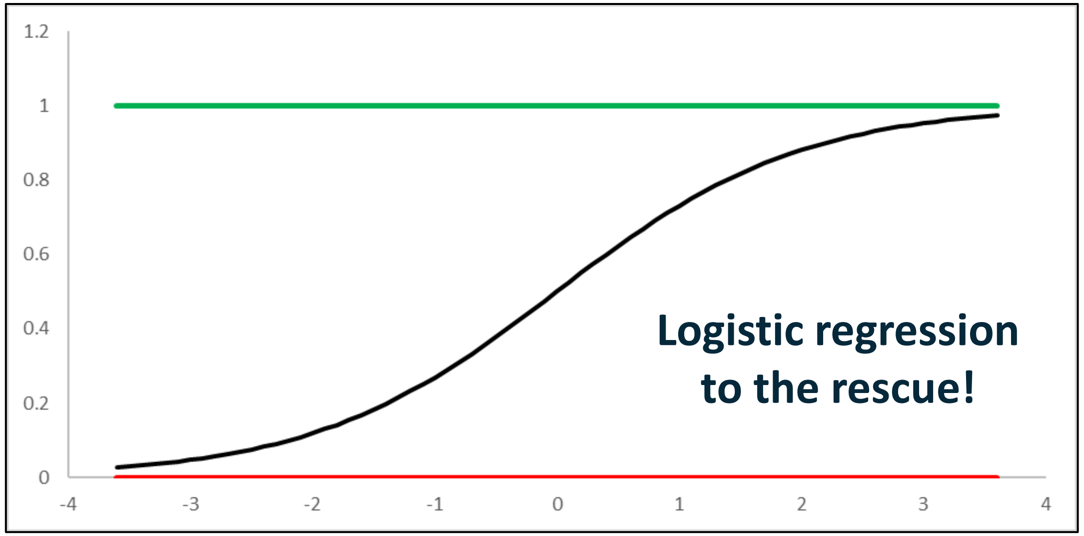
LOGISTIC REGRESSION WITH PYTHON

A FREE CRASH COURSE

Dave ON DATA




```python
from patsy import dmatrices
import statsmodels.api as sm

y, X = dmatrices('HeartDisease ~ Age + Male + ChestPainType',
                 data = heart, return_type = 'dataframe')

heart_model_1 = sm.Logit(y, X)
model_1_results = heart_model.fit()

print(model_1_results.summary())
```

About Me

I’ve been in tech for 26 years and doing hands-on analytics for 12+ years.

I’ve supported all manner of business functions and advised leaders.

I have successfully trained 1000+ professionals in a live classroom setting.

Trained 1000s more via my online courses and tutorials.

Hands-on analytics consultant and instructor.

schedulicity

Microsoft

tdwi
Transforming Data With Intelligence™

datasciencedojo
data science for everyone


Tyler Henderson • 3rd+
Manager, CX Quantitative Insights at Asurion
2h • Edited •

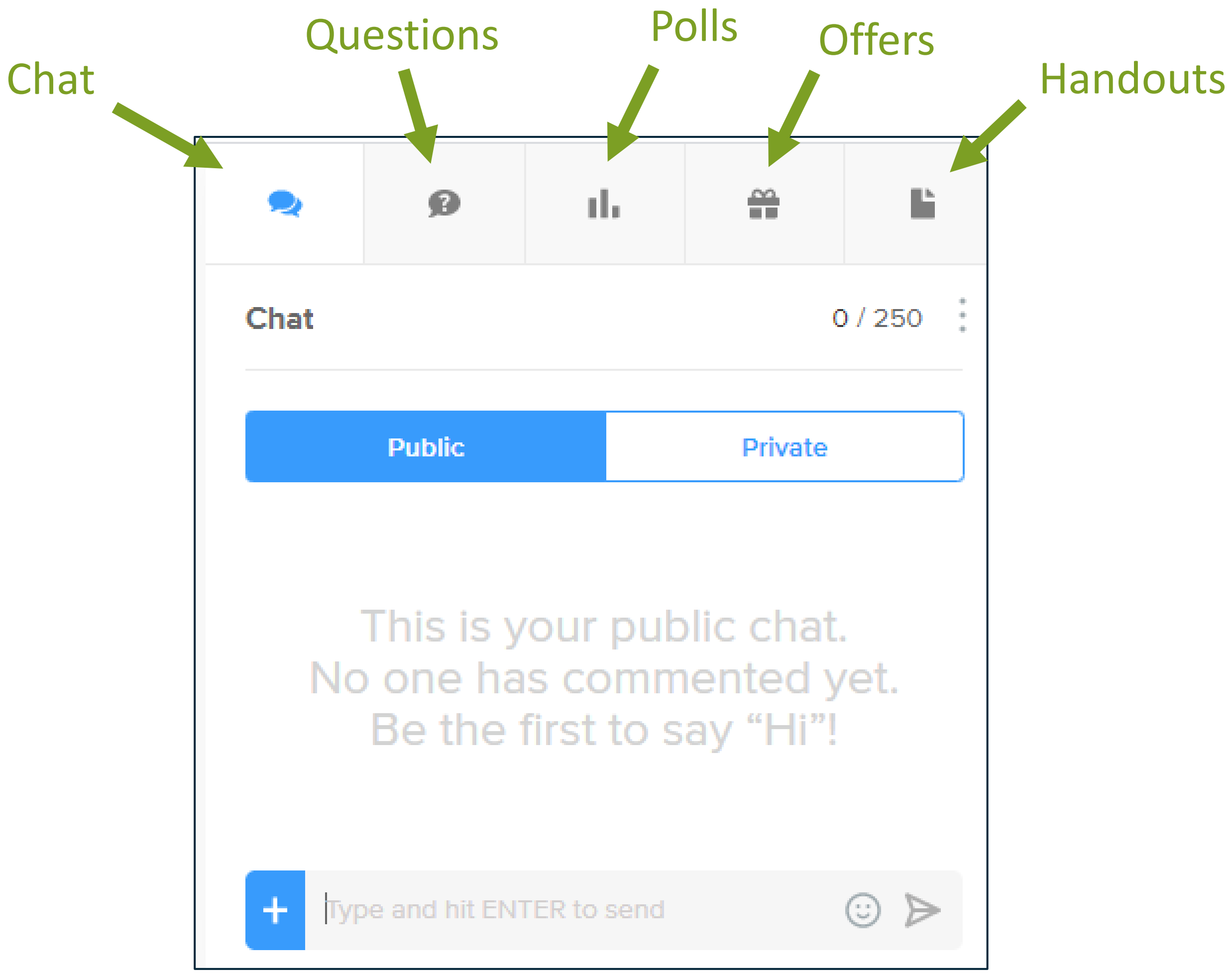
+ Follow

#TDWINashville was a blast!

Grateful for the opportunity to spend three days with David Langer on the topics of data wrangling in R, Random Forrest machine learning, and Python clustering analysis. All three courses were engaging and built in a way to give you the tools to “go do” which I appreciated!

Looking forward to 2024! 🤓
#dataiscool

Housekeeping



Why Learn
Logistic Regression?

The 4 Level of Analytics

Diagnostic Analytics is the minimum level required.

True Data Analysis

Descriptive Analytics

What happened in the past.

Excel Reports
Executive Dashboards
PowerPoint

Diagnostic Analytics

Why it happened in the past.

BI Dashboards
Excel Workbooks
Analytics Tools
PowerPoint

Predictive Analytics

What is likely to happen in the future.

BI Dashboards
Excel Workbooks
Analytics Tools
PowerPoint

Prescriptive Analytics

How to optimize the business.

Excel Workbooks
Analytics Tools
PowerPoint

Logistic Regression in Action

This webinar is a crash course in *binary logistic regression* analysis.

This form of analysis is useful to ANY professional:

[HR] - What factors are associated with high performers quitting?

[Non-Profit] – What is the chance that a prospect will become a donor?

[Product Management] - What feature usage(s) are highly predictive of a sticky customer?

[Marketing] - What are the demographic factors that predict conversion?

[Finance] – What is chance that this transaction will result in a chargeback?

The list of impactful analyses is endless!

The Heart Dataset

Dave ON DATA

Statlog (Heart)

This dataset is a heart disease database similar to a database already present in the repository (Heart Disease databases) but in a slightly different form

Dataset Characteristics	Subject Area	Associated Tasks
Multivariate	Life	Classification
Attribute Type	**# Instances**	**# Attributes**
Categorical, Real	270	13

Information

Additional Information

Cost Matrix

______ abse pres...

SHOW MORE

Has Missing Values

Symbol: 0

DOWNLOAD

CITE

DOI

10.24432/C57303

The Heart Dataset

Dependent Variable → HeartDisease

Independent Variables: Age through Thal

Feature	Type	Categorical Levels
HeartDisease	Categorical	Class label. Value of 1 if heart disease is present, 0 otherwise
Age	Numeric	
Male	Categorical	Value of 1 if male, 0 otherwise
ChestPainType	Categorical	Chest pain type. Values are 1 thru 4
BloodPressure	Numeric	
Cholesterol	Numeric	
BloodSugar	Categorical	Value of 1 if fasting blood sugar > 120 mg/dl, 0 otherwise
EEG	Categorical	Values of 0 thru 2
MaxHR	Numeric	
Angina	Categorical	Value of 1 if angina induced, 0 otherwise
OldPeak	Numeric	
PeakST	Numeric	
Flourosopy	Numeric	
Thal	Categorical	3 = normal; 6 = fixed defect; 7 = reversable defect

What Is a Model?

Model Defined

“A **model** is an informative representation of an object, person or system…Models can be divided into **physical models** (e.g., a model plane) and **abstract models** (e.g., mathematical expressions describing behavioral patterns).”

- Wikipedia

This crash course is all about abstract models.

Mathematical Models

Logistic regression models are *mathematical models*.

Mathematical models usually look like this:

$$Y = b_0 + b_1 x_1 + b_2 x_2^2 + b_3 x_1 x_2 + \varepsilon$$

As discussed previously, logistic regression models represent systems with *binary outcomes*:

- Yes/No
- True/False
- Approve/Deny
- Legit/Fraudulent

Model Equations 101

Set the Way Back Machine…

As discussed, logistic regression models are mathematical models (i.e., *equations*).

The good news is that you use Python to craft logistic regression models.

There will be some math you need to learn, but Python does the heavy lifting for you.

As the data analyst, you will learn to *interpret the equations*.

Remember this?

The slope

The y-intercept

$$y = mx + b$$

The equation for a line!

This is the same thing.

The y-intercept

The slope

$$y = b_0 + b_1x_1$$

Taking it up a notch.

Intercept

Coefficients

$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3$$

Dependent Variable

Independent Variables

An Example

Dave ON DATA

Imagine you are a real estate agent and want to be able to predict the selling prices of houses.

In this scenario *price (y)* is the *dependent variable* - the value you are trying to predict.

Additionally, your *independent variables* are:

- The size of the house in square feet (x_1)
- The number of bedrooms (x_2)
- The number of bathrooms (x_3)

Estimated from the data.

Let's say your model is as follows:

$$y = 135000 + 23.76x_1 + 4323.59x_2 + 1881.13x_3$$

Using the model to predict the price for this house:

- 2,300 square feet
- 4 bedrooms
- 2.5 bathrooms

$$y = 135000 + (23.76 \times 2300) + (4323.59 \times 4) + (1881.13 \times 2.5)$$

$$y = 135000 + 54648 + 17294.36 + 4702.83$$

$$y = 211{,}645.19$$

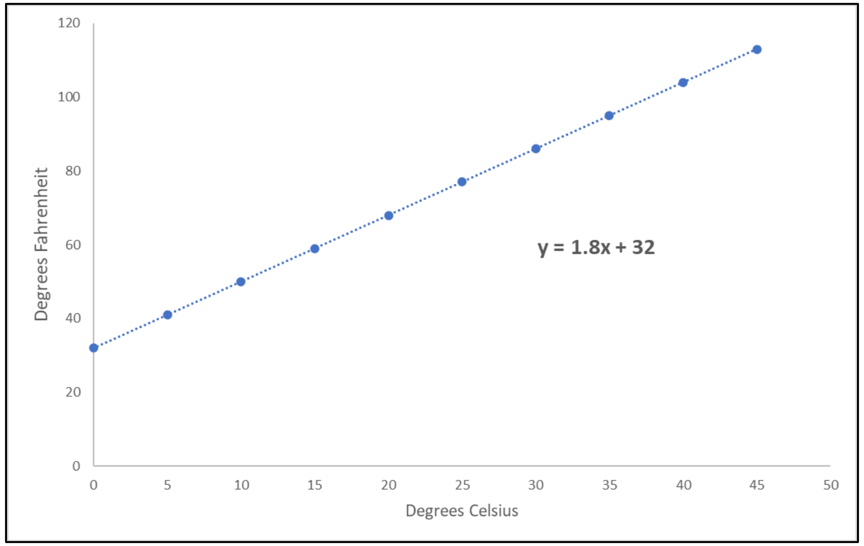
Wait! That's not a binary outcome!

The Logistic Curve

The Problem

The equations from the previous slides produce *straight lines*.

For example, the equation to predict degrees Fahrenheit from degrees Celsius.

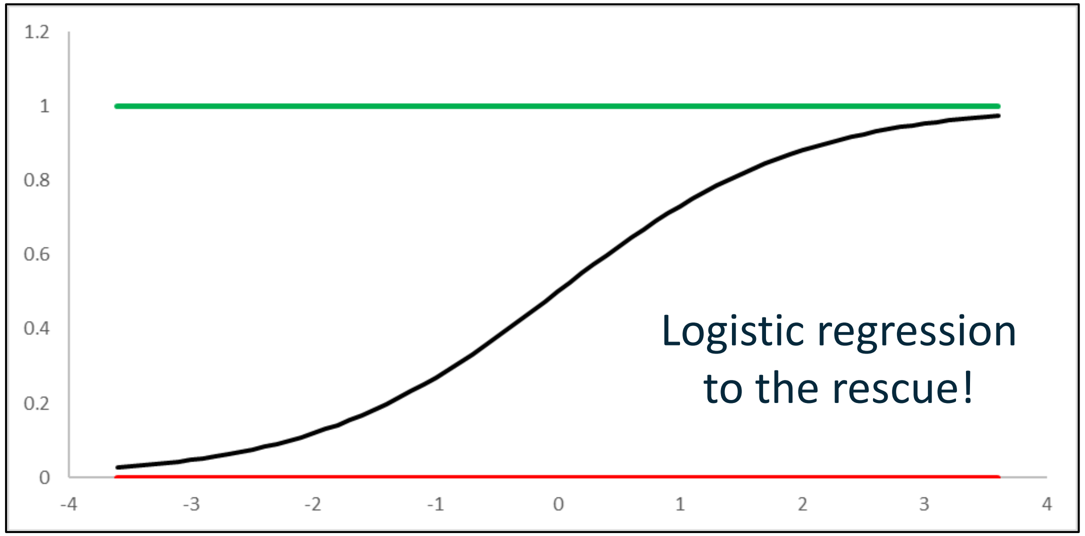


However, we need an equation to predict *binary outcomes*.

In this crash course we will focus on predicting *binary labels*:

- Legitimate/Fraudulent
- Approve/Deny
- True/False (i.e., 1/0)

We need an equation that produces values between 0 and 1.



Putting It All Together

The Problem

We know logistic regression models are math equations: $\boxed{y = b_0 + b_1x_1 + b_2x_2 + b_3x_3}$

We also know these math equations do not produce *labels* (i.e., binary outcomes).

We can use the logistic curve to get the labels we need: $\frac{1}{1 + e^{-X}}$

The logistic curve gives us the *probability* of a label occurring given the data.

When we combine these two, we get the magic of logistic regression: $P(y) = \frac{1}{1 + e^{-(b_0 + b_1x_1 + b_2x_2 + b_3x_3)}}$

Let's cement these ideas with your first model.

Your First Model

Dave ON DATA

You have become convinced there is an association between being male and heart disease.

Using the *Heart Data Set*, you can perform a logistic regression analysis.

You craft a logistic regression model with the following *dependent/independent variables*:

- *HeartDisease* (y) – value of 1 or 0
- *Male* (x_1) – value of 1 or 0

$$b_0 + b_1x_1$$

Using the *Heart* data, the model *intercept* and *coefficient* are estimated:

- $b_0 = -1.2090$
- $b_1 = 1.3953$

$$-1.2090 + 1.3953x_1$$

Combining the equation with the logistic curve give the full model:

$$P(y) = \frac{1}{1 + e^{-(-1.2090 + (1.3953x_1))}}$$

When male, per the model, what's the probability of heart disease?

$$P(y) = \frac{1}{1 + e^{-(-1.2090 + (1.3953 \times 1))}} = \frac{1}{1 + e^{-(-1.2090 + 1.39553)}}$$

$$P(y) = \frac{1}{1 + e^{-(0.1883)}} = \frac{1}{1 + 0.82837} = 0.5469 = 54.69\%$$

Is this a good model?

What’s the Baseline?

Your First Model

Let's return to your first model – the association of being male with heart disease.

To establish a baseline, you look at the *Heart* data used to craft your logistic regression model.

The following summarizes the data:

- There are 270 rows of data (i.e., *observations*)
- Of the 270 observations, 120 have *HeartDisease* indicated (i.e., a *label* of 1)

The *baseline model* is the proportion of labels: $P(y) = 120 \div 270 = 0.4444 = 44.44\%$

The baseline model provides the simplest predictive model.

With only the label data, we would predict heart disease 44.44% of the time.

Using a simple decision threshold (e.g., >50%), we would never predict heart disease.

Here's the question – is the logistic regression model more useful?

Baseline Usefulness

Analyzing data with logistic regression is an iterative process.

You typically craft many models during a single data analysis (e.g., using different independent variables).

Throughout the process, your goal is to craft more accurate models than the baseline.

Revisiting the *Heart* data example baseline and logistic regression models:

Baseline model

$$P(y) = 120 \div 270 = 0.4444$$

The baseline predicts no heart disease!

"Male model"

($x_1 = 1$ male, 0 otherwise)

$$P(y) = \frac{1}{1 + e^{-(-1.2090 + (1.3953 x_1))}}$$

$$P(y_{male}) = \frac{1}{1 + e^{-(-1.2090 + (1.3953 \times 1))}} = 0.5469$$

$$P(y_{otherwise}) = \frac{1}{1 + e^{-(-1.2090 + (1.3953 \times 0))}} = 0.2299$$

This model predicts only males have heart disease.

Which model is more accurate?

That is, which model fits the data better?

Goodness of Fit

Goodness for Everyone

Dave ON DATA

Comparing models for usefulness (i.e., accuracy) cannot be left to gut feel.

Luckily for us, statisticians figured out a way to compare logistic regression models in a standardized way.

Not surprisingly, they crafted a mathematical calculation to quantify a model's *goodness of fit*.

The goodness of fit calculation is called the *log-likelihood*.

Before looking at the math, it best to think about the intuition that drives the math.

Think of the labels we try to predict:

1. When the label in the data is 0, models fit well when they predict a probability close to 0
2. When the label in the data is 1, models fit well when they predict a probability close to 1

"Go through all the data and add up"

The model's prediction

$$\text{log-likelihood} = \boxed{\sum_{i=1}^{n}} [y_i \ln(P(y_i)) + (1 - y_i) \ln(1 - P(y_i))]$$

Each label (i.e., 0 or 1) in the data

Plug & Chug

$$\text{log-likelihood} = \sum_{i=1}^{n} [y_i \ln(P(y_i)) + (1 - y_i) \ln(1 - P(y_i))]$$

The label in the data

The model's prediction

i	y_i	$P(y_i)$
1	0	0.00001
2	0	0.99999
3	1	0.00001
4	1	0.99999

$\ln(P(y_i))$	$\ln(1 - P(yi))$
-11.51293	-0.00001
-0.00001	-11.51293
-11.51293	-0.00001
-0.00001	-11.51293

Correct predictions

$y_i\ln(P(y_i))$	$(1 - y_i)\ln(1 - P(y_i))$	row total
0.00000	-0.00001	-0.00001
0.00000	-11.51293	-11.51293
-11.51293	0.00000	-11.51293
-0.00001	0.00000	-0.00001

1. The log-likelihood is always negative.
2. The log-likelihood calculation scores correct predictions close to zero.
3. The log-likelihood calculation scores incorrect predictions away from zero.
4. The most useful (i.e., accurate) models have grand totals as close to zero as possible.

Logistic Regression with Python

Your First Model

Dave ON DATA

```python
import pandas as pd

# Load the Heart dataset
heart = pd.read_csv('Heart.csv')
heart.head()
```

	HeartDisease	Age	Male	ChestPainType	BloodPressure	Cholesterol	BloodSugar	EEG	MaxHR	Angina	OldPeak	PeakST	Flourosopy	Thal
0	1	70	1	4	130	322	0	2	109	0	2.4	2	3	3
1	0	67	0	3	115	564	0	2	160	0	1.6	2	0	7
2	1	57	1	2	124	261	0	0	141	0	0.3	1	0	7
3	0	64	1	4	128	263	0	0	105	1	0.2	2	1	7
4	0	74	0	2	120	269	0	2	121	1	0.2	1	1	3

Your first model will predict
HeartDisease **using** ***Male***

Your First Model

```
import statsmodels.formula.api as smf

# Craft a logistic regression model to predict HeartDisease based on being Male
heart_model_1 = smf.logit(formula = 'HeartDisease ~ Male', data = heart)

# Train the model from the data
model_1_results = heart_model_1.fit()

# What are the model results?
print(model_1_results.summary())
```

"Predict *HeartDisease* by *Male*"

Your First Model

```
Optimization terminated successfully.
         Current function value: 0.640593
         Iterations 5
                           Logit Regression Results
==============================================================================
Dep. Variable:            HeartDisease   No. Observations:                  270
Model:                           Logit   Df Residuals:                      268
Method:                            MLE   Df Model:                            1
Date:                 Sun, 23 Jul 2023   Pseudo R-squ.:                 0.06750
Time:                         11:24:48   Log-Likelihood:                -172.96
converged:                        True   LL-Null:                       -185.48
Covariance Type:             nonrobust   LLR p-value:                 5.618e-07
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     -1.2090      0.255     -4.745      0.000      -1.708      -0.710
Male           1.3953      0.295      4.731      0.000       0.817       1.973
==============================================================================
```

← Your model

← Baseline model

Your model is better than the baseline, but not by much!

When Simple Won’t Do

Adding Complexity

The last section covered *simple logistic regression* where models take the form of: $b_0 + b_1x_1$

However, modeling systems sufficiently usually requires more than a single *independent variable*.

Multiple logistic regression allows you to use more than one independent variable: $b_0 + b_1x_1 + b_2x_2 + b_3x_3$

We could try to improve upon the "male model" by adding a 2nd independent variable…

HeartDisease	Age	Male
1	70	1
0	67	0
1	57	1
0	64	1
0	74	0

$$b_0 + \underset{\textit{Male}}{b_1x_1} + \underset{\textit{Age}}{b_2x_2}$$

Predictions

$$P(y) = \frac{1}{1 + e^{-(b_0 + b_1x_1 + b_2x_2)}}$$

Goodness of fit

$$\sum_{i=1}^{n}[y_i \ln(P(y_i)) + (1 - y_i)\ln(1 - P(y_i))]$$

The process of adding variables, expanding models, and all calculations remains the same whether you have 2, 3, 4, or 15 independent variables.

This makes crafting multiple logistic regression models in Python quite easy.

Your Second Model

```
# A logistic regression model to predict HeartDisease using Male and Age
heart_model_2 = smf.logit(formula = 'HeartDisease ~ Male + Age', data = heart)

# Train the model from the data
model_2_results = heart_model_2.fit()

# What are the model results?
print(model_2_results.summary())
```

Your Second Model

```
Optimization terminated successfully.
         Current function value: 0.607039
         Iterations 5
                           Logit Regression Results
==============================================================================
Dep. Variable:           HeartDisease   No. Observations:                  270
Model:                          Logit   Df Residuals:                      267
Method:                           MLE   Df Model:                            2
Date:                Sun, 23 Jul 2023   Pseudo R-squ.:                  0.1163
Time:                        17:01:08   Log-Likelihood:                -163.90
converged:                       True   LL-Null:                       -185.48
Covariance Type:            nonrobust   LLR p-value:                 4.249e-10
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     -4.8637      0.959     -5.071      0.000      -6.744      -2.984
Male           1.6222      0.315      5.156      0.000       1.006       2.239
Age            0.0639      0.016      4.060      0.000       0.033       0.095
==============================================================================
```

← Your 2nd model

← Baseline model

Your 2nd model is better than the first!

Interpreting Your Models

Your Third Model

Dave ON DATA

```
# A logistic regression model to predict HeartDisease using Male, Age, & Angina
heart_model_3 = smf.logit(formula = 'HeartDisease ~ Male + Age + Angina', data = heart)

# Train the model from the data
model_3_results = heart_model_3.fit()

# What are the model results?
print(model_3_results.summary())
```

Your Third Model


```
Optimization terminated successfully.
         Current function value: 0.538839
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:           HeartDisease   No. Observations:                  270
Model:                          Logit   Df Residuals:                      266
Method:                           MLE   Df Model:                            3
Date:                Sun, 23 Jul 2023   Pseudo R-squ.:                  0.2156
Time:                        17:10:56   Log-Likelihood:                -145.49
converged:                       True   LL-Null:                       -185.48
Covariance Type:            nonrobust   LLR p-value:                 3.090e-17
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     -5.2011      1.030     -5.051      0.000      -7.219      -3.183
Male           1.4648      0.333      4.404      0.000       0.813       2.117
Age            0.0614      0.017      3.643      0.000       0.028       0.094
Angina         1.7952      0.312      5.761      0.000       1.184       2.406
==============================================================================
```

← Your 3rd model

← Baseline model

Confidence Intervals

Confidence Intervals

Think of confidence intervals as the logistic regression model estimating the range of plausible values for your model's coefficients.

You are looking for any confidence intervals that do not include the value of 0.0…

"95% Confidence Interval"

[0.025	0.975]	
-7.219	-3.183	
0.813	2.117	← **Zero is not included**
0.028	0.094	← **Zero is not included**
1.184	2.406	← **Zero is not included**

When the confidence interval **does not** include 0.0, the coefficient is *statistically significant*.

Based on the data used to craft the logistic regression model, statistically significant coefficients are important predictors.

NOTE – Confidence intervals are estimated for a particular dataset and a particular set of independent variables!

Your Third Model

```
Optimization terminated successfully.
         Current function value: 0.538839
         Iterations 6
                           Logit Regression Results
==============================================================================
Dep. Variable:           HeartDisease   No. Observations:                  270
Model:                          Logit   Df Residuals:                      266
Method:                           MLE   Df Model:                            3
Date:                Sun, 23 Jul 2023   Pseudo R-squ.:                  0.2156
Time:                        17:10:56   Log-Likelihood:                -145.49
converged:                       True   LL-Null:                       -185.48
Covariance Type:            nonrobust   LLR p-value:                 3.090e-17
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept     -5.2011      1.030     -5.051      0.000      -7.219      -3.183
Male           1.4648      0.333      4.404      0.000       0.813       2.117
Age            0.0614      0.017      3.643      0.000       0.028       0.094
Angina         1.7952      0.312      5.761      0.000       1.184       2.406
==============================================================================
```

Coefficients

Interpreting Coefficients

Logistic regression model coefficients require a transformation to make them useful for interpretation.

The transformation allows the coefficients to be interpreted as *odds ratios*.

Odds ratios represent the *strength of association* between two *events*.

As it turns out, there is a very specific mathematical relationship between the coefficients of your models and the odds ratio calculation.

This relationship provides a shortcut for your calculations:

$$odds\ ratio = e^{Coefficient\ Value}$$

Given your third model and the *Male* independent variable:

$$odds\ ratio = e^{1.4648} = 4.3265$$

In Python code:

```
from math import exp

print(exp(model_3_results.params['Male']))
```

```
4.326528261966836
```

Interpreting Your Third Model

Dave ON DATA

```
                     coef
-------------------------
Intercept         -5.2011
Male               1.4648
Age                0.0614
Angina             1.7952
```

```python
from math import exp

print(exp(model_3_results.params['Male']))
```

```
4.326528261966836
```

The model estimates that males are 4.33 times more likely to have heart disease

```python
# Get the odds ratio for the Age coefficient
print(exp(model_3_results.params['Age']))
```

```
1.063317182375142 6
```

The model estimates that each year of age raises the relative risk of heart disease by 6.3%

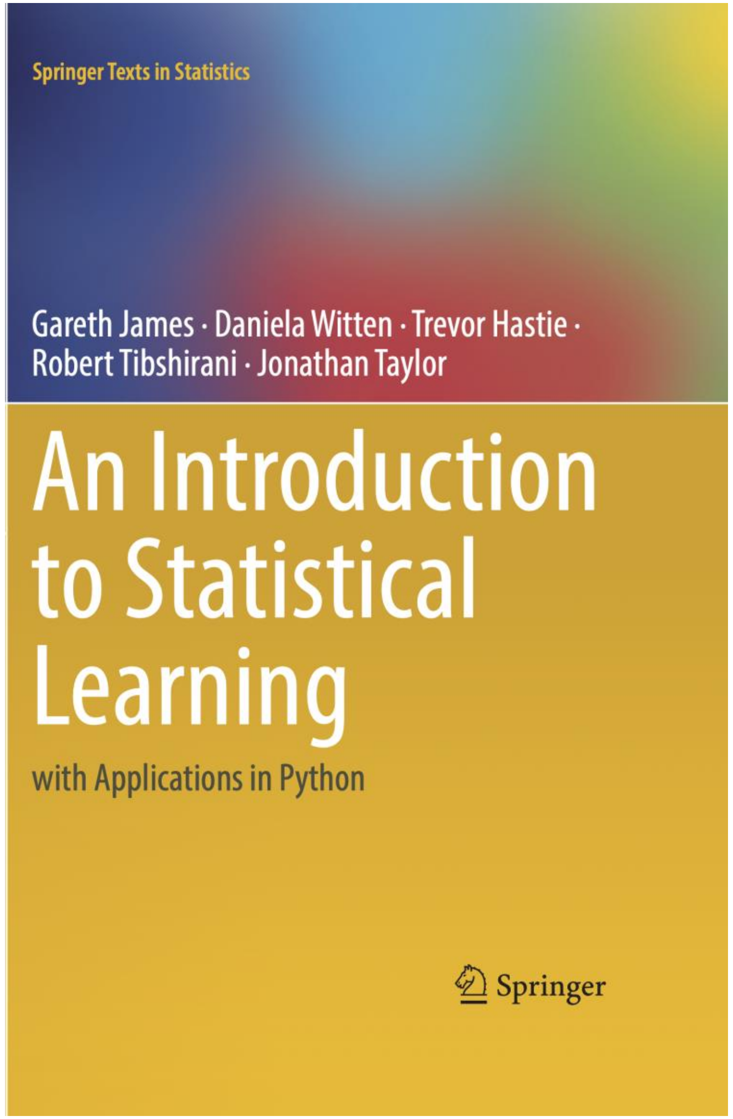
```python
# Get the odds ratio for the Angina coefficient
print(exp(model_3_results.params['Angina']))
```

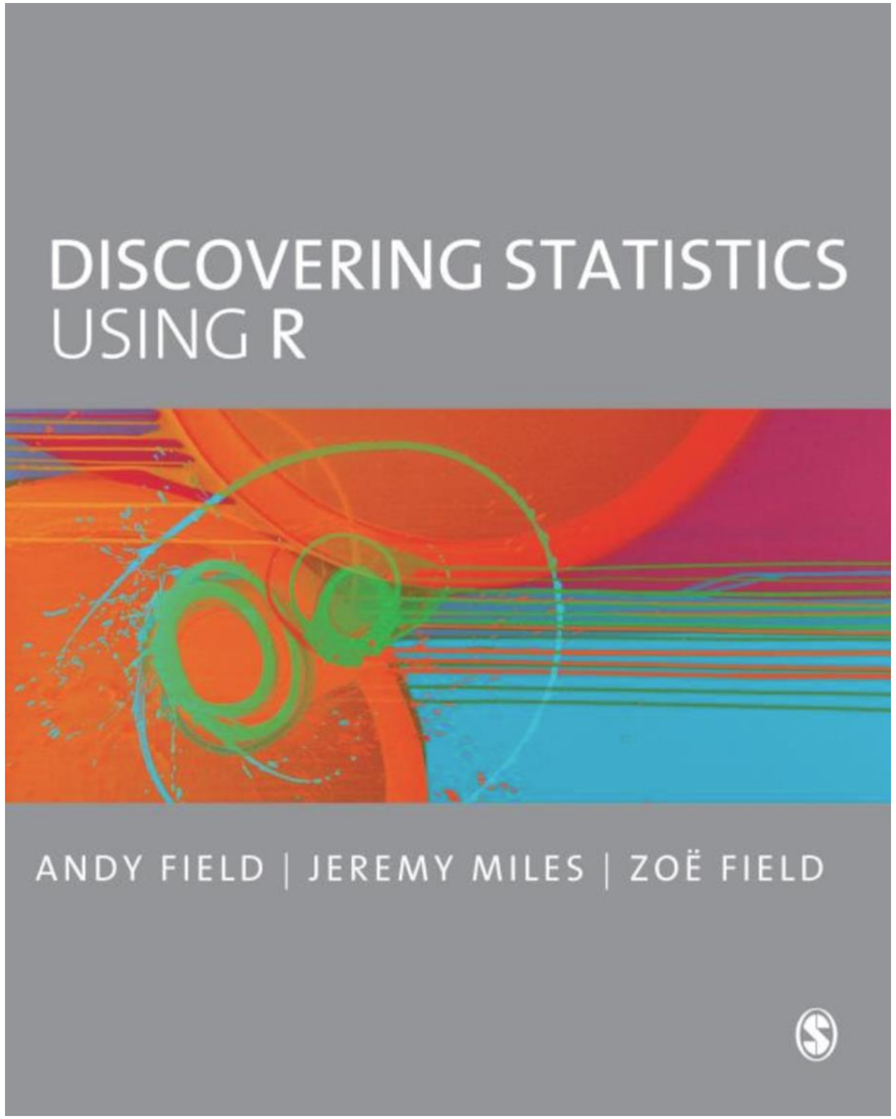
```
6.020938842080951
```

The model estimates that patients with angina are 6 times more likely to have heart disease.

Wrap-Up

Continue Your Learning





Thank You!