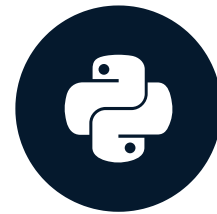


# A tale of two variables

INTRODUCTION TO REGRESSION WITH STATSMODELS IN PYTHON



**Maarten Van den Broeck**

Content Developer at DataCamp

# Swedish motor insurance data

- Each row represents one geographic region in Sweden.
- There are 63 rows.

n_claims	total_payment_sek
108	392.5
19	46.2
13	15.7
124	422.2
40	119.4
...	...

# Descriptive statistics

```
import pandas as pd  
print(swedish_motor_insurance.mean())
```

```
n_claims          22.904762  
total_payment_sek  98.187302  
dtype: float64
```

```
print(swedish_motor_insurance['n_claims'].corr(swedish_motor_insurance['total_payment_sek']))
```

```
0.9128782350234068
```

# What is regression?

- Statistical models to explore the relationship between a response variable and some explanatory variables.
- Given values of explanatory variables, you can predict the values of the response variable.

n_claims	total_payment_sek
108	3925
19	462
13	157
124	4222
40	1194
200	???

# Jargon

## **Response variable (a.k.a. dependent variable)**

The variable that you want to predict.

## **Explanatory variables (a.k.a. independent variables)**

The variables that explain how the response variable will change.

# Linear regression and logistic regression

## Linear regression

- The response variable is numeric.

## Logistic regression

- The response variable is logical.

## Simple linear/logistic regression

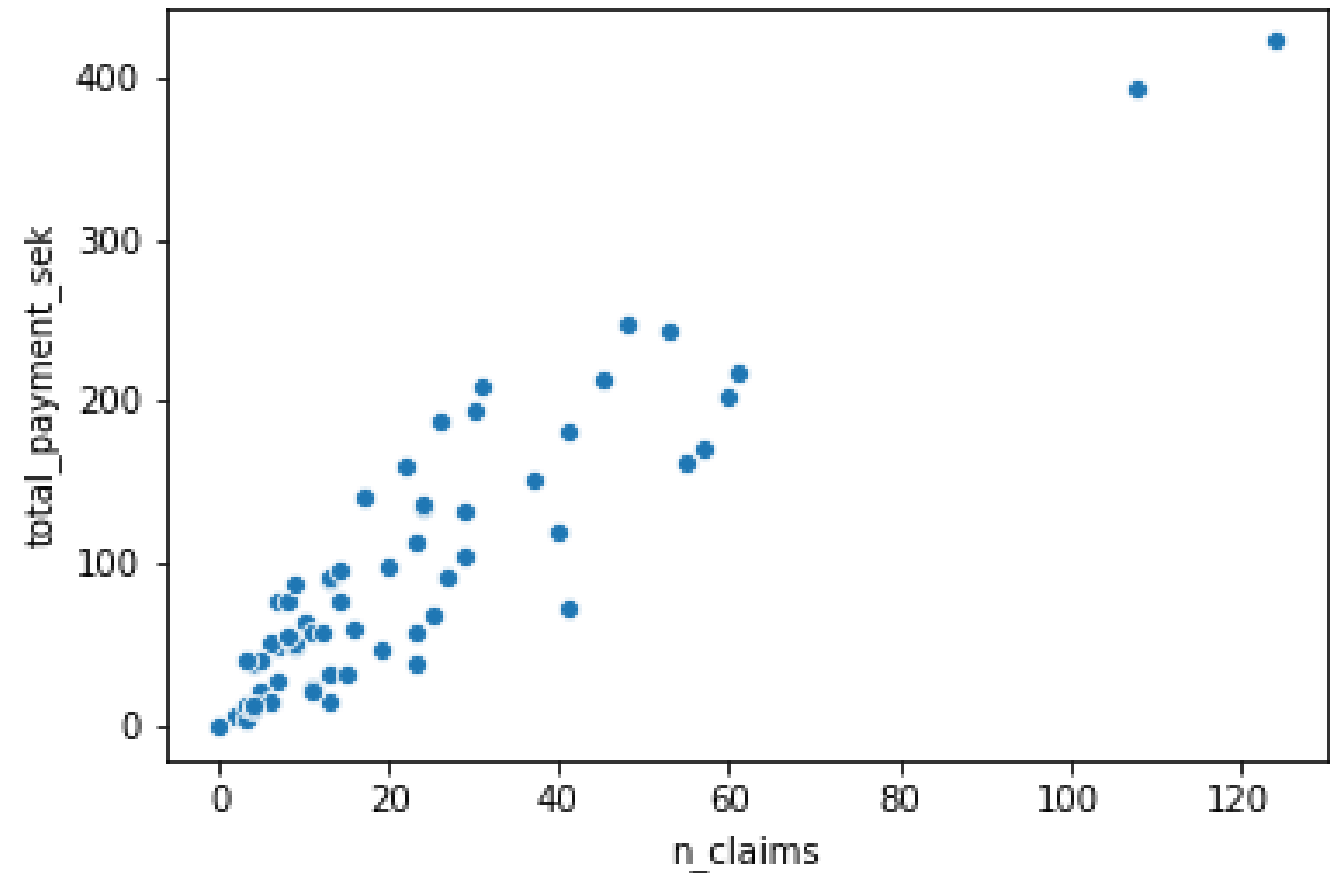
- There is only one explanatory variable.

# Visualizing pairs of variables

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

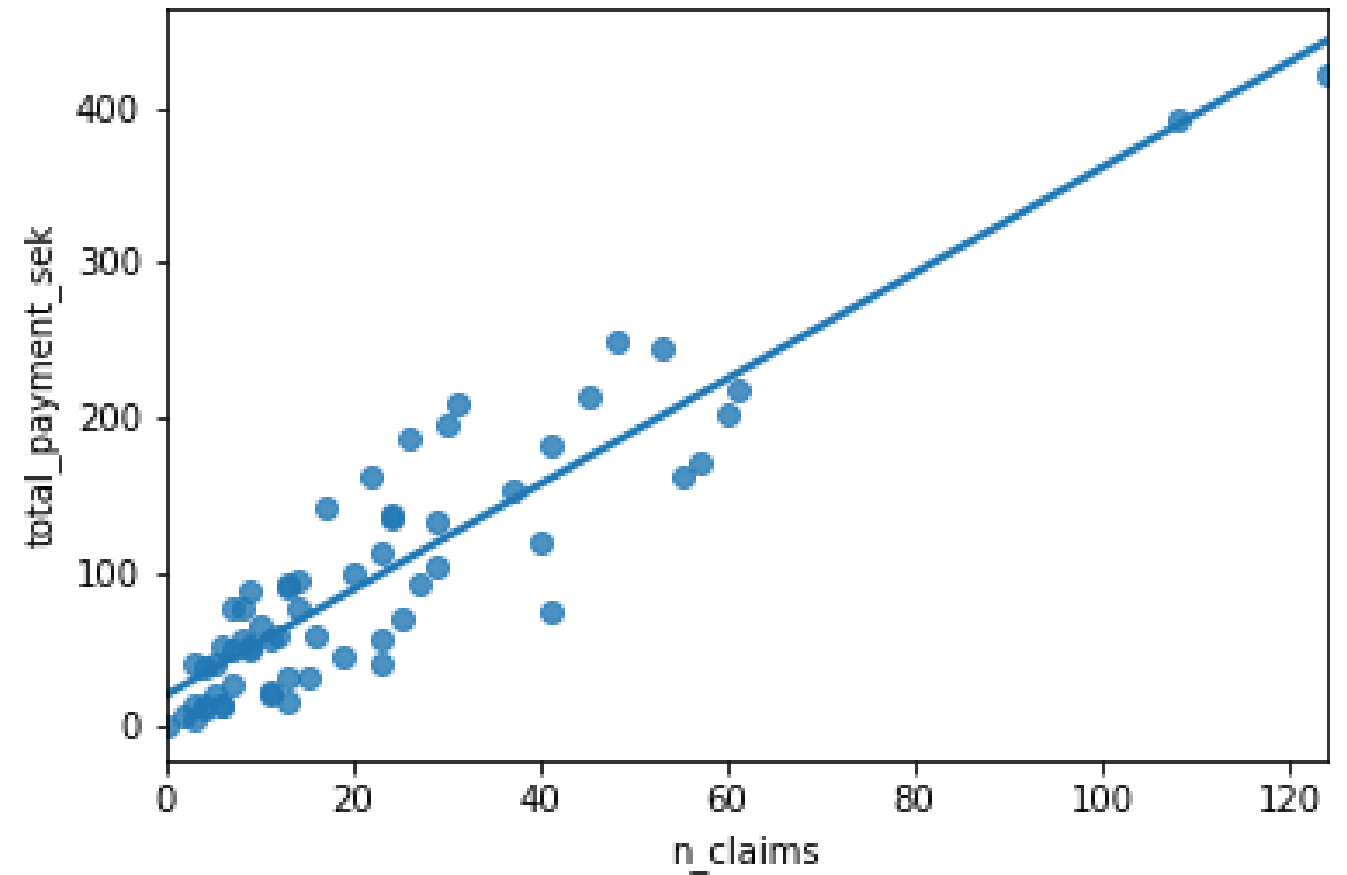
sns.scatterplot(x="n_claims",
                y="total_payment_sek",
                data=swedish_motor_insurance)

plt.show()
```



# Adding a linear trend line

```
sns.regplot(x="n_claims",  
            y="total_payment_sek",  
            data=swedish_motor_insurance,  
            ci=None)
```





# Course flow

## Chapter 1

Visualizing and fitting linear regression models.

## Chapter 2

Making predictions from linear regression models and understanding model coefficients.

## Chapter 3

Assessing the quality of the linear regression model.

## Chapter 4

Same again, but with logistic regression models

# Python packages for regression

## `statsmodels`

- Optimized for insight (focus in this course)

## `scikit-learn`

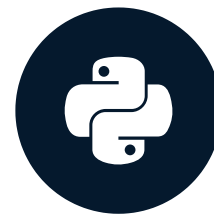
- Optimized for prediction (focus in other DataCamp courses)

# Let's practice!

INTRODUCTION TO REGRESSION WITH STATSMODELS IN PYTHON

# Fitting a linear regression

INTRODUCTION TO REGRESSION WITH STATSMODELS IN PYTHON



**Maarten Van den Broeck**

Content Developer at DataCamp

# Straight lines are defined by two things

## Intercept

The  $y$  value at the point when  $x$  is zero.

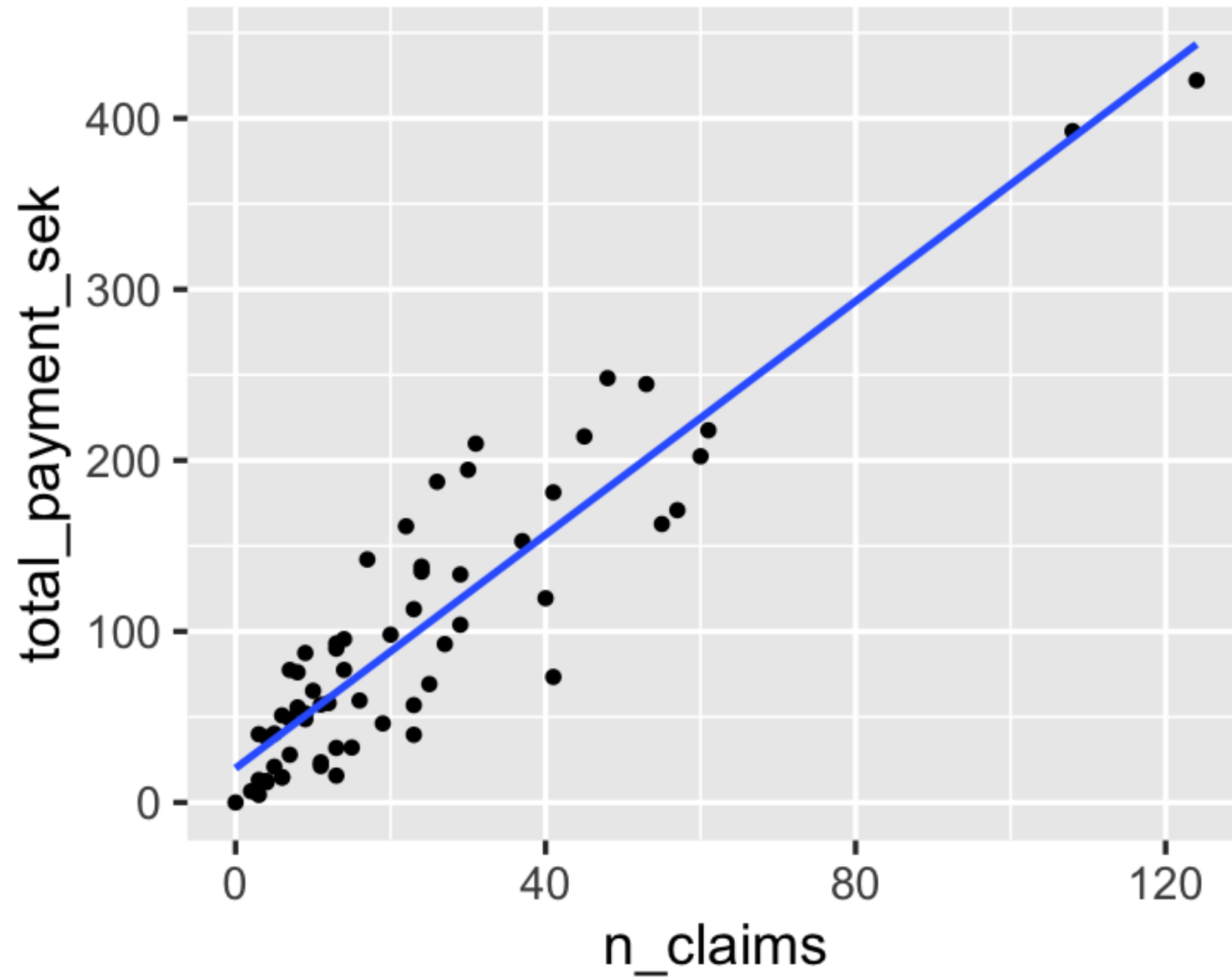
## Slope

The amount the  $y$  value increases if you increase  $x$  by one.

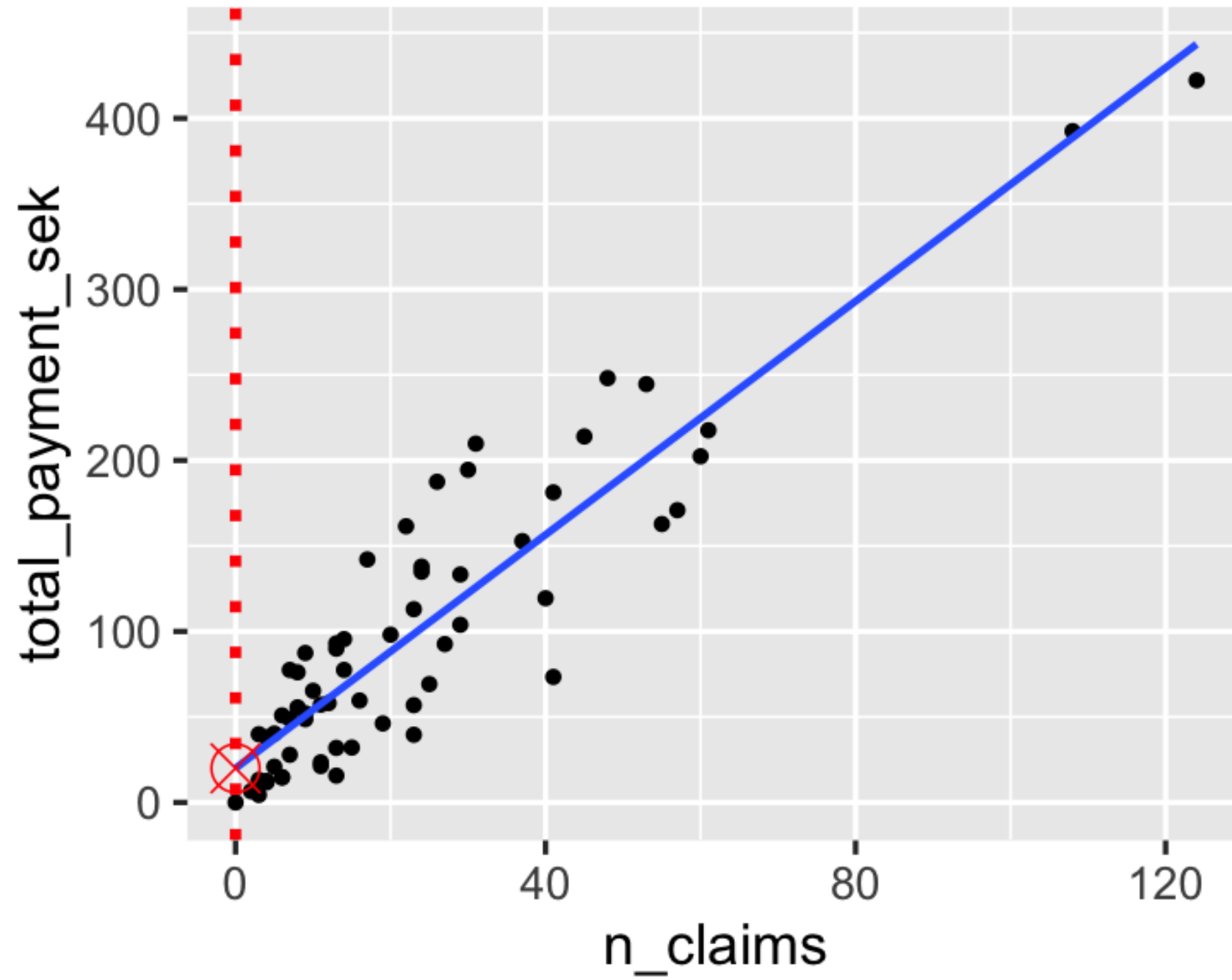
## Equation

$$y = \text{intercept} + \text{slope} * x$$

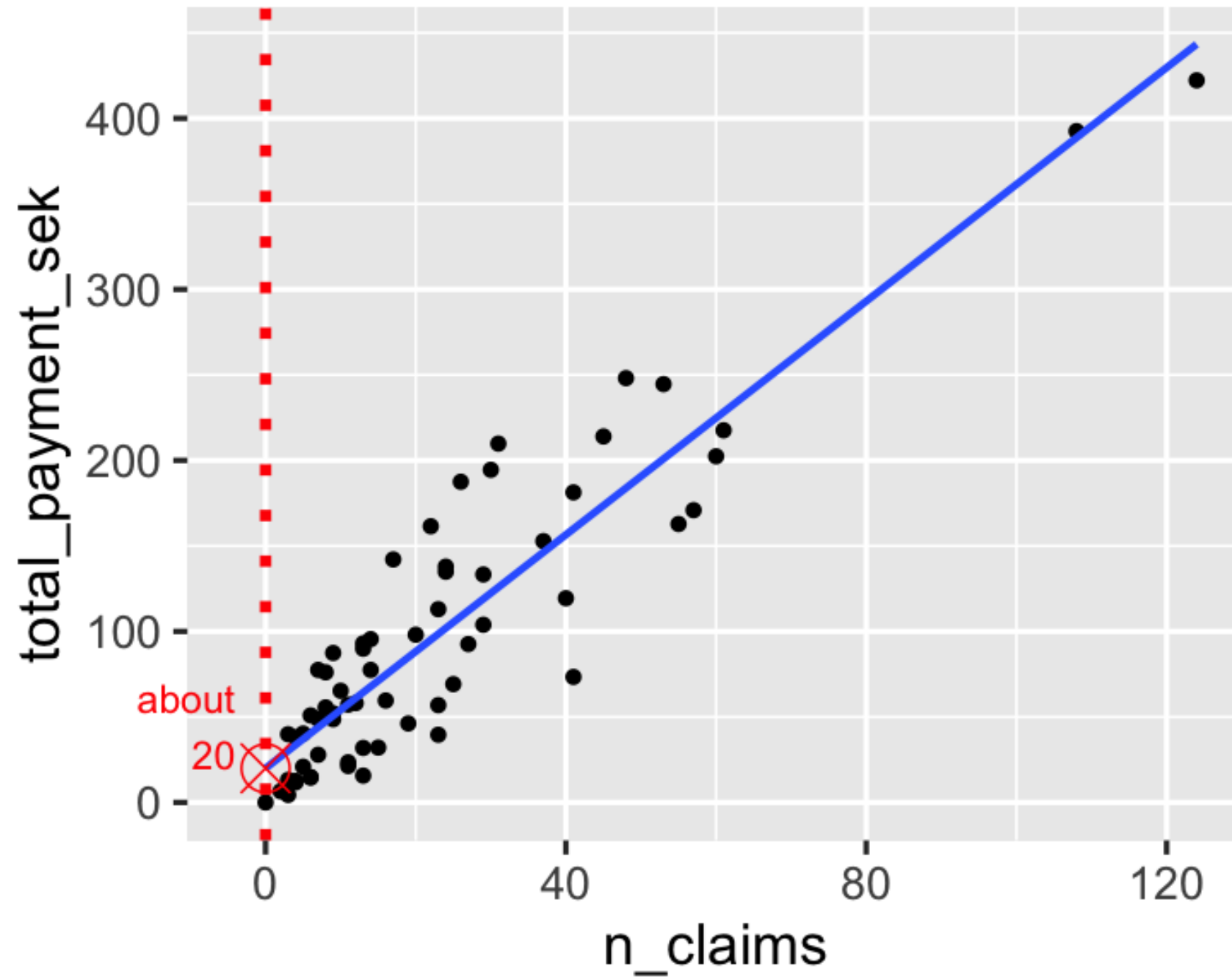
# Estimating the intercept



# Estimating the intercept

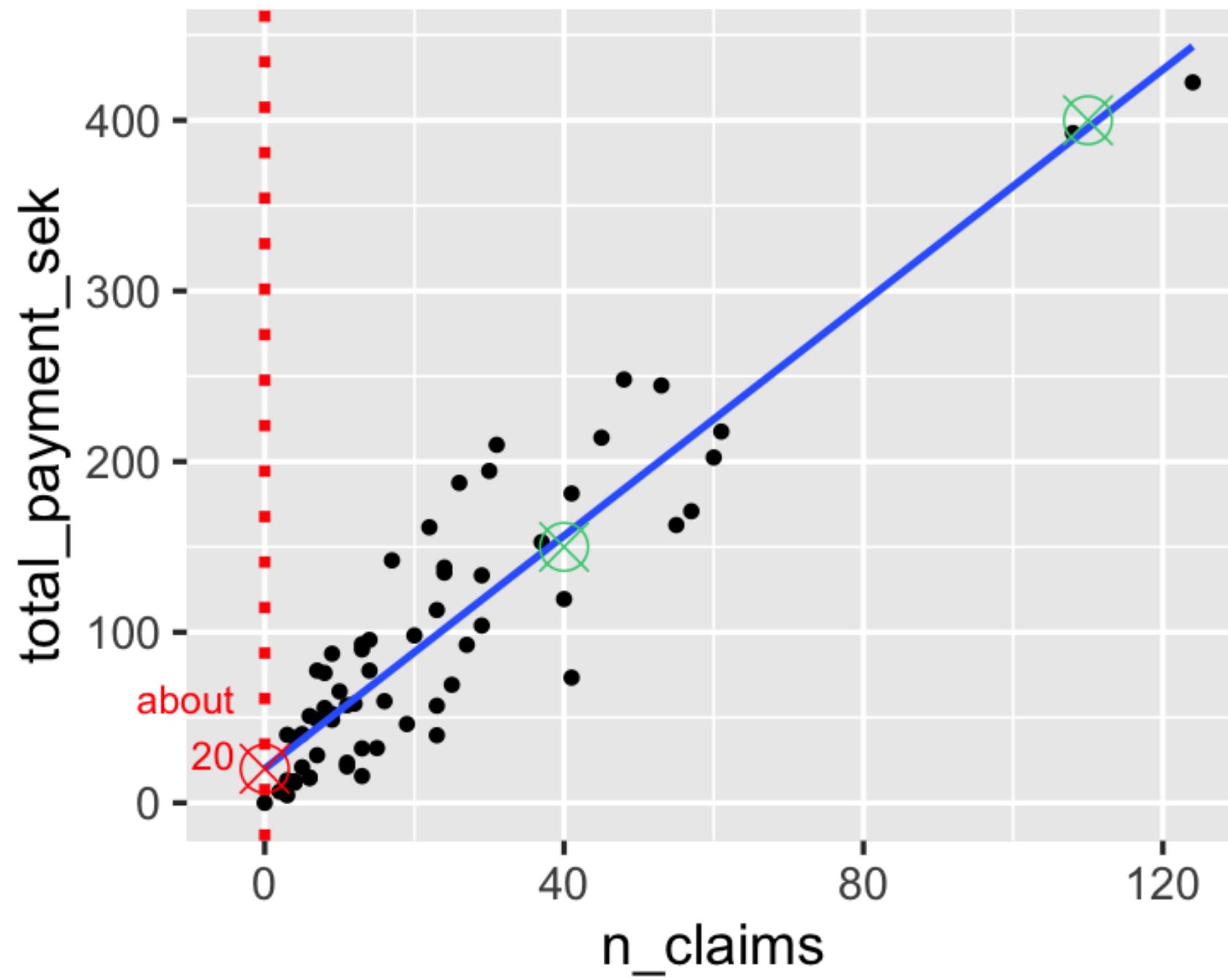


# Estimating the intercept

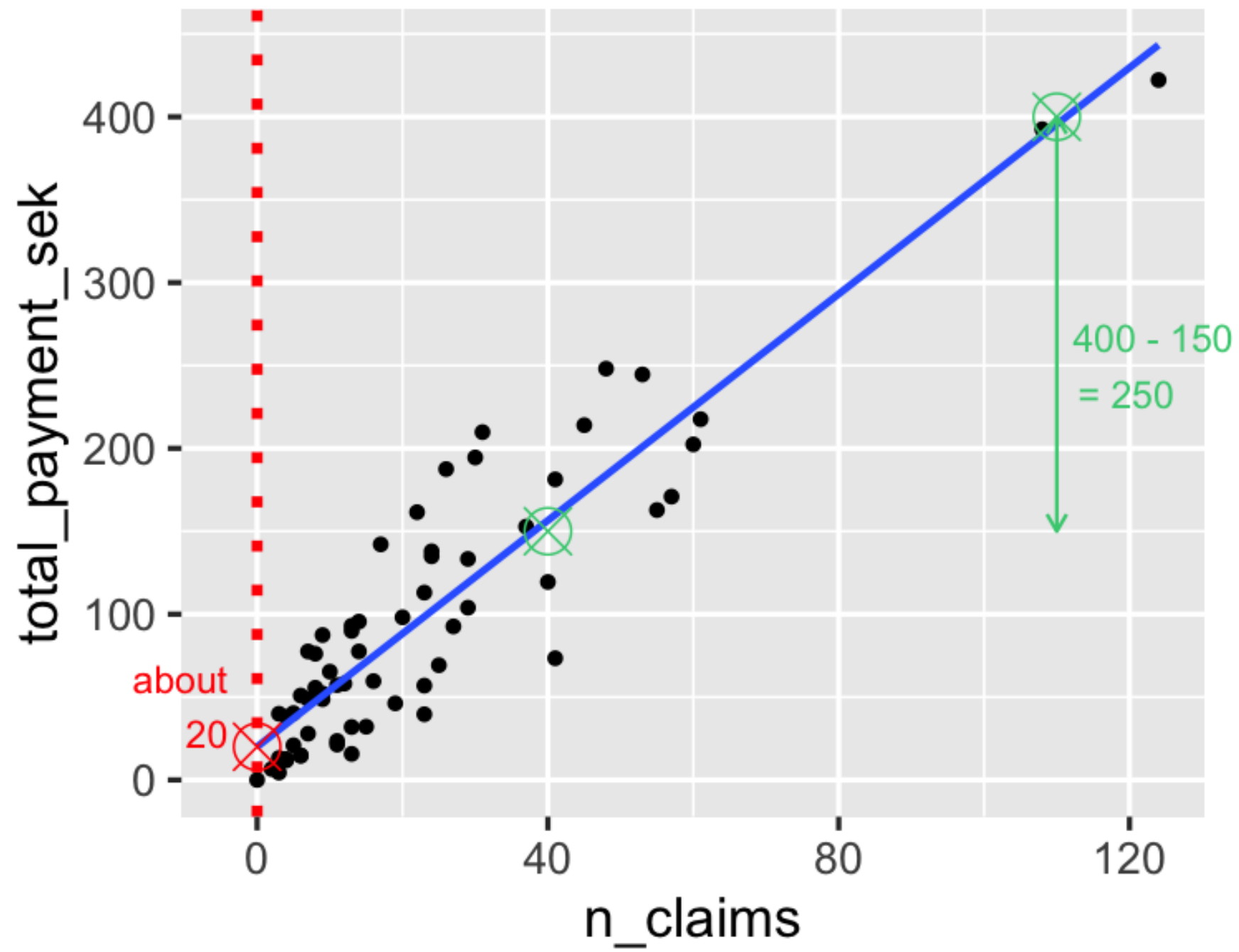




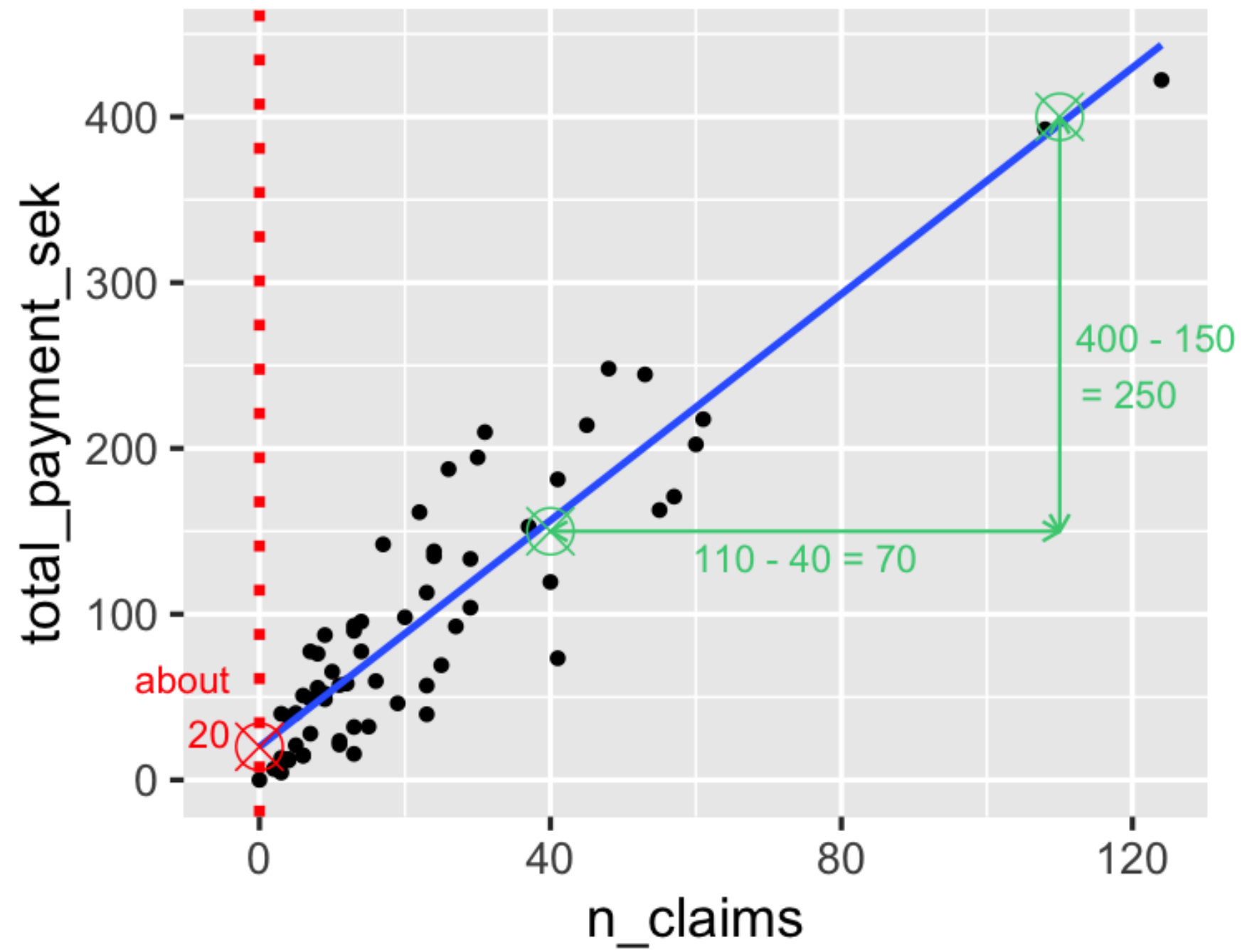
# Estimating the slope



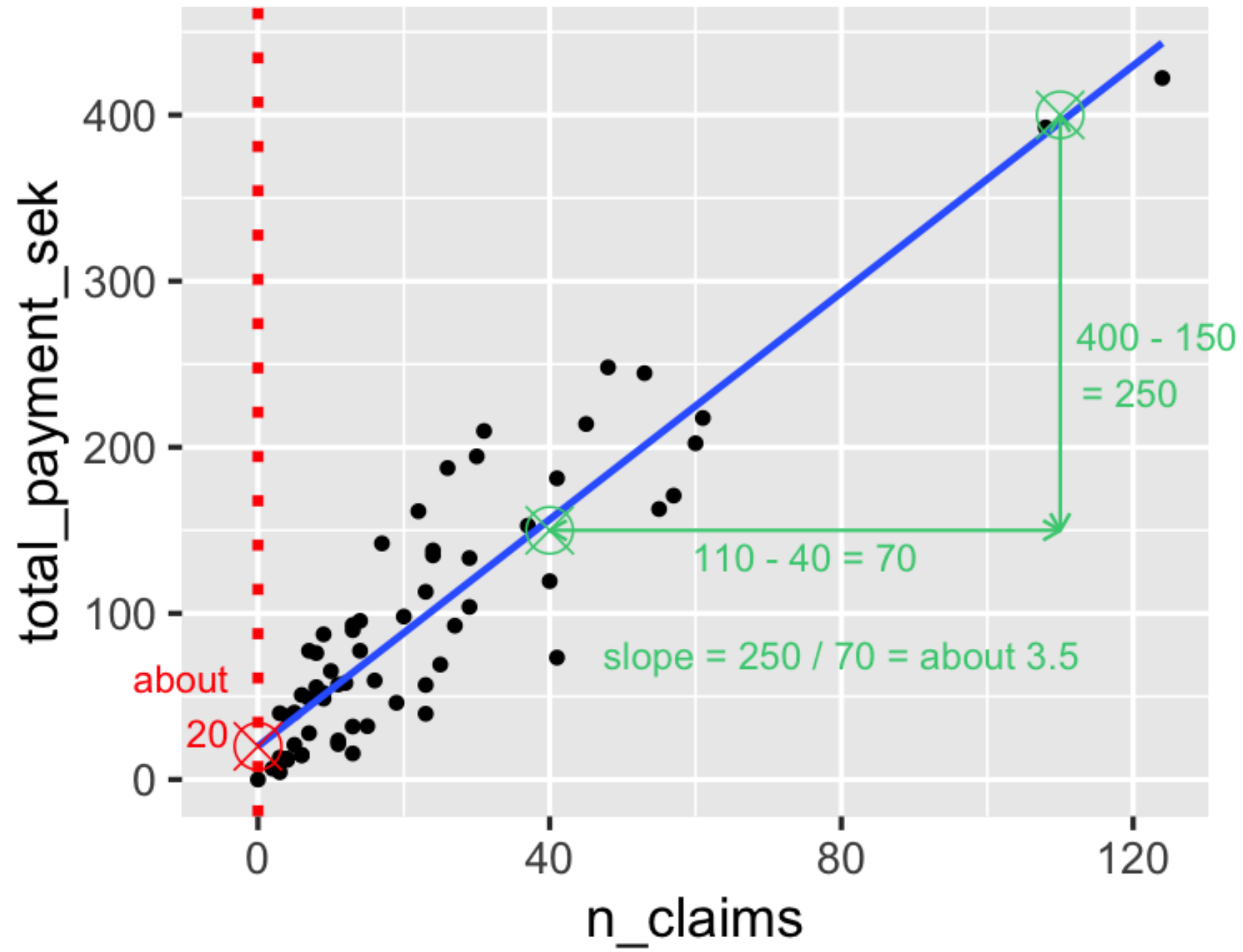
# Estimating the slope



# Estimating the slope



# Estimating the slope



# Running a model

```
from statsmodels.formula.api import ols
mdl_payment_vs_claims = ols("total_payment_sek ~ n_claims",
                             data=swedish_motor_insurance)

mdl_payment_vs_claims = mdl_payment_vs_claims.fit()
print(mdl_payment_vs_claims.params)
```

```
Intercept    19.994486
n_claims      3.413824
dtype: float64
```

# Interpreting the model coefficients

```
Intercept    19.994486  
n_claims     3.413824  
dtype: float64
```

## Equation

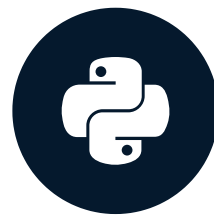
$$\text{total\_payment\_sek} = 19.99 + 3.41 * \text{n\_claims}$$

# Let's practice!

INTRODUCTION TO REGRESSION WITH STATSMODELS IN PYTHON

# Categorical explanatory variables

INTRODUCTION TO REGRESSION WITH STATSMODELS IN PYTHON



**Maarten Van den Broeck**

Content Developer at DataCamp



# Fish dataset

- Each row represents one fish.
- There are 128 rows in the dataset.
- There are 4 species of fish:
  - Common Bream
  - European Perch
  - Northern Pike
  - Common Roach

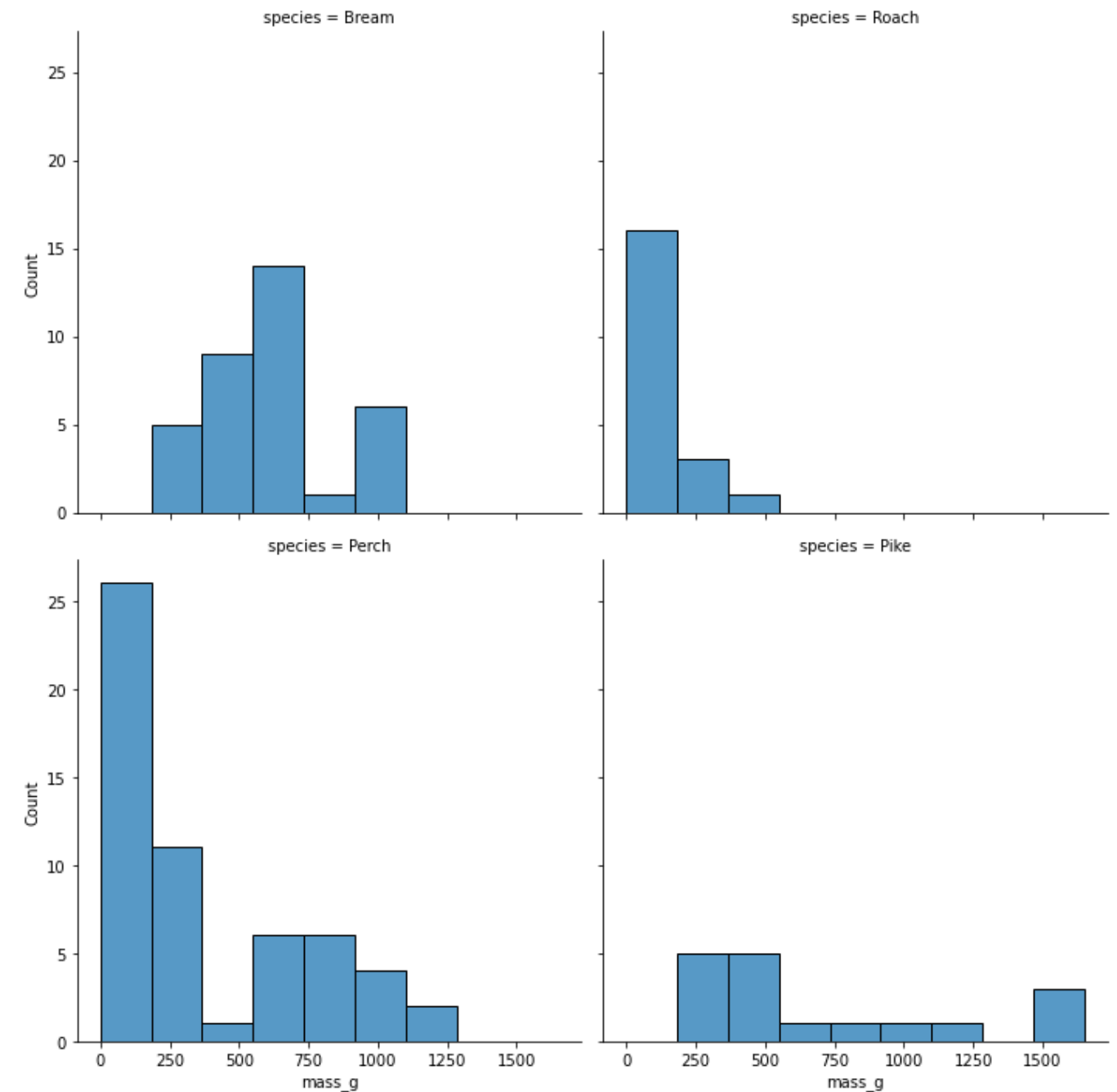
species	mass_g
Bream	242.0
Perch	5.9
Pike	200.0
Roach	40.0
...	...

# Visualizing 1 numeric and 1 categorical variable

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

```
sns.displot(data=fish,
            x="mass_g",
            col="species",
            col_wrap=2,
            bins=9)
```

```
plt.show()
```



# Summary statistics: mean mass by species

```
summary_stats = fish.groupby("species")["mass_g"].mean()  
print(summary_stats)
```

```
species  
Bream      617.828571  
Perch      382.239286  
Pike       718.705882  
Roach      152.050000  
Name: mass_g, dtype: float64
```

# Linear regression

```
from statsmodels.formula.api import ols
mdl_mass_vs_species = ols("mass_g ~ species", data=fish).fit()
print(mdl_mass_vs_species.params)
```

```
Intercept          617.828571
species[T.Perch]   -235.589286
species[T.Pike]     100.877311
species[T.Roach]   -465.778571
```

# Model with or without an intercept

From previous slide, model with intercept

```
mdl_mass_vs_species = ols(  
    "mass_g ~ species", data=fish).fit()  
print(mdl_mass_vs_species.params)
```

Intercept	617.828571
species[T.Perch]	-235.589286
species[T.Pike]	100.877311
species[T.Roach]	-465.778571

The coefficients are relative to the intercept:  
 $617.83 - 235.59 = 382.24!$

Model without an intercept

```
mdl_mass_vs_species = ols(  
    "mass_g ~ species + 0", data=fish).fit()  
print(mdl_mass_vs_species.params)
```

species[Bream]	617.828571
species[Perch]	382.239286
species[Pike]	718.705882
species[Roach]	152.050000

In case of a single, categorical variable,  
coefficients are the means.

# Let's practice!

INTRODUCTION TO REGRESSION WITH STATSMODELS IN PYTHON