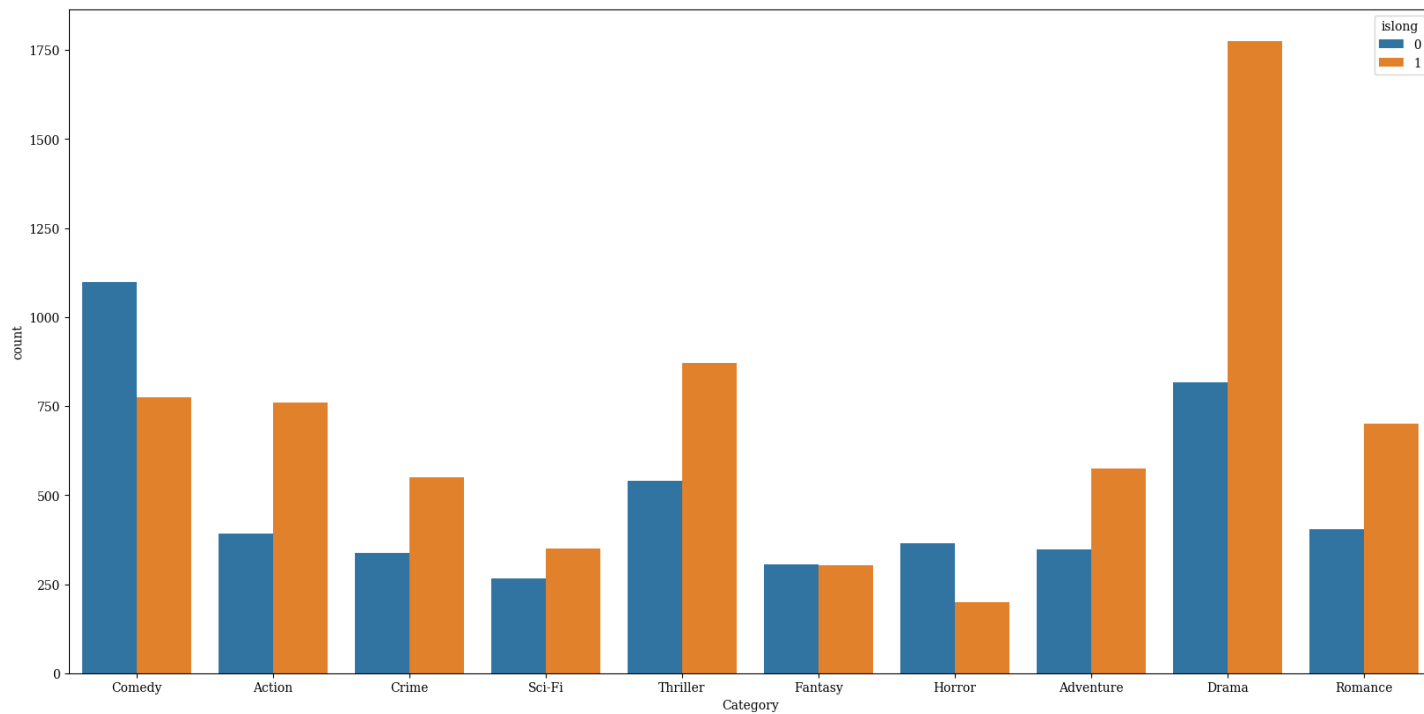
Basic plot

```
p = sns.countplot(data=df, x = 'Category')
```



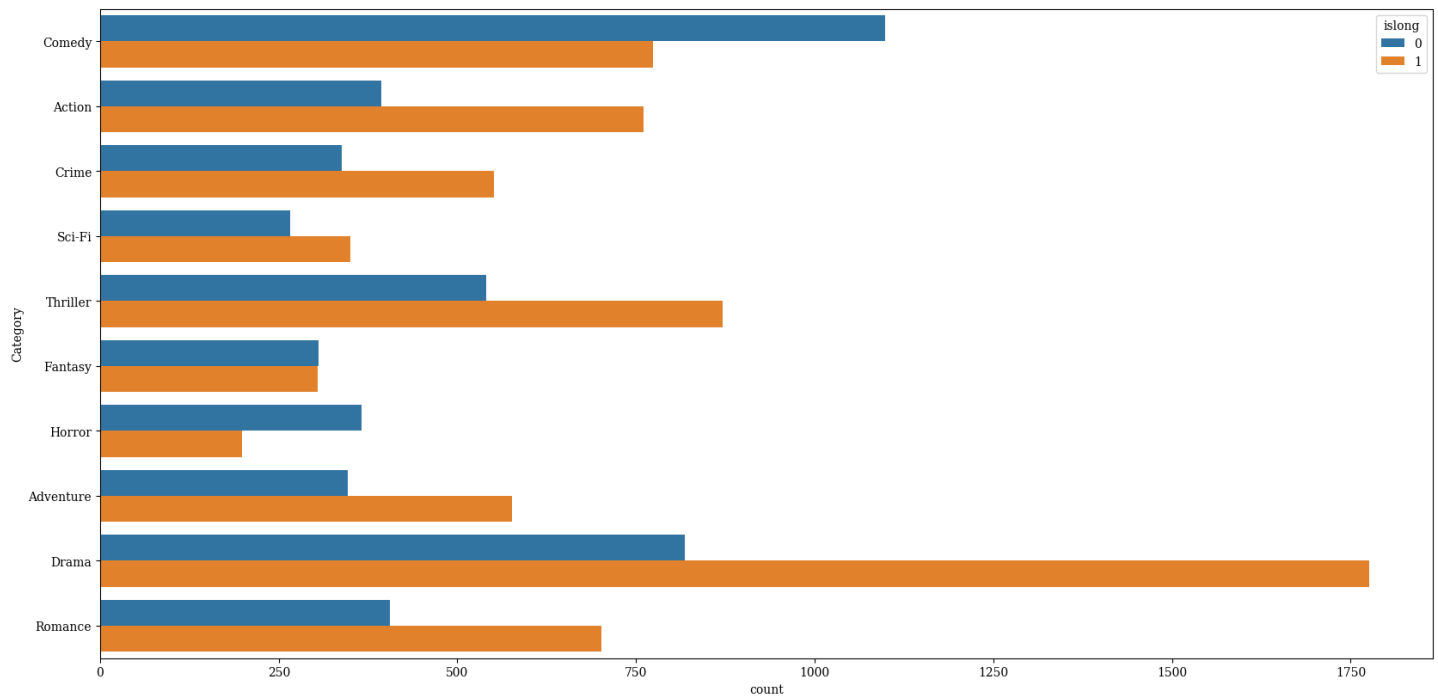
color by a category

```
p = sns.countplot(data=df,  
                  x = 'Category',  
                  hue = 'islong')
```



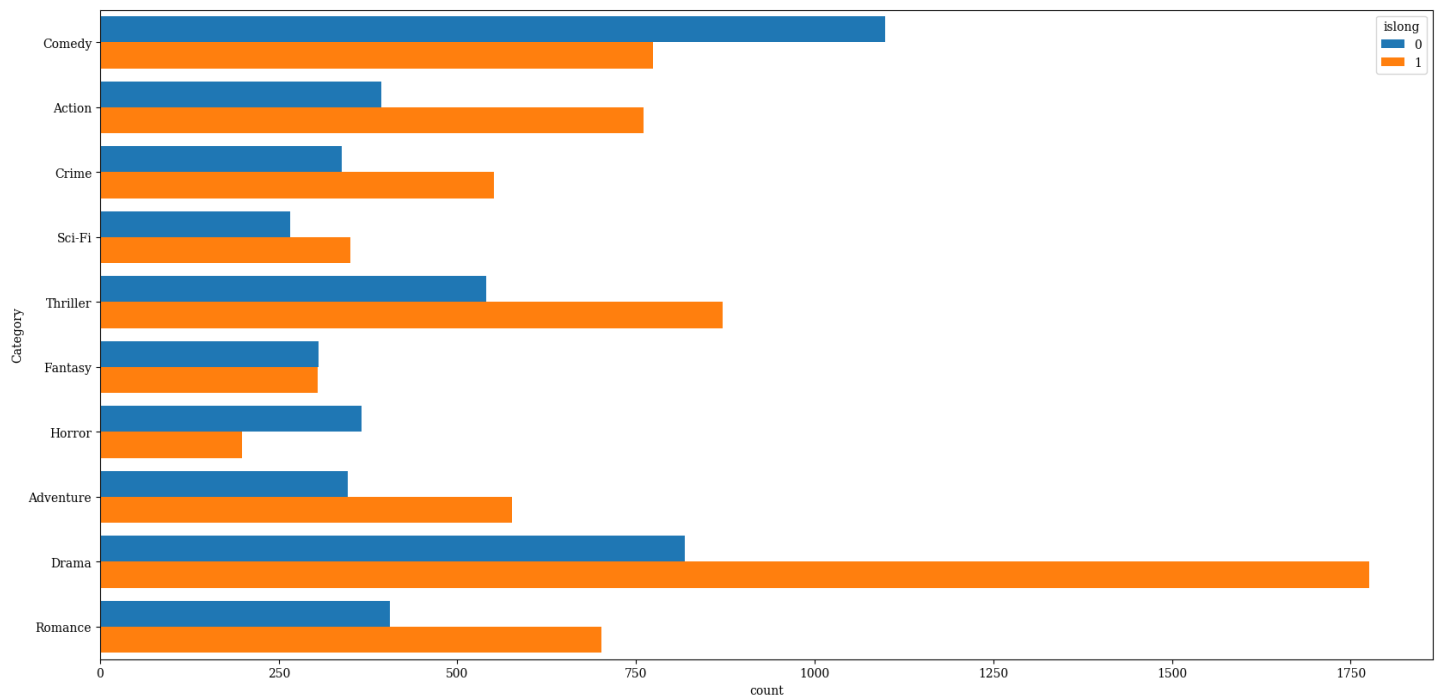
make plot horizontal

```
p = sns.countplot(data=df,  
                  y = 'Category',  
                  hue = 'islong')
```



## Saturation

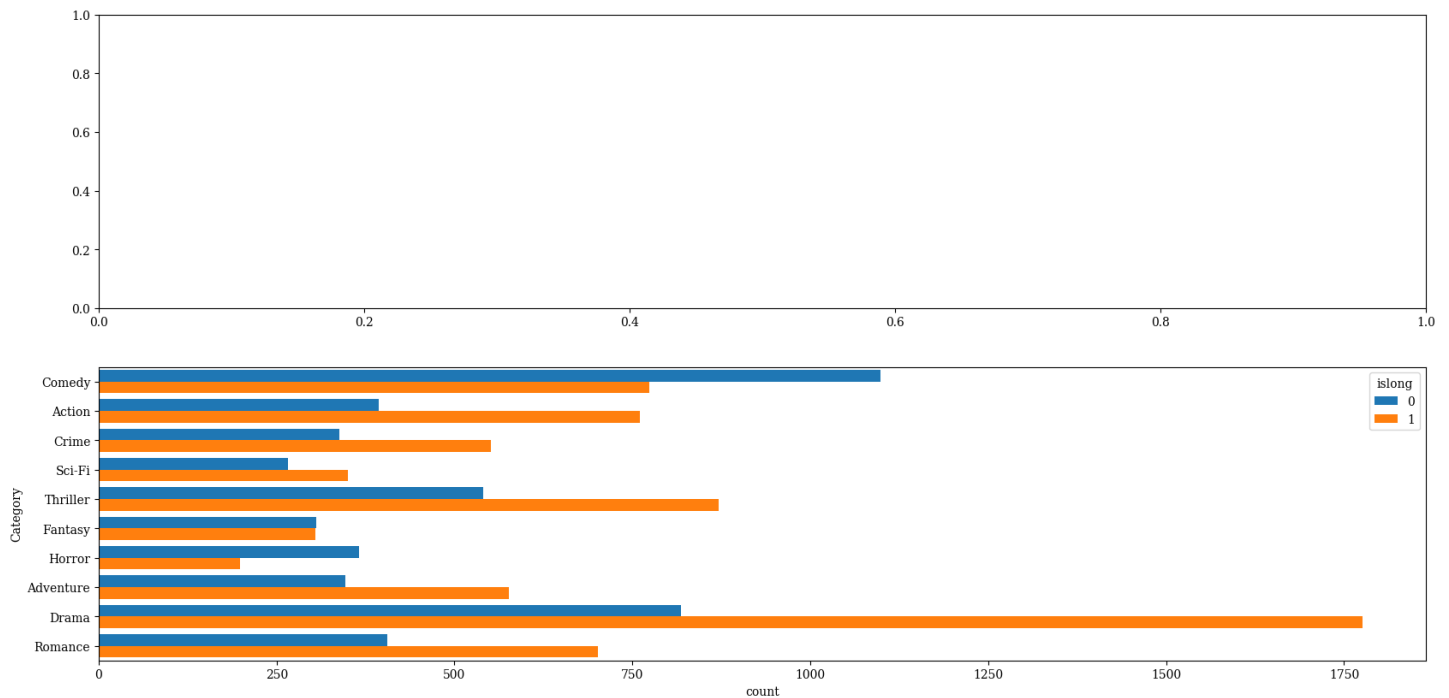
```
p = sns.countplot(data=df,  
                  y = 'Category',  
                  hue = 'islong',  
                  saturation=1)
```



## Targeting a non-default axes

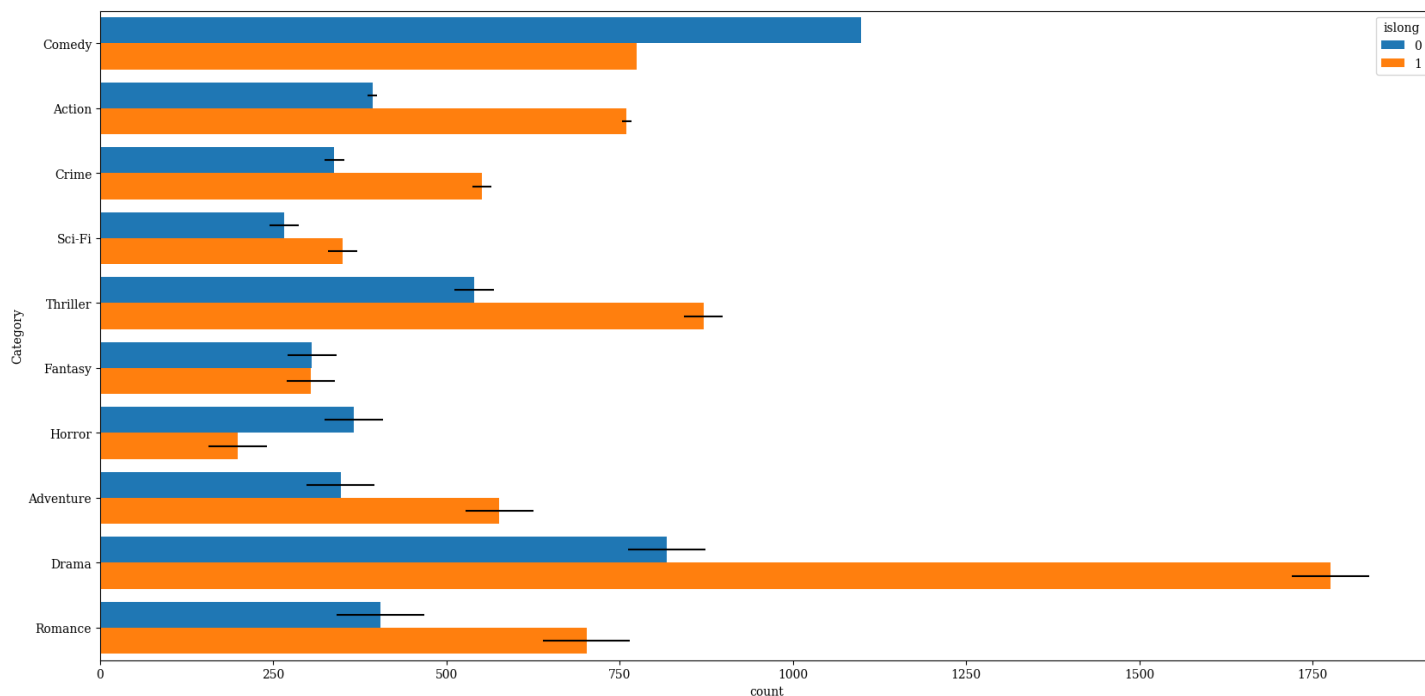
```
import matplotlib.pyplot as plt
fig, ax = plt.subplots(2)
sns.countplot(data=df,
              y = 'Category',
              hue = 'islong',
              saturation=1,
              ax=ax[1])
```

<matplotlib.axes.\_subplots.AxesSubplot at 0x111017278>



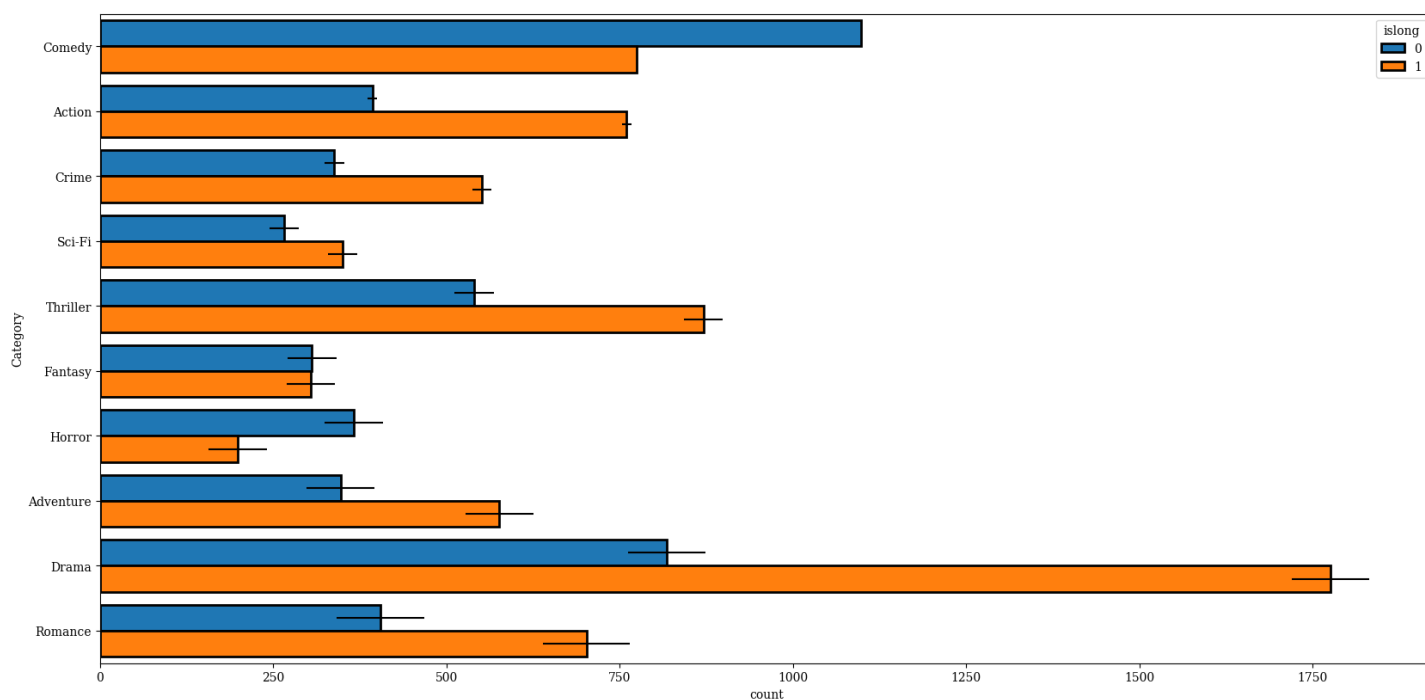
### Add error bars

```
import numpy as np
num_categories = df.Category.unique().size
p = sns.countplot(data=df,
                  y = 'Category',
                  hue = 'islong',
                  saturation=1,
                  xerr=7*np.arange(num_categories))
```



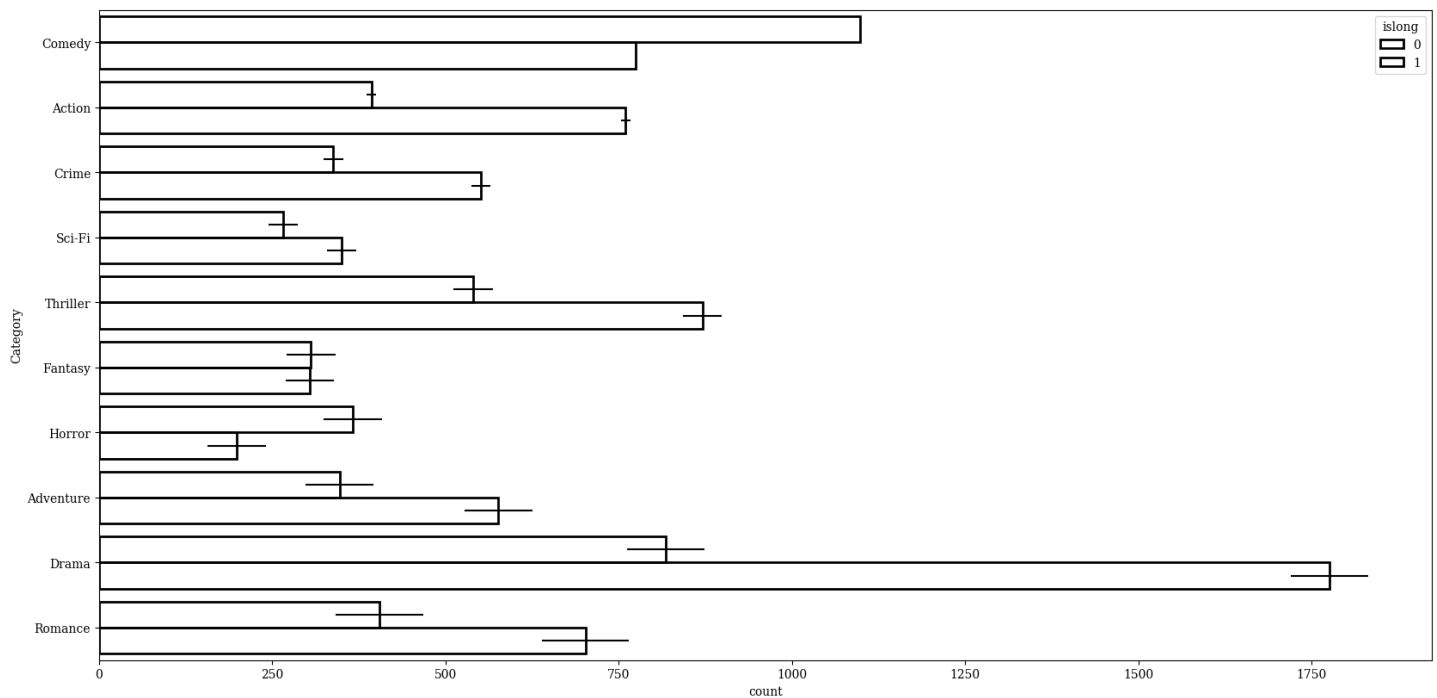
add black bounding lines

```
import numpy as np
num_categories = df.Category.unique().size
p = sns.countplot(data=df,
                  y = 'Category',
                  hue = 'islong',
                  saturation=1,
                  xerr=7*np.arange(num_categories),
                  edgecolor=(0,0,0),
                  linewidth=2)
```

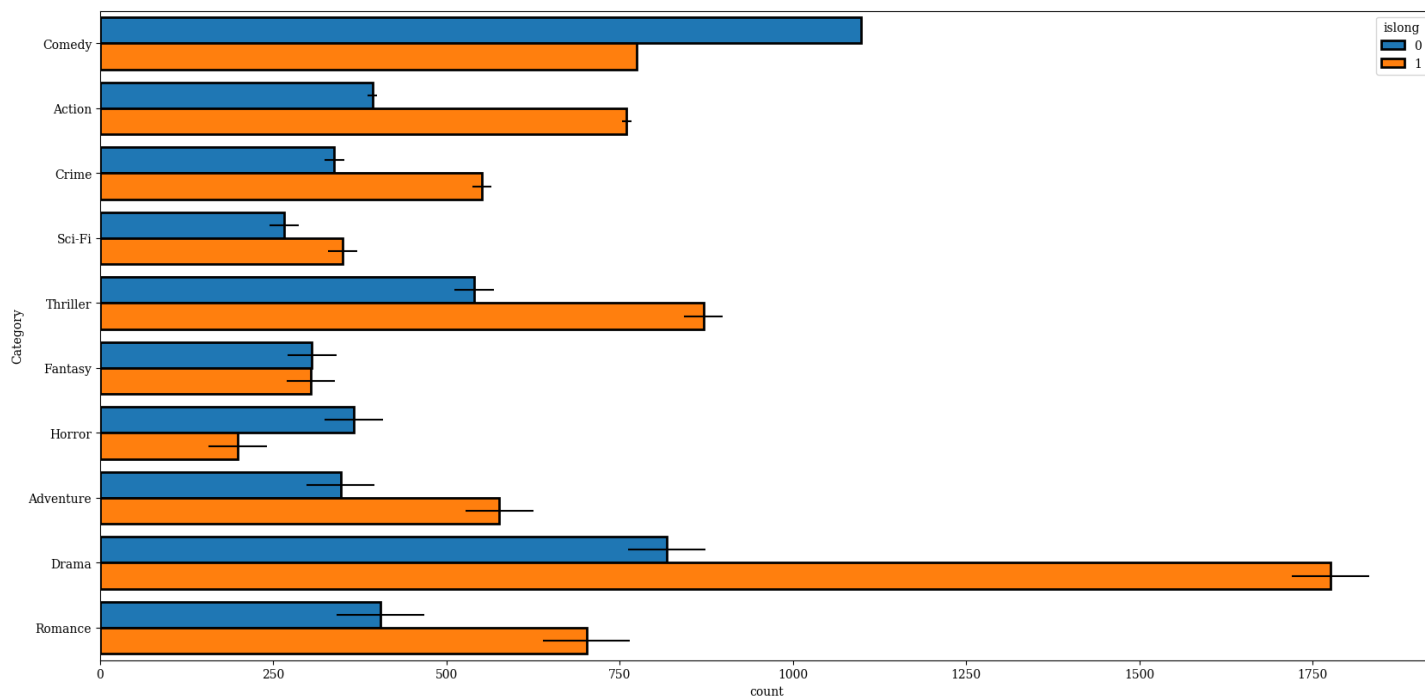


Remove color fill

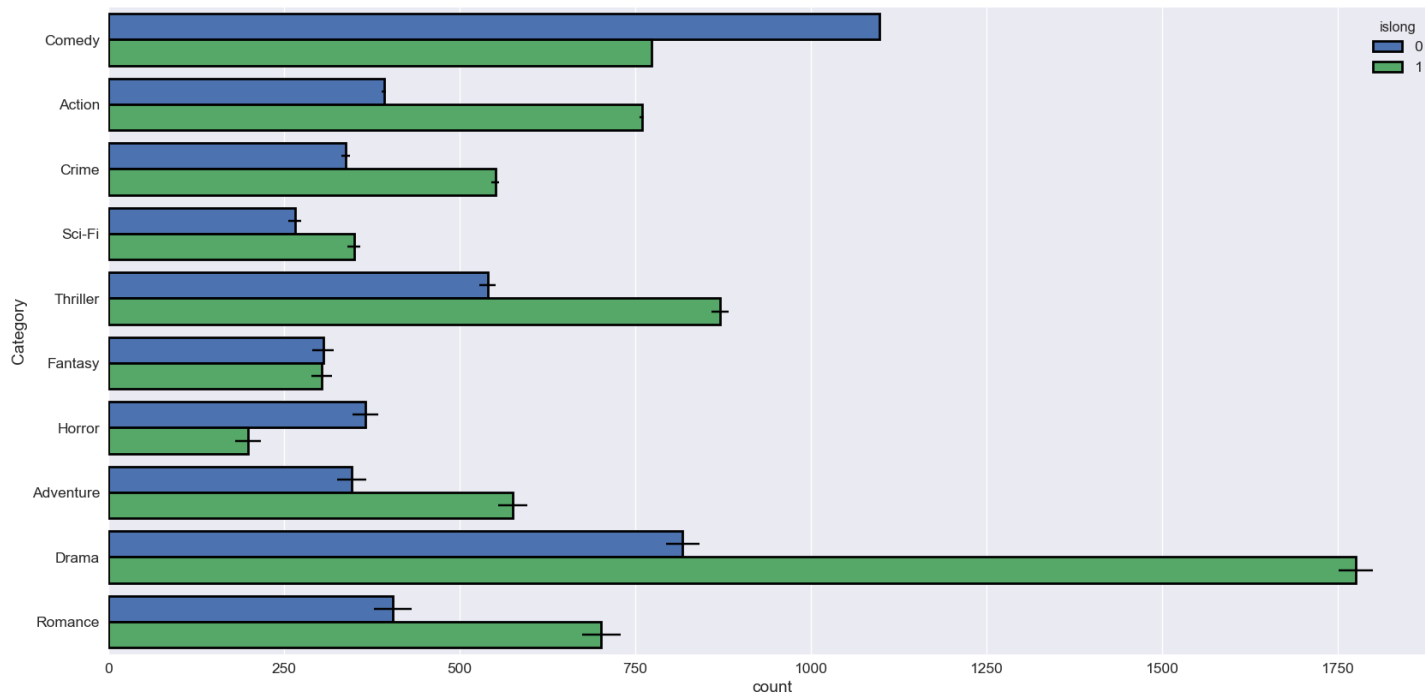
```
import numpy as np
num_categories = df.Category.unique().size
p = sns.countplot(data=df,
                  y = 'Category',
                  hue = 'islong',
                  saturation=1,
                  xerr=7*np.arange(num_categories),
                  edgecolor=(0,0,0),
                  linewidth=2,
                  fill=False)
```



```
import numpy as np
num_categories = df.Category.unique().size
p = sns.countplot(data=df,
                  y = 'Category',
                  hue = 'islong',
                  saturation=1,
                  xerr=7*np.arange(num_categories),
                  edgecolor=(0,0,0),
                  linewidth=2)
```

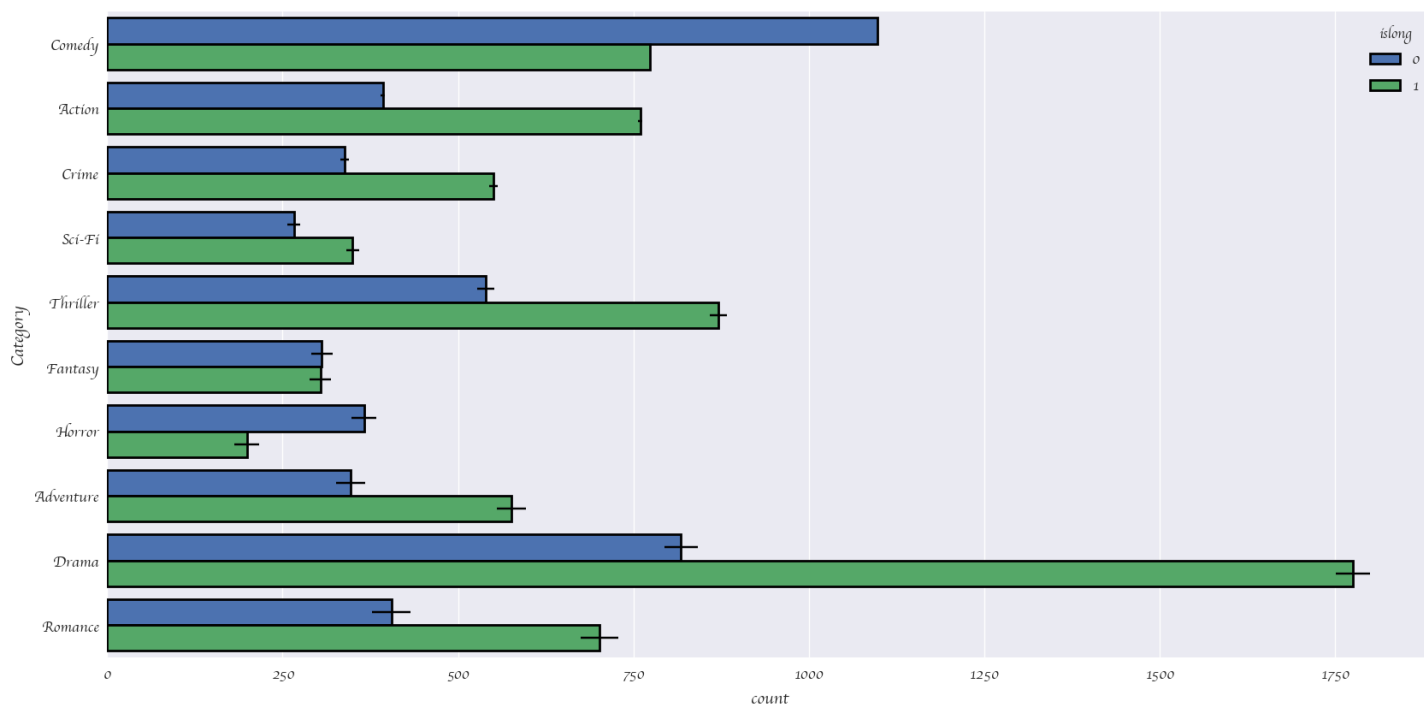


```
sns.set(font_scale=1.25)
num_categories = df.Category.unique().size
p = sns.countplot(data=df,
                  y = 'Category',
                  hue = 'islong',
                  saturation=1,
                  xerr=3*np.arange(num_categories),
                  edgecolor=(0,0,0),
                  linewidth=2)
```

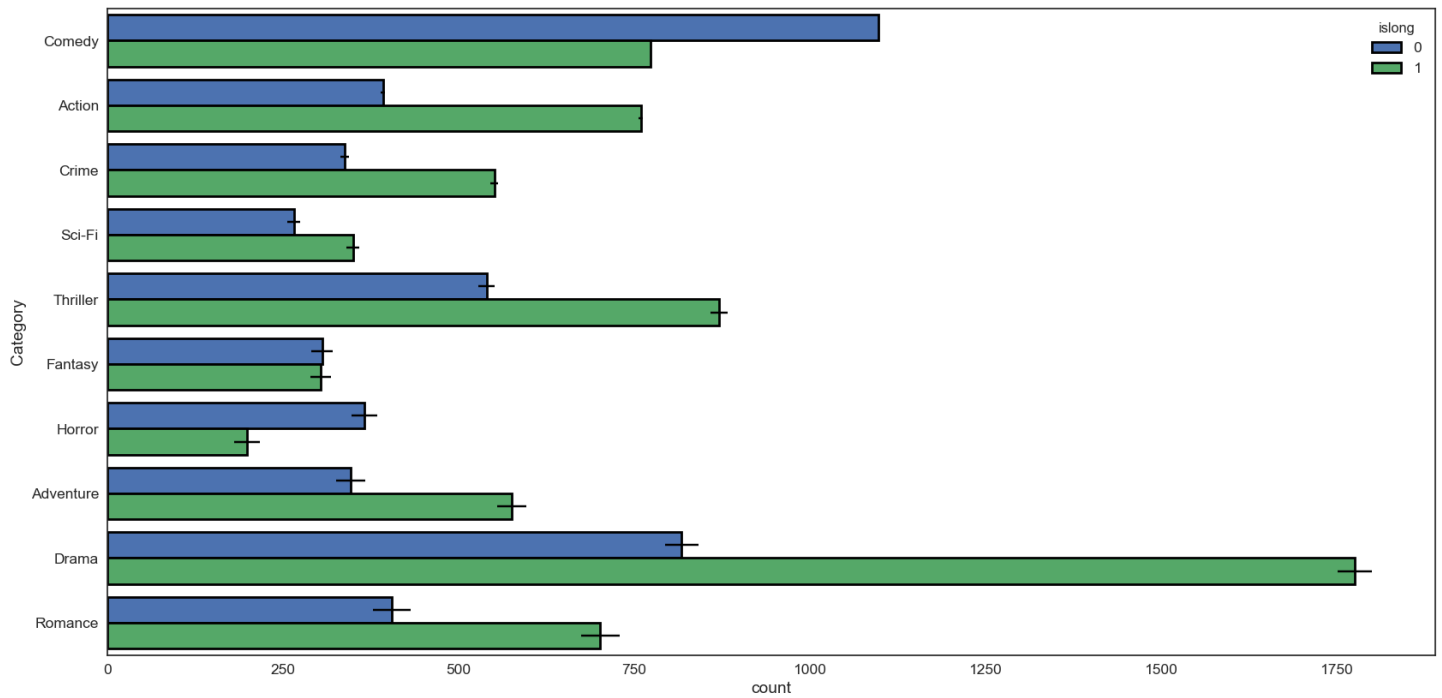




```
plt.rcParams['font.family'] = "cursive"
#sns.set(style="white", font_scale=1.25)
num_categories = df.Category.unique().size
p = sns.countplot(data=df,
                  y = 'Category',
                  hue = 'islong',
                  saturation=1,
                  xerr=3*np.arange(num_categories),
                  edgecolor=(0,0,0),
                  linewidth=2)
```



```
plt.rcParams['font.family'] = 'Times New Roman'
#sns.set_style({'font.family': 'Helvetica'})
sns.set(style="white", font_scale=1.25)
num_categories = df.Category.unique().size
p = sns.countplot(data=df,
                  y = 'Category',
                  hue = 'islong',
                  saturation=1,
                  xerr=3*np.arange(num_categories),
                  edgecolor=(0,0,0),
                  linewidth=2)
```



```

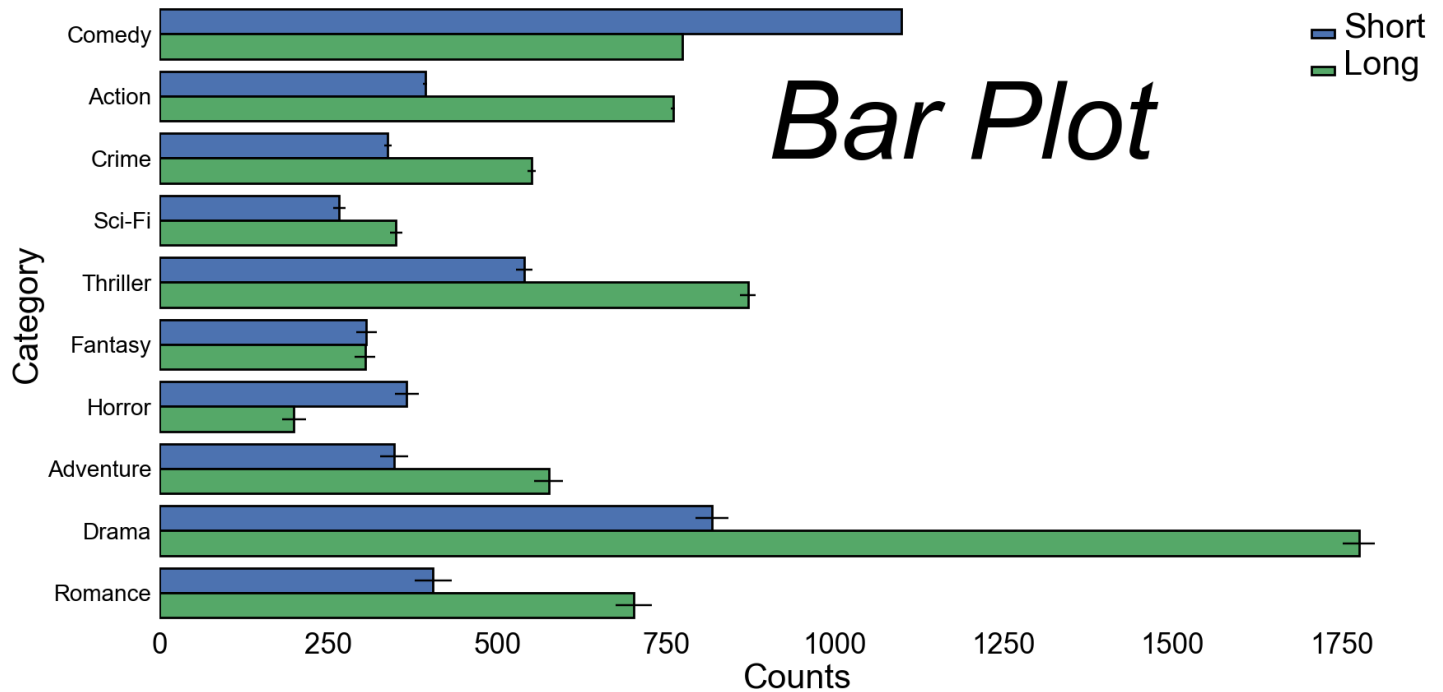
bg_color = 'white'
sns.set(rc={"font.style": "normal",
           "axes.facecolor": bg_color,
           "figure.facecolor": bg_color,
           "text.color": "black",
           "xtick.color": "black",
           "ytick.color": "black",
           "axes.labelcolor": "black",
           "axes.grid": False,
           'axes.labelsize': 30,
           'figure.figsize': (20.0, 10.0),
           'xtick.labelsize': 25,
           'font.size': 20,
           'ytick.labelsize': 20})

#sns.set_style({'font.family': 'Helvetica'})
#sns.set(style="white", font_scale=1.25)
num_categories = df.Category.unique().size
p = sns.countplot(data=df,
                  y = 'Category',
                  hue = 'islong',
                  saturation=1,
                  xerr=3*np.arange(num_categories),
                  edgecolor=(0,0,0),
                  linewidth=2)

leg = p.get_legend()
leg.set_title("")
labs = leg.texts
labs[0].set_text("Short")
labs[0].set_fontsize(25)
labs[0].set_size(30)
labs[1].set_text("Long")
leg.get_title().set_color('black')
p.axes.xaxis.label.set_text("Counts")
plt.text(900,2, "Bar Plot", fontsize = 95, color='Black', fontstyle='italic')

```

```
<matplotlib.text.Text at 0x112bbc400>
```



```
p.get_figure().savefig('../figures/barplot.png')
```

<https://github.com/apryor6> <mailto:apryor6@gmail.com> <https://linkedin.com/in/alan-pryor-02a52b57>

Alan (AJ) Pryor, Jr. • 2017 • [alanpryorjr.com](http://alanpryorjr.com) (<http://apryor6.github.io>)

Theme by beautiful-jekyll (<http://deanattali.com/beautiful-jekyll/>)