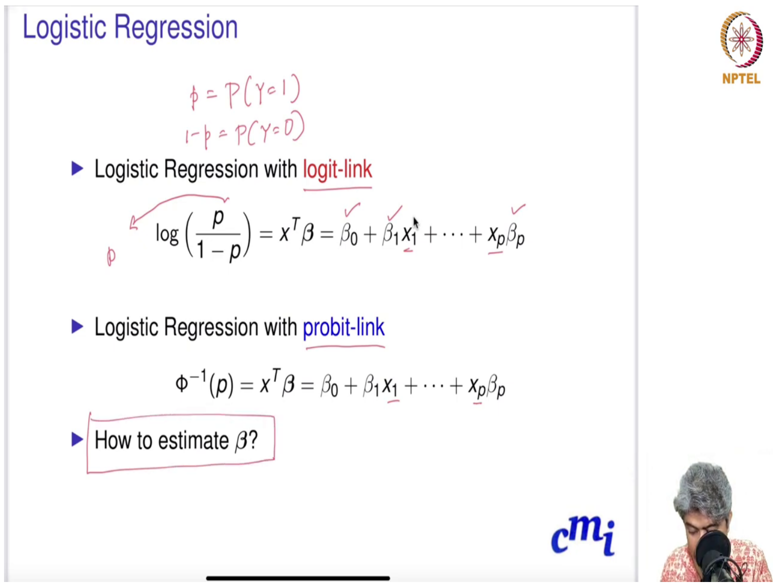
Predictive Analytics - Regression and Classification

Prof. Sourish Das
Department of Mathematics
Chennai Mathematical Institute

Lecture - 37
Maximum Likelihood Estimate for Logistic Regression

(Refer Slide Time: 00:24)



Logistic Regression

$p = P(Y=1)$
$1-p = P(Y=0)$

- Logistic Regression with logit-link

$$\log\left(\frac{p}{1-p}\right) = x^T\beta = \beta_0 + \beta_1 x_1 + \cdots + x_p\beta_p$$

- Logistic Regression with probit-link

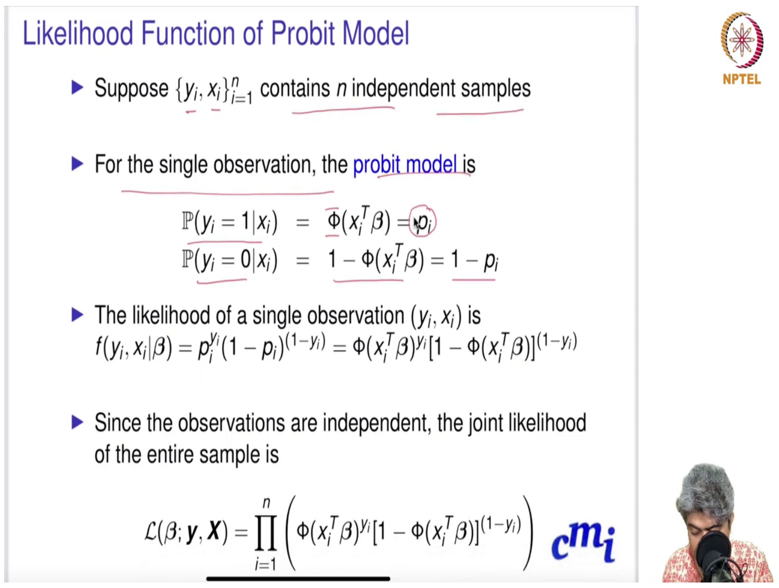
$$\Phi^{-1}(p) = x^T\beta = \beta_0 + \beta_1 x_1 + \cdots + x_p\beta_p$$

- $\boxed{\text{How to estimate } \beta?}$

Hello, all welcome back to the part B of lecture 11. We will continue on logistic regression and in logistic regression we found there are two variant of the Logistic Regression model. One is Logistic Regression with logit-link, another is the Logistic Regression with the probit-link.

And what we found that we know the value of x 1 x 2 x p, we can somehow estimate beta naught, beta 1, beta p, the beta coefficients then given a x 1 x 2 x p we can estimate the value of p. And that is what we have our final objective that we want to estimate probability y equal to 1 or 1 minus p which is probability y equal to 0. This is the probability that we want to estimate. So, question now boils down to how we estimate beta, how we estimate beta?

(Refer Slide Time: 01:25)



Likelihood Function of Probit Model

- Suppose $\{y_i, x_i\}_{i=1}^{n}$ contains n independent samples
- For the single observation, the probit model is

$$\begin{aligned} \mathbb{P}(y_i = 1|x_i) &= \Phi(x_i^T\beta) = p_i \\ \mathbb{P}(y_i = 0|x_i) &= 1 - \Phi(x_i^T\beta) = 1 - p_i \end{aligned}$$

- The likelihood of a single observation (y_i, x_i) is $f(y_i, x_i|\beta) = p_i^{y_i}(1 - p_i)^{(1-y_i)} = \Phi(x_i^T\beta)^{y_i}[1 - \Phi(x_i^T\beta)]^{(1-y_i)}$
- Since the observations are independent, the joint likelihood of the entire sample is

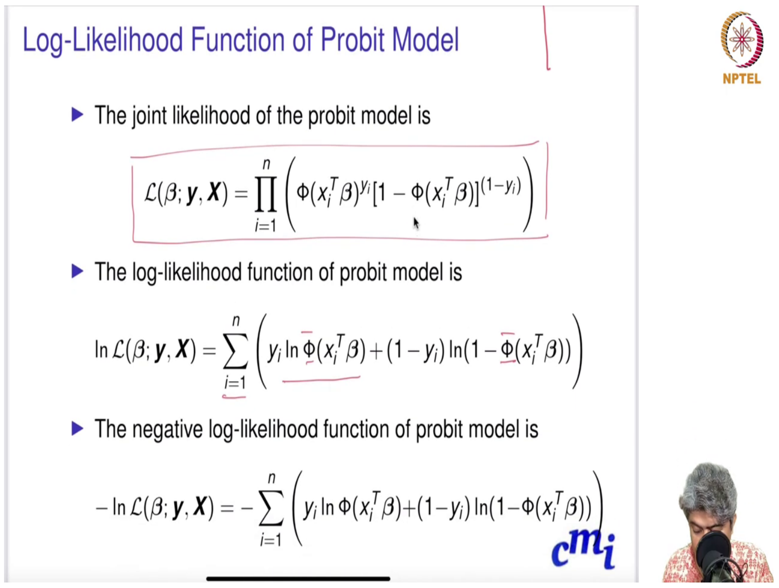
$$\mathcal{L}(\beta; \boldsymbol{y}, \boldsymbol{X}) = \prod_{i=1}^{n}\left(\Phi(x_i^T\beta)^{y_i}[1 - \Phi(x_i^T\beta)]^{(1-y_i)}\right)$$

So, what we do, we try to write the likelihood function of probit model. Suppose y i x i i equal to 1 to n contains n independent samples and for single observation the probit model is probability y i equal to 1 given x i is phi of x i transpose beta which is p i and probability y equal to 0 is 1 minus phi of x i transpose beta which is 1 minus p i.

Now, what is the likelihood of a single observation y i x i? The likelihood of single observation of f y i x i, f y i given there is a mistake this given should be given x i transpose beta should be p i to the power y i and 1 minus p i to the power 1 minus y i. Now, instead of p i you just plug in phi of x i transpose beta and 1 minus phi of x i transpose beta to the power 1 minus y i.

And since all observations are independent the joint likelihood of the entire sample is just product over i equal to 1 to n. So, that is the like this is the likelihood function this is the likelihood function of probit model.

(Refer Slide Time: 03:09)



Log-Likelihood Function of Probit Model

- The joint likelihood of the probit model is

$$\mathcal{L}(\beta; \mathbf{y}, \mathbf{X}) = \prod_{i=1}^{n} \left(\Phi(x_i^T \beta)^{y_i} [1 - \Phi(x_i^T \beta)]^{(1-y_i)} \right)$$

- The log-likelihood function of probit model is

$$\ln \mathcal{L}(\beta; \mathbf{y}, \mathbf{X}) = \sum_{i=1}^{n} \left(y_i \ln \Phi(x_i^T \beta) + (1 - y_i) \ln(1 - \Phi(x_i^T \beta)) \right)$$

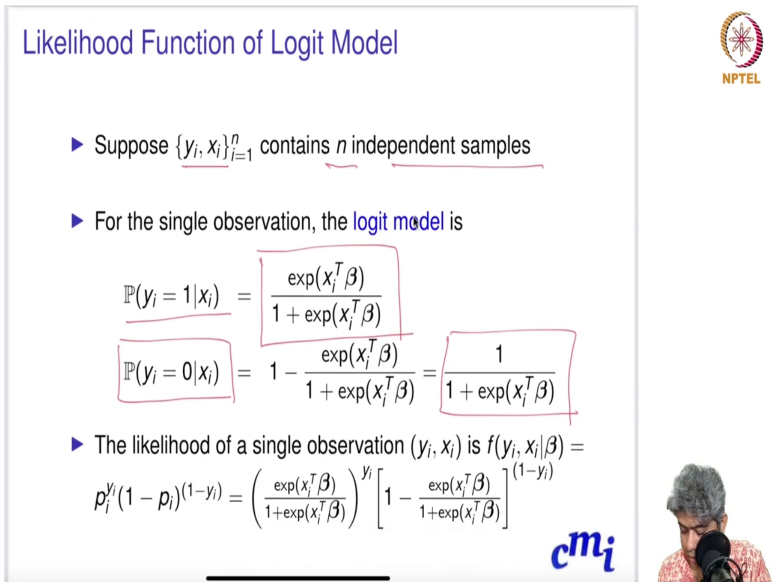
- The negative log-likelihood function of probit model is

$$-\ln \mathcal{L}(\beta; \mathbf{y}, \mathbf{X}) = -\sum_{i=1}^{n} \left(y_i \ln \Phi(x_i^T \beta) + (1 - y_i) \ln(1 - \Phi(x_i^T \beta)) \right)$$

Now, once you have the likelihood function of the probit model you can write the log likelihood the joint likelihood of the probit model is this. Now, you can take the log of the likelihood this will give you summation y i l ln phi of x i transpose beta and 1 minus y i times ln 1 minus phi of x i transpose beta.

Now, once you have that you can also write the negative log likelihood of the probit model which is just you multiply the this one this log likelihood function by minus 1 that will give you the negative log likelihood function.

(Refer Slide Time: 03:57)



Likelihood Function of Logit Model

- Suppose $\{y_i, x_i\}_{i=1}^n$ contains n independent samples
- For the single observation, the logit model is

$$\mathbb{P}(y_i = 1|x_i) = \boxed{\frac{\exp(x_i^T\beta)}{1+\exp(x_i^T\beta)}}$$

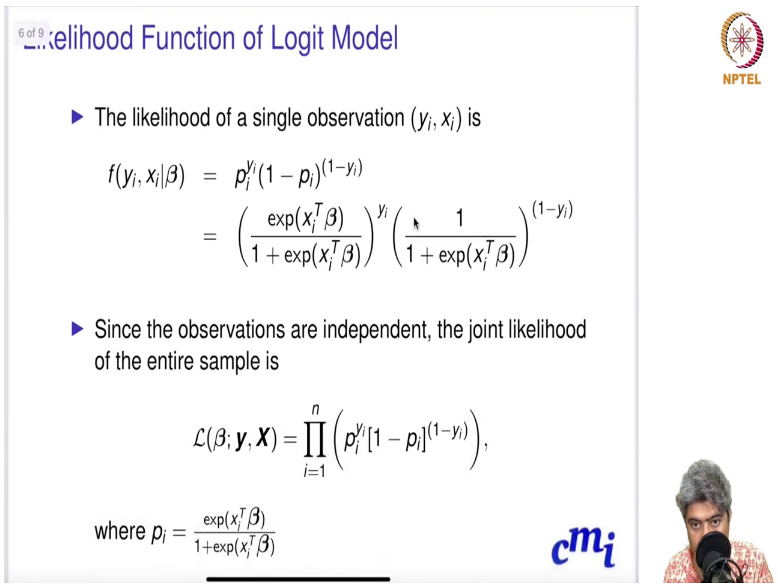
$$\boxed{\mathbb{P}(y_i = 0|x_i)} = 1 - \frac{\exp(x_i^T\beta)}{1+\exp(x_i^T\beta)} = \boxed{\frac{1}{1+\exp(x_i^T\beta)}}$$

- The likelihood of a single observation (y_i, x_i) is $f(y_i, x_i|\beta) =$ $p_i^{y_i}(1-p_i)^{(1-y_i)} = \left(\frac{\exp(x_i^T\beta)}{1+\exp(x_i^T\beta)}\right)^{y_i}\left[1 - \frac{\exp(x_i^T\beta)}{1+\exp(x_i^T\beta)}\right]^{(1-y_i)}$

Similarly, we can define the likelihood function of logit model. Suppose y i x i is equal to 1 and contains the n independent samples. Then for a single observation the logit model is probability y i equal to 1 given the x i is e to the power x i transpose beta divided by 1 plus e to the power x i transpose beta and y i probability y i equal to 0 given x i can be written as 1 over 1 plus e to the power x i transpose beta.

Now, likelihood of single observations can be written as p i to the power y i 1 minus p i to the power 1 minus y i. Now, instead of p i you just plug in this functional form and in place of 1 minus p i you plug in this functional form that will give you the likelihood of the single observations.

(Refer Slide Time: 05:07)



Likelihood Function of Logit Model

- The likelihood of a single observation (y_i, x_i) is

$$f(y_i, x_i|\beta) = p_i^{y_i}(1-p_i)^{(1-y_i)}$$
$$= \left(\frac{\exp(x_i^T\beta)}{1+\exp(x_i^T\beta)}\right)^{y_i}\left(\frac{1}{1+\exp(x_i^T\beta)}\right)^{(1-y_i)}$$

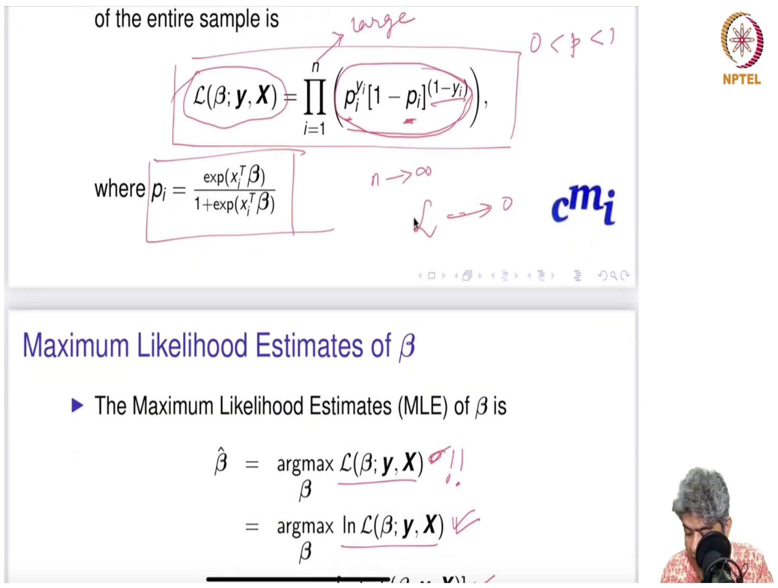
- Since the observations are independent, the joint likelihood of the entire sample is

$$\mathcal{L}(\beta; \boldsymbol{y}, \boldsymbol{X}) = \prod_{i=1}^{n}\left(p_i^{y_i}[1-p_i]^{(1-y_i)}\right),$$

where $p_i = \frac{\exp(x_i^T\beta)}{1+\exp(x_i^T\beta)}$

And then you can define the joint likelihood of as this this product of p i to the power on y i times 1 minus y i p i to the power 1 minus y i and where p i is e to the power x i transpose beta divided by 1 plus e to the power x i transpose beta.

(Refer Slide Time: 05:33)



of the entire sample is

large

$0 < p < 1$

$$\mathcal{L}(\beta; \mathbf{y}, \mathbf{X}) = \prod_{i=1}^{n} \left(p_i^{y_i}[1 - p_i]^{(1-y_i)}\right),$$

where $p_i = \frac{\exp(x_i^T\beta)}{1+\exp(x_i^T\beta)}$

$n \rightarrow \infty$

$\mathcal{L} \rightarrow 0$

Maximum Likelihood Estimates of β

- The Maximum Likelihood Estimates (MLE) of β is

$$\begin{aligned}\hat{\beta} &= \underset{\beta}{\operatorname{argmax}}\ \mathcal{L}(\beta; \mathbf{y}, \mathbf{X}) \\ &= \underset{\beta}{\operatorname{argmax}}\ \ln \mathcal{L}(\beta; \mathbf{y}, \mathbf{X})\end{aligned}$$

So, now once you define the log likelihood function you can estimate the beta through maximum likelihood estimates of beta which is argmax of either likelihood function or log likelihood function or negative log likelihood function. Now, in order to do the in most of the typical optimization sub routine it is a minimization problems.

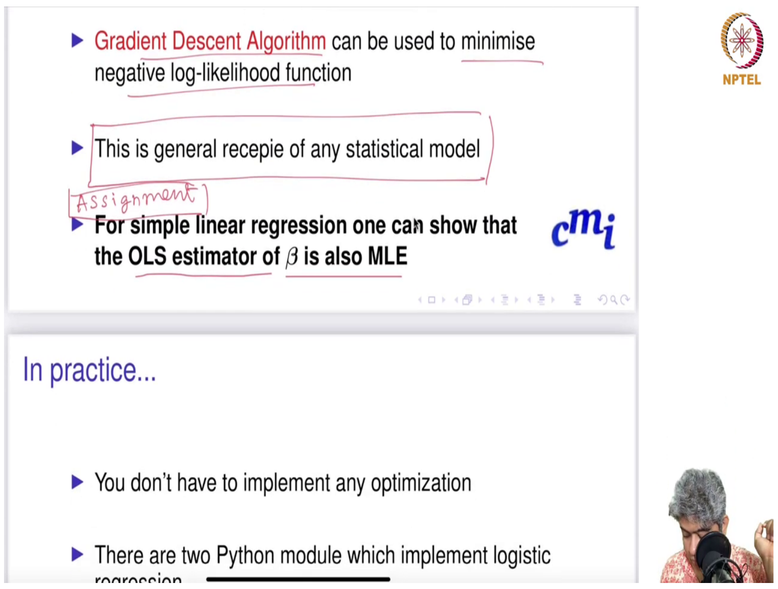
So, if you are solve so, you have to when you are applying a optimization routine you have to be careful whether you are the routine is doing it minimization solving it as a minimization problem or a maximization problem. If it is solving it as a minimization problem then you should try to minimize the negative log likelihood function.

If it is solving at as a maximization problem then you have to solve it as a maximizing log likelihood function. Maximizing directly likelihood is bit of a questionable why? Because you see if you look into it what is p i p i is a probability 1 minus p i is also probability. So, these points are this is going to be a probability which is some probability between 0 and 1 and then you are taking product.

So, for if n is very large then what happens for very large n this likelihood function value of likelihood function will start tend to 0, as n tends to infinity become larger large the value of l becomes smaller and smaller and after a while what will happen it will just your computer will just truncate to 0 and that is the problem.

So, if you want to avoid that. So, instead of working with likelihood function you it is better to work with either log likelihood function or negative log likelihood function.

(Refer Slide Time: 07:49)


- Gradient Descent Algorithm can be used to minimise negative log-likelihood function
- This is general recepie of any statistical model

Assignment
- For simple linear regression one can show that the OLS estimator of β is also MLE

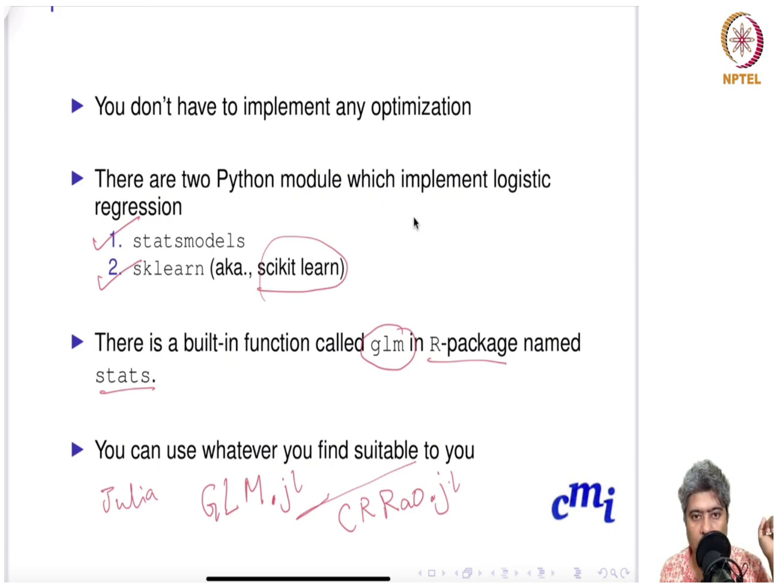
In practice...

- You don't have to implement any optimization
- There are two Python module which implement logistic regression

Gradient descent algorithm can be used to minimize the negative log likelihood function. This is general recipe for any statistical model. Remember that this is a general's recipe for any statistical model. So, once you have the statistical model you write down the likelihood function then you write down either log likelihood or negative log likelihood then pass it to the gradient descent algorithm and try to estimate the beta.

For simple linear regression one can show OLS estimator of beta is MLE. This is a I am leaving it as an assignment I am leaving it as an assignment that for you guys that you know that show that simple linear regression is OLS estimator of beta is the maximum likelihood estimator.

(Refer Slide Time: 08:57)



- You don't have to implement any optimization
- There are two Python module which implement logistic regression
  1. `statsmodels`
  2. `sklearn` (aka., scikit learn)
- There is a built-in function called `glm` in `R`-package named `stats`.
- You can use whatever you find suitable to you

Julia GLM.jl ~~CRRaO.jl~~

So, now in practice you do not have to implement any optimization. There are two Python module which implement the logistic regression one is statsmodels, another is the sklearn also known as scikit learn. There is a built in function called GLM in R-package named stats. So, in R also you do not have to worry in Julia. Also, there is a function called there is a package called GLM dot jl you just need to install this package GLM dot jl or you can install CRRao dot jl.

So, CRRao dot jl also implement logistic regression with both logit and probit link. So, in either you can do it in Julia or in R or in Python all standard packages do implement logistic regression. So, you can use any one of them.

Thank you very much in the next video we will do hands on.