

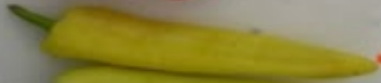
## Classification

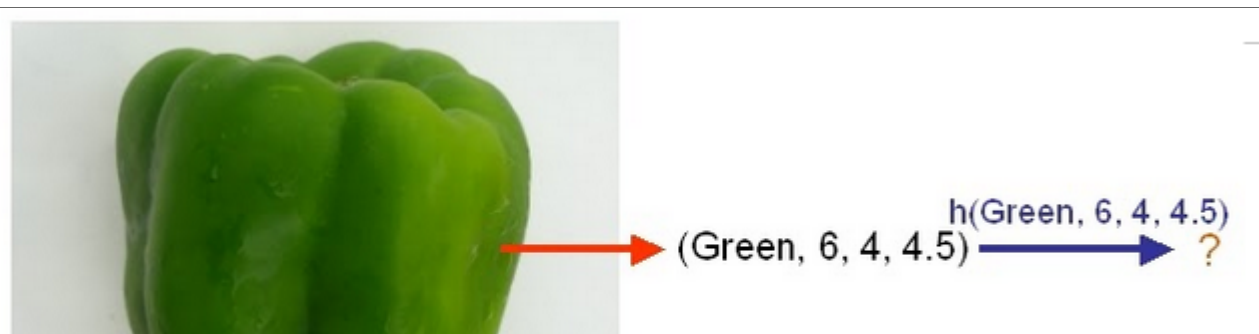
- Classification, is an area of *supervised learning* that addresses the problem of how to systematically assign unlabeled (**classes** unknown) novel data to their labels (**classes** or groups or types) by using knowledge of their **features** (characteristics or attributes) that are obtained from observation and/or measurement.
- A classifier algorithm is a specific technique or method for performing classification.
- To learn to classify, the classifier algorithm first uses labeled (classes are known) training data to train a model (i.e., fit parameters), and then it uses a function known as its classification rule (or for short, the **classifier**) to assign

a label to each new data point given the feature values.

- A simple measure of classification performance is **accuracy**, that is what fraction of the test data is labeled accurately.

## The automated checkout problem

Objects	Features (X)	Labels (Y)
	→ (Green, 6, 4, 4.5) →	<b>Green Pepper</b>
	→ (Green, 7, 4.5, 5) →	<b>Green Pepper</b>
	→ (Red, 6, 3, 3.5) →	<b>Red Pepper</b>
	→ (Red, 4.5, 4, 4.5) →	<b>Red Pepper</b>
	→ (Yellow, 1.5, 8, 2) →	<b>Hot Pepper</b>
	→ (Yellow, 1.5, 7, 2.5) →	<b>Hot Pepper</b>





## The intuitive importance of Classification

- In conventional statistics courses, and experimental psychology or neuroscience courses, emphasis is placed on the notion of finding a \textit{significant} difference between two (or more) subject groups or experimental conditions.
- For example in clinical research we ask questions such as
  - **Is the patient data different than the control data?**
- But in our minds (and definitely in the patient and in the physicians mind) perhaps we should ask a different question -
  - **Based on the characteristics of the patients data, can we determine if the data comes from a patient or a control?**
- The first approach is built around hypothesis testing for differences, the second approach is classification of data.

## What is a Classifier?

- At the simplest level a Classifier is a decision rule as in Signal Detection Theory (SDT), that allows us to categorize data. In fact, many of the ideas we will discuss closely mirror SDT.
- For example, when you go to the doctor they take your blood pressure, and you get a pair of numbers like 120/80 for the systolic/diastolic pressure.
- The doctor has a decision rule. If the systolic pressure is above 130, the patient receives a stern lecture about diet and exercise, and if the systolic pressure is above 140, medication is prescribed to lower blood pressure.
- Thus, there are 3 classes of patients based on the systolic blood pressure reading.
  1. below 130 - healthy
  2. 130-140 - borderline

### 3. above 140 - medication

- **This is a 3-class classifier**
- How were these critical values found? Hopefully, huge amounts of data are collected to look at patient cardiovascular health and blood pressure, and the data says if blood pressure remains above 140, the heart walls thicken and secondary cardiovascular diseases can emerge. (There are other bad effects too).

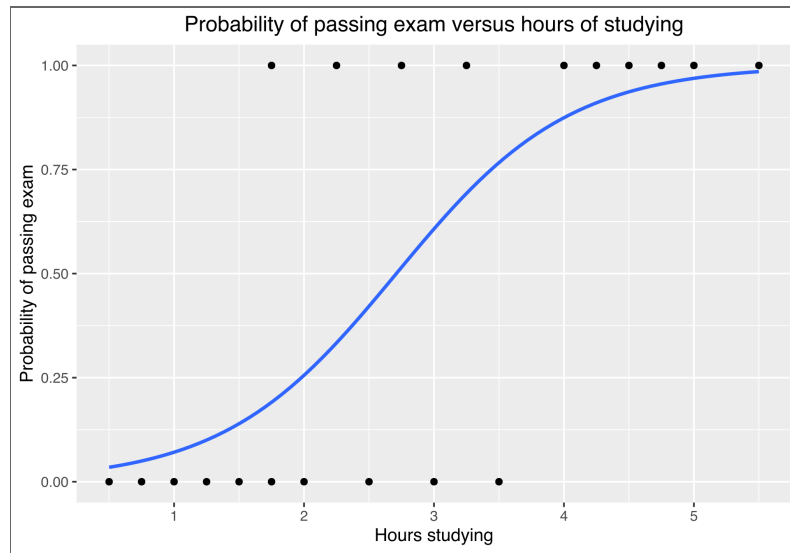
## Examples in Cognitive Science/Cognitive Neuroscience

1. Categorization
2. Automatic Speech Recognition
3. Face Recognition
4. Brain-Computer Interfaces
5. Biomarkers for Mental Health
6. Single-trial analysis of Neural Signals
  - In data science/machine learning applications, we are solely interested in making classifiers work as accurately as possible.
  - In scientific applications, we want to know how the classifier was able to work. We want to know what *features of the data* were useful and what *transformations of the data* produced the accurate classification.



## Logistic Regression

- The first classifier we will discuss in this class is **Logistic Regression**.
- In Linear Regression, we fit a line to data.
- In a simple (two-class) Logistic Regression we will fit a curve to the probability that the data comes from one **class**



## Logistic Function

Logistic Regression addresses the problem of estimating a probability,  $P(Y = 1)$ . The logistic regression model uses a function, called the logistic function:

$$P(Y = 1 \mid x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}} =$$

In [1]:

```
import numpy as np
import matplotlib.pyplot as plt

def logistic(x, beta0=0, beta1=1):
    p = 1 / (1 + np.exp(-(beta0 + beta1 * x)))
    return p

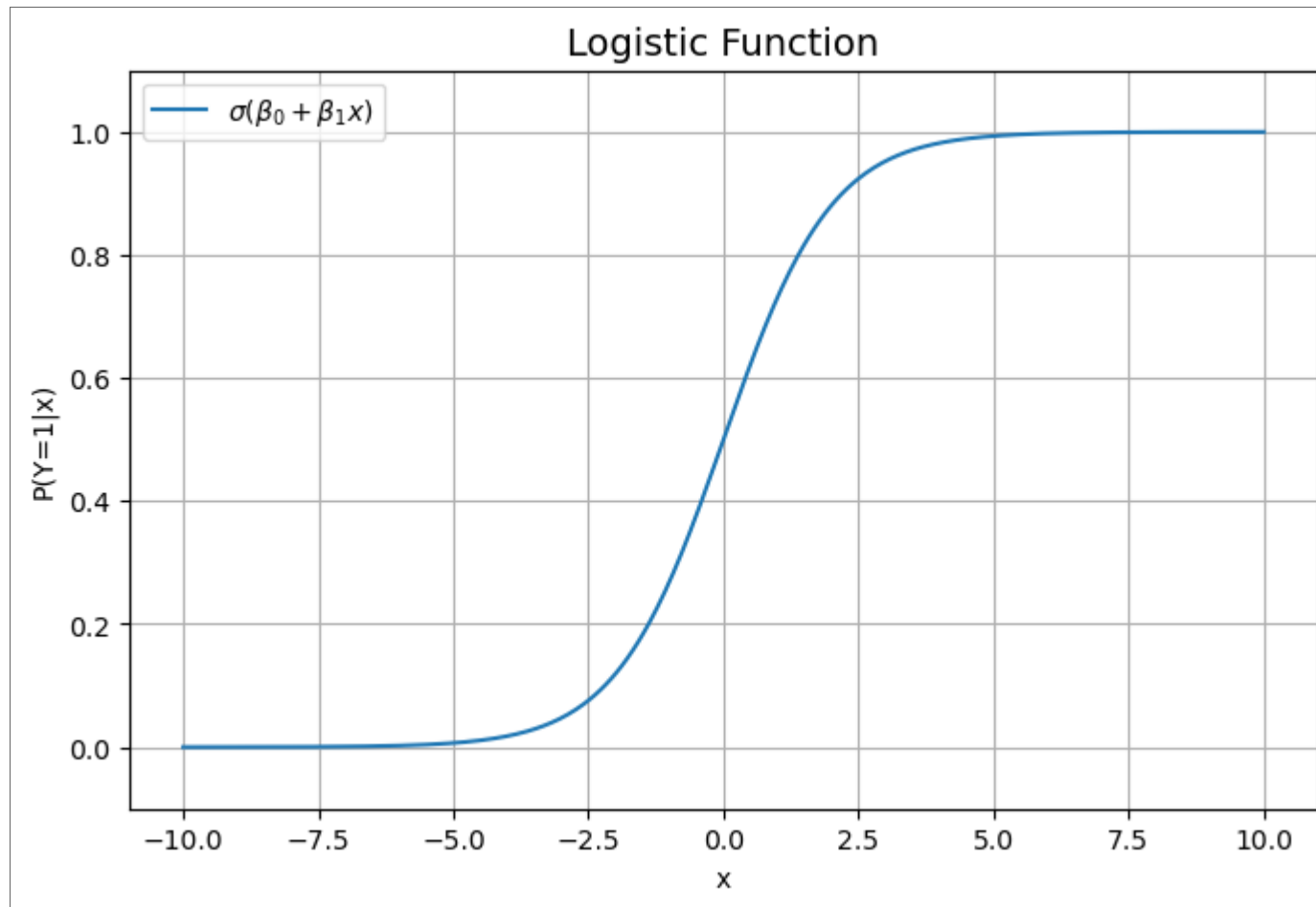
# Generate a range of x values
x = np.linspace(-10, 10, 400)

# Parameters
beta0 = 0    # Intercept
beta1 = 1    # Slope

# Compute logistic values
y = logistic(x, beta0, beta1)

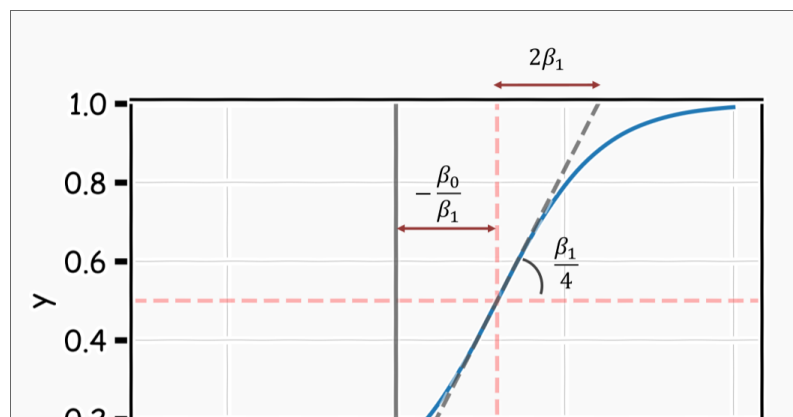
# Plot
plt.figure(figsize=(8, 5))
plt.plot(x, y, label=r'$\sigma(\beta_0 + \beta_1 x)$')
plt.title("Logistic Function", fontsize=14)
plt.xlabel("x")
plt.ylabel("P(Y=1|x)")
plt.grid(True)
plt.legend()
plt.ylim(-0.1, 1.1)
```

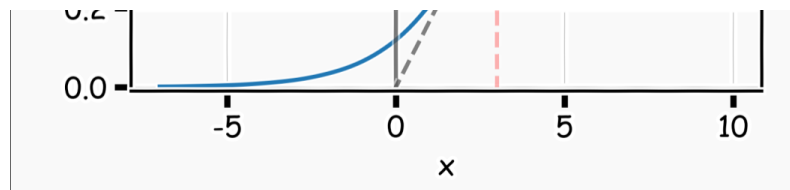
```
plt.show()
```



The probability model will predict  $P(Y = 1|x)$  with an S-shaped curve:

- $\beta_0$  shifts the curve right or left by  $c = \frac{-\beta_0}{\beta_1}$
- $\beta_1$  controls the steepness the S-shaped curve. Distance from  $\frac{1}{2}$  to almost 1 or  $\frac{1}{2}$  to almost 0 to  $\frac{1}{2}$  is  $\frac{2}{\beta_1}$
- if  $\beta_1$  is positive, then the predicted  $P(Y = 1|x)$  goes from zero for small values of  $x$  to one for large values of  $X$  if  $\beta_1$  is negative, then the predicted  $P(Y = 1|x)$  goes from one for small values of  $x$  to zero for large values of  $X$





- It's useful to rewrite the logistic regression model, in terms of odds. This is called the **logit** function by statisticians

$$\text{logit}(P(Y = 1 \mid x)) = \ln\left(\frac{P(Y = 1 \mid x)}{1 - P(Y = 1 \mid x)}\right) = \beta_0 + \beta_1 x$$

- The ratio shown is the **odds** ratio between the probability of  $Y = 1$  with the probability  $Y = 0$ , if  $Y$  can only be 1 or 0
- What happens with the odds ratio is 1, i.e.,  
 $P(Y = 1) = 0.5$