بسم الله الرحمن الرحيم

انهارده هنتكلم عن ال binary logistic regression .. هنتكلم بعدها ال multinomial logistic regression وده مختلف عن ال BLR ... آخر مره قلنا البرنسبل بتاع ال MLE .. ان عندنا داتا من شوية أمثله Xi واننا هنختار البارمترز اللي بت maximize likelihood of data تحت ال iid assumption فده معناه اننا هنختار البارمترز Theta اللي هت argmax ال L(theta) اللي هي ال product of the probability of each of the examples .. ولو عاوز تفكر ف الكلام ده من ناحية الماشين ليرننج برسبيكتيف اللي هو ال classification .. هنا ال Xi هي ال general data examples انما في الماشين ليرننج ال Xi بتبقا pairs of feature vector (Xi, Yi) ... واللي بنعملو لما بنحاول نحل مشكلة ال argmax .. ان عندنا بارمترز و عندنا فانكشن اسمها L(theta) وبنحاول نلاقي الثيتا اللي بت maximize الفانكشن ديه .. الدكتور بيتكلم في السلايد و الرسمتين اللي تحت .. بيقول ان اللي علي الشمال ديه في ال 1D .. اللي علي اليمين في ال 2D .. في ال 2D عندك ثيتا 1 و ثيتا 2 .. والفانكشن بتاعتك في ال D2 .. واحنا بنحاول ن climb to the top of the nearest hill .. ده هو ال setting for finding the maximum likelihood estimate .. الي احنا هنعملو قدام هو اننا هنقول احنا ضيعنا كل الوقت ده في عشان نحط ال setting بتاع ال optimization algorithm .. والي هي اصلا بتعرف ازاي ت walk downhill .. بس هنا انت عاوز ت maximize .. فازاي تقدر تستخدم ألجورذم بيعرف يتحرك ناحية ال nearest valley عشان يحل مشكله شكلها عامل زي ال hill .. المنطقي انك تاخد الفانكشن بتاعتك اللي عاوز ت maximize .. وتقوم مديها اشاره سالب .. يبقا عملت ال flip اللي يخليك تتحرك downhill فانت برضو هنا بقي هدفك انك ت minimize the negative function ..

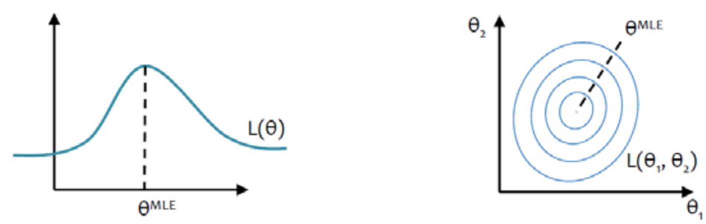
MLE

Suppose we have data $\mathcal{D} = \{x^{(i)}\}_{i=1}^N$

Principle of Maximum Likelihood Estimation:
Choose the parameters that maximize the likelihood of the data.

$$\boldsymbol{\theta}^{\text{MLE}} = \underset{\boldsymbol{\theta}}{\text{argmax}} \prod_{i=1}^{N} p(\mathbf{x}^{(i)}|\boldsymbol{\theta})$$

Maximum Likelihood Estimate (MLE) $(\vec{x}^{(i)}, \vec{y}^{(i)})$



ال MLE بيحاول انو يحجز as much probability mass as possible للحاجات اللي احنا شفناها.. الفكره كلها انك تحاول ت fit the data .. و MLE هيوديك ناحية الأوفر فيتتنج للداتا اللي انت لاحظتها ..

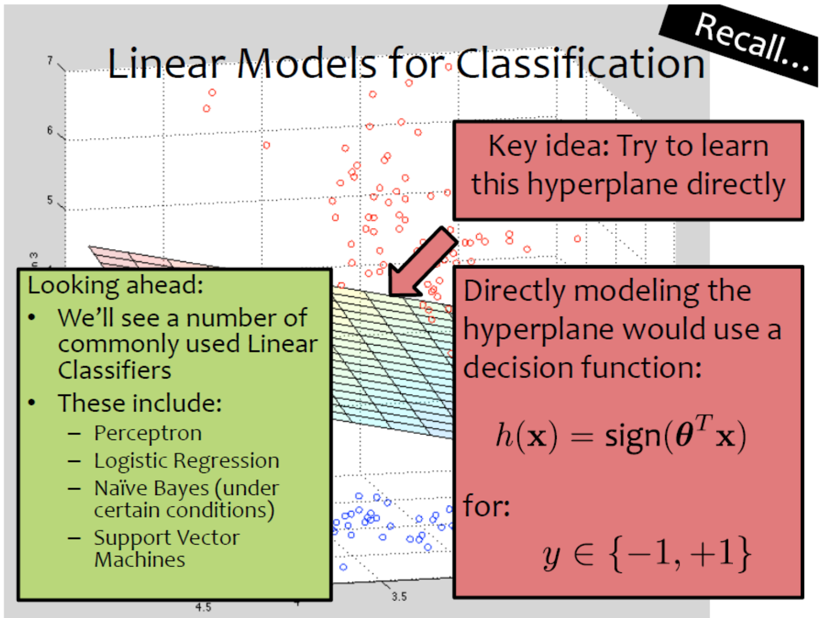
MLE

What does maximizing likelihood accomplish?

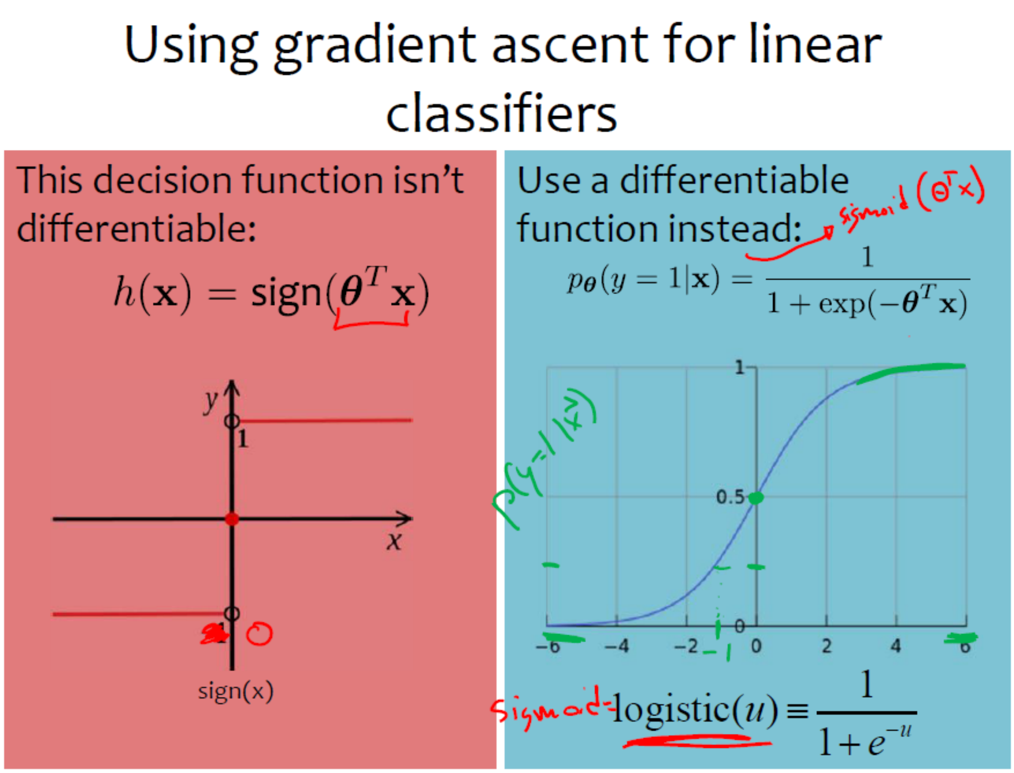
- There is only a finite amount of probability mass (i.e. sum-to-one constraint)
- MLE tries to allocate **as much** probability mass **as possible** to the things we have observed...

  ...**at the expense** of the things we have **not** observed

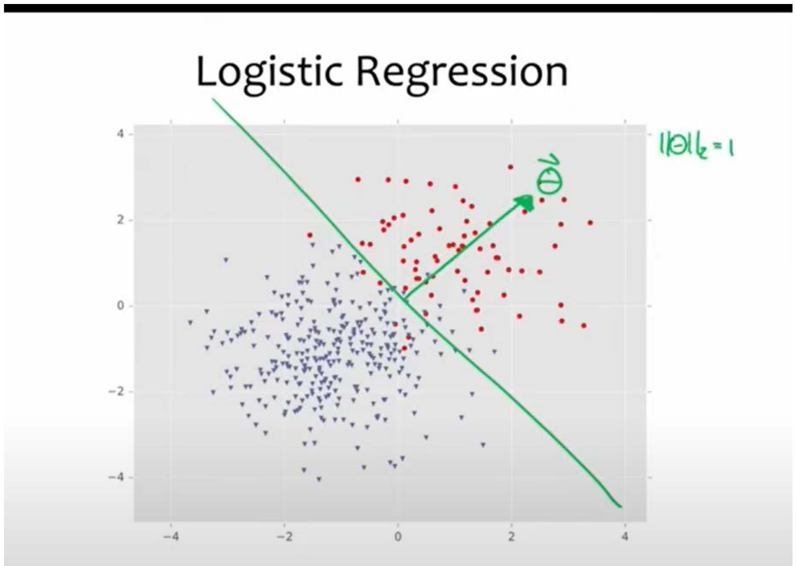
احنا هنستخدم ال MLE عشان نتعلم logistic regression model .. احنا هنفترض ان الأوتبوتس هتبقا باينري .. ف في ال Logistic regression .. خد بالك اننا دلوقت برضو بنتكلم علي classification.. بغض النظرعن الاسم يعني اللي هو regression .. وهنا ال classification هيتعلم linear model زي البرسيبترون كدا ... و علي طول استخدم مثلا حاجه زي ال sign(Theta.T x) .. احنا انهارده هنعرّف ال linear classifier اللي هو ال linear regression .. وبعدين هنعرّف ال objective function .. وديه هتيجي من ال MLE وبعدين optimize using gradient descent عشان تتعلم البارمترز .. وبعدين ت predict the class with highest probability under the model ..



طيب في الطريق ده احنا هيقابلنا مشكله .. ال decision function not differentiable ... بس احنا قلنا اننا هنن ال gradient descent علي الفانكشن ديه .. فهنحتاج نحسب ال partial deriv. .. بس هنا مش هنعرف نشتق .. خلاص هنستخدم a differentiable function .. و هي ديه ال sigmoid .. او اللوجستيك فانكشن ..

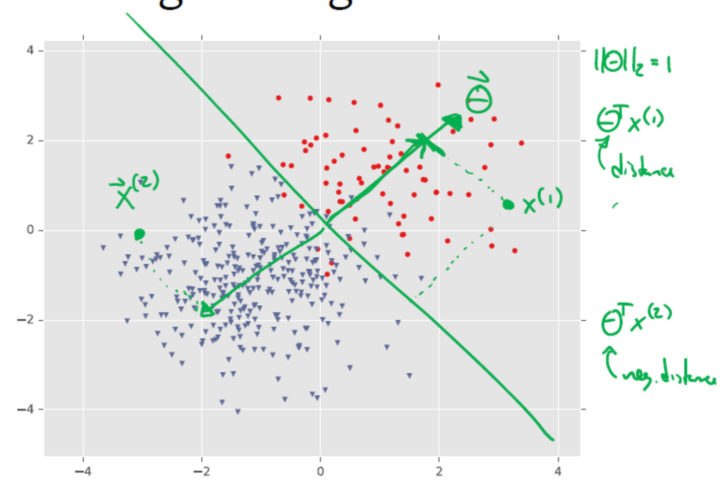


تعال نبص علي الداتا سيت .. وتخيل ان عندنا DB لينير وعندك الفيكتور ثيتا .. في الدايركشن بتاع ال + إكزامبلز .. .



افترض عندك نقطه اسمها X1 .. وانت عاوز تحسب قيمة ال theta.T x(1) .. الدوت برودكت ده بيعبر عن الماجنتيود بتاع البروجكشن ... وده فالدوت برودكت هو المسافه ما بين الإكس 1 للثيتا فيكتور .. طيب تعال نحط نقطه تانيه اسمها X2 .. تعال نوقعها علي الثيتا .. هيديلك فيكتور في الاتجاه التاني .. في الحاله ديه الثيتا ترانسبوز إكس 2 هيبقا في النيجاتيف دايركشن ..

Logistic Regression



Logistic Regression

Data: Inputs are continuous vectors of length M. Outputs are discrete.

$$\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{N} \text{ where } \mathbf{x} \in \mathbb{R}^M \text{ and } y \in \{0,1\}$$

Model: Logistic function applied to dot product of parameters with input vector.

$$p_{\boldsymbol{\theta}}(y=1|\mathbf{x}) = \frac{1}{1+\exp(-\boldsymbol{\theta}^T\mathbf{x})}$$

Learning: finds the parameters that minimize some objective function. $\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}}\, J(\boldsymbol{\theta})$

Prediction: Output is the most probable class.

$$\hat{y} = \underset{y \in \{0,1\}}{\operatorname{argmax}}\, p_{\boldsymbol{\theta}}(y|\mathbf{x})$$

23

تعال نشوف هنعمل ايه في BLR:

1. احنا المره اللي فاتت قلنا مش هنستخدم x .. خلينا بقا نستخدم x المره ديه
2. الموديل : هنعرف الموديل هنا .. ال y يا صفر ياواحد ... فبالتالي احتاملية ان الواي تبقا بواحد أو صفر هيبقا واخد bernolli distr. اللي هو عندك بارمتر إسمو فاي .. يقلك how likely we are to get y = 1 و 1- phi هيبقا هو الاحتمالية ان الواي تبقا بصفر ... احنا المره اللي فاتت احنا اتعلمنا البارمتر فاي علي طول .. المره ديه هنعرف الفاي انها فانكشن في الثيتا و الإكس .. وهنشتغل علي السجمويد في الحاله ديه .. sigmoid(u) = 1 / (1 + exp(-u)) بحيث ان ال u = Theta.T*X .... تعال بقا نحط p(y|x) هيبقا ليه قيمتين .. لو الواي بواحد هتبقا السجمويد .. والواي بصفر هتبقا واحد ماينص السجمويد ... ده كدا الموديل بتاعنا
3. الأوبجكتيف .. –ve conditional log likelihood .. هتكتب ال L(theta) ... هتبقا زي الصوره اللي محطوطه .. خد بالك ان p(y|x,theta) هي بتتقال بروبابلتي أوف واي جيفن الإكس و البارمترز ثيتا .. الدكتور ساعات ممكن يكتبها زي الي باللون الاخضر اللي هو يحط الثيتا في ال subscript ده عشان يعني كإنك بتقول ان الثيتا هي البارمترز بتاعت البروبابلتي دستريبيوشن ..

$\boxed{\text{Binary Logistic Regression}}$

① Assume: Use $\vec{x}$

② Model: $\phi = \sigma(\vec{\theta}^T \vec{x})$ where $\sigma(u) = \frac{1}{1+\exp(-u)}$

$y \sim \text{Bernoulli}(\phi)$

$$p(y|\vec{x}) = \begin{cases} \sigma(\theta^T x) & \text{if } y=1 \\ 1-\sigma(\theta^T x) & \text{if } y=0 \end{cases}$$

③ Objective:

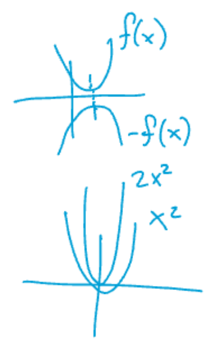
$$\ell(\vec{\theta}) = \sum_{i=1}^{N} \log p(y^{(i)}|x^{(i)},\vec{\theta})$$

$p_{\vec{\theta}}(y^{(i)}|x^{(i)})$

$$J(\theta) = -\frac{1}{N}\ell(\theta) = \frac{1}{N}\sum_{i=1}^{N} \underbrace{-\log p(y^{(i)}|x^{(i)},\theta)}_{J^{(i)}(\theta) \text{ for SGD}}$$

★ negative average conditional log-likelihood for Log. Reg. is convex

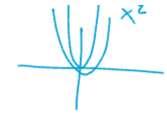
$$\theta_{MLE} = \arg\max_{\theta} \ell(\theta) = \arg\min_{\theta} -\ell(\theta) = \arg\min_{\theta} -\frac{1}{N}\ell(\theta)$$



تعال نكتب الأوبجكتيف فانكشن J(theta) .. هتبقا 1/N- مضروبه في ال L(theta) ...هنشتغل بال –ve average conditional log likelihood ..... وده هيبقا convex function .. الدكتور حط السالب جوا السامشن .. وبعدين سأل ليه .. هنستفاد ايه .. عشان نقدر ن minimize ال J(theta) . بس ليه حطيناه جوا السامشن . ده عشان انت اللي عاوز تعملو انك تعرّف small quantity لكل تريننج إكزامبل اللي نقدر نقللو .. السبب ف حاجه زي كدا .. هو ان عندنا single quantity نقدر نسميها Ji(theta) و بعدين نقدر نستخدم ال Ji(theta) لل SGD ... تعال نفكر في ازاي هنستخدم الفانكشن ديه .. انت هتحاول تلاقي البارمترز الثيتا .. اللي هي ال argmax L(theta) ... لو خدت النيجاتيف بتاع الفانكشن ... فانت بتاخد ال argmin ... المهم تعال ناخد المشتقه ونحسب الجريدينتس

$$J(\theta) = -\frac{1}{N}\ell(\theta) = \frac{1}{N}\sum_{i=1}^{N} \underbrace{-\log p(y^{(i)}|x^{(i)},\theta)}_{J^{(i)}(\theta) \text{ for SGD}}$$

$$\theta_{MLE} = \underset{\theta}{\operatorname{argmax}}\ \ell(\theta) = \underset{\theta}{\operatorname{argmin}}\ -\ell(\theta) = \underset{\theta}{\operatorname{argmin}}\ -\frac{1}{N}\ell(\theta)$$



★ need derivatives for learning

$$\frac{\partial J^{(i)}(\theta)}{\partial \theta_m} = \frac{\partial}{\partial \theta_m}\left(-\log p(y^{(i)}|x^{(i)},\theta)\right)$$

$$= \begin{cases} \frac{\partial}{\partial \theta_m} -\log[\sigma(\theta^T x^{(i)})] & \text{if } y^{(i)} = 1 \\ \frac{\partial}{\partial \theta_m} -\log[1-\sigma(\theta^T x^{(i)})] & \text{if } y^{(i)} = 0 \end{cases}$$

$$\left.\begin{aligned} &= \dots \\ &= \dots \\ &= \dots \end{aligned}\right] \leftarrow \text{recitation}$$

$$= -(\underbrace{y^{(i)}}_{\text{truth}} - \underbrace{\sigma(\theta^T x^{(i)})}_{\text{prob of } y=1})\ \underbrace{x_m^{(i)}}_{m^{th} \text{ feature}}$$

$$\nabla J^{(i)}(\theta) = \begin{bmatrix} \vdots \end{bmatrix} = -(y^{(i)} - \sigma(\theta^T x^{(i)}))\ \vec{x}^{(i)}$$

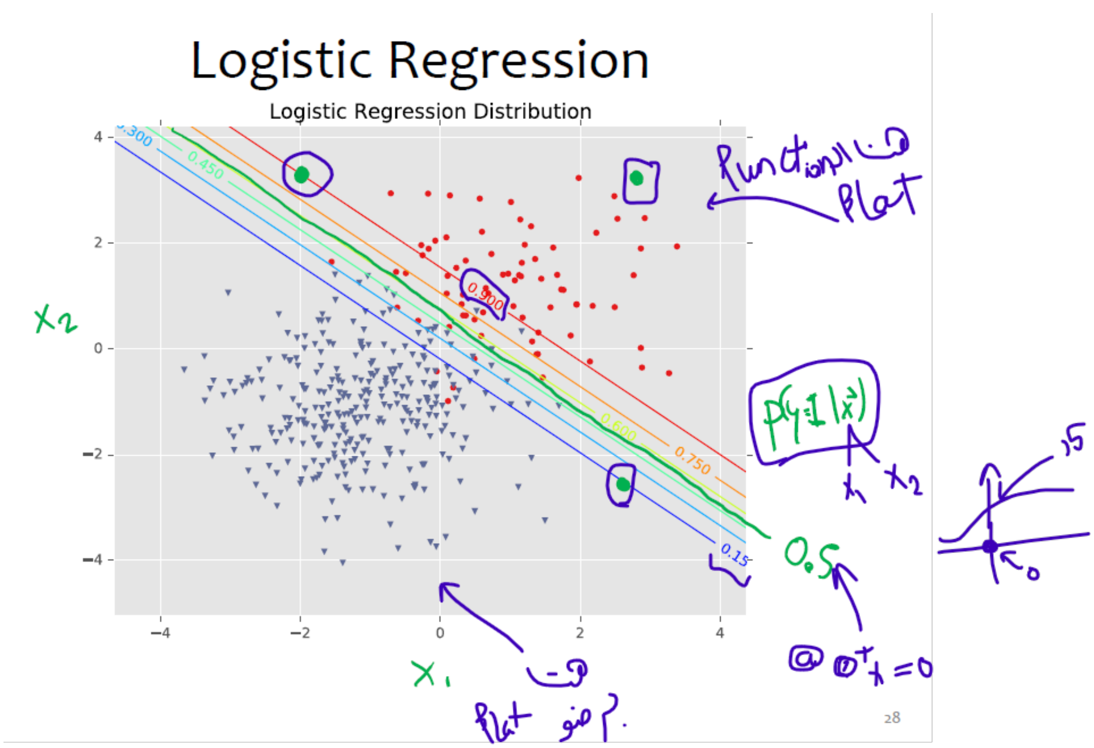
$$\nabla J(\theta) = \begin{bmatrix} \vdots \end{bmatrix} = \frac{1}{N}\sum_{i=1}^{N} \nabla J^{(i)}(\theta)$$

④ Find $\hat{\theta}$ by gradient descent or SGD

⑤ Predict

$$\hat{y} = \underset{y \in \{0,1\}}{\operatorname{argmax}}\ p(y|\vec{x}) = \begin{cases} 1 & \text{if } p(y=1|\vec{x}) \geq 0.5 \\ 0 & \text{otherwise} \end{cases} = \operatorname{sign}(\theta^T x) \in \{0,1\}$$

الدكتور راح يشوف مثال علي اللوجستيك ريجريشن علي الدات سيت ..



فانت عشان تعمل بريدكشن في الموديل ده .. هو ال argmax of y, either 0 or 1 for which it maximizes (p(y|x)) .... ده هنا غير المجاورتي فووت ... القيمه الأعلي هنا بتعتمد علي الإكس .. عشان الفانكشن فاي ... بتعتمد علي الثيتا والإكس .. في طرق تانيه تقدر تكتب بيها الإكسبرشن ده ..

④ Find $\hat{\theta}$ by gradient descent or SGD

⑤ Predict

$$\hat{y} = \underset{y \in \{0,1\}}{\operatorname{argmax}} \; p(y|\vec{x})$$

$$= \begin{cases} 1 & \text{if } p(y=1|\vec{x}) \geq 0.5 \\ 0 & \text{otherwise} \end{cases}$$

$$= \operatorname{sign}(\theta^T x) \in \{0,1\}$$

فده برضو لينير كلاسيفير ..

Maximum **Conditional** Likelihood Estimation

Learning: finds the parameters that minimize some objective function.

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \; J(\boldsymbol{\theta})$$

We minimize the *negative* log conditional likelihood:

$$J(\boldsymbol{\theta}) = -\log \prod_{i=1}^{N} p_{\boldsymbol{\theta}}(y^{(i)}|\mathbf{x}^{(i)})$$

Why?

1. ~~We can't maximize likelihood (as in Naïve Bayes) because we don't have a joint model p(x,y)~~
2. It worked well for Linear Regression (least squares is MCLE)

31

Maximum **Conditional** Likelihood Estimation

Learning: Four approaches to solving $\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmin}} \; J(\boldsymbol{\theta})$

Approach 1: Gradient Descent
(take larger – more certain – steps opposite the gradient)

Approach 2: Stochastic Gradient Descent (SGD)
(take many small steps opposite the gradient)

Approach 3: Newton's Method
(use second derivatives to better follow curvature)

~~**Approach 4:** Closed Form???~~
~~(set derivatives equal to zero and solve for parameters)~~

Logistic Regression does not have a closed form solution for MLE parameters.

33

خد بالك ان مفيش هنا كلوزد فورم سولوشن للوجستيك ريجريشن .. فهنروح لل SGD ..

SGD for Logistic Regression

Question:

Which of the following is a correct description of SGD for Logistic Regression?

Answer:

At each step (i.e. iteration) of SGD for Logistic Regression we…

5 A. ~~(1)~~ compute the gradient of the log-likelihood for all examples (2) update all the parameters using the gradient

calmly B. ~~(1) ask Matt for a description of SGD for Logistic Regression, (2) write it down, (3) report that answer~~

6 C. ~~(1)~~ compute the gradient of the log-likelihood for all examples (2) randomly pick an example (3) update only the parameters for that example

18 D. (1) randomly pick a parameter, (2) compute the partial derivative of the log-likelihood with respect to that parameter, (3) update that parameter for all examples

56 E. (1) randomly pick an example, (2) compute the gradient of the log-likelihood for that example, (3) update all the parameters using that gradient

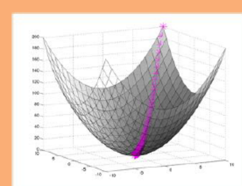
16 F. (1) randomly pick a parameter and an example, (2) compute the gradient of the log-likelihood for that example with respect to that parameter, (3) update that parameter using that gradient

Gradient Descent

Recall…

Algorithm 1 Gradient Descent

1: **procedure** GD($\mathcal{D}$, $\boldsymbol{\theta}^{(0)}$)
2: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$
3: **while** not converged **do**
4: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \gamma \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$
5: **return** $\boldsymbol{\theta}$



In order to apply GD to Logistic Regression all we need is the **gradient** of the objective function (i.e. vector of partial derivatives).

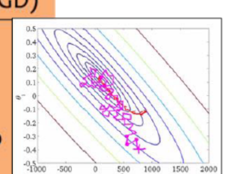
$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \begin{bmatrix} \frac{d}{d\theta_1} J(\boldsymbol{\theta}) \\ \frac{d}{d\theta_2} J(\boldsymbol{\theta}) \\ \vdots \\ \frac{d}{d\theta_M} J(\boldsymbol{\theta}) \end{bmatrix}$$

Stochastic Gradient Descent (SGD)

Recall…

Algorithm 1 Stochastic Gradient Descent (SGD)

1: **procedure** SGD($\mathcal{D}$, $\boldsymbol{\theta}^{(0)}$)
2: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}^{(0)}$
3: **while** not converged **do**
4: **for** $i \in$ shuffle($\{1, 2, \ldots, N\}$) **do**
5: $\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \gamma \nabla_{\boldsymbol{\theta}} J^{(i)}(\boldsymbol{\theta})$
6: **return** $\boldsymbol{\theta}$



We can also apply SGD to solve the MCLE problem for Logistic Regression.

We need a per-example objective:

$$\text{Let } J(\boldsymbol{\theta}) = \sum_{i=1}^{N} J^{(i)}(\boldsymbol{\theta})$$
$$\text{where } J^{(i)}(\boldsymbol{\theta}) = -\log p_{\boldsymbol{\theta}}(y^i|\mathbf{x}^i).$$

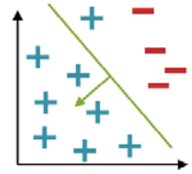
Logistic Regression vs. Perceptron

Question:

True or False: Just like Perceptron, **one step** (i.e. iteration) **of SGD for Logistic Regression** will result in a change to the parameters **only** if the current example is **incorrectly** classified.

Answer:

A=calamity
B=T
C=F 90%



newname
9/29/2021 1:20:57 PM

الاجابه لا .. عشان في الجريدينت في فرق ما بين السجمويد فانكشن ... السجمويد فانكشن مش بتوصل للصفر ولا للواحد .. فدايماً هيبقا في أبديت صغير للبارمترز

يلا نتكلم علي ال mini patch SGD .. في ال gradient descent احنا حسبنا ال gradient من كل الأمثله .. بالنسبه لل SGD احنا قرّبنا ال true gradient بالنسبه لل gradient of one randomly chosen example .. في ال mini-patch SGD احنا بنقرّب ال true gradient عن طريق اننا ناخد الأفريدج بتاع ال k randomly chosen examples ..

Mini-Batch SGD

- **Gradient Descent:**
  Compute true gradient exactly from all N examples
- **Stochastic Gradient Descent (SGD):**
  Approximate true gradient by the gradient of one randomly chosen example
- **Mini-Batch SGD:**
  Approximate true gradient by the average gradient of ~~K~~ randomly chosen examples
  S

لكل الألجورذمز احنا عندنا نفس الفورمات .. أبديت الثيتا بالقيمه بتاعتهاماينص الجاما في ال g .. في الجريدينت ديسسنت .. كانت ال g هي الفوولل جريدينت .. بالنسبه لل mini-batch SGD .. عندنا الأفريدج بتاع شوية أمثله Ji و ال i ديه هي ارقام 1و 2 و 3و 4 .. لحد N . وديه راندوملي تشوزن إكزامبلز .. وبعدين تاخد الداتا ديه و تحسب منهم ال gradient ..

Mini-Batch SGD

$$\boxed{\textbf{while not converged: } \boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \cancel{\lambda}\mathbf{g}}$$

Three variants of first-order optimization:

Gradient Descent: $\mathbf{g} = \nabla J(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} \nabla J^{(i)}(\boldsymbol{\theta})$

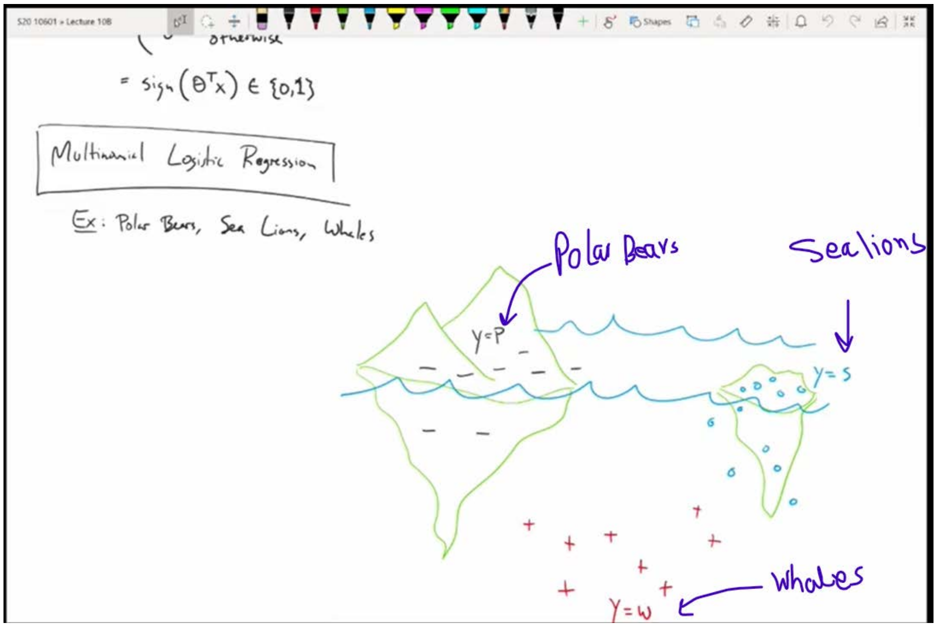
SGD: $\mathbf{g} = \nabla J^{(i)}(\boldsymbol{\theta})$ where i sampled uniformly

Mini-batch SGD: $\mathbf{g} = \frac{1}{S}\sum_{s=1}^{S} \nabla J^{(i_s)}(\boldsymbol{\theta})$ where i_s sampled uniformly $\forall s$

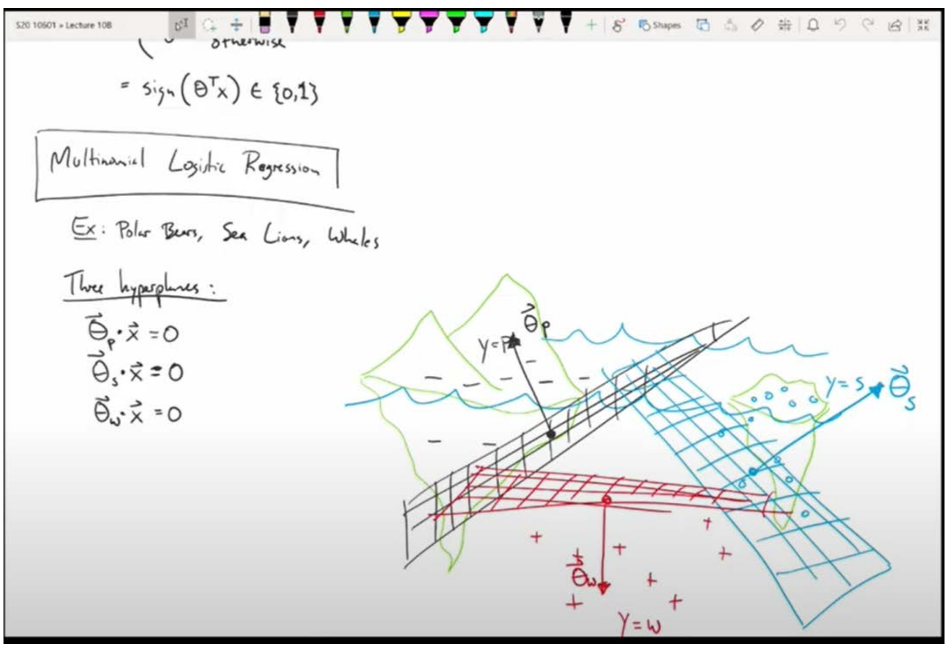
#samples → N

دلوقت هنعرف موديل جديد اسمو Multinomial logistic regression .. وعشان نعمل كدا ... تعال نوجه الشكر لعم dave و عم kevin عشان حرفياً لولا مجهودهم لا كان زمان المحاضرات اتسجلت ولا كان زماني قعدت اذاكر المنهج بتاعهم .. ربنا يباركلهم ويوفقهم ..

الدكتور بيقول بقا لو عندنا دروون و عاوزين نعمل classification ل 3 حاجات .. Polar bears, sea lions, whales ..



الدكتور حط هايبر بلينز بتفصل كل واحد عن الآخرين .. فانت عندك 3 هايبر بلينز ..

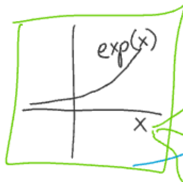


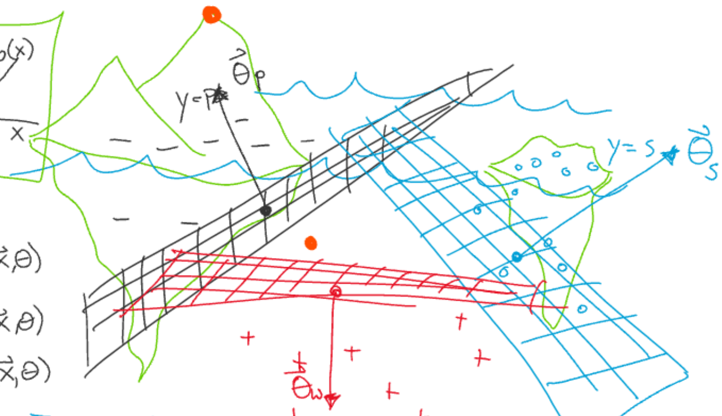
Multinomial Logistic Regression

Ex: Polar Bears, Sea Lions, Whales

Three hyperplanes:

$$\vec{\Theta}_p \cdot \vec{x} = 0$$
$$\vec{\Theta}_s \cdot \vec{x} = 0$$
$$\vec{\Theta}_w \cdot \vec{x} = 0$$





Y=w

$$p(y=P|\vec{x}) = \exp(\vec{\Theta}_p \cdot \vec{x}) / Z(\vec{x},\Theta)$$
$$p(y=S|\vec{x}) = \exp(\vec{\Theta}_s \cdot \vec{x}) / Z(\vec{x},\Theta)$$
$$p(y=W|\vec{x}) = \exp(\vec{\Theta}_w \cdot \vec{x}) / Z(\vec{x},\Theta)$$

$$Z(\vec{x},\Theta) = \exp(\Theta_p \cdot x) + \exp(\Theta_s \cdot x) + \exp(\Theta_w \cdot x)$$

Ex: Normalize These

$p(x=1) \propto 1 \quad = 1/5$

$p(x=2) \propto 3 \quad = 3/5$

$p(x=3) \propto \underline{1} \quad = 1/5$

$\quad\quad\quad\quad\quad 5$

Def: Multiclass Classification $\vec{x} \in \mathbb{R}^M$, $y \in \{1,2,\ldots,K\}$

② Model:

$$p(y|\vec{x}) = \frac{\exp(\vec{\Theta}_y \cdot \vec{x})}{\underbrace{\sum_{k \in \mathcal{Y}} \exp(\vec{\Theta}_k \cdot \vec{x})}_{Z(\vec{x},\Theta) \text{ normalization constant}}}$$

where $\Theta = K \times M$ matrix $= \begin{array}{r} \text{polar} \\ \text{sea} \\ \\ \text{whale} \end{array} \begin{bmatrix} \text{—} \vec{\Theta}_1 \text{—} \\ \text{—} \vec{\Theta}_2 \text{—} \\ \vdots \\ \text{—} \vec{\Theta}_K \text{—} \end{bmatrix}$

$$\phi = [\phi_1, \phi_2, \ldots, \phi_K]^T$$
$$\phi_k = p(y=k|\vec{x})$$
$$y \sim \text{Categorical}(\vec{\phi})$$

same as Multinomial with 1 sample] weighted die roll

<u>Def</u>: <u>Multiclass Classification</u> $\vec{x} \in \mathbb{R}^M$, $y \in \{1, 2, \dots, K\}$

② <u>Model</u>: $p(y|\vec{x}) = \dfrac{\exp(\vec{\theta}_y \cdot \vec{x})}{\underbrace{\sum_{k \in \mathcal{Y}} \exp(\vec{\theta}_k \cdot \vec{x})}_{Z(\vec{x},\theta) \text{ normalization constant}}}$

where $\Theta = K \times M$ matrix $= \begin{array}{l} \\ \text{polar} \\ \text{sea} \\ \\ \text{whale} \end{array} \begin{bmatrix} - \vec{\theta}_1 - \\ - \vec{\theta}_2 - \\ \vdots \\ - \vec{\theta}_K - \end{bmatrix}$

$\phi = [\phi_1, \phi_2, \dots, \phi_K]^T$

$\phi_k = p(y = k|\vec{x})$

$y \sim \text{Categorical}(\vec{\phi})$ ← same as Multinomial with 1 sample] weighted die roll

③ Learning by MLE

<u>Conditional Log-Likelihood</u>:

$$\ell(\theta) = \log\Big[\prod_{i=1}^{N} p(y^{(i)}|x^{(i)}, \theta)\Big] = \sum_{i=1}^{N} \log p(\cdots)$$

$$J(\theta) = -\frac{1}{N}\ell(\theta) \quad \longleftarrow \text{☆ convex ☆}$$

$\hat{\theta} = \underset{\theta}{\operatorname{argmin}}\, J(\theta)$ use GD or SGD

<u>Gradient</u>:

$$\frac{\partial J^{(i)}(\theta)}{\partial \theta_{km}} = \frac{\partial}{\partial \theta_{km}}\Big(-\log p(y^{(i)}|x^{(i)}, \theta)\Big)$$

$$= \cdots \quad \longleftarrow \text{HW4}$$

$\begin{cases} 1 & \text{if } y^{(i)} = k \\ 0 & \text{otherwise} \end{cases}$

$$= -\Big(\mathbb{1}(y^{(i)} = k) - p(y^{(i)}|x^{(i)}, \theta)\Big) x_m^{(i)}$$

$$\frac{\partial J^{(i)}(\theta)}{\partial \vec{\theta}_k} = \nabla_{\vec{\theta}_k} J^{(i)}(\theta) = -\big(\mathbb{1}(y^{(i)} = k) - p(y^{(i)}|x^{(i)}, \theta)\big)\, \vec{x}^{(i)}$$

$$= \begin{bmatrix} \partial \cdot / \partial \theta_{k1} \\ \partial \cdot / \partial \theta_{k2} \\ \vdots \\ \partial \cdot / \partial \theta_{kM} \end{bmatrix}$$ compute for each k

④ SGD for learning/opt.

⑤ predict the most probable class.