CRA Week 4:
Logistic Regression with Generalized Linear Models

MAISI

Michigan Data Science Team
Fall 2025

Session 4 Agenda

01
Fun Icebreaker!!
Get to know your projectmates!

02
Intro to Logistic Regression
What is logistic regression?

03
Coding Logistic Regression
The functions you need to know to accomplish our goals.

04
Interpreting Logistic Regression
What do our findings tell us about our dataset?

05
Practice Time!
Work on the Logistic Regression notebook!

Quick Icebreaker!!

If you could have any animal for a pet (real or mythical), what would it be? (don't be basic)

Also, what were your highlights from fall break?

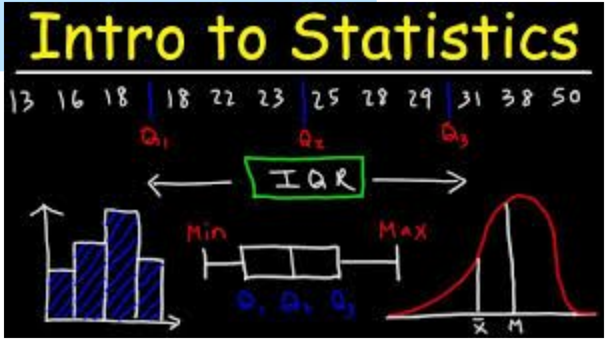
Share with the people around you :)

But first, a quick review!

Continuous variable: A value in a given range. Can represent measurement, count, etc. Something like distance, temperature, speed, and so on.

Categorical variable: Represents a category - there are no “in-between” values. Examples include race and gender (hmm… how do these relate to COMPAS…)

Probability: a value between 0 and 1 that tells us the likelihood of an event happening



Log-Odds of the Outcome

- The **log-odds** is the **natural logarithm** of the odds of an event happening
- The **odds** of an event is the ratio of the probability

$$\text{Odds} = \frac{p}{1-p}$$

Where p is the probability that an event happens. (1 - p) is the **complement** of p.

- **Logistic regression** models a linear relationship between **predictors** and the **log-odds** of the outcome

What are Log-Odds?

We can also express the equation as:

$$\operatorname{logit}(p) = \ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k$$

- This is the **linear relationship** between independent variable **predictors** $x_1, \ldots, x_n$ and the **log-odds** of the outcome

- Each coefficient **β** represents the effect of a **one-unit increase** in the predictor on the log-odds
  - β_0 is the “intercept” or the “bias”

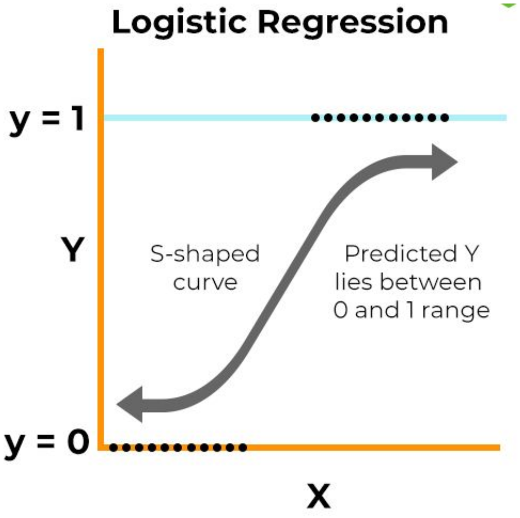
…

What is Logistic Regression?

- Logistic Regression estimates the probability of a binary event y happening based on predictors $x_1, \dots, x_n$.

$$p(y=1|X) = \frac{1}{1+e^{-(\beta_0+\beta_1 x_1+\beta_2 x_2+\dots+\beta_n x_n)}}$$

- It "squashes" the linear combination of predictors into the range (0, 1), making it suitable for probabilities



Logits and Probabilities

$$z = \operatorname{logit}(p) = \log\left(\frac{p}{1-p}\right)$$

To convert a **logit z** (the log odds of an event) back into a probability, you can use the function:

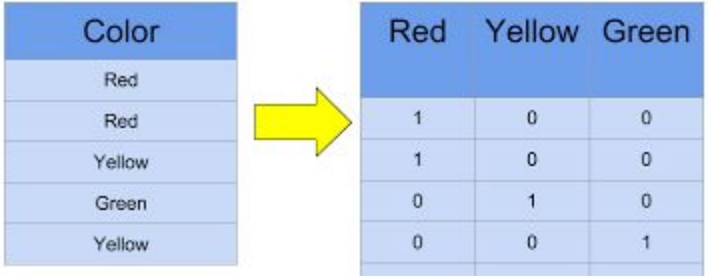
$$p = \frac{1}{1+e^{-z}}$$

...

Logistic Regression with Categorical Variables

- In this lesson, we will use logistic regression to predict the probability of the binary event: (Low COMPAS score vs. High COMPAS score.)

- Our predictors will be **categorical variables** (Race, Gender, etc.). Since the default logistic regression equation's predictors are continuous, we will have to use **one-hot encoding** to modify the equation to fit categorical predictors.



Color
Red
Red
Yellow
Green
Yellow

Red	Yellow	Green
1	0	0
1	0	0
0	1	0
0	0	1

How does one-hot encoding work?

Suppose we have a categorical predictor with three categories: "A," "B," and "C." After one-hot encoding, we will create two binary variables:

- x_A = 1 if the category is "A" and 0 otherwise.
- x_B = 1 if the category is "B" and 0 otherwise.
- The category "C" is the reference category, so if $x_A = x_B = 0$, the observation is assigned to "C."
- We can say A and B are compared in reference to C. γ_A, γ_B are coefficients of A and B.

$$\mathrm{logit}(p) = \ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k + \gamma_A x_A + \gamma_B x_B$$

Why use Logistic Regression?

We're using Logistic Regression to understand how different independent variable "predictors" lead to low score vs. medium/high score.

You may be asking: *"But couldn't we do this with linear regression?"*

- Logistic Regression gives us a new ability: We can **adjust for confounders (confounding variables)**

Adjusting for Confounders

- In general, when we're trying to **predict the probability** of an event happening in real life, there are countless other that **affect the final outcome**.
- With logistic regression, we can **keep the other variables (confounders) constant** and only look at the effect on probability when changing a single predictor. This will help us understand how different groups have different score distributions.
- In mathematical terms, we'll keep all of the terms with **gamma coefficients = 0** except for one term that is different, looking at the probability for a situation where all categorical variables are referenced except for one.

$$\operatorname{logit}(p) = \ln\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k + \gamma_A x_A + \gamma_B x_B$$

Commands for Coding Logistic Regression

Change to categorical:

- **df['col_cat'] = df['col'].astype('category')**
    - Ensures the column is treated as categorical (essential for GLM models with factors)

Relevel categories:

- **df['col_cat'] = df['col_cat'].cat.reorder_categories(['A', 'B', ‘C’])**
    - Reorders categories, where the first item in the list is your reference category, the rest don’t matter
    - i.e., Setting “Male” as the reference gender variable allows all comparisons to be made against “Male”

Generalized Linear Model Syntax

Import libraries:

- **import statsmodels.formula.api as smf:** This allows us to use formula-based syntax for specifying the model (similar to R-style)
- **import statsmodels.api as sm:** Imports the main **statsmodels** library, including GLM families like **Binomial()** for logistic regression

Generalized Linear Model Syntax

Define the model:

- model = **smf.glm**(formula='dependent_variable ~ predictor1 + predictor2 + predictor3', data=df, family=sm.families.Binomial()).fit()
  - Creates a generalized linear model (GLM)
  - Specify the relationship between the **dependent variable** (outcome) and **predictors**
  - Specify the model as **binomial** (logistic) regression
  - **.fit()**s the model to the data using maximum likelihood estimation (finding the best coefficients for the predictors -> NumPy!)

And then **print(model.summary()):** coefficients, standard errors, z/p-values

How to Interpret

Coefficients: Represents the **change** in the **log-odds** of the outcome for a **one-unit increase** in the predictor variable

- **Positive/Negative**: increases/decreases the log-odds (and probability) of the outcome
- **Exponentiating** the coefficients gives us the **odds ratio**, which makes it easier to interpret the **effect** of each predictor in terms of how much it increases or decreases the odds of the outcome occurring

e^{β_1} represents the **odds ratio** associated with a one-unit increase in x_1.

Std. Error: A measure of the **precision** of the coefficient estimate

- **Smaller** standard errors suggest **more confidence** in the coefficient.
- **Larger** standard errors indicate that the estimate may be **less reliable**.

$$SE = \frac{\sigma}{\sqrt{n}}$$

Standard deviation

Number of samples

How to Interpret

z: The **ratio** of the coefficient to its standard error (**Coefficient / Std. Error**)

- Shows how many **standard deviations** the coefficient is from **zero**
- **Higher** absolute values indicate that the **predictor** is **likely significant**.

P>|z|: Represents the **probability** of observing the effect seen in the data (or something **more extreme**) if there were actually **no true effect** (i.e., if the null hypothesis were true, being that this variable has no effect on COMPAS scoring)

- **Low p-value (< 0.05)**: Suggests that the predictor is statistically significant.
- **High p-value (≥ 0.05)**: Indicates the predictor may **not** significantly impact the outcome.
    - Common thresholds for significance are 0.05 or 0.01

Practice Time!

Time to investigate what factors have a significant impact on COMPAS scoring!

Hands-On Data Science!! :O

Next Steps:

1. Find the F25 CRA repo in the MDST GitHub and download all the files in the Week 4 directory (GLM.ipynb is all you need)
2. Split into teams of 2-3 (new ones or same as last week)
   a. If you're working with new people, introduce yourselves!!
3. Work on the exercises in the notebooks!
   a. Ask us if you need help!

Pandas Cheat Sheet Seaborn Cheat Sheet

Reminders

- Don't share colab notebooks with teammates if you are working at the same time

- Where to put csv and data files
  - Google Drive
    - Need to include:

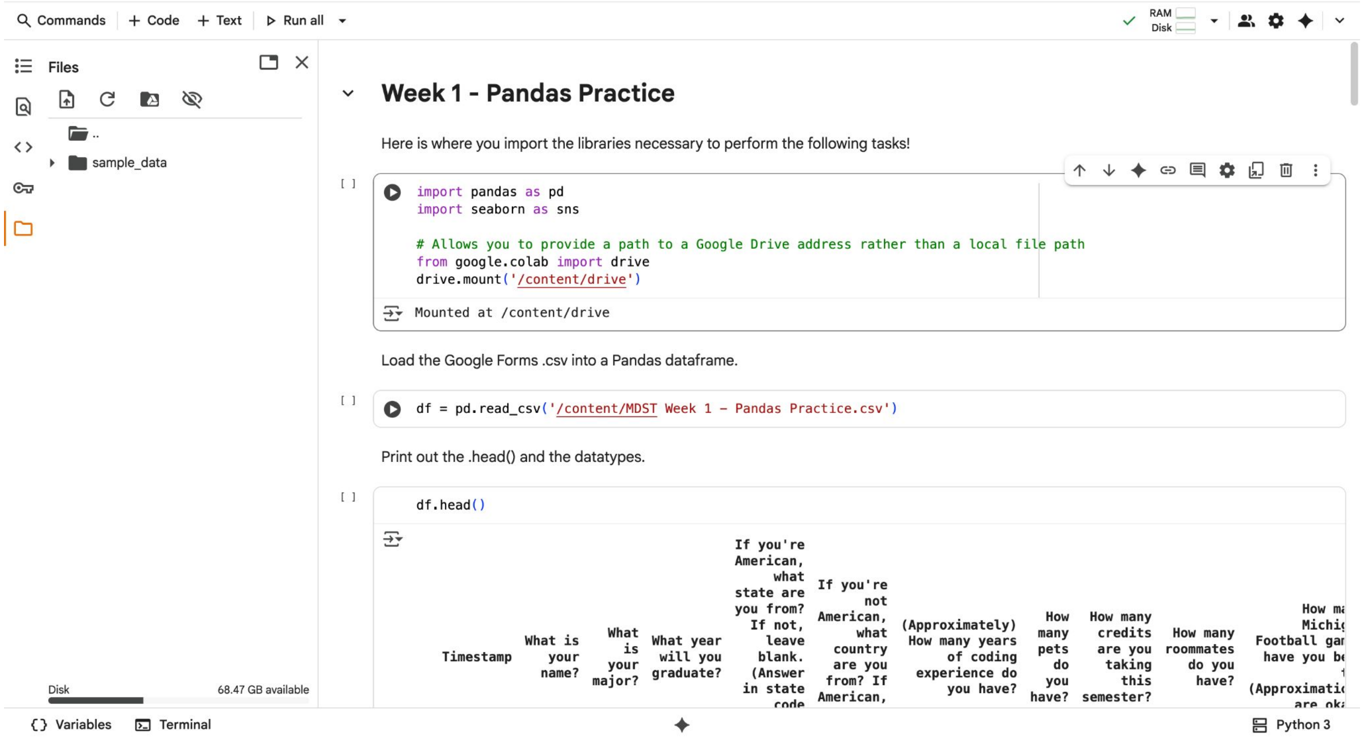
```
from google.colab import drive
drive.mount('/content/drive')
```

```
pd.read_csv('/content/drive/MyDrive/[FILE NAME]')
```

  - Colab Files
    - See next slides

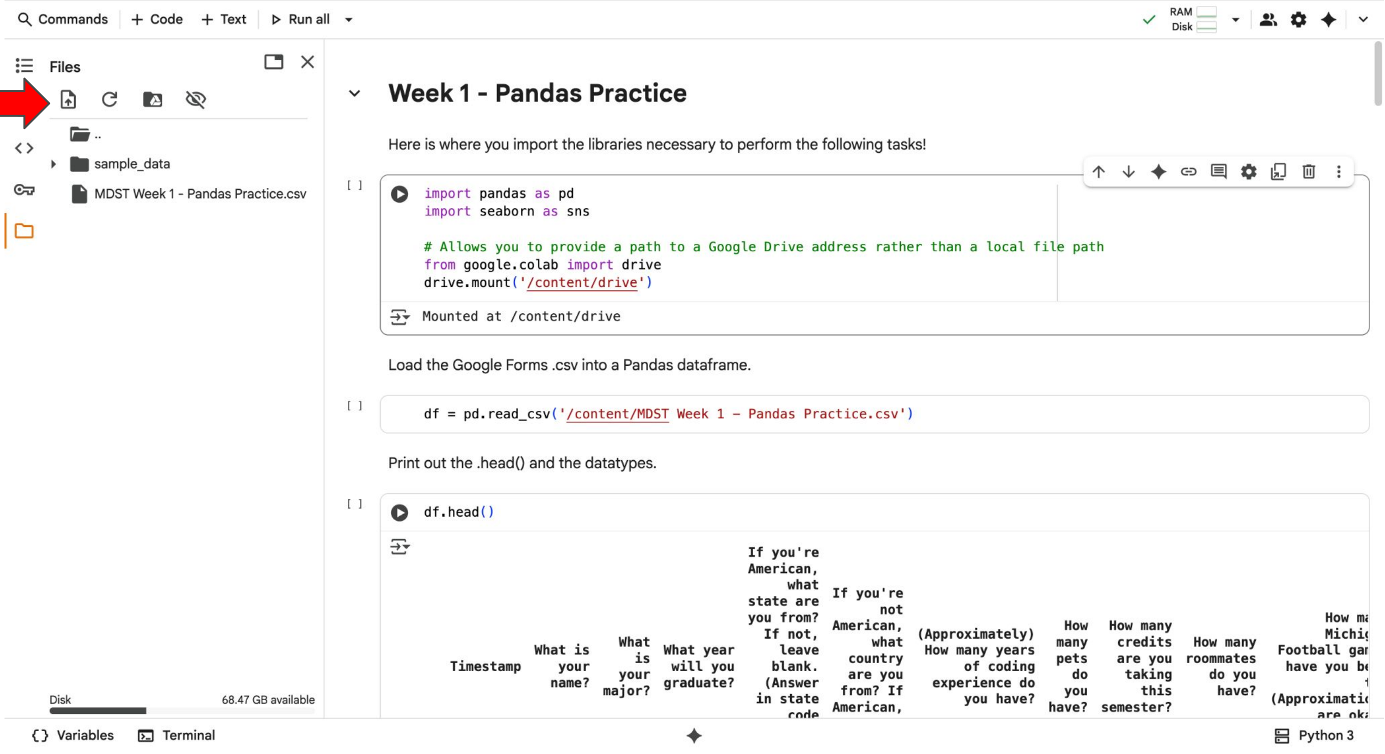
Reminders

Click on the folder in the sidebar



Reminders

Click the upload button and select the file you want to upload



Reminders

Click the three dots and copy the path. Put this in your read function