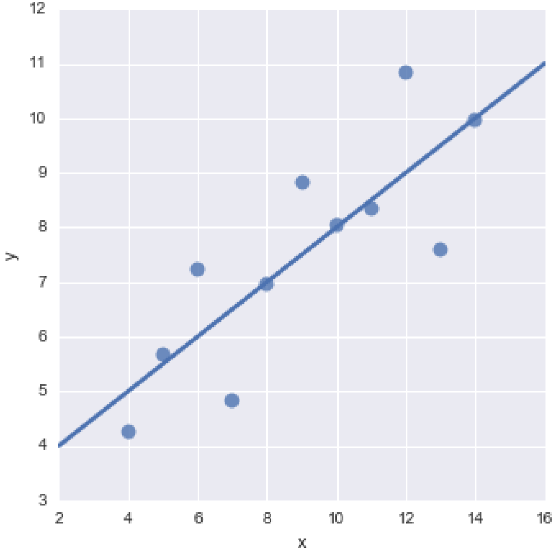
## Seaborn : Visualizing Linear Relationships

- The simple linear regression model used above is very simple to fit, however, it is not appropriate for some kinds of datasets.
- The Anscombes quartet dataset shows a few examples where simple linear regression provides an identical estimate of a relationship where simple visual inspection clearly shows differences.
- For example, in the first case, the linear regression is a good model:

## Seaborn : Visualizing Linear Relationships

```
anscombe = sns.load_dataset("anscombe")
sns.lmplot(x="x", y="y", data=anscombe.query("dataset
           ci=None, scatter_kws={"s": 80});
```

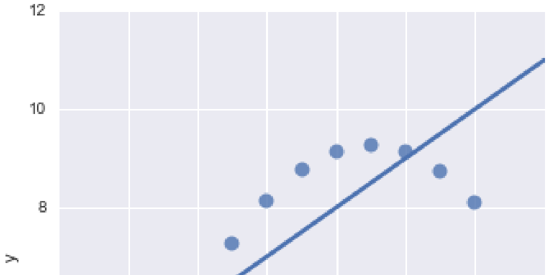
# Seaborn : Visualizing Linear Relationships



## Seaborn : Visualizing Linear Relationships

The linear relationship in the second dataset is the same, but the plot clearly shows that this is not a good model:

```
sns.lmplot(x="x", y="y",
      data=anscombe.query("dataset == 'II'"),
      ci=None, scatter_kws={"s": 80});
```

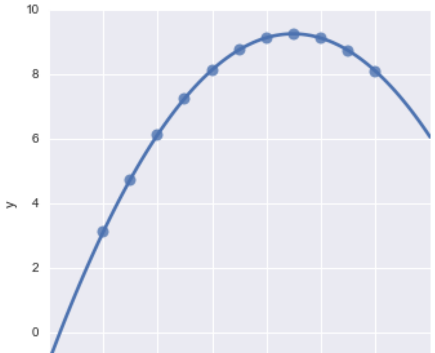


## Seaborn : Visualizing Linear Relationships

In the presence of these kind of higher-order relationships, `lmplot()` and `regplot()` can fit a polynomial regression model to explore simple kinds of nonlinear trends in the dataset:

## Seaborn : Visualizing Linear Relationships

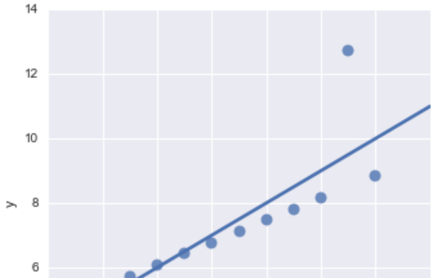
```
sns.lmplot(x="x", y="y",
    data=anscombe.query("dataset == 'II'"),
    order=2, ci=None, scatter_kws={"s": 80});
```



## Seaborn : Visualizing Linear Relationships

A different problem is posed by outlier observations that deviate for some reason other than the main relationship under study:

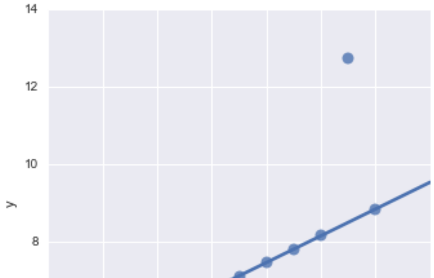
```
sns.lmplot(x="x", y="y", data=anscombe.query("dataset
           ci=None, scatter_kws={"s": 80});
```



## Seaborn : Visualizing Linear Relationships

In the presence of outliers, it can be useful to fit a robust regression, which uses a different loss function to downweight relatively large residuals:

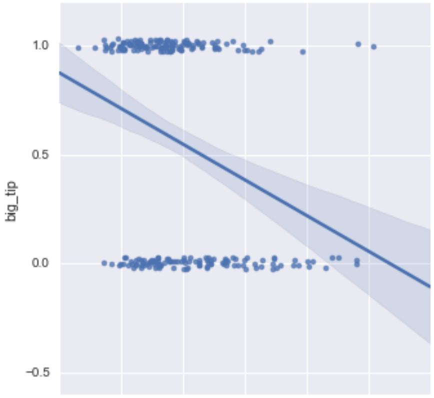
```
sns.lmplot(x="x", y="y", data=anscombe.query("dataset
               robust=True, ci=None, scatter_kws={"s": 80
```



## Seaborn : Visualizing Linear Relationships

When the y variable is binary, simple linear regression also works but provides implausible predictions:

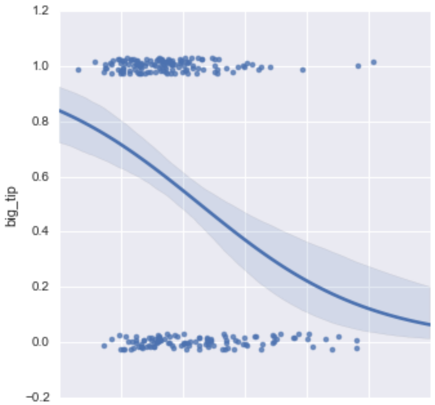
```
tips["big_tip"] = (tips.tip / tips.total_bill) > .15
sns.lmplot(x="total_bill", y="big_tip", data=tips,
           y_jitter=.03);
```



## Seaborn : Visualizing Linear Relationships

The solution in this case is to fit a logistic regression, such that the regression line shows the estimated probability of y = 1 for a given value of x:

```
sns.lmplot(x="total_bill", y="big_tip", data=tips,
           logistic=True, y_jitter=.03);
```

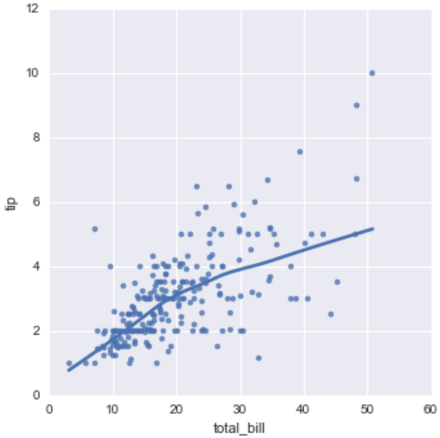


## Seaborn : Visualizing Linear Relationships

- Note that the logistic regression estimate is considerably more computationally intensive (this is true of robust regression as well) than simple regression, and as the confidence interval around the regression line is computed using a bootstrap procedure, you may wish to turn this off for faster iteration (using `ci=False`).
- An altogether different approach is to fit a nonparametric regression using a lowess smoother. This approach has the fewest assumptions, although it is computationally intensive and so currently confidence intervals are not computed at all:

## Seaborn : Visualizing Linear Relationships

```
sns.lmplot(x="total_bill", y="tip", data=tips,
               lowess=True);
```

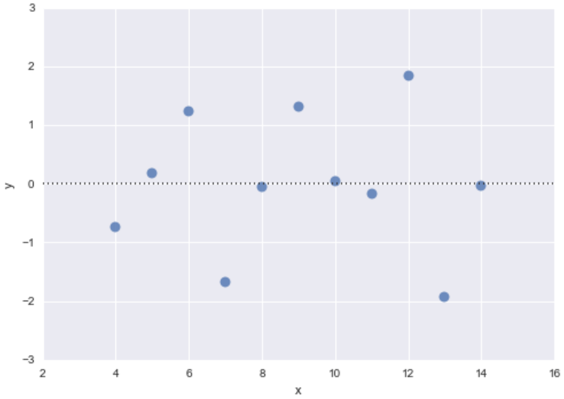


## Seaborn : Visualizing Linear Relationships

- The `residplot()` function can be a useful tool for checking whether the simple regression model is appropriate for a dataset.
- It fits and removes a simple linear regression and then plots the residual values for each observation.
- Ideally, these values should be randomly scattered around $y = 0$:

## Seaborn : Visualizing Linear Relationships

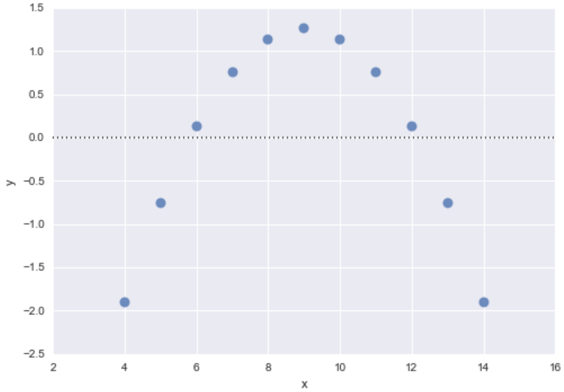
```
sns.residplot(x="x", y="y", data=anscombe.query("datase
scatter_kws={"s": 80});
```



## Seaborn : Visualizing Linear Relationships

If there is structure in the residuals, it suggests that simple linear regression is not appropriate:

```
sns.residplot(x="x", y="y", data=anscombe.query("datase
              scatter_kws={"s": 80});
```



1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2.5
y
2
4
6
8
10
12
14
16
x