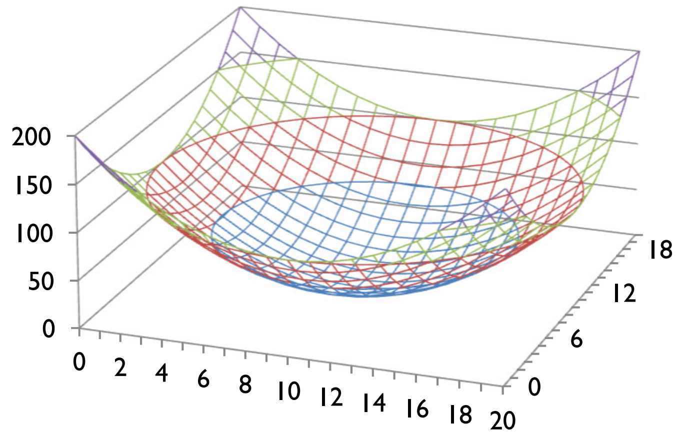
Regression and Logistic Regression

Regression

- Linear regression
  - Objective : predict real values
  - Training set
    - $(\boldsymbol{x}^1, y^1), \dots, (\boldsymbol{x}^N, y^N)$
    - $\boldsymbol{x} \in R^n, y \in R$: single output regression
  - Linear model
    - $F(x) = \boldsymbol{w}.\boldsymbol{x} = \sum_{i=0}^{n} w_i x_i$ with $x_0 = 1$
  - Loss function
    - Mean square error
      - $C = \frac{1}{2}\sum_{i=1}^{N}\left(y^i - \boldsymbol{w}.\boldsymbol{x}^i\right)^2$
  - Steepest descent gradient (batch)
    - $\boldsymbol{w} = \boldsymbol{w}(t) - \epsilon \nabla_w C \text{ , } \nabla_w C = \left(\frac{\partial C}{\partial w_1}, \dots, \frac{\partial C}{\partial w_n}\right)^T$
    - $\frac{\partial C}{\partial w_k} = \frac{1}{2}\sum_{i=1}^{N}\frac{\partial}{\partial w_k}\left(y^i - \boldsymbol{w}.\boldsymbol{x}^i\right)^2 = -\sum_{i=1}^{N}\left(y^i - \boldsymbol{w}.\boldsymbol{x}^i\right)x_{\boldsymbol{k}}^i$ for component w_k
    - $\boldsymbol{w} = \boldsymbol{w}(t) + \epsilon\sum_{i=1}^{N}\left(y^i - \boldsymbol{w}.\boldsymbol{x}^i\right)\boldsymbol{x}^i$ in vector form

Regression

- Geometry of mean squares



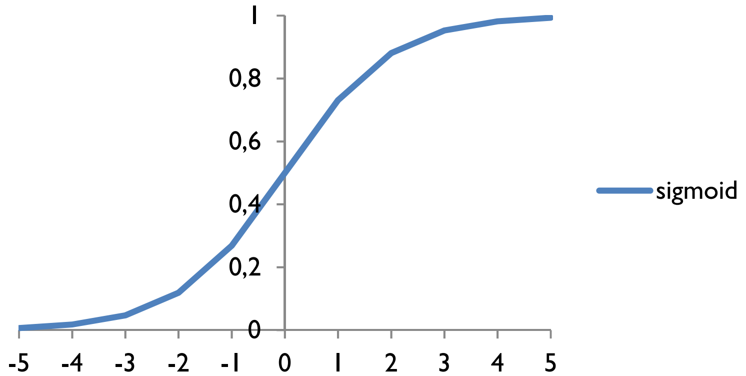
- Regression with multiple outputs $\mathbf{y} \in R^p$
  - Simple extension: p independent linear regressions

Probabilistic Interpretation

- Statistical model of linear regression
  - $y = \boldsymbol{w}.\boldsymbol{x} + \epsilon$, where ϵ is a random variable (error term)
  - Hypothesis ϵ is i.i.d. Gaussian
    - $\epsilon \sim N(0, \sigma^2)$, $p(\epsilon) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{\epsilon^2}{2\sigma^2})$
    - The posterior distribution of y is then
    - $p(y \mid \boldsymbol{x}; \boldsymbol{w}) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(y-\boldsymbol{w}.\boldsymbol{x})^2}{2\sigma^2})$
  - Likelihood
    - $L(w) = \prod_{i=1}^{N} p(y^i \mid \boldsymbol{x}^i; \boldsymbol{w})$
      - Likelihood is a function of $\boldsymbol{w}$, it is computed on the training set
  - Maximum likelihood principle
    - Choose the parameters $\boldsymbol{w}$ maximizing $L(\boldsymbol{w})$ or any incresing function of $L(\boldsymbol{w})$
  - In practice, one optimizes the log likelihood $l(\boldsymbol{w}) = logL(\boldsymbol{w})$
    - $l(\boldsymbol{w}) = Nlog\left(\frac{1}{\sqrt{2\pi}\sigma}\right) - \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(y^i - \boldsymbol{w}.\boldsymbol{x}^i\right)^2$
    - This is the MSE criterion
- This provides a probabilistic interpretation of regression
  - Under a gaussian hypothesis max likelihood is equivalent to MSE minimization

Logistic regression – 2 classes

- Linear regression can be used (in practice) for regression or classfication
- For classification a proper model is logistic regression
  - $F_w(\boldsymbol{x}) = \sigma(\boldsymbol{w}.\boldsymbol{x}) = \frac{1}{1+\exp(-\boldsymbol{w}.\boldsymbol{x})}$
  - Logistic (or sigmoid) function
    - $\sigma(z) = \frac{1}{1+\exp(-z)}$
      - hint
        - $\sigma'(z) = \sigma(z)(1-\sigma(z))$



  - Hyp: $y \in \{0,1\}$

Logistic regression – 2 classes
Probabilistic interpretation

- Since $y \in \{0,1\}$, we make a Bernoulli hypothesis for the posterior distribution
  - $p(y=1|\boldsymbol{x};\boldsymbol{w}) = F_w(\boldsymbol{x})$ et $p(y=0|\boldsymbol{x};\boldsymbol{w}) = 1 - F_w(\boldsymbol{x})$
  - In compact format
    - $p(y|\boldsymbol{x};\boldsymbol{w}) = \left(F_w(\boldsymbol{x})\right)^y \left(1 - F_w(\boldsymbol{x})\right)^{1-y}$ with $y \in \{0,1\}$
- Likelihood
  - $L(\boldsymbol{w}) = \prod_{i=1}^{N} \left(F_w(\boldsymbol{x}^i)\right)^{y^i} \left(1 - F_w(\boldsymbol{x}^i)\right)^{1-y^i}$
- Log-likelihood
  - $l(\boldsymbol{w}) = \sum_{i=1}^{N} y^i log F_w(\boldsymbol{x}^i) + (1 - y^i)(\log(1 - F_w(\boldsymbol{x}^i))$
    - This is minus the cross-entropy between the target and the estimated posterior distribution
  - Steepest descent algorithm (batch) for minimizing cross entropy
    - Componentwise: $\frac{\partial l(\boldsymbol{w})}{\partial w_k} = \sum_{i=1}^{N} \left(y^i - F_w(\boldsymbol{x}^i)\right) \boldsymbol{x}_k^i$
    - **Vector** form: $\nabla_w l = \sum_{i=1}^{N} \left(y^i - F_w(\boldsymbol{x}^i)\right) \boldsymbol{x}^i$
    - Algorithm
      - $\boldsymbol{w} = \boldsymbol{w} - \epsilon \nabla_w C = \boldsymbol{w} + \epsilon \sum_{i=1}^{N} \left(y^i - F_w(\boldsymbol{x}^i)\right) \boldsymbol{x}^i$

Multivariate logistic regression

- Consider a p class classification problem
- Classes are encoded by "one hot" indicator vectors. Each vector is of dimension p
  - Class 1: $\mathbf{y} = (1,0, \dots, 0)^T$
  - Class 2 : $\mathbf{y} = (0,1, \dots, 0)^T$
  - …
  - Class p: $\mathbf{y} = (0,0, \dots, 1)^T$
- $F_W(x)$ is a vector valued function with values in R^p
  - Its component i is a **softmax function** (generalizes the sigmoid)
    - $\hat{\boldsymbol{y}}_i = F_W(\boldsymbol{x})_i = \frac{\exp(\boldsymbol{w}_i.\boldsymbol{x})}{\sum_{j=1}^{p} \exp(\boldsymbol{w}_j.\boldsymbol{x})}$
      - Note : here $\boldsymbol{w}_j \in R^n$ is a vector, $\hat{\boldsymbol{y}}_i \in R$ is the i^{th} component of $\hat{\boldsymbol{y}}$
- The probabilistic model for the posterior is a multinomial distribution
  - $p(Class = i|\boldsymbol{x}; \boldsymbol{w}) = \frac{\exp(\boldsymbol{w}_i.\boldsymbol{x})}{\sum_{j=1}^{p} \exp(\boldsymbol{w}_j.\boldsymbol{x})} = softmax(\boldsymbol{w}_i.\boldsymbol{x})$

Multivariate logistic regression

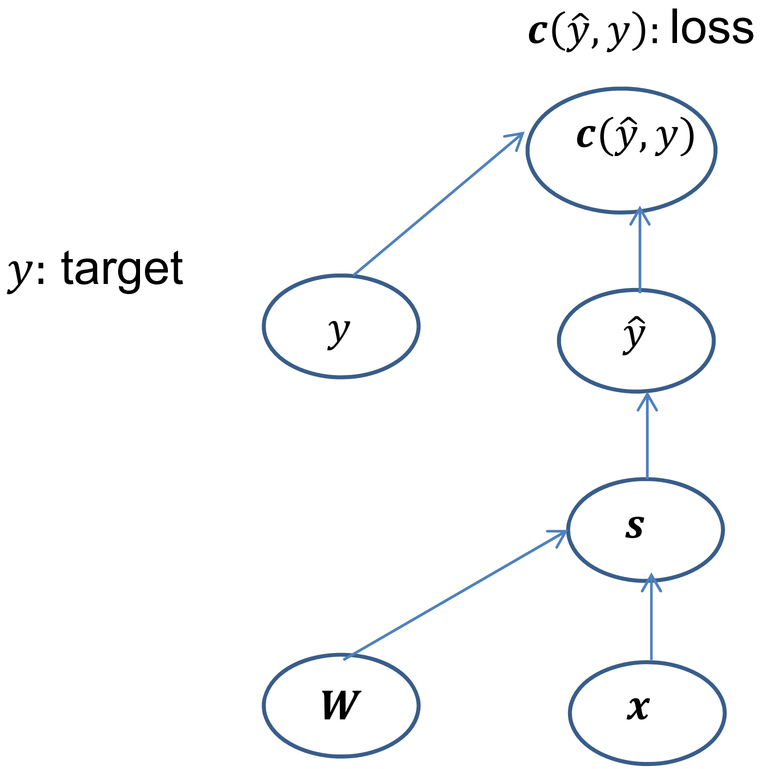
- Notations
  - $\boldsymbol{s^i} = W\boldsymbol{x}^i$ is the logit for input $\boldsymbol{x}^i$
    - $W = (\boldsymbol{w}_1, \dots, \boldsymbol{w}_p)^T$ is a pxn matrix of weights
    - $\boldsymbol{s^i} = (s_1^i, \dots, s_p^i)^T \in R^p$
  - $\widehat{\boldsymbol{y}}^i = softmax(\boldsymbol{s^i})$ is the output for input $\boldsymbol{x}^i$ (here σ applies component-wise, i.e. $\widehat{\boldsymbol{y}}_j^i = softmax(s_j^i)$)
    - $\widehat{\boldsymbol{y}}^i = (\hat{y}_1^i, \dots, \hat{y}_p^i)^T \in R^p$
- Let $\hat{y}$ be a computed output for input x (we drop the index i for simplicity), then
  - $\frac{\partial \hat{y}_j}{\partial s_i} = \hat{y}_j(I_{ji} - \hat{y}_i)$ with I_{ji} elements of the identity matrix (1)
- Likelihood
  - $L(W) = p(Y|X;W) = \prod_{i=1}^{N} \prod_{j=1}^{p} (\hat{y}_j^i)^{y_j^i}$, X and Y are the column wise matrices of input and output vector
- Log likelihood
  - $l(W) = \sum_{i=1}^{N} \sum_{j=1}^{p} y_j^i \ln \hat{y}_j^i$ again this is minus the cross entropy for the multiclass classification problem
- Gradient of the log likelihood
  - $\nabla_{w_k} l(W) = -\sum_{i=1}^{N} (\hat{y}_k^i - y_k^i)\boldsymbol{x}^{\mathrm{i}}$ by using identity (1)
- Training algorithm
  - As before, one may use a gradient method for maximizing the log likelihood.
  - When the number of classes is large, computing the soft max is prohibitive, alternatives are required

Probabilistic interpretation for non linear models

- These results extend to non linear models, e.g. when $F_w(x)$ is a NN
- Non linear regression
  - Max likelihood is equivalent to MSE loss optimization under the Gaussian hypothesis
    - For multivariate $(y \in R, x \in R^n)$ non linear regression we have
    - $y = F_w(\boldsymbol{x}) + \epsilon, \epsilon \sim N(0, \sigma^2)$
    - $p(y\,|\boldsymbol{x}; \boldsymbol{w}) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(y-\boldsymbol{F}(\boldsymbol{x}))^2}{2\sigma^2})$
  - $\log - \text{likelihood } l(w)$
    - $l(\boldsymbol{w}) = N log\left(\frac{1}{\sqrt{2\pi}\sigma}\right) - \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(y^i - \boldsymbol{F}(\boldsymbol{x}^i)\right)^2$
- Classification
  - Max likelihood is equivalent to cross entropy maximization under Bernoulli/ multinomial distribution
    - 2 classes: if y is binary and we make the hypothesis that it is conditionnally Bernoulli with probability $F(x) = p(y = 1|\boldsymbol{x})$ we get the cross entropy loss
    - More than 2 classes: same as logistic regression with multiple outputs

Logistic regression – Computational graph -SGD

- Forward pass



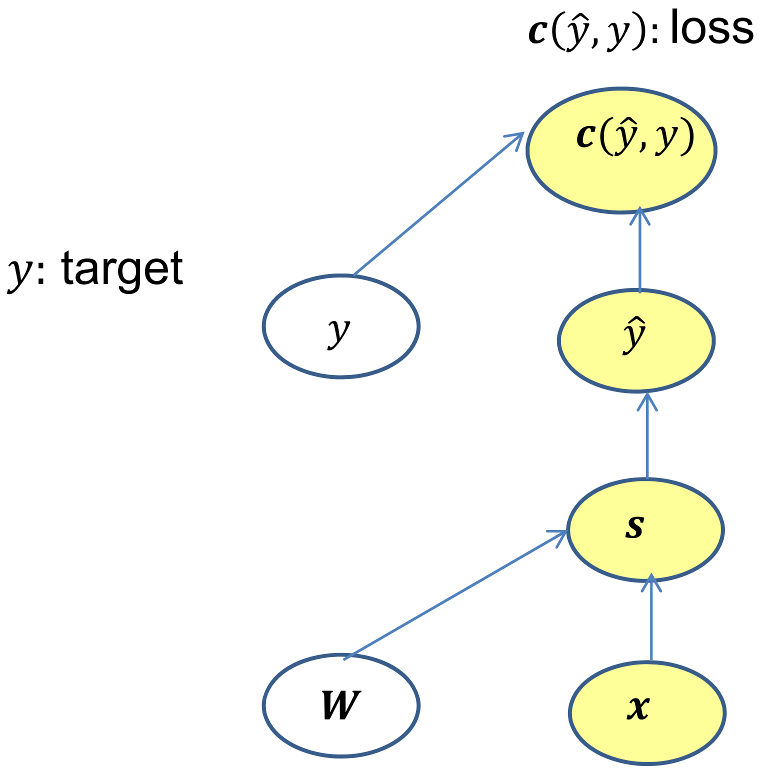
Forward propagation:

$s = \boldsymbol{w}.\boldsymbol{x}$

$\hat{y} = \sigma(s)$

Logistic regression – Computational graph - SGD

- Forward pass



Forward propagation:

$s = \boldsymbol{w}.\boldsymbol{x}$

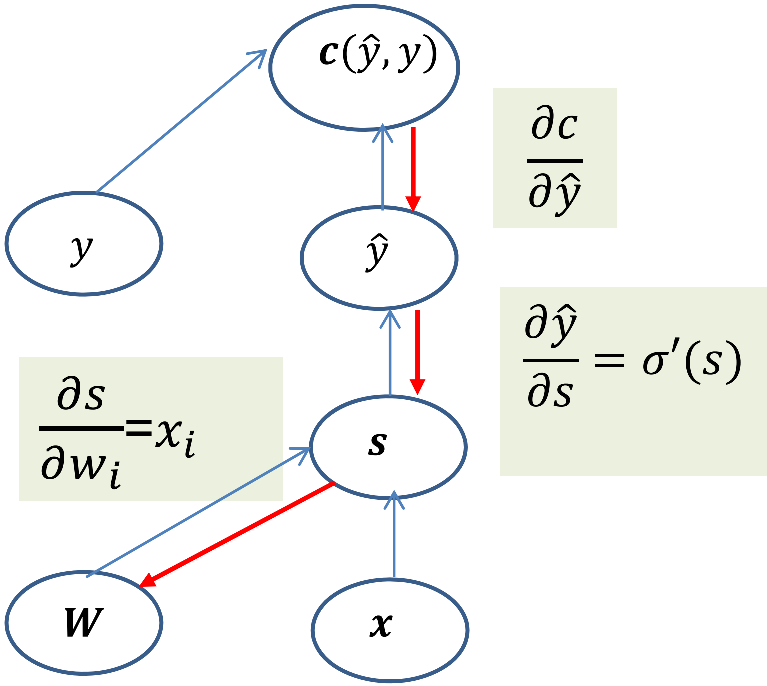
$\hat{y} = \sigma(s)$

Logistic regression – Computational graph - SGD

- Backward pass

$\boldsymbol{c}(\hat{y}, y)$: loss

y: target



Backward propagation:

$$\frac{\partial c}{\partial s} = \frac{\partial c}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial s}$$

$$\frac{\partial c}{\partial w_i} = \frac{\partial c}{\partial s} \frac{\partial s}{\partial w_i}$$

$$\frac{\partial c}{\partial w_i} = \frac{\partial c}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial s} \frac{\partial s}{\partial w_i}$$

Chain Rule

Logistic regression – Computational graph - SGD

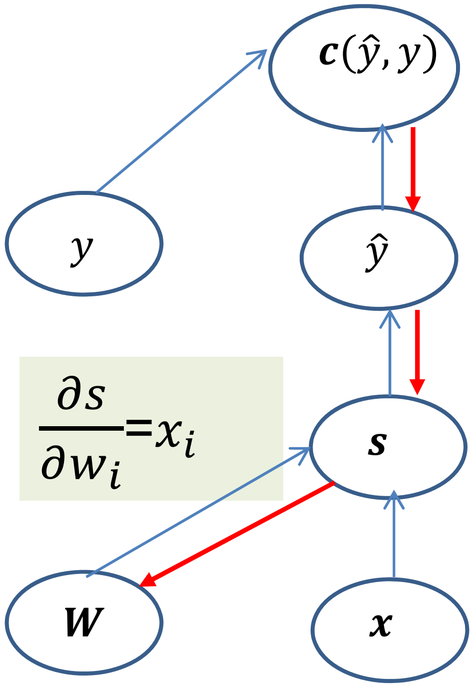
- Backward pass

For the cross entropy loss

$$l(\boldsymbol{w}) = \sum_{i=1}^{N} y^i log \hat{y}^i + (1 - y^i)\log(1 - \hat{y}^i) = \sum_{i=1}^{N} \boldsymbol{c}(\hat{y}^i, y^i)$$

$\boldsymbol{c}(\hat{y}, y)$: loss

y: target



$$\frac{\partial c}{\partial \hat{y}} = \frac{\mathrm{y}}{\hat{y}} - \frac{1 - \mathrm{y}}{1 - \hat{y}}$$

$$\frac{\partial \hat{y}}{\partial s} = \sigma'(s)$$

$$\frac{\partial s}{\partial w_i} = x_i$$

Backward propagation:

$$\frac{\partial c}{\partial s} = \frac{\partial c}{\partial \hat{y}} \sigma'(s)$$

$$\frac{\partial c}{\partial w_i} = \frac{\partial c}{\partial s} x_i$$

$$\frac{\partial c}{\partial w_i} = \left(\frac{\mathrm{y}}{\hat{y}} - \frac{1 - \mathrm{y}}{1 - \hat{y}}\right) \sigma'(s) x_i$$

Probabilistic interpretation of NN outputs
Mean Square loss

- Derived here for multivariate regression (1 output), trivial extension to multiple outputs
- Holds for any continuous functional (regression, logistic regression, NNs, etc)
- Risk $R = E_{x,y}\left[\left(y - h(\boldsymbol{x})\right)^2\right]$
- The minimum of R, $\mathrm{Min}_h R$, is obtained for $h^*(\boldsymbol{x}) = E_y[y|\boldsymbol{x}]$
- The risk R pour the family of functions $F_w(\boldsymbol{x})$ decomposes as follows:
  - $R = E_{x,y}[\left(y - F_w(\boldsymbol{x})\right)^2]$
  - $R = E_{x,y}\left[\left(y - E_y[y|\boldsymbol{x}]\right)^2\right] + E_{x,y}\left[\left(E_y[y|\boldsymbol{x}] - F_w(\boldsymbol{x})\right)^2\right]$
- Let us consider $E_y\left[\left(y - E_y[y|\boldsymbol{x}]\right)^2\right]$
  - This term is independent of the model $F_w(.)$ and only depends on the problem characteristics (the data distribution).
  - It represents the min error that could be obtained for this data distribution
  - $h^*(x) = E_y[y|\boldsymbol{x}]$ is the optimal solution to $\mathrm{Min}_h R$
- Minimizing $E_{x,y}[\left(y - F_w(\boldsymbol{x})\right)^2]$ is equivalent to minimizing $E_{x,y}\left[\left(E_y[y|\boldsymbol{x}] - F_w(\boldsymbol{x})\right)^2\right]$
  - The optimal solution $F_{w*}(\boldsymbol{x}) = \mathrm{argmin}_w E_{x,y}\left[\left(E_y[y|\boldsymbol{x}] - F_w(\boldsymbol{x})\right)^2\right]$ is the best mean square approximation of $E[y|\boldsymbol{x}]$

Probabilistic interpretation of NN outputs

- Classification
  - Let us consider multi-class classification with one hot encoding of the target outputs
    - i.e. $\boldsymbol{y} = (0, \dots, 0, 1, 0, \dots, 0)^T$ with a 1 at position i if the target is class i and zero everywhere else
    - $h_i^* = E_y[y|x] = 1 * P(C_i|x) + 0 * \left(1 - P(C_i|x)\right) = P(C_i|x)$
    - i.e. $F_{w^*}()$ is the best LMS approximation of the Bayes discriminant function (which is the optimal solution for classification with $0/1$ loss)
  - More generally with binary targets
    - $h_i^* = P(y_i = 1|x)$
- Note
  - Similar results hold for the cross entropy criterion
  - Precision on the computed outputs depends on the task
    - Classification: precision might not be so important (max decision rule, one wants the correct class to be ranked above all others)
    - Posterior probability estimation: precision is important