Berkeley Engineering | BerkeleyHaas

PROFESSIONAL CERTIFICATE IN MACHINE LEARNING AND ARTIFICIAL INTELLIGENCE

Office Hour #13 with Matilde D'Amelio

Logistic Regression

Difference with other classification models

KNN	Logistic Regression
Non Parametric	Parametric
Label Observations	Label Observations
Does not give Level of Confidence	Give Level of Confidence

Naive Bias	Logistic Regression
No Collinearity (independent features)	Deal well with Collinearity
Label Observations	Label Observations
Categories are created by the model	Categories are given to the model

Decision Tree	Logistic Regression
Deal well with Collinearity	Does not deal as well with Collinearity
Label Observations	Label Observations
Does not give Level of Confidence	Give Level of Confidence

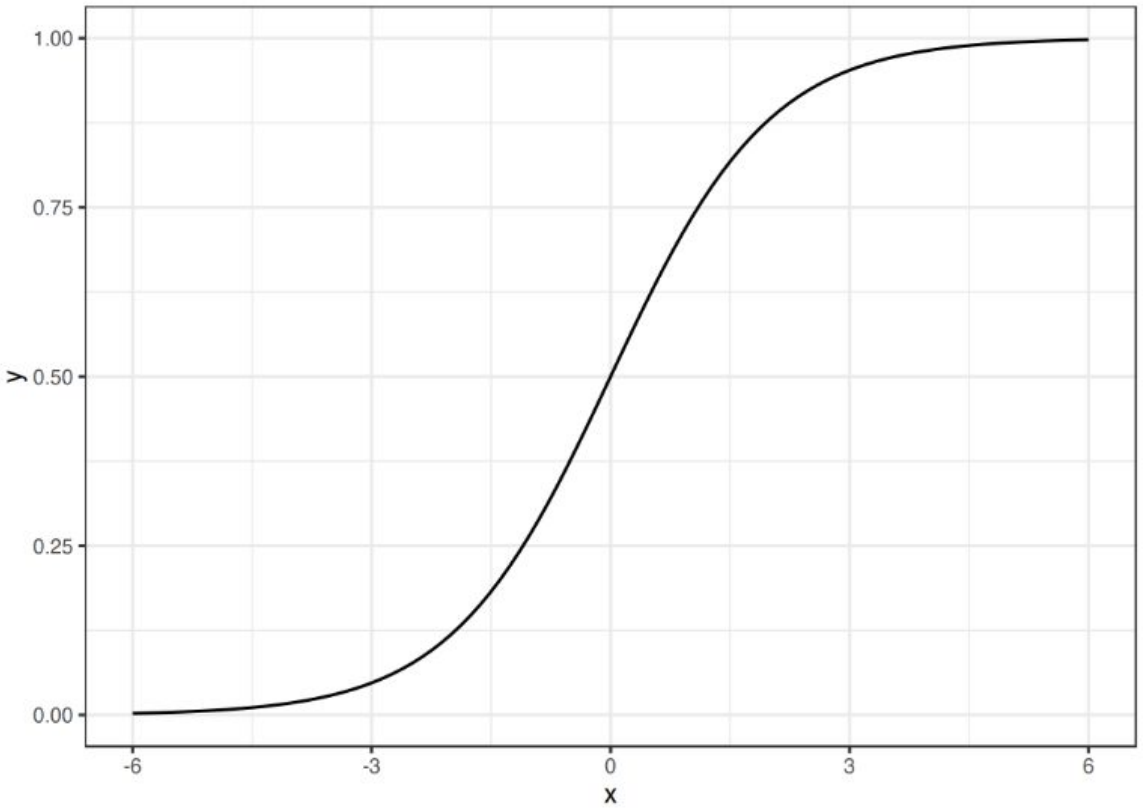
Support Vector Machine (SVM)	Logistic Regression
Parametric	Parametric
Label Observations	Label Observations
Give Level of Confidence	Give Level of Confidence

Logistic Regression

It is the go-to method for binary classification problems
For example, if we are modeling people's sex as male or female from their height, then the first class could be male and the logistic regression model could be written as the probability of male given a person's height (Maximum-likelihood estimation)

The logistic regression model uses the logistic function to squeeze the output of a linear equation between 0 and 1

$$\text{logistic}(\eta) = \frac{1}{1 + exp(-\eta)}$$



Logistic Regression

Simplest Classification Model: Binary Response

Given a training set,The simplest form of classification is when the response variable y has only two categories, and then an ordering of the categories is natural. For example, an Imperial student could be categorized as (note, the y=0 category is a "catch-all" so it would involve all other situations):

$$y = \begin{cases} 1 & \text{If lives on Berkeley campus} \\ 0 & \text{otherwise} \end{cases}$$

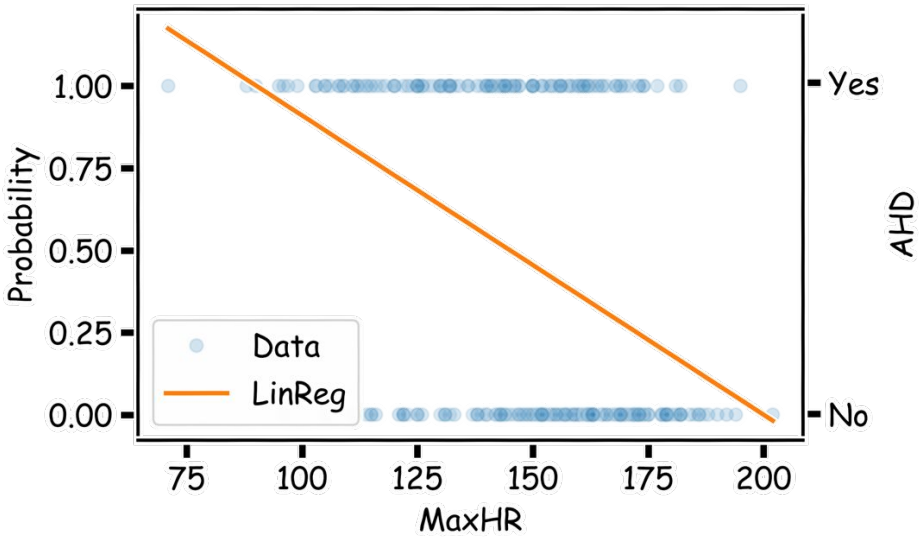
Logistic regression could be used to predict y directly from a set of covariates (like gender, whether an athlete or not, concentration, GPA, etc.), and if $\hat{y} \geq 0.5$, we could predict the student lives on campus and predict other houses if $\hat{y} < 0.5$.

Prepare Data for Logistic Regression

- **Binary Output Variable**: Logistic regression is intended for binary (two-class) classification problems. It will predict the probability of an instance belonging to the default class, which can be snapped into a 0 or 1 classification.
- **Remove Noise**: Logistic regression assumes no error in the output variable (y), consider removing outliers and possibly misclassified instances from your training data.
- **Gaussian Distribution**: Logistic regression is a linear algorithm (with a non-linear transform on output). It does assume a linear relationship between the input variables with the output. Data transforms of your input variables that better expose this linear relationship can result in a more accurate model. For example, you can use log, root, Box-Cox and other univariate transforms to better expose this relationship.
- **Remove Correlated Inputs**: Like linear regression, the model can overfit if you have multiple highly-correlated inputs. Consider calculating the pairwise correlations between all inputs and removing highly correlated inputs.
- **Fail to Converge**: It is possible for the expected likelihood estimation process that learns the coefficients to fail to converge. This can happen if there are many highly correlated inputs in your data or the data is very sparse (e.g. lots of zeros in your input data).

Logistic Regression

What could go wrong with this linear regression model?

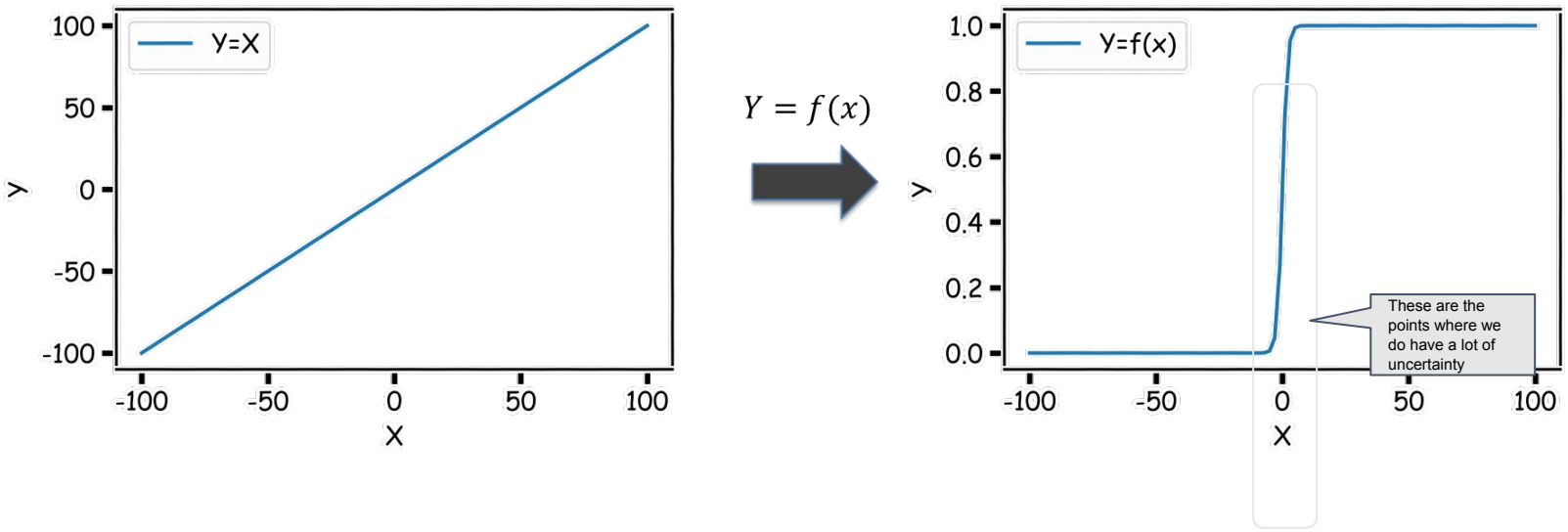


The main issue is you could get nonsensical values for y. Since this is modeling P(y=1) , values for yˆbelow 0 and above 1 would be at odds with the natural measure for y. Linear regression can lead to this issue.

Logistic Regression

Think of a function that would do this for us

Think of a function that would do this for us



Adapted from Harvard CS109A

Logistic Regression

Think of a function that would do this for us

Logistic Regression addresses the problem of estimating a probability, P(y=1), to be outside the range of [0,1]. The logistic regression model uses a function, called the ***logistic*** function, to model P(y=1):

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

Adapted from Harvard CS109A

Logistic Regression

Think of a function that would do this for us

As a result the model will predict P(y=1) with an S-shaped curve, which is the general shape of the logistic function.

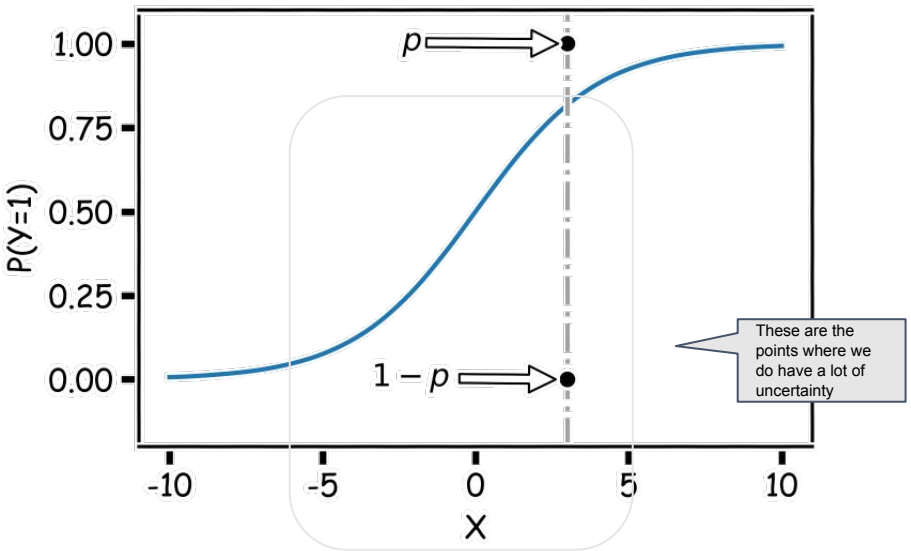
β0 shifts the curve right or left by $c = -\frac{\beta_0}{\beta_1}$

β1 controls how steep the S-shaped curve is. Distance from ½ to almost 1 or ½ to almost 0 to ½ is 2/β1

Note: if β1 is positive, then the predicted P(y=1) goes from zero for small values of X to one for large values of X and if β1 is negative, then the P(y= 1) has opposite association.

Logistic Regression

Estimation in Logistic Regression



Probability $Y = 1$: p
Probability $Y = 0$: $1 - p$

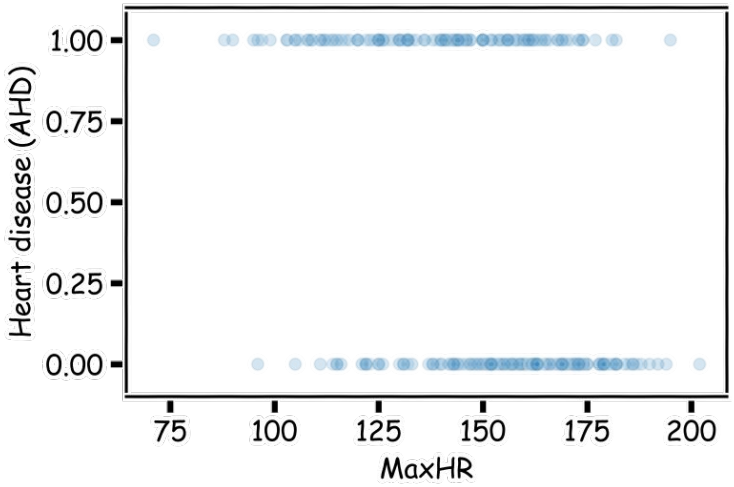
$$P(Y = y) = p^y(1 - p)^{(1-y)}$$

where:
$p = P(Y = 1|X = x)$ and therefore p depends on X.
Thus not every p is the same for each individual measurement.

Logistic Regression

Predict whether a person has heart disease

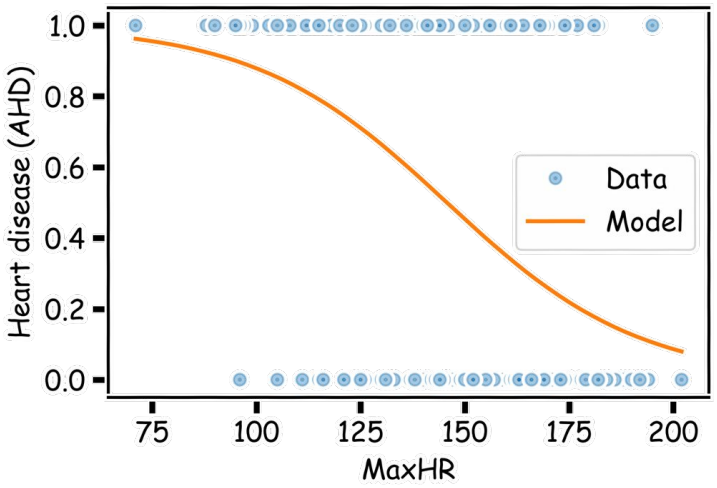
How should we visualize these data?



Logistic Regression

Predict whether a person has heart disease

How should we visualize these data?



Adapted from Harvard CS109A

Logistic Regression

We'll start with a real simple equation. The probability P of a binary event occurring is:

$$P$$

The probability of a binary event not occurring must then be:

$$1 - P$$

The Odds is defined as the **ratio of success to the ratio of failure**.

$$O = \frac{P}{1 - P}$$

It should be clear from the above equation that if P can vary in the range $[0, 1]$ then O can vary in the range $[0, \infty]$. The higher the odds, the higher the chance of success. This should sound very familiar to gamblers, who use this term frequently.

Source: Geoffrey Hunter

Logistic Regression

The reason we introduce the $ln()$ function into the equation begins to make sense once you understand basic linear regression, which can be used to predict the probability of continuous target variables. The basic equation defining linear regression involving just one predictor x and the outcome y is:

$$y = \beta_0 + \beta_1 x$$
$$-\infty < x < \infty$$
$$-\infty < y < \infty$$

where:

β_n are the coefficients

x is the predictor

y is the outcome

The problem with using linear regression for making binary categorical predictions (i.e. `true/false`) is that y can vary from $-\infty$ to $+\infty$. We really want y to range from 0 to 1. When varying between 0 and 1, this tells us the probability of the target being `true` or `false`. For example, if $y = 0.7$ this would say there is a 70% chance of the target being `true`, and conversely a 30% chance of the target being `false`.

To make things less confusing, we will replace y which is used to represent a continuous target variable with P (for probability), which is used to represent the probability:

$$P = \beta_0 + \beta_1 x$$
$$-\infty < x < \infty$$
$$0 \leq P \leq 1$$

Source: Geoffrey Hunter

Logistic Regression

Notice a problem? This limits of the LHS and RHS of the equation don't match up! This is where we begin to understand why the log function is introduced. We will try and modify the RHS such that it has the same range as the LHS ($-\infty$ to $+\infty$). What if we replace the probability P on the LHS with the odds O:

$$O = \beta_0 + \beta_1 x$$
$$-\infty \leq x < \infty$$
$$0 \leq O < \infty$$

where:

O are the odds

We are getting closer! Now the RHS varies from 0 to ∞ rather than from just 0 to 1. So how do we modify a number which ranges from 0 to 1 to range from $-\infty$ to $+\infty$? One way is to use the `ln()` function! (to recap some mathematics, the `ln` of values between 0 and 1 map from $-\infty$ and 0, and the `ln` of values from 1 to ∞ map to 0 to ∞.)

$$\ln(O) = \beta_0 + \beta_1 x$$
$$-\infty < x < \infty$$
$$-\infty < \ln(O) < \infty$$

The ranges on both sides of the equation now match! The base of the logarithm does not actually matter. We choose to use the natural logarithm ($\ln$) but you could use any other base such as base 10 (typically written as log_{10} or just log).

Now we can see why it's called logistic regression, and why it is useful.

Source: Geoffrey Hunter

Logistic Regression

However, the equation is usually re-arranged with P on the LHS.

$$\ln(O) = \beta_0 + \beta_1 x \tag{1}$$

$$O = e^{\beta_0+\beta_1 x} \quad \text{Take the exponential of both sides} \tag{2}$$

$$\frac{P}{1-P} = e^{\beta_0+\beta_1 x} \quad \text{Substitute } O \text{ as per equation XXX} \tag{3}$$

$$P = e^{\beta_0+\beta_1 x}(1-P) \quad \text{Multiply both sides by } (1-P) \tag{4}$$

$$P = e^{\beta_0+\beta_1 x} - Pe^{\beta_0+\beta_1 x} \quad \text{Expand RHS} \tag{5}$$

$$P + Pe^{\beta_0+\beta_1 x} = e^{\beta_0+\beta_1 x} \quad \text{Move all terms with } P \text{ to the LHS} \tag{6}$$

$$P(1 + e^{\beta_0+\beta_1 x}) = e^{\beta_0+\beta_1 x} \quad \text{Factor the } P \tag{7}$$

$$P = \frac{e^{\beta_0+\beta_1 x}}{1 + e^{\beta_0+\beta_1 x}} \quad \text{Divide both sides of equation by } 1 + e^{\beta_0+\beta_1 x} \tag{8}$$

$$P = \frac{1}{\frac{1}{e^{\beta_0+\beta_1 x}} + 1} \quad \text{Divide top and bottom of RHS fraction by } e^{\beta_0+\beta_1 x} \tag{9}$$

$$P = \frac{1}{1 + e^{-(\beta_0+\beta_1 x)}} \quad \text{Use rule } \frac{1}{e^x} = e^{-x} \tag{10}$$

As you can see from above, P is now a form of a sigmoid function.

Source: Geoffrey Hunter

Logistic Regression

So we have the basic logistic function equation:

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

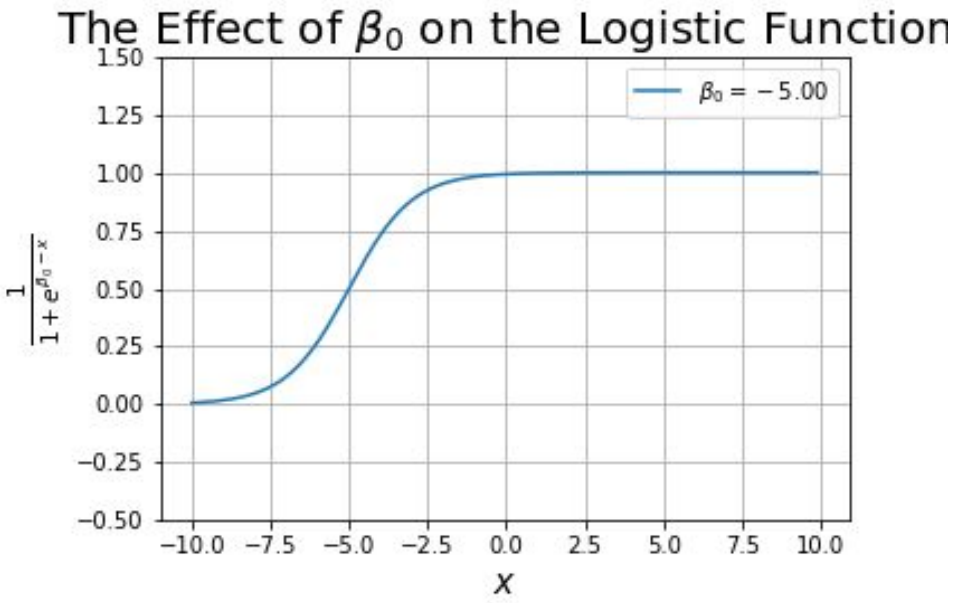
where:

β_0 and β_1 are constants

Source: Geoffrey Hunter

Logistic Regression

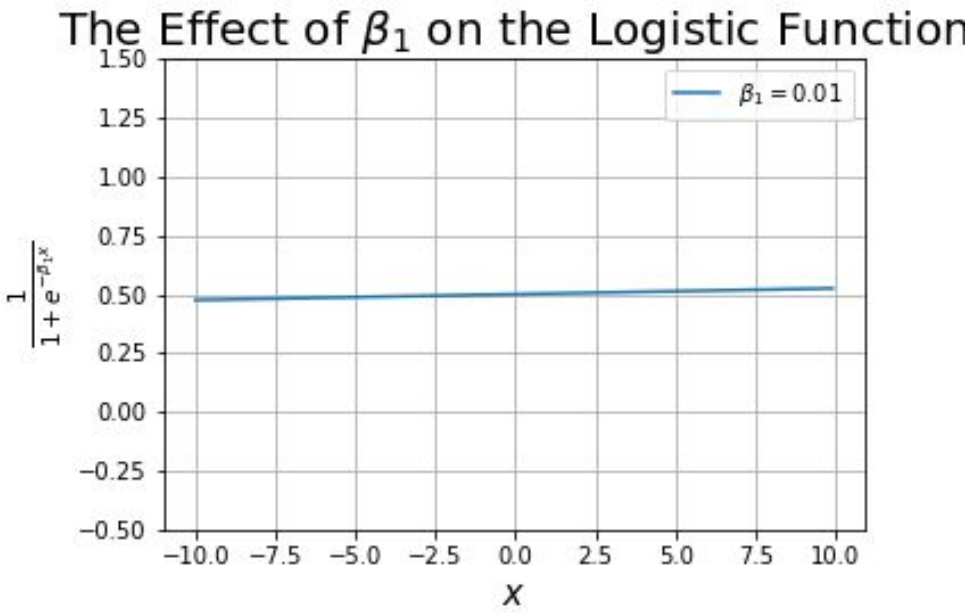
$$P(Y = 1) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 X)}}$$



Source: Geoffrey Hunter

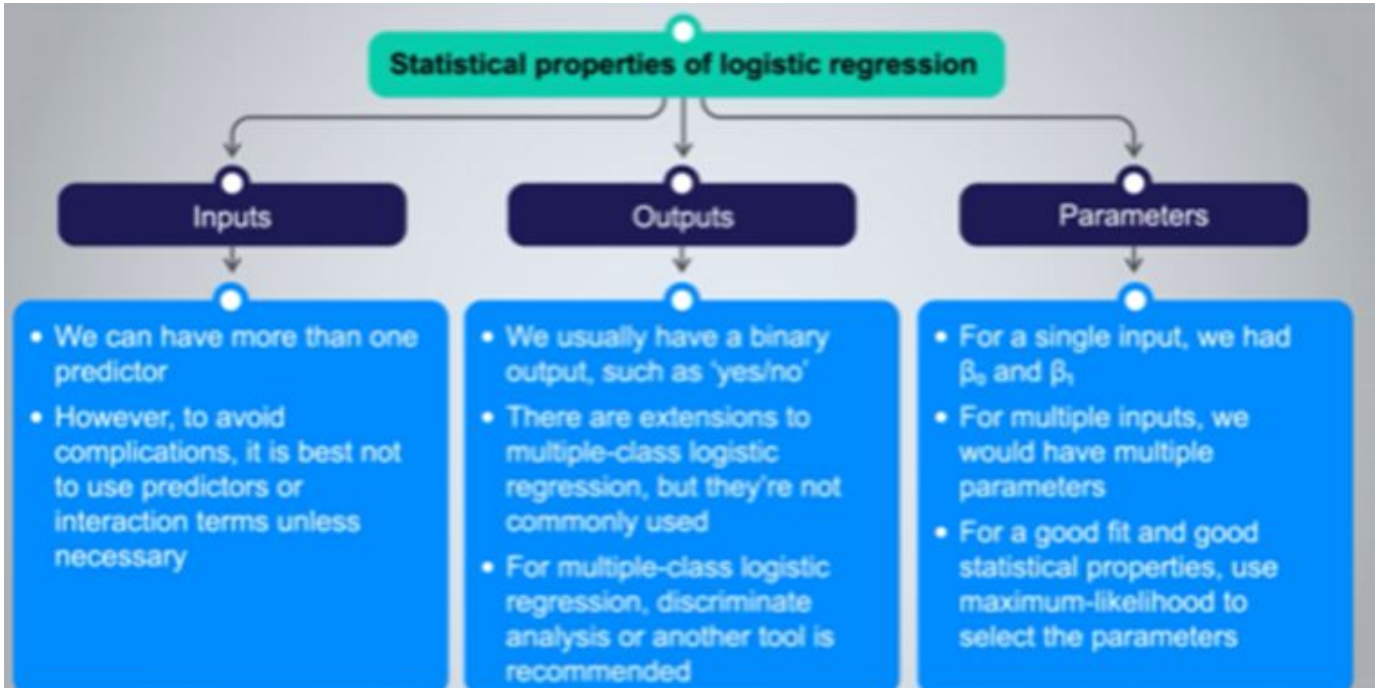
Logistic Regression

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



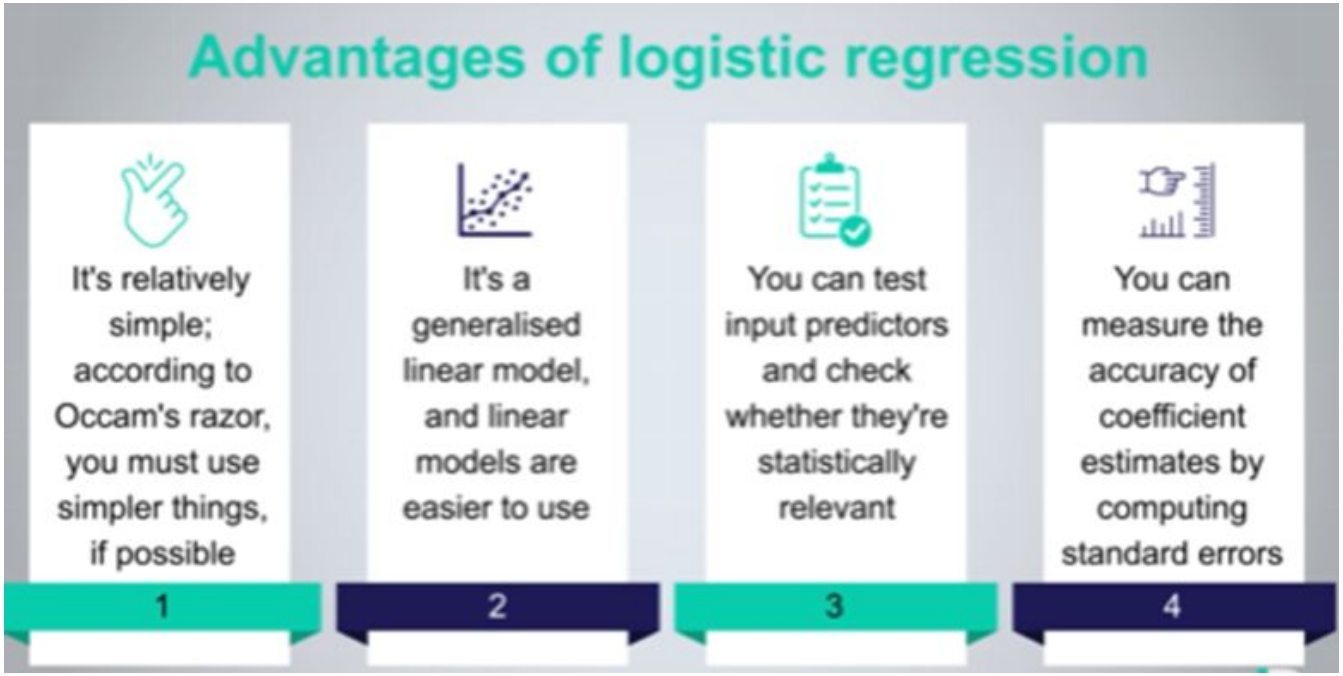
Source: Geoffrey Hunter

Statistical Properties



Logistic Regression

Advantages



Logistic Regression

```
import seaborn as sns
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report
from sklearn.model_selection import train_test_split
```

```
data = sns.load_dataset('iris')
print(data.shape[0])
# 150
data.head())
```

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

The first 5 rows of the Iris dataset.

Source: Geoffrey Hunter

Logistic Regression

Split the data into the features `x` and the target `y`:

```
x = data.iloc[:, 0:-1] # All columns except "species"
y = data.iloc[:, -1] # The "species" column
```

Now split the data into training and test data:

```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2, random_state=0)
```

Let's train the model:

```
model = LogisticRegression()
model.fit(x_train, y_train) # Training the model
```

Make predictions:

```
predictions = model.predict(x_test)
print(predictions)
```

```
['virginica' 'versicolor' 'setosa' 'virginica' 'setosa' 'virginica'
 'setosa' 'versicolor' 'versicolor' 'versicolor' 'virginica' 'versicolor'
 'versicolor' 'versicolor' 'versicolor' 'setosa' 'versicolor' 'versicolor'
 'setosa' 'setosa' 'virginica' 'virginica' 'setosa' 'setosa' 'virginica'
 'setosa' 'setosa' 'versicolor' 'versicolor' 'setosa']
```

The predictions of the type of Iris for the test data.

Source: Geoffrey Hunter

Logistic Regression

```
print(classification_report(y_test, predictions))
```

	precision	recall	f1-score	support
setosa	1.00	1.00	1.00	11
versicolor	1.00	1.00	1.00	13
virginica	1.00	1.00	1.00	6
accuracy			1.00	30
macro avg	1.00	1.00	1.00	30
weighted avg	1.00	1.00	1.00	30

The classification report for our logistic regression model.

For someone new to categorization, these terms in the classification report can be confusing. This is what they mean[1]:

- The *precision* is how well the classifier is at labelling an instance positive when it was actually positive. It can be thought of as: "For all instances labelled positive, what percentage of them are actually correct?

$$precision = \frac{true\,positive}{true\,positive + false\,positive}$$

- The *recall* is the ability for a classifier to find all true positives. It can be though of as: "For all instances that where actually positive, what percentage were labelled correctly?"

$$recall = \frac{true\,positive}{true\,positive + false\,negative}$$

- The *f1-score* is the harmonic mean of the precision and recall. Personally I find this the most difficult metric to understand intuitively. It is a score which incorporates both the *precision* and *recall*, and varies between 0 and 1.
- The *support* is the number of actual occurrences of a class in a specific dataset.

And let's print the accuracy score:

```
print(accuracy_score(y_test, predictions))
# 0.9666666666666667
```

Source: Geoffrey Hunter

Group Discussion

What can be applications of logistic regressions?

.

Logistic Regression Applications

- **Credit scoring:** It's difficult if you have more than 15 variables in your model. For logistic regression, it is easy to find out which variables affect the final result of the predictions more and which ones less. It is also possible to find the optimal number of features and eliminate redundant variables with methods like recursive feature elimination
- An **e-commerce** company that mails expensive promotional offers to customers, for example, would like to know whether a particular customer is likely to respond to the offers or not: i.e., whether that consumer will be a "responder" or a "non-responder." In marketing, this is called propensity to respond modeling.
- **Healthcare**: discrimination between type 1 and type 2 diabetes in young adult

More details: https://activewizards.com/blog/5-real-world-examples-of-logistic-regression-application

Logistic Regression Many Features

Techniques to Reduce the Number of Features

1. **PCA**: this creates "new" linear combinations of your data where each preceding component explains as much variance in the data as possible. The disadvantage here is that because the components are combinations of your original variables you lose some interpretability with your regression model. It should however produce very good accuracy.
2. **Lasso Regression** uses an L_1 penalization norm that shrinks the coefficients of features effectively eliminating some of them.You can include this L_1 norm into your logistic regression model. It seems Note: Lasso will not explicitly set variable coefficients to zero, but will shrink them allowing you to select the largest coefficients.

Multi-Class Logistic Regression

Eg. If we have to predict whether the weather is sunny, rainy, or windy, we are dealing with a Multi-class problem. We turn this problem into three binary classification problem i.e whether it is sunny or not, whether it is rainy or not and whether it is windy or not. We run all three classifications **independently** on input. The classification for which the value of probability is maximum relative to others, is the solution.

Assumptions:

The Dependent variable should be either nominal or ordinal variable.

Set of one or more Independent variables can be continuous, ordinal or nominal.

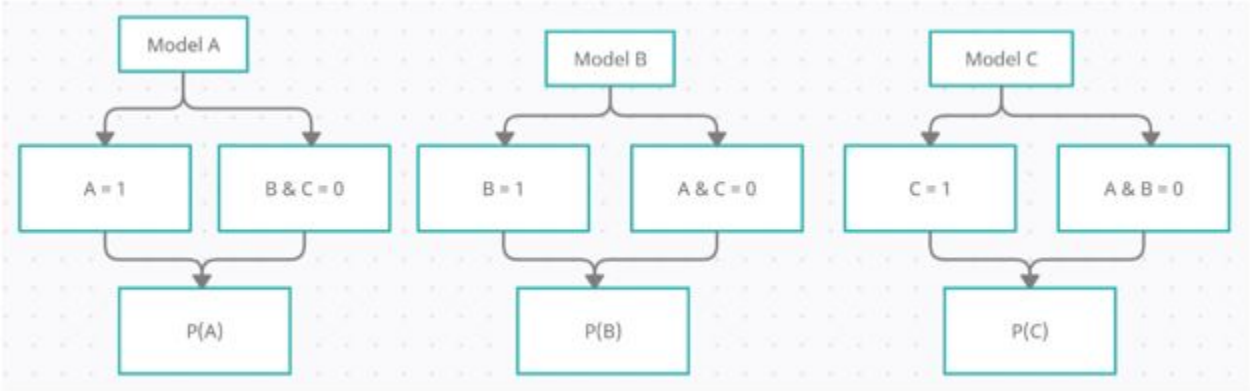
The Observations and dependent variables must be mutually exclusive and exhaustive.

No Multicollinearity between Independent variables.

There should be no Outliers in the data points.

Multi-Class Logistic Regression

K models for K classes



QUESTIONS?