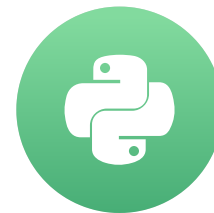


Count data and Poisson distribution

GENERALIZED LINEAR MODELS IN PYTHON



Ita Cirovic Donev
Data Science Consultant

Count data

- Count the **number of occurrences** in a specified **unit of** time, distance, area or volume

Examples:

- Goals in a soccer match
- Number of earthquakes
- Number of crab satellites
- Number of awards won by a person
- Number of bike crossings over the bridge

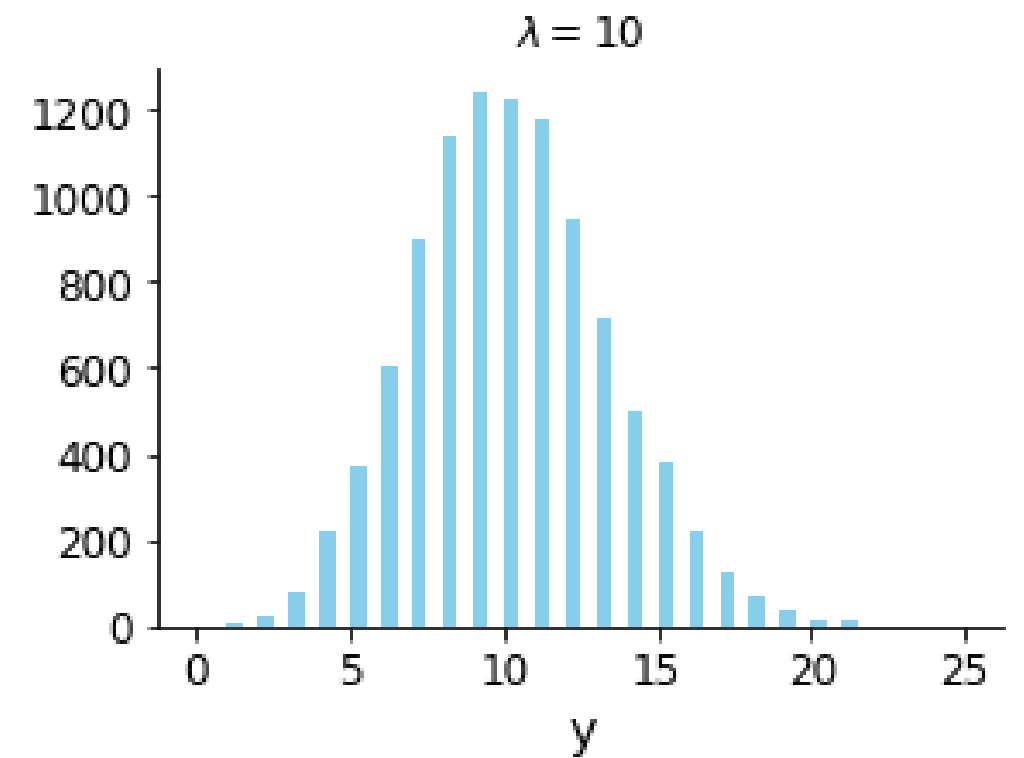
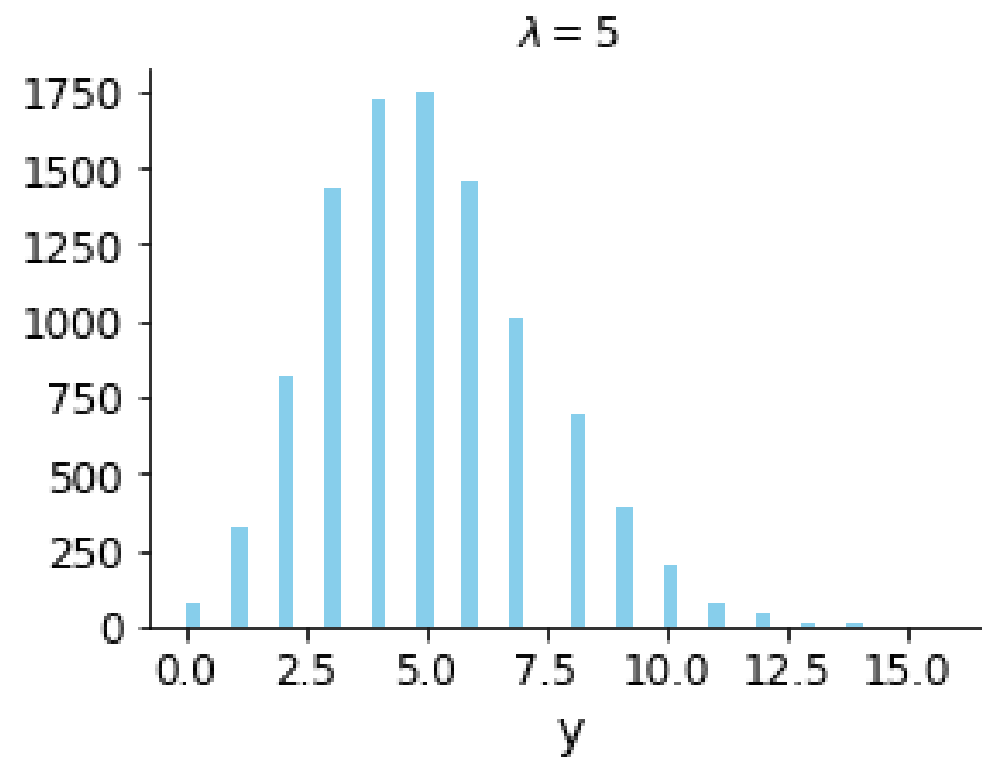
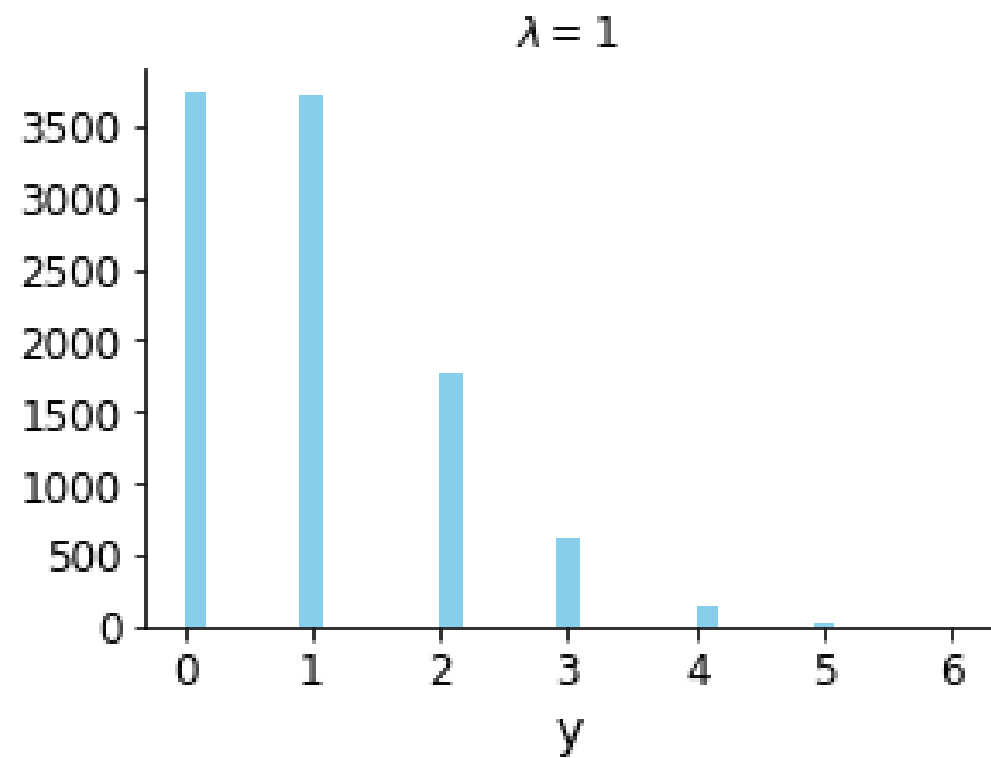
Poisson random variable

- Events occur independently and randomly
- Poisson distribution

$$P(y) = \frac{\lambda^y e^{-\lambda}}{y!}$$

- λ : mean and variance
- $y = 0, 1, 2, 3, \dots$
 - Always positive
 - Discrete (*not continuous*)
 - Lower bound at zero, but no upper bound

Understanding the parameter of the Poisson distribution



Visualizing the response

```
import seaborn as sns
```

```
sns.distplot('y')
```

Poisson regression

- Response variable

$$y \sim \text{Poisson}(\lambda)$$

- Mean of the response

$$E(y) = \lambda$$

- Poisson regression model

$$\log(\lambda) = \beta_0 + \beta_1 x_1$$

Explanatory variables

- Continuous and/or categorical → **Poisson regression model**
- Categorical → **log-linear model**

GLM with Poisson in Python

```
import statsmodels.api as sm
from statsmodels.formula.api import glm
```

```
glm('y ~ x',
    data = my_data,
    family = sm.families.Poisson())
```


Let's practice!

GENERALIZED LINEAR MODELS IN PYTHON

Interpreting model fit

GENERALIZED LINEAR MODELS IN PYTHON



Ita Cirovic Donev
Data Science Consultant

Parameter estimation

- Maximum likelihood estimation (MLE)
- Iteratively reweighted least squares (IRLS)

The response function

- Poisson regression model

$$\log(\lambda) = \beta_0 + \beta_1 x_1$$

- The response function:

$$\lambda = \exp(\beta_0 + \beta_1 x_1)$$

or

$$\lambda = \exp(\beta_0) \times \exp(\beta_1 x_1)$$

The response function

- Poisson regression model

$$\log(\lambda) = \beta_0 + \beta_1 x_1$$

- The response function:

$$\lambda = \exp(\beta_0 + \beta_1 x_1)$$

or

$$\lambda = \exp(\beta_0) \times \exp(\beta_1 x_1)$$

Interpretation of parameters

- $\exp(\beta_0)$
 - The effect on the mean λ when $x = 0$
- $\exp(\beta_1)$
 - The **multiplicative effect** on the mean λ for a **1-unit increase** in x

Interpreting coefficient effect

- If $\beta_1 > 0$
 - $\exp(\beta_1) > 1$
 - λ is $\exp(\beta_1)$ times larger than when $x = 0$
- If $\beta_1 = 0$
 - $\exp(\beta_1) = 1$
 - $\lambda = \exp(\beta_0)$
 - Multiplicative factor is 1
 - y and x are not related
- If $\beta < 0$
 - $\exp(\beta_1) < 1$
 - λ is $\exp(\beta_1)$ times smaller than when $x = 0$

Example

```
model = glm('sat ~ weight', data = crab,  
            family = sm.families.Poisson()).fit()
```

Generalized Linear Model Regression Results (print cut)

```
=====
```

	coef	std err	z	P> z	[0.025	0.975]
Intercept	-0.4284	0.179	-2.394	0.017	-0.779	-0.078
weight	0.5893	0.065	9.064	0.000	0.462	0.717

```
=====
```


Example - interpretation of beta

- Extract model coefficients

```
model.params
```

```
Intercept    -0.428405  
weight        0.589304
```

- Compute the effect

```
np.exp(0.589304)
```

```
1.803
```

Confidence interval for ...

- β_1

```
print(model.conf_int())
```

	0	1
Intercept	-0.779112	-0.077699
weight	0.461873	0.716735

- The multiplicative effect on mean

```
print(np.exp(crab_fit.conf_int()))
```

	0	1
Intercept	0.458813	0.925243
weight	1.587044	2.047737

Let's practice!

GENERALIZED LINEAR MODELS IN PYTHON

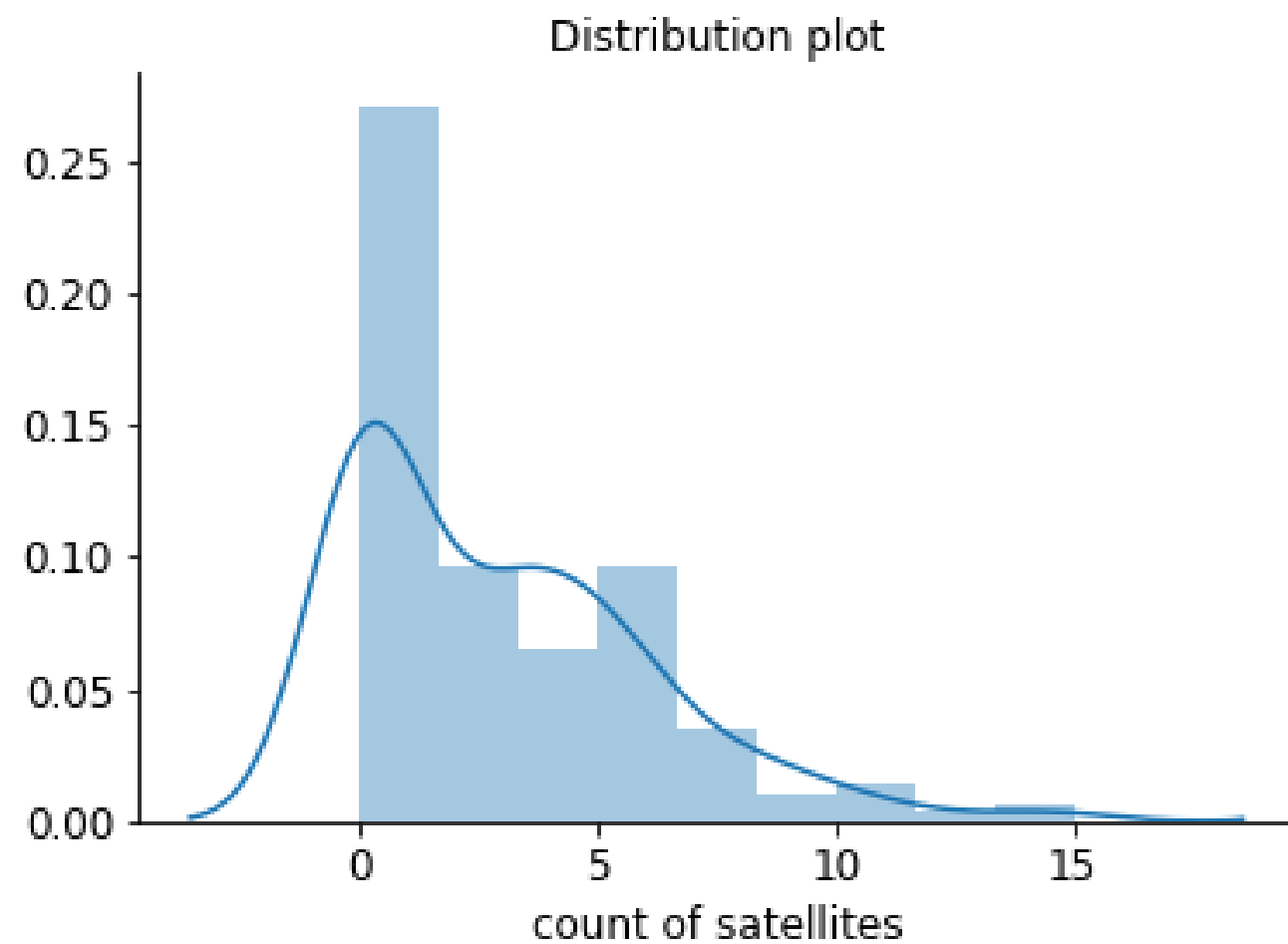
The Problem of Overdispersion

GENERALIZED LINEAR MODELS IN PYTHON



Ita Cirovic Donev
Data Science Consultant

Understanding the data



```
# mean of y  
y_mean = crab['sat'].mean()
```

```
2.919
```

```
# variance of y  
y_variance = crab['sat'].var()
```

```
9.912
```

Mean not equal to variance

- $\text{variance} > \text{mean} \rightarrow$ overdispersion
- $\text{variance} < \text{mean} \rightarrow$ underdispersion

Consequences:

- Small standard errors
- Small p-value

How to check for overdispersion?

```
Generalized Linear Model Regression Results
=====
Dep. Variable:          sat      No. Observations:          173
Model:                  GLM      Df Residuals:              171
Model Family:          Poisson  Df Model:                  1
Link Function:         log      Scale:                    1.0000
Method:                 IRLS     Log-Likelihood:           -458.08
Date:                  Tue, 05 Mar 2019    Deviance:                 560.87
Time:                  18:21:13    Pearson chi2:             536.
No. Iterations:        5          Covariance Type:          nonrobust
=====
               coef      std err          z      P>|z|      [0.025      0.975]
-----
Intercept    -0.4284      0.179     -2.394     0.017    -0.779    -0.078
weight       0.5893      0.065      9.064     0.000      0.462      0.717
=====
```

Compute estimated overdispersion

```
ratio = crab_fit.pearson_chi2 / crab_fit.df_resid  
print(ratio)
```

```
3.134
```

- $\text{Ratio} = 1 \rightarrow$ approximately Poisson
- $\text{Ratio} < 1 \rightarrow$ underdispersion
- $\text{Ratio} > 1 \rightarrow$ overdispersion

Negative Binomial Regression

- $E(y) = \lambda$
- $Var(y) = \lambda + \alpha\lambda^2$
- α - dispersion parameter

GLM negative Binomial in Python

```
import statsmodels.api as sm
from statsmodels.formula.api import glm
```

```
model = glm('y ~ x', data = my_data,
            family = sm.families.NegativeBinomial(alpha = 1)).fit()
```

Let's practice!

GENERALIZED LINEAR MODELS IN PYTHON

Plotting a regression model

GENERALIZED LINEAR MODELS IN PYTHON



Ita Cirovic Donev
Data Science Consultant

Import libraries

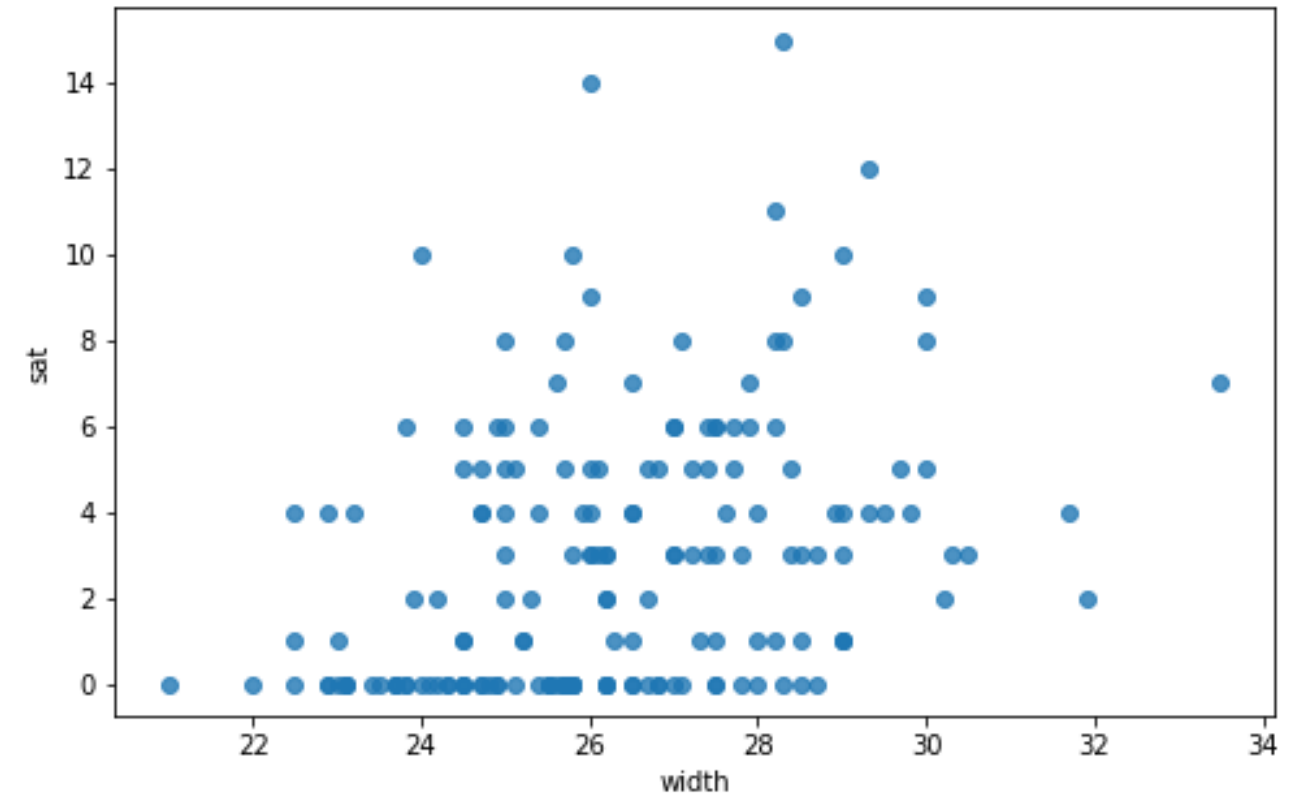
```
import seaborn as sns
import matplotlib.pyplot as plt
```

- Crab model `'sat ~ width'` is saved as `model`

Plot data points

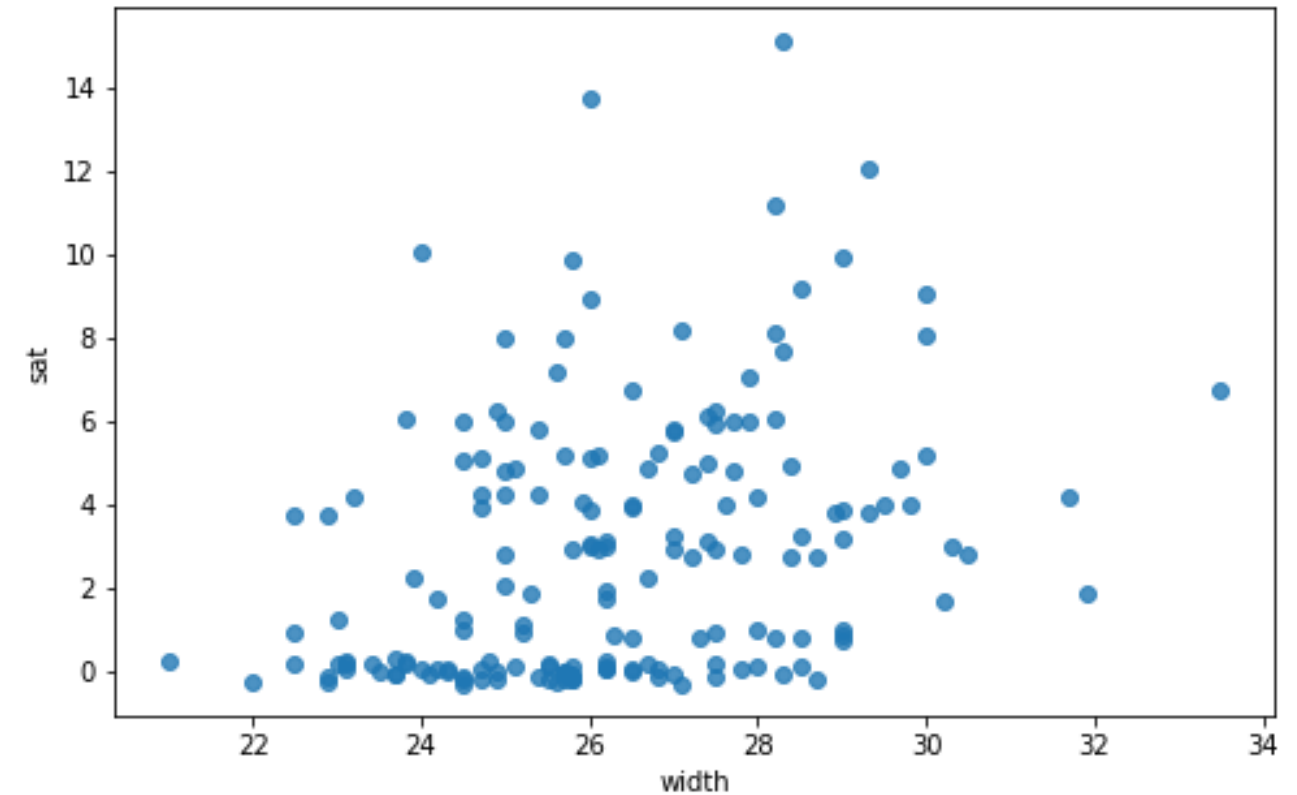
```
# Adjust figure size  
plt.subplots(figsize = (8, 5))
```

```
# Plot data points  
sns.regplot('width', 'sat',  
            data = crab,  
            fit_reg = False)
```



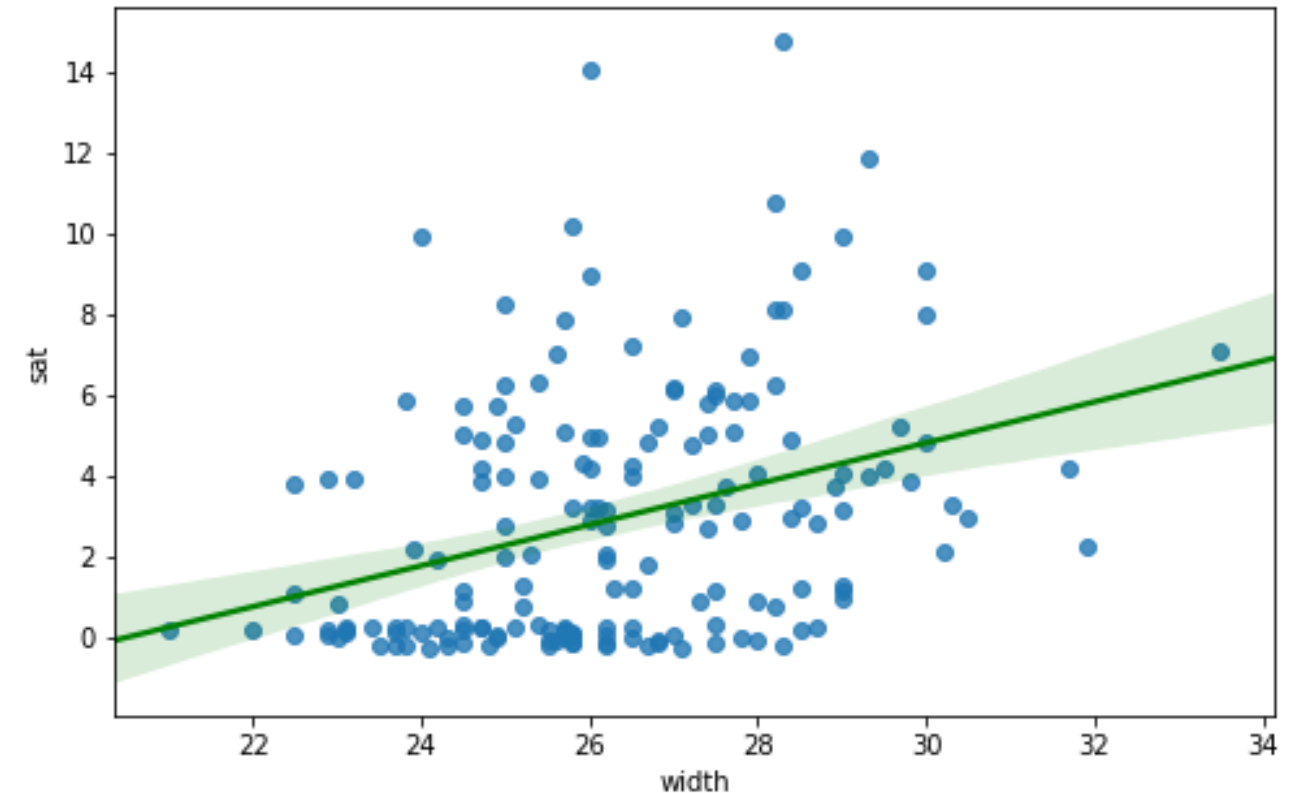
Add jitter

```
sns.regplot('width', 'sat',  
            data = crab,  
            fit_reg = False,  
            y_jitter = 0.3)
```



Add linear fit

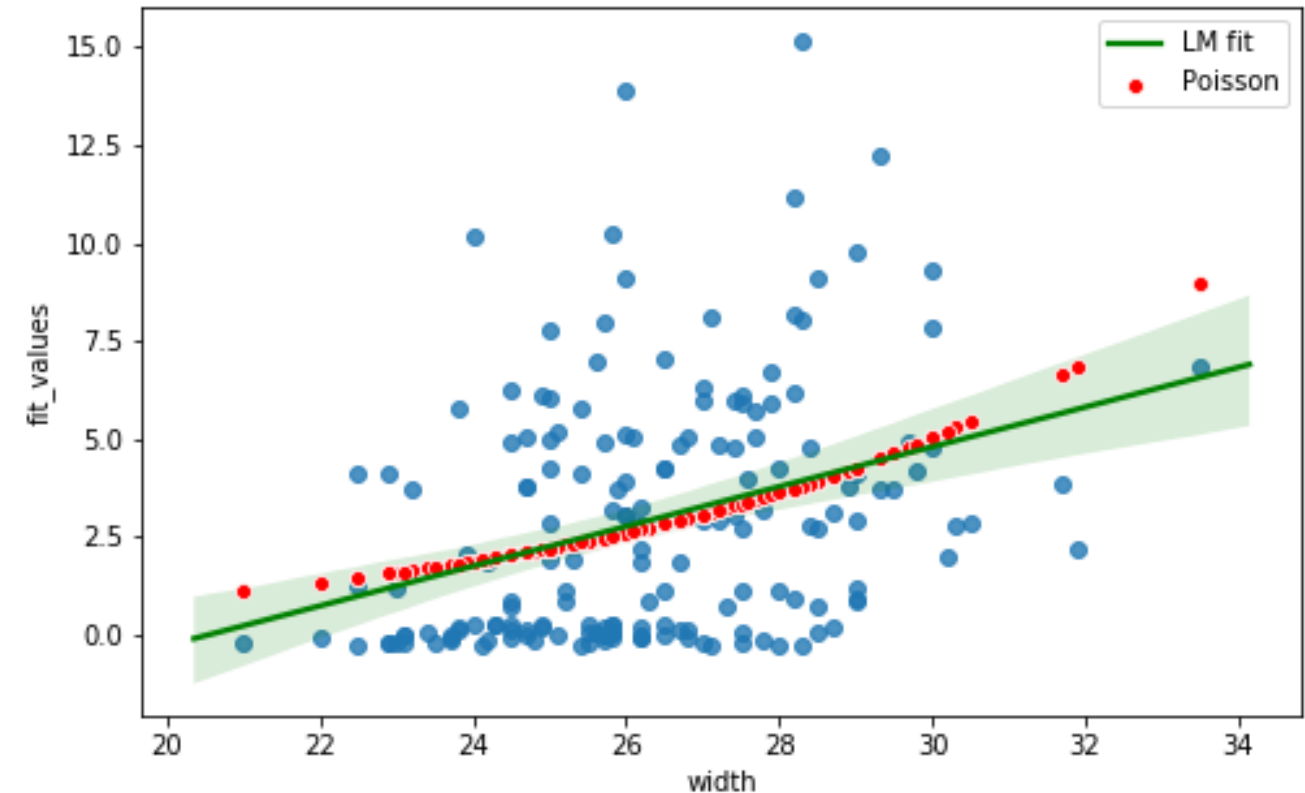
```
sns.regplot('width', 'sat',  
            data = crab,  
            y_jitter = 0.3,  
            fit_reg = True,  
            line_kws = {'color': 'green',  
                        'label': 'LM fit'})
```



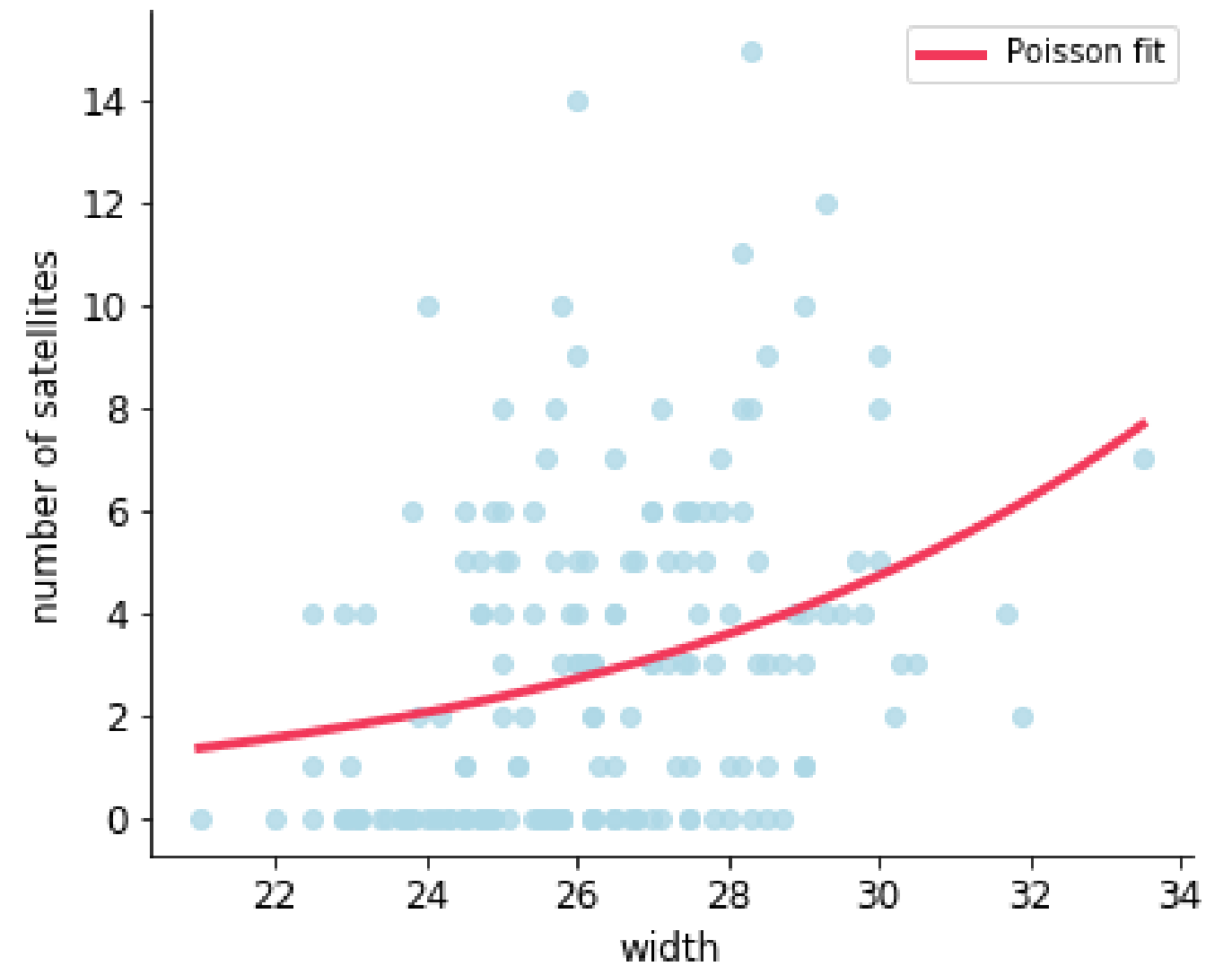
Add Poisson GLM estimated values

```
crab['fit_values'] = model.fittedvalues
```

```
sns.scatterplot('width', 'fit_values',  
                data = crab,  
                color = 'red',  
                label = 'Poisson')
```



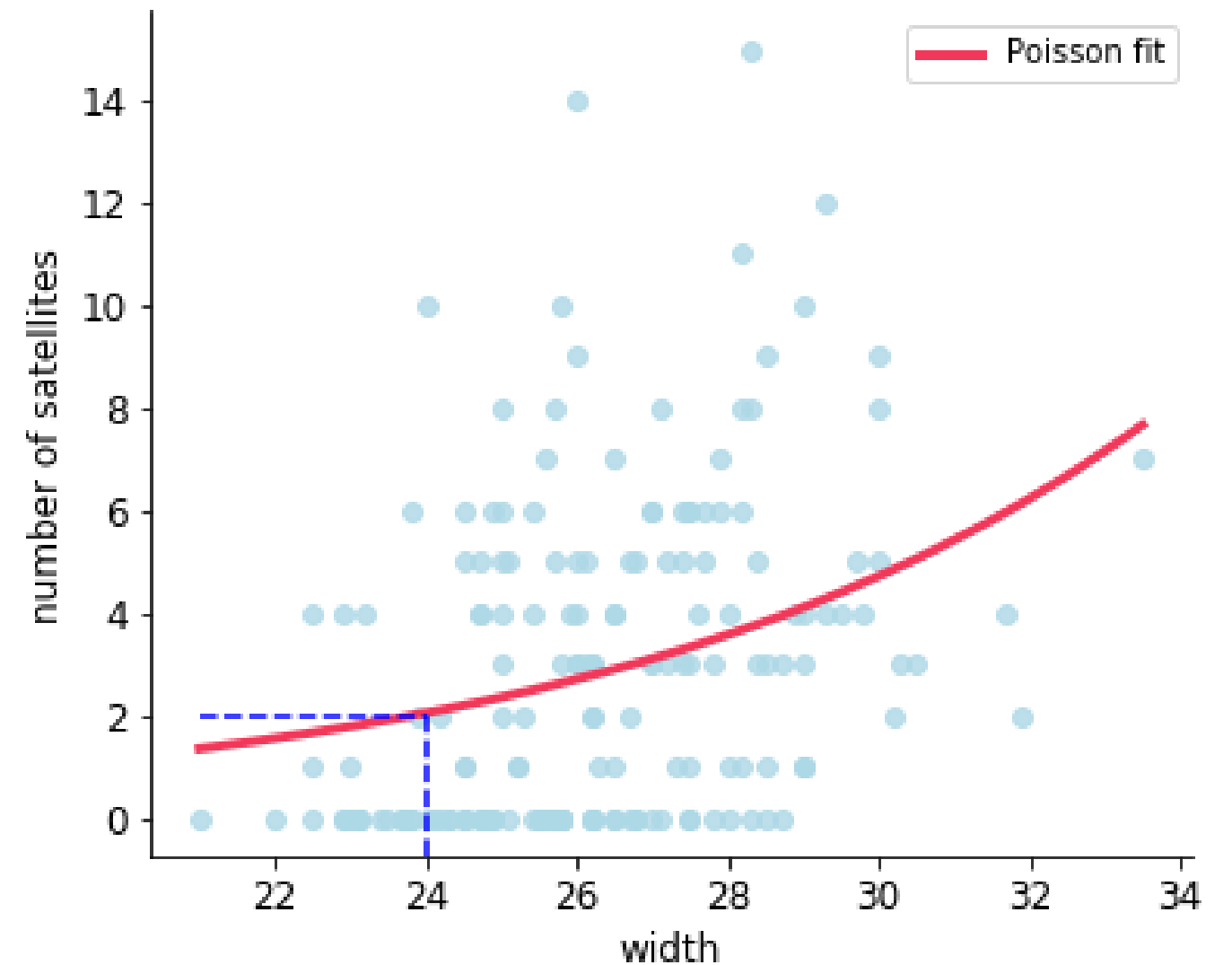
Predictions



Predictions

```
new_data = pd.DataFrame({'width': [24, 28, 32]})  
model.predict(new_data)
```

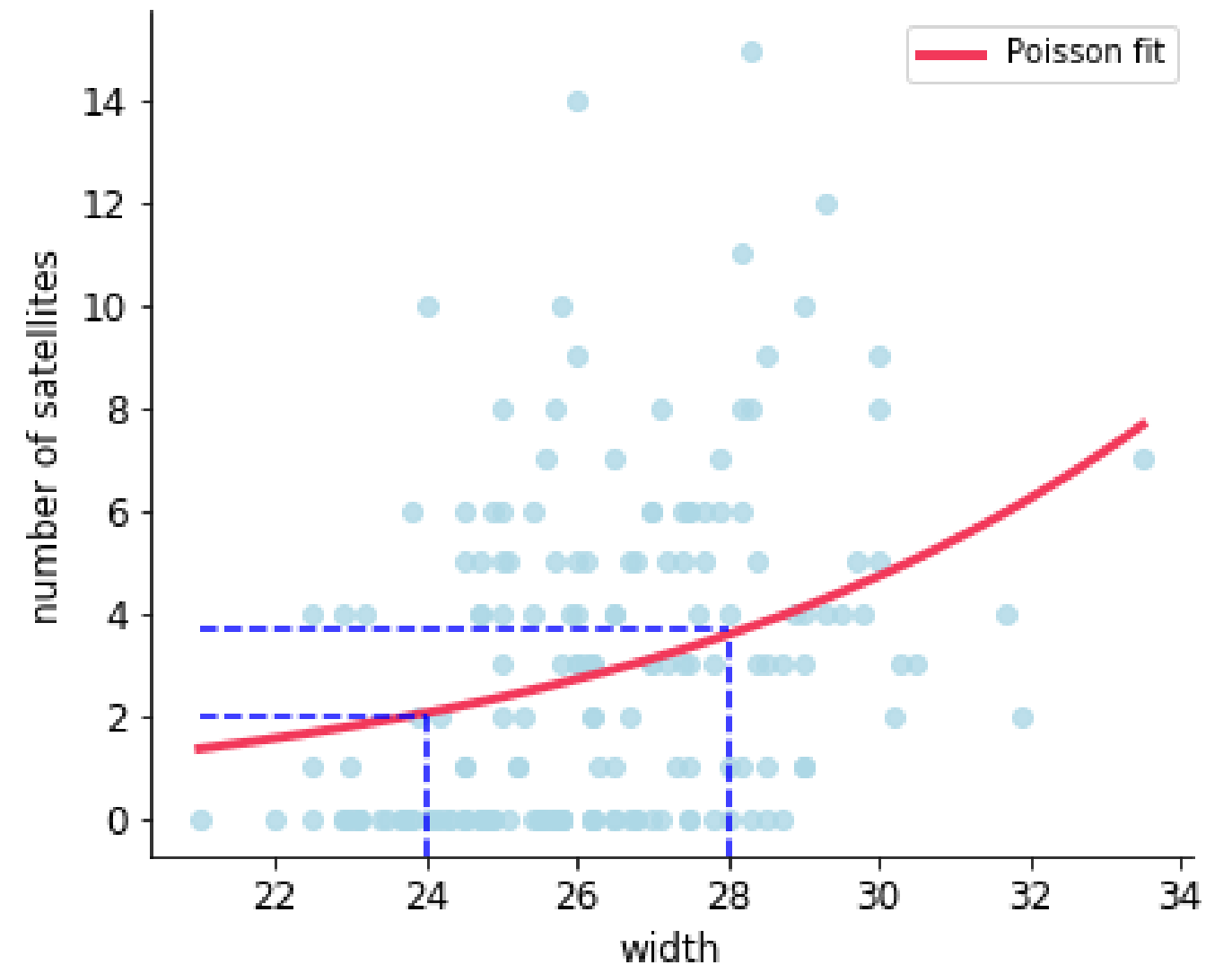
```
0    1.881981
```



Predictions

```
new_data = pd.DataFrame({'width': [24, 28, 32]})  
model.predict(new_data)
```

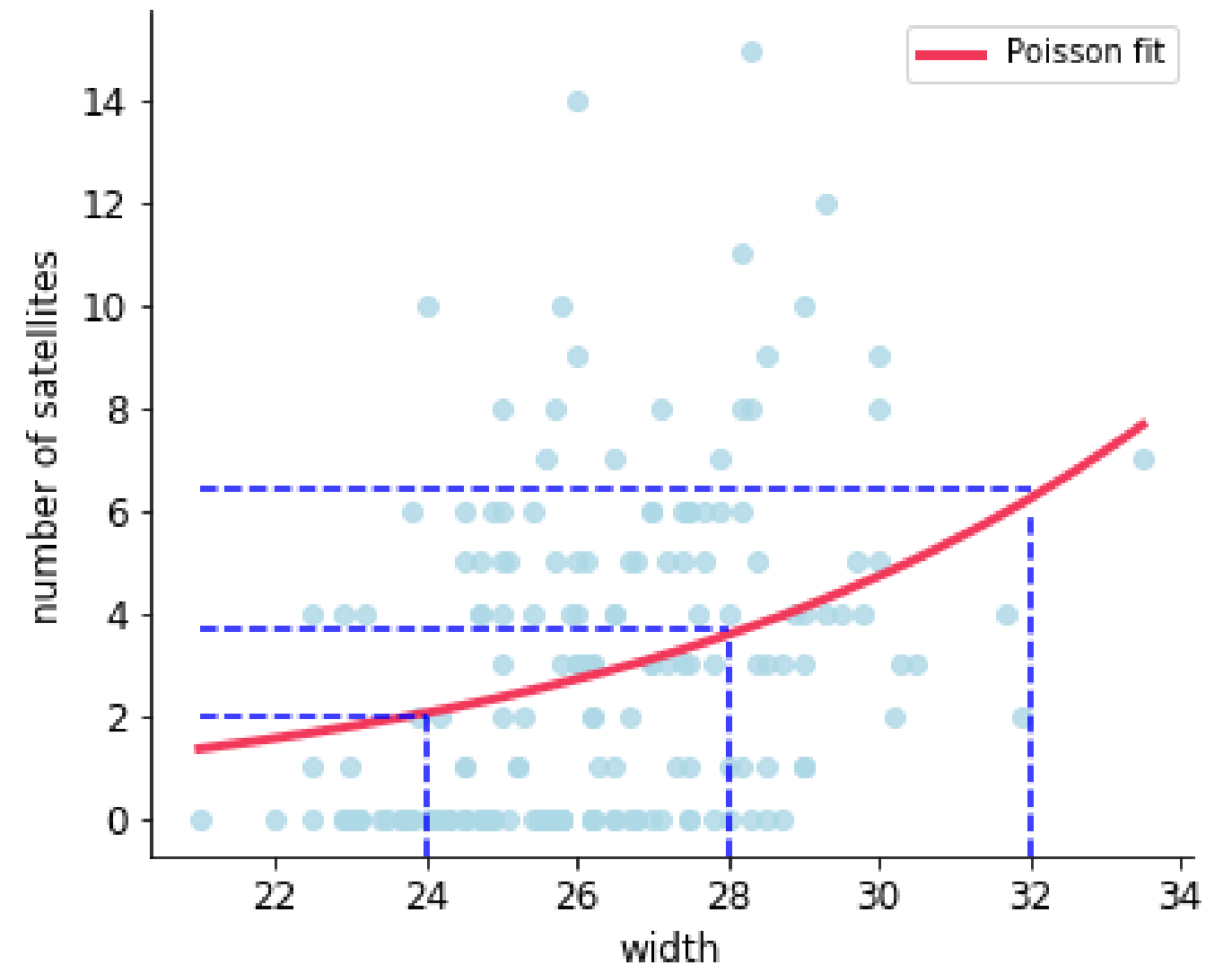
```
0    1.881981  
1    3.627360
```



Predictions

```
new_data = pd.DataFrame({'width': [24, 28, 32]})  
model.predict(new_data)
```

```
0    1.881981  
1    3.627360  
2    6.991433
```



Let's practice!

GENERALIZED LINEAR MODELS IN PYTHON