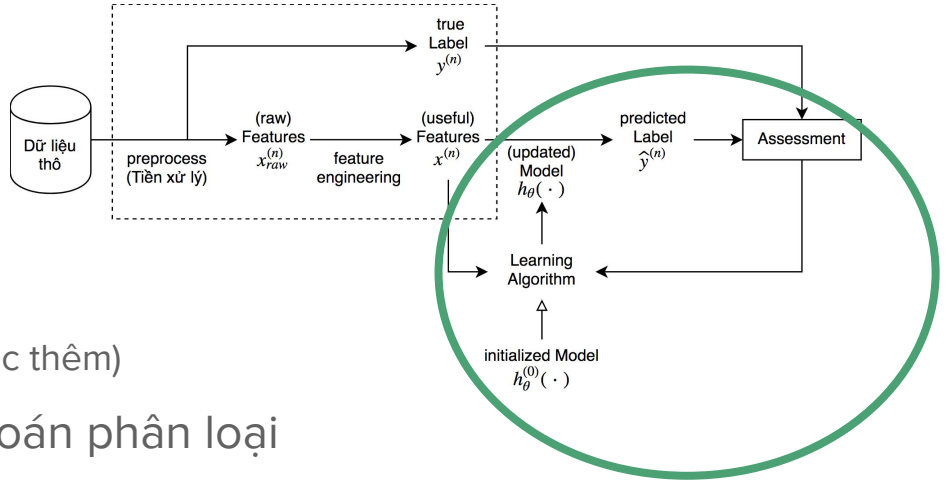
VietAI

Bài 6: Bài toán Phân loại & Mô hình Logistic/Softmax Regression

Tuần 3B

Nội dung chính

1. Mô hình phân loại
2. Binary Classifiers
   a. Mô hình Logistic Regression
3. Multi-class (single-label) Classifiers
   a. Mô hình Softmax Regression
   b. Một số lưu ý về mô hình phân loại (đọc thêm)
4. Một số độ đo đánh giá trong bài toán phân loại



Bài toán phân loại

Giới thiệu chung

Binary classification - Phân loại nhị phân

Image

$x^{(i)}$

$[0,255]^{600\times400\times3}$

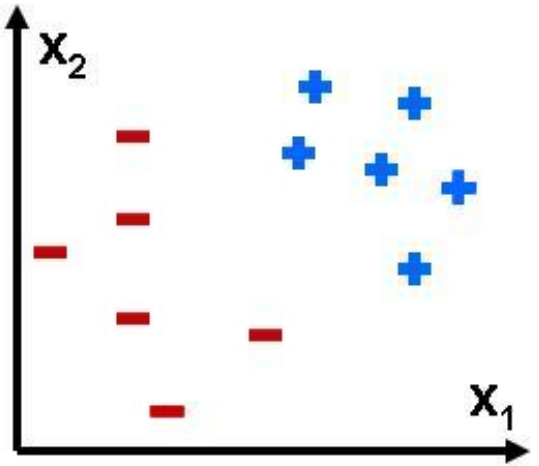
Cat?
Not cat?

Prediction

$\hat{y}^{(i)}$

True / False

$\{1, 0\}$ hoặc $\{1, -1\}$



- "1" được gọi là ***positive class*** - lớp dương; "0", hoặc "-1", được gọi là ***negative class*** - lớp âm
- Dữ liệu có nhãn - *label* - "1" tức là dữ liệu thuộc lớp "1"

Giới thiệu chung

Multiclass classification - Phân loại đa lớp (2+ lớp)

(single-label)

Image

$x^{(i)}$

$[0,255]^{600\times400\times3}$

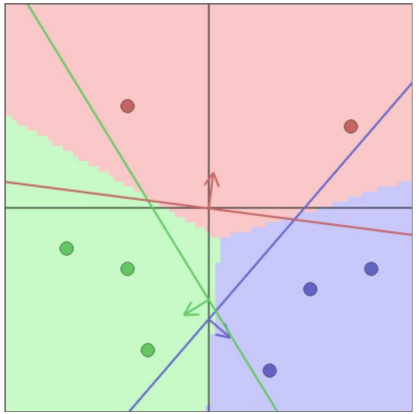
cat
flower
dog
Jet
ground
grass

Prediction

$\hat{y}^{(i)}$

$\{1,2,3,4,5,6\}$

one-hot



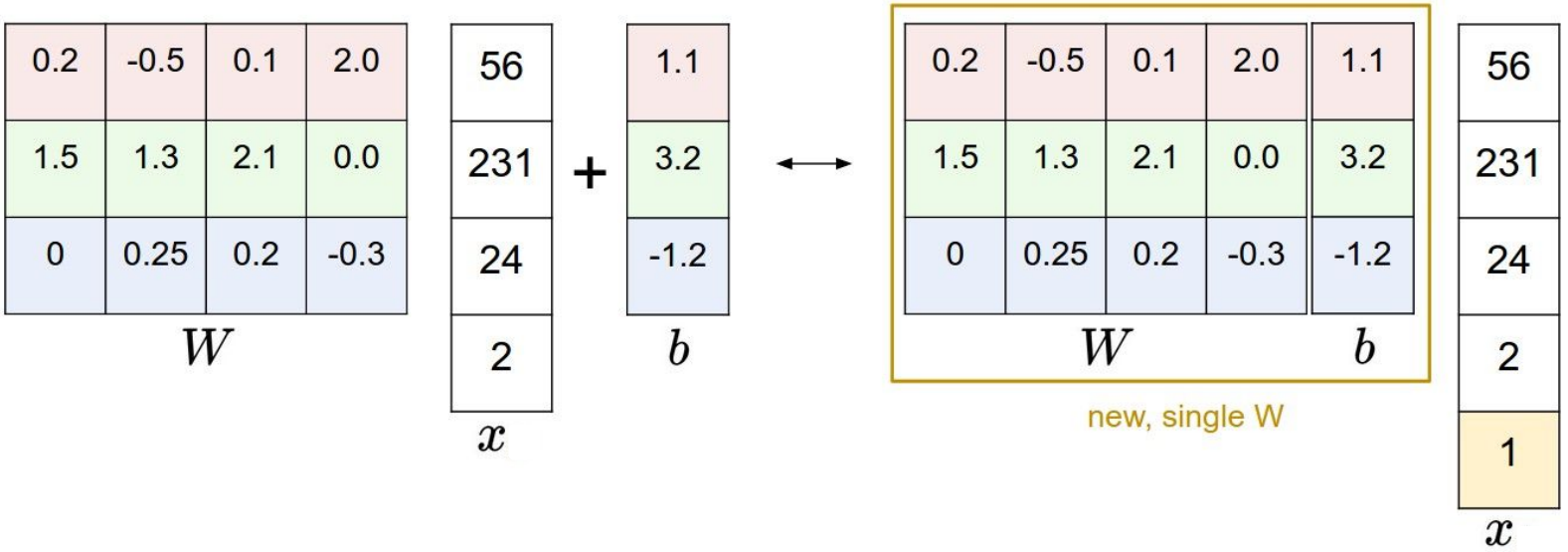
(image credits: Stanford's CS231n)

- Giả sử các nhãn trong mỗi ảnh là mutually exclusive (nếu có nhãn "cat" thì không thể đồng thời có nhãn khác)
  - Một điểm dữ liệu có thể thuộc nhiều lớp - *multi-labelled data* *(không thuộc nội dung bài học)*
- Nhãn y cần được biểu diễn dưới dạng one-hot

Mô hình Phân loại nhị phân

Case-study: Mô hình Logistic Regression

Quy ước



- Viết tắt $x^T W + b \rightarrow x^T W \rightarrow x^T \theta$
- Trong tuần 2-3, ta có thể dùng θ thay cho *W* (và *b*) vì các mô hình được giới thiệu đều là mô hình *tuyến tính* (theo *W*). Kể từ tuần 4, ta đi vào các mô hình *phi tuyến* theo *W* nên sẽ không sử dụng cách viết này nữa.

(image credits: Stanford's CS231n)

Mô hình

Intuition

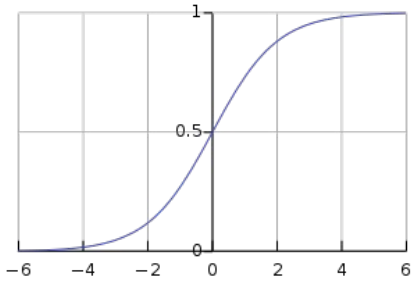
- Tiếp cận bài toán phân loại theo hướng Discriminative modelling
- **Ước lượng *P(y|x)*** - phân bố xác suất của label *y* khi biết data *x*

$$P(y=1|x) = h_\theta(x) = g\left(\theta^{\mathrm{T}}x\right) = \frac{1}{1+\exp(-\theta^{\top}x)}$$

$$P(y=0|x) = 1 - P(y=1|x) = 1 - h_\theta(x)$$

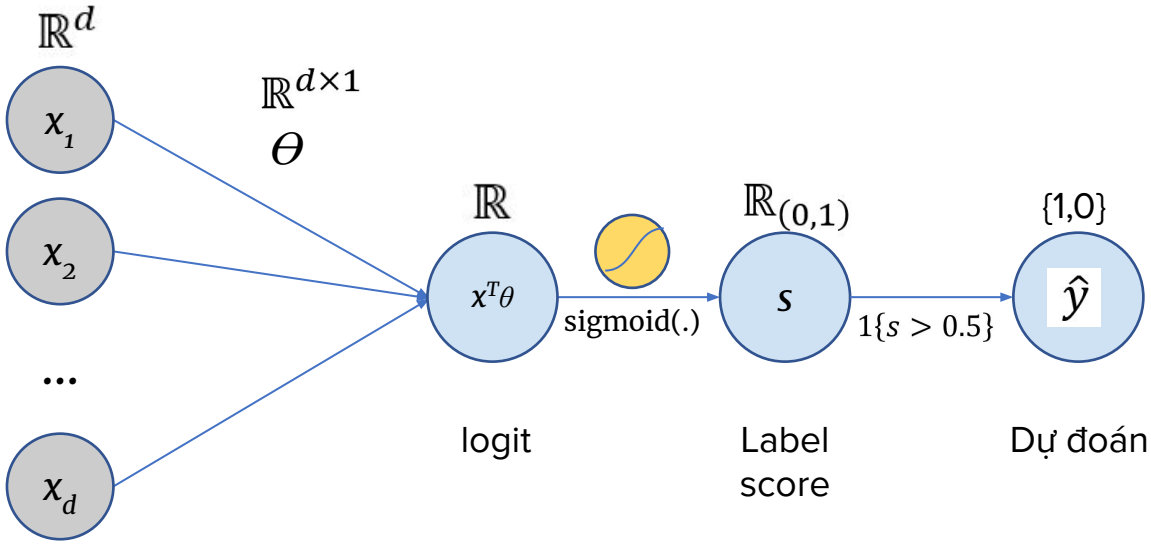
- Hàm *g* được gọi là hàm ***sigmoid***, map giá trị đầu vào về các giá trị trong khoảng [0, 1]

$$g(z) = \frac{1}{1+e^{-z}}$$



Mô hình

Dự đoán khi có x và W



Chú ý: Đường / Siêu phẳng (hyperplane) phân định 2 lớp là $x^T\theta = 0$

(tương đương với $s_{threshold} = 0.5$)

Học mô hình

Cost Function

- Cost function được xây dựng để *cực đại hóa* hàm ***likelihood*** - thể hiện khả năng quan sát được tập (nhãn, dữ liệu) $Y_1^m = \{y^{(m)}, \dots, y^{(m)}\}, X_1^m = \{x^{(1)}, \dots, x^{(m)}\}$ của một giá trị Θ nhất định

$$\mathcal{L}(\theta) = p(Y_1^m | X_1^m, \theta) = \prod_{i=1}^{m} p_\theta\left(y^{(i)} | x^{(i)}\right)$$

- Vì likelihood có giá trị rất bé nên ta làm việc với $\log \mathcal{L}(\theta)$

=> Cost function cần <u>cực tiểu hóa</u> trong mô hình Logistic Regression như sau:

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \left(y^{(i)} \log(h_\theta(x^{(i)})) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) \right)$$

tên thường gọi: ***binary cross-entropy***

Học mô hình

Gradient Descent

- Đạo hàm của hàm sigmoid

$$g'(z) = g(z)\,(1 - g(z))$$

- Đạo hàm của Cost function

$$\frac{\partial J(\theta)}{\partial \theta_j} = \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)x_j^{(i)}$$

- Thuật toán Gradient Descent được áp dụng như sau:

$$\theta_j := \theta_j - \alpha\frac{\partial J(\theta)}{\partial \theta_j}$$

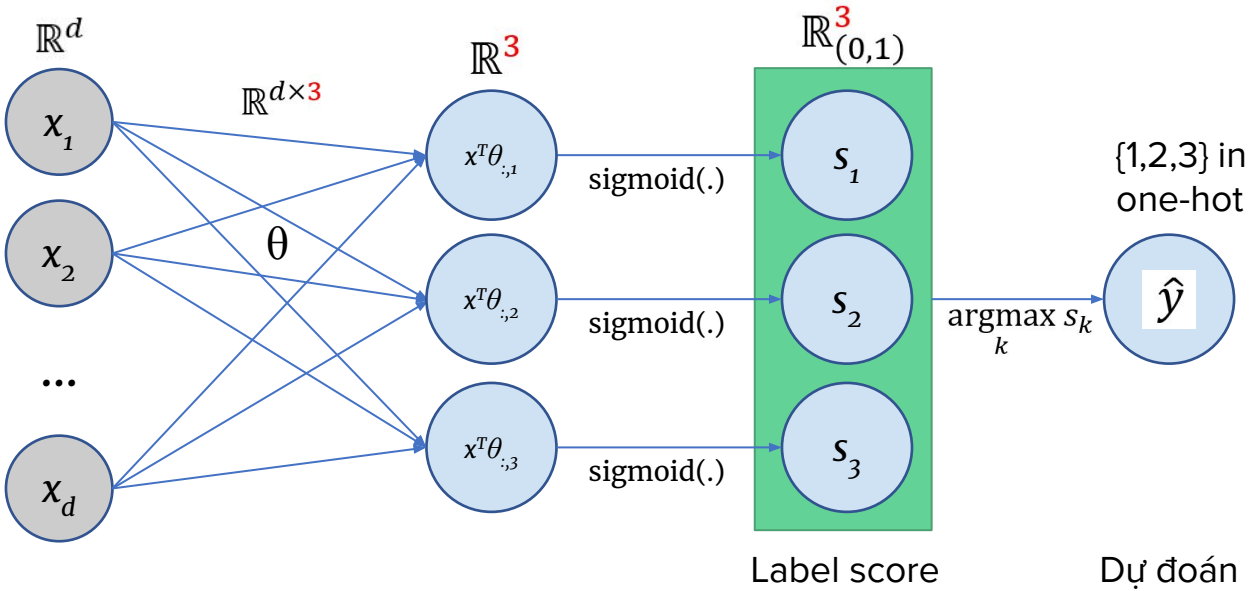
Mô hình Phân loại đa lớp (đơn nhãn)

Case-study: Mô hình Softmax Regression

Mô hình One-vs-Rest Logistic Regression

Intuition: Mở rộng trực tiếp từ mô hình Logistic Regression

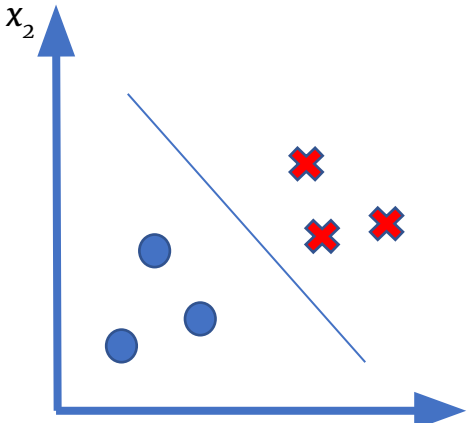
- Huấn luyện K mô hình LogReg cho **từng** class k để dự đoán xác suất $p(y=k|x)$
- Dự đoán: lớp cho giá trị $p(y=k|x)$ lớn nhất $\hat{y} = \underset{k}{\text{argmax}} p(y = k|x)$



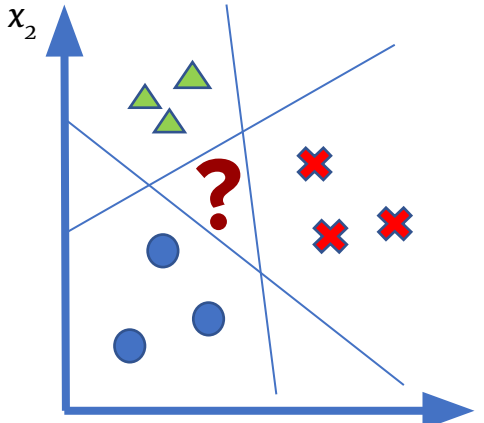
Mô hình One-vs-Rest Logistic Regression

Nhược điểm

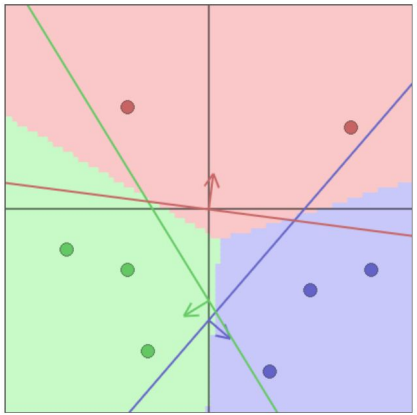
- Các đường / siêu phẳng phân định $x^T\theta_{:,k} = 0$ tạo ra vùng "không xác định"
- Không xét điều kiện các nhãn là mutually exclusive trong bài toán phân loại đơn nhãn - single-label ("đã có 1 nhãn k thì không thể có nhãn k' khác")



Logistic Regression cho phân loại nhị phân



OvR Logistic Regression cho phân loại đa lớp



Softmax Regression vs OvR Logistic Regression

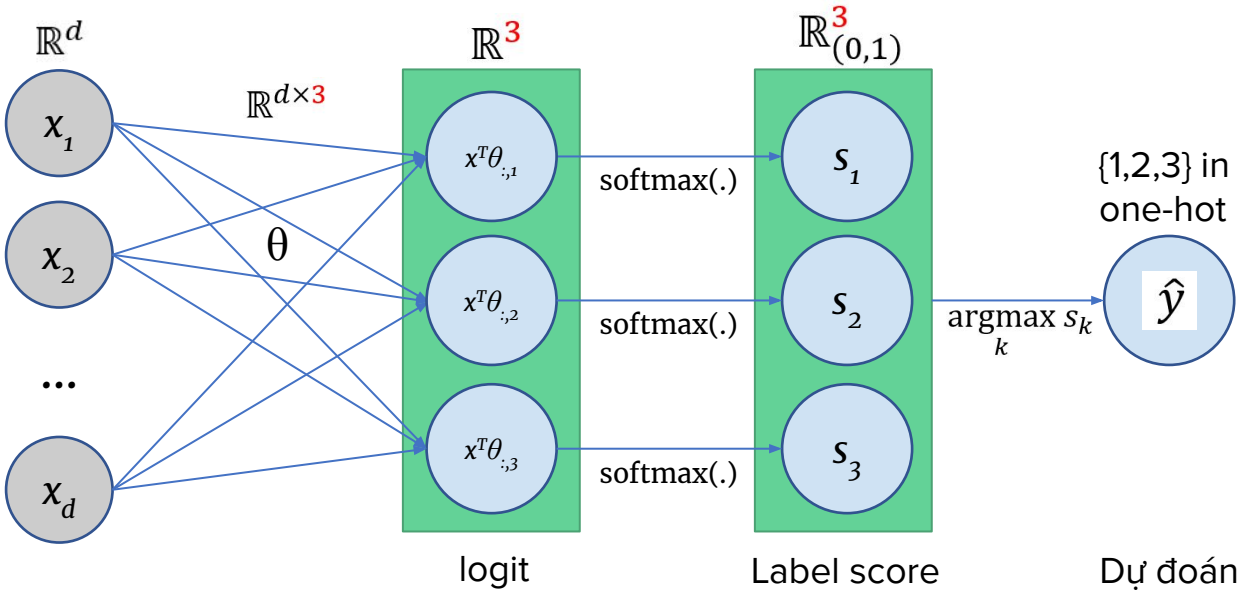
(image credits: Stanford's CS231n)

Mô hình Softmax Regression

Mô hình

Thay vì dùng hàm sigmoid, ta dùng hàm **softmax** để chuẩn hóa các giá trị sao cho các giá trị label score s_k tạo thành *một phân phối xác suất (tổng các giá trị bằng 1)*

$$s_k = P(y^{(i)} = k | x^{(i)}; \theta) = \frac{\exp(\theta^{(k)\top} x^{(i)})}{\sum_{j=1}^{K} \exp(\theta^{(j)\top} x^{(i)})}$$



Mô hình Softmax Regression

Cost Function

- Cách xây dựng tương tự như Cost function của Logistic Regression
- Cost function được xây dựng để *cực đại hóa* hàm ***likelihood*** - thể hiện khả năng quan sát được tập (nhãn, dữ liệu) $Y_1^m = \{y^{(m)}, \dots, y^{(m)}\}, X_1^m = \{x^{(1)}, \dots, x^{(m)}\}$ của một giá trị Θ nhất định

$$\mathcal{L}(\theta) = p(Y_1^m | X_1^m, \theta) = \prod_{i=1}^{m} p_\theta\left(y^{(i)} | x^{(i)}\right)$$

- Vì likelihood có giá trị rất bé nên ta làm việc với $\log \mathcal{L}(\theta)$

=> Cost function cần <u>cực tiểu hóa</u> trong mô hình Softmax Regression như sau:

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \sum_{k=1}^{K} \left(1\{y^{(i)} = k\} \log s_k\right)$$

tên thường gọi: ***categorical cross-entropy***

Mô hình Softmax Regression

Mối quan hệ với Logistic Regression

- Cost function của Logistic Regression có thể được viết lại như sau:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left((1-y^{(i)})\log(1-h_\theta(x^{(i)})) + y^{(i)}\log h_\theta(x^{(i)})\right)$$

$$= -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=0}^{1} 1\left\{y^{(i)}=k\right\}\log P(y^{(i)}=k|x^{(i)};\theta)$$

- Với trường hợp $K = 2$, Softmax Regression có thể được viết như sau:

$$\begin{bmatrix} P(y=1|x) \\ P(y=2|x) \end{bmatrix} = \frac{1}{\exp(\theta^{(1)\top}x) + \exp(\theta^{(2)\top}x^{(i)})} \begin{bmatrix} \exp(\theta^{(1)\top}x) \\ \exp(\theta^{(2)\top}x) \end{bmatrix}$$

Mô hình Softmax Regression

Mối quan hệ với Logistic Regression

- Như vậy có thể thấy, khi đó Softmax Regression chính là Logistic Regression với $\theta' = \theta^{(2)} - \theta^{(1)}$

$$\begin{bmatrix} P(y=1|x) \\ P(y=2|x) \end{bmatrix} = \frac{1}{\exp((\theta^{(1)} - \theta^{(2)})^\top x^{(i)}) + \exp(\vec{0}^\top x)} \begin{bmatrix} \exp((\theta^{(1)} - \theta^{(2)})^\top x) \\ \exp(\vec{0}^\top x) \end{bmatrix}$$

$$= \begin{bmatrix} \frac{\exp((\theta^{(1)} - \theta^{(2)})^\top x)}{1+\exp((\theta^{(1)} - \theta^{(2)})^\top x^{(i)})} \\ \frac{1}{1+\exp((\theta^{(1)} - \theta^{(2)})^\top x^{(i)})} \end{bmatrix}$$

$$= \begin{bmatrix} 1 - \frac{1}{1+\exp((\theta^{(1)} - \theta^{(2)})^\top x^{(i)})} \\ \frac{1}{1+\exp((\theta^{(1)} - \theta^{(2)})^\top x^{(i)})} \end{bmatrix}$$

Đọc thêm

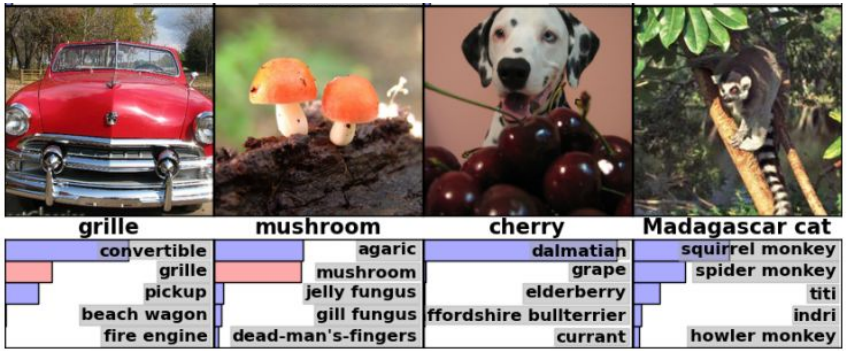
- Một mô hình phân loại theo cách tiếp cận Discriminative modelling **không** nhất thiết phải ước lượng phân bố *P(y|x)* - tức *không* nhất thiết phải có ý nghĩa xác suất - mà chỉ cần ước lượng đường phân định (decision boundary)
- Có thể thay cross-entropy loss bằng *hinge loss* để huấn luyện mô hình bằng Gradient Descent
  - Khi đó mô hình không còn có ý nghĩa xác suất, nhưng vẫn học ra các đường / siêu phẳng phân định $x^T\theta_{:,k} = 0$
  - Tham khảo http://cs231n.github.io/linear-classify/
- Một số mô hình phân loại phổ biến khác:
  - k-Nearest Neighbours
  - Classification tree, Random Forest
  - Support Vector Machine

Một số độ đo Đánh giá trong bài toán phân loại

Chỉ số đánh giá chung

Accuracy, hay 0-1 Error

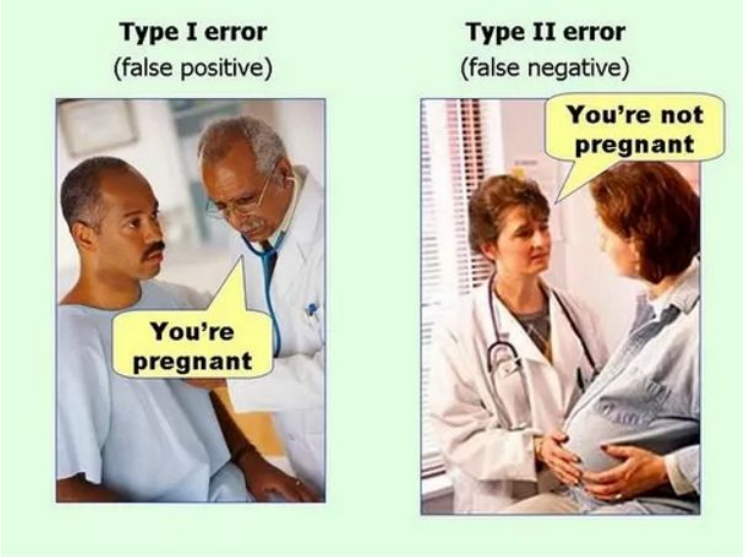
- ***Accuracy*** là độ đo thường gặp nhất trong bài toán phân loại, được tính dựa trên số mẫu dữ liệu được phân lớp đúng chia tổng số mẫu.
  - vd: classifier thực hiện dự đoán 100 mẫu dữ liệu, trong đó có 75 mẫu dự đoán đúng lớp của nó, khi đó classifier này có accuracy là 75%.
  - Biến thể: Top-k Accuracy / Top-k Error



(image credits: ImageNet Classification with Deep Convolutional Neural Networks (Krizhevsky, 2012))

Đánh giá bài toán Binary Classification

Precision, Recall



		Thực tế +	Thực tế -	
Dự đoán/ Quyết định	+	True positive	False positive (Lỗi loại I)	P'
	-	False Negative (Lỗi loại II)	True Negative	
		P		

(image credits: Duke University)

- Định lượng được chất lượng của dự đoán trên *từng* lớp khi có hiện tượng **class imbalance**
  - Biến thể: Precision@M, Recall@M
- Lưu ý:
  - Recall còn được gọi là True Positive Rate (TPR)
  - Precision/Recall cũng thường được sử dụng để đánh giá trong bài toán Truy vấn dữ liệu (Information Retrieval)

$$Precision = \frac{TP}{P'} = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{P} = \frac{TP}{TP + FN}$$

Đánh giá bài toán Binary Classification

F1-score, AUC-PR

- Nếu muốn so sánh chất lượng của các mô hình trên đồng thời 2 chỉ số Precision và Recall, ta cần một chỉ số chung vd. F1-score
- F1-score được tính bằng ***trung bình điều hòa*** (***harmonic mean***) của Precision và Recall

$$F_1 = \frac{2}{\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}}$$

  - Biến thể: F_β-score
  - Các độ đo liên quan:
    - AUC-PR: Area Under Precision-Recall Curve
    - AUC-ROC: Area Under ROC curve, hay Area Under TPR-FPR curve (không khuyến nghị sử dụng khi bài toán có hiện tượng class imbalance

Mở rộng cho đánh giá bài toán Multiclass Classification

Cách kết hợp kết quả Precision, Recall, F1-Score cho nhiều lớp:

- ***Macro-averaged***: Tính tổng các đại lượng TP, FP, TN, FN trên tất cả các lớp, sau đó mới tính các độ đo Precision, Recall, F1-Score dựa trên TP, FP, TN, FN thu được.
- ***Micro-averaged***: Tính các độ đo Precision, Recall, F1-Score cho từng lớp rồi lấy kết quả trung bình
- ***Weighted***: Tính các độ đo Precision, Recall, F1-Score cho từng lớp rồi tính trung bình theo trọng số cho trước (thường là dựa theo tỉ lệ của lớp tương ứng trong tập dữ liệu)

Tài liệu tham khảo

1. Stanford's CS231n on Linear Classifier
2. UFLDL Tutorials on Logistic and Softmax Regression
3. Machine Learning - CS229 - Stanford University

https://bit.ly/2k134zX