Machine Learning for Computational Linguistics

Classification

Çağrı Çöltekin

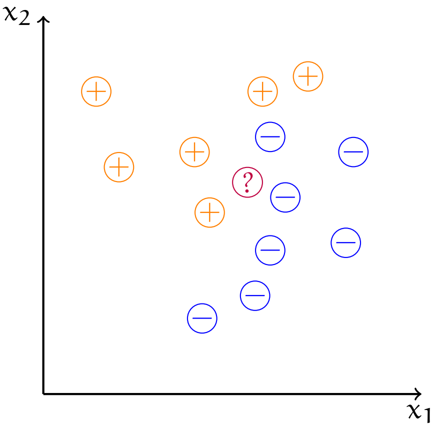
University of Tübingen
Seminar für Sprachwissenschaft

May 3, 2016

Practical issues

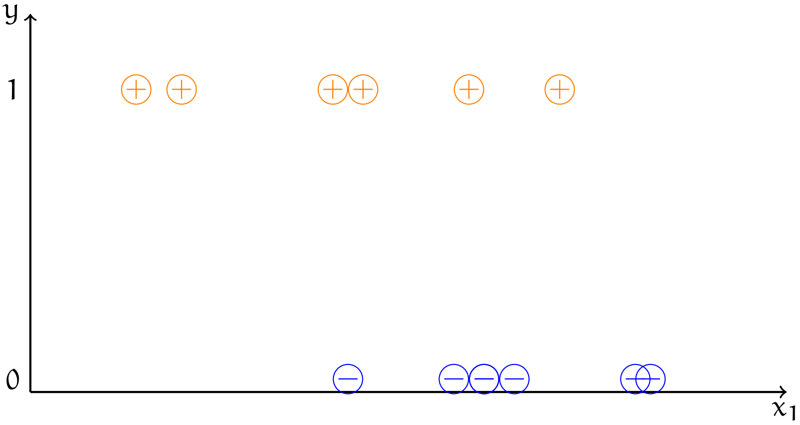
- Homework 1: try to program it without help from specialized libraries (like NLTK)
- Time to think about projects. A short proposal towards the end of May.

The problem



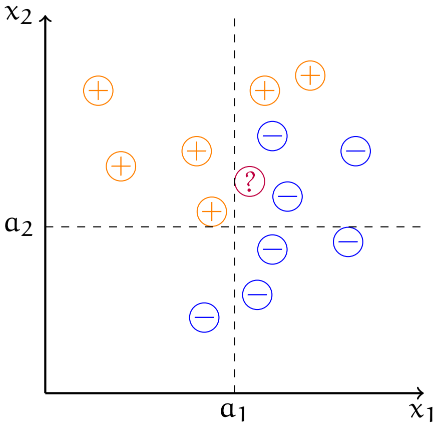
- The response (outcome) is a label. In the example: positive ⊕ or negative ⊖
- Given the features (x_1 and x_2), we want to predict the label of an unknown instance ?
- Note: regression is not a good idea here

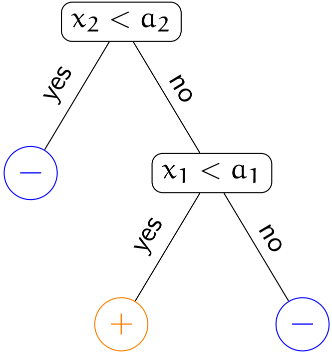
The problem (with a single predictor)



A quick survey of some solutions

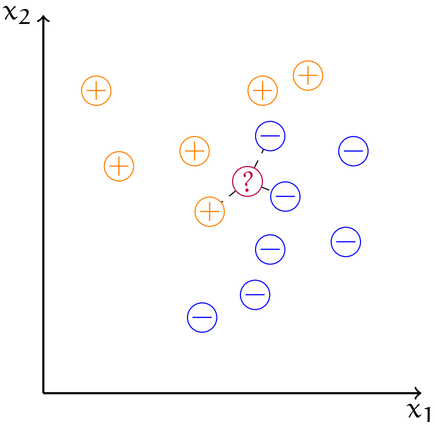
Decision trees





A quick survey of some solutions

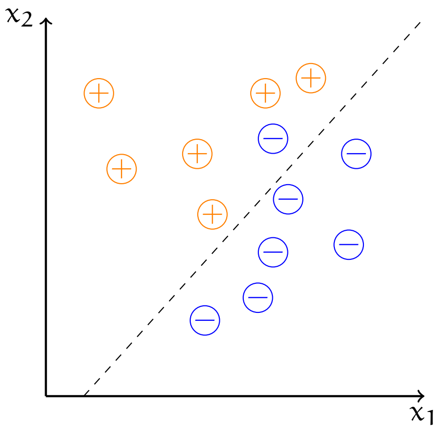
Instance/memory based methods



- No training: just memorize the instances
- During test time, decide based on the k nearest neighbors
- Like decision trees, kNN is non-parametric
- It can also be used for regression

A quick survey of some solutions

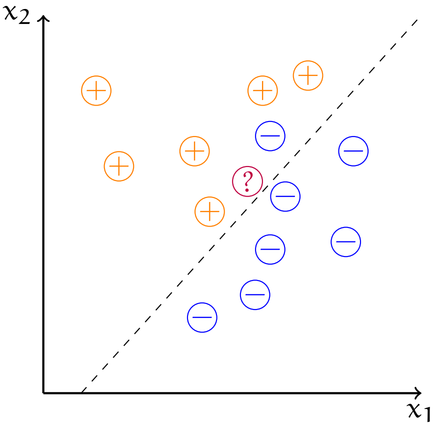
(Linear) discriminant functions



- Find a discriminant function (f) that separates the training instance best (for a definition of 'best')

A quick survey of some solutions

(Linear) discriminant functions

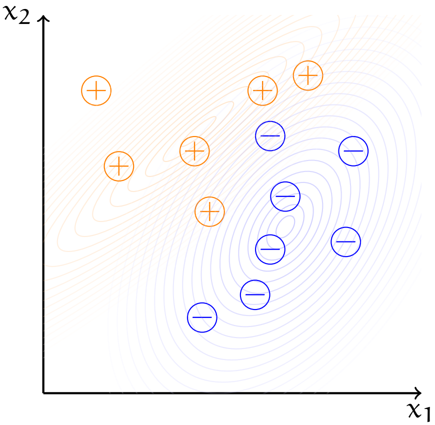


- Find a discriminant function (f) that separates the training instance best (for a definition of 'best')
- Use the discriminant to predict the label of unknown instances

$$\hat{y} = \begin{cases} \oplus & f(\boldsymbol{x}) > 0 \\ \ominus & f(\boldsymbol{x}) < 0 \end{cases}$$

A quick survey of some solutions

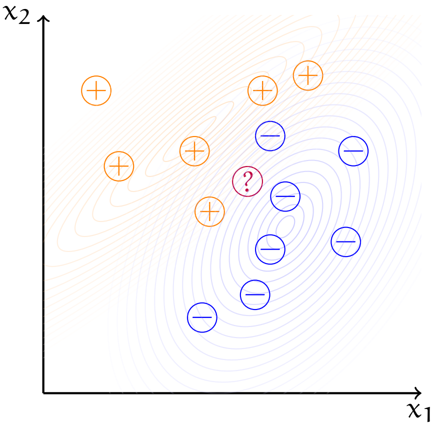
Probability-based solutions



- Estimate distributions of $p(\boldsymbol{x}|y = \oplus)$ and $p(\boldsymbol{x}|y = \ominus)$ from the training data
- Assign the new items to the class c with the highest $p(\boldsymbol{x}|y = c)$

A quick survey of some solutions

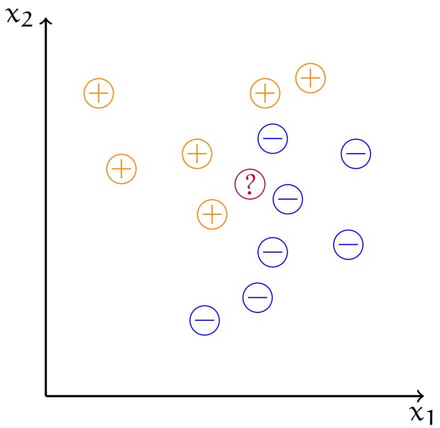
Probability-based solutions

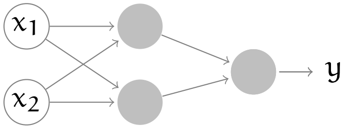


- Estimate distributions of $p(\boldsymbol{x}|y = \oplus)$ and $p(\boldsymbol{x}|y = \ominus)$ from the training data
- Assign the new items to the class c with the highest $p(\boldsymbol{x}|y = c)$

A quick survey of some solutions

Artificial neural networks





Logistic regression

- Logistic *regression* is a *classification* method
- In logistic regression, we fit a model that predicts $P(y|x)$
- Alternatively, logistic regression is an extension of linear regression. It is a member of the family of models called generalized linear models

A simple example

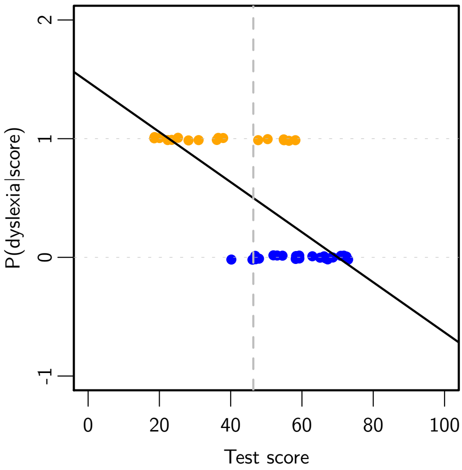
We would like to guess whether a child would develop dyslexia or not based on a test applied to pre-verbal children. Here is a simplified problem:

- We test children when they are less than 2 years of age.
- We want to predict the diagnosis from the test score
- The data looks like

Test score	Dyslexia
82	0
22	1
62	1
⋮	⋮

* The research question is from a real study by Ben Maasen and his colleagues. Data is fake as usual.

Example: fitting ordinary least squares regression



Problems:

- The probability values are not bounded between 0 and 1
- Residuals will be large for correct predictions
- Residuals are not distributed normally

Example: transforming the output variable

Instead of predicting the probability p, we predict logit(p)

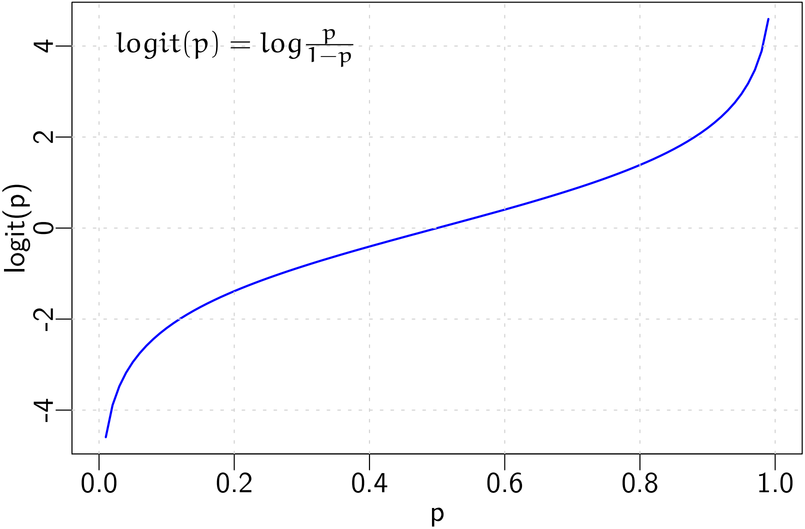
$$\hat{y} = \text{logit}(p) = \log \frac{p}{1-p} = w_0 + w_1 x$$

- $\frac{p}{1-p}$ (odds) is bounded between 0 and ∞
- $\log \frac{p}{1-p}$ (log odds) is bounded between $-\infty$ and ∞
- we can estimate $\mathrm{logit}(p)$ with regression, and convert it to a probability using the inverse of logit

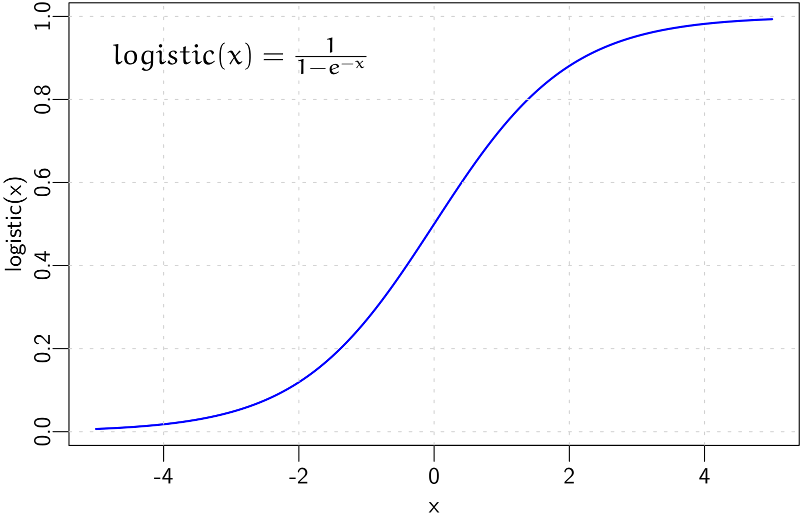
$$\hat{p} = \frac{e^{w_0 + w_1 x}}{1 + e^{w_0 + w_1 x}} = \frac{1}{1 + e^{-w_0 - w_1 x}}$$

which is called logistic function (or sometimes sigmoid function, with some ambiguity).

Logit function



Logit function



Logistic regression as a generalized linear model

Logistic regression is a special case of generalized linear models (GLM). GLMs are expressed with,

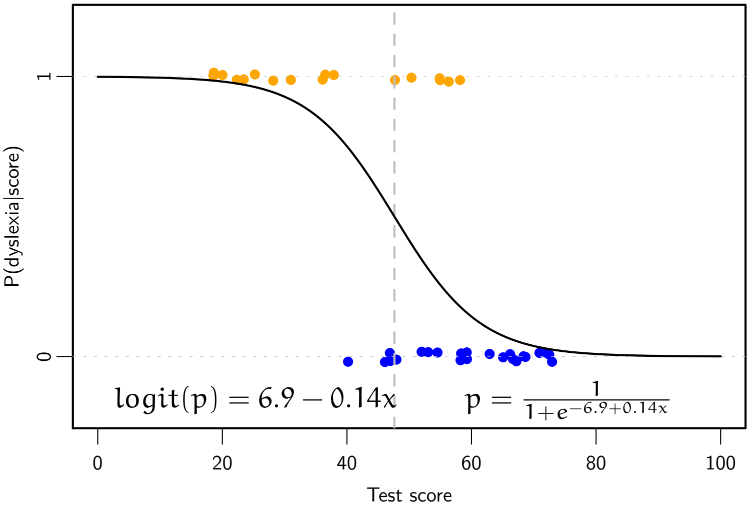
$$g(\mathbf{y}) = \mathbf{X}\boldsymbol{w} + \boldsymbol{\epsilon}$$

- The function $g()$ is called the *link function*
- $\boldsymbol{\epsilon}$ is distributed according to a distribution from *exponential family*
- For logistic regression, $g()$ is the logit function, $\boldsymbol{\epsilon}$ is distributed binomially
- For linear regression $g()$ is the identity function, $\boldsymbol{\epsilon}$ is distributed normally

Interpreting the dyslexia example

```
glm(formula = diag ~ score, family = binomial, data = dys)
Coefficients:
          Estimate Std. Error z value Pr(>|z|)
(Intercept) 6.90079  2.31737  2.978 0.00290 **
score     -0.14491  0.04493 -3.225 0.00126 **
---
(Dispersion parameter for binomial family taken to be 1)
   Null deviance: 54.548 on 39 degrees of freedom
Residual deviance: 30.337 on 38 degrees of freedom
AIC: 34.337
Number of Fisher Scoring iterations: 5
```

Interpreting the dyslexia example



How to fit a logistic regression model

Reminder:

$$P(y = 1|\boldsymbol{x}) = p = \frac{1}{1 + e^{-\boldsymbol{w}\boldsymbol{x}}} \qquad P(y = 0|\boldsymbol{x}) = 1 - p = \frac{e^{-\boldsymbol{w}\boldsymbol{x}}}{1 + e^{-\boldsymbol{w}\boldsymbol{x}}}$$

The likelihood of the training set is,

$$\mathcal{L}(\boldsymbol{w}) = \prod_i P(y_i|x_i) = \prod_i p^{y_i}(1-p)^{1-y_i}$$

In practice, maximizing $\log$ likelihood is more practical:

$$\hat{\boldsymbol{w}} = \arg\max_{\boldsymbol{w}} \log \mathcal{L}(\boldsymbol{w}) = \sum_i P(y_i|x_i) = \sum_i y_i \log p + (1 - y_i) \log(1 - p)$$

To maximize, we find the gradient:

$$\nabla \log \mathcal{L}(\boldsymbol{w}) = \sum_i (y_i - \frac{1}{1 + e^{-\boldsymbol{w}\boldsymbol{x}}}) \boldsymbol{x_i}$$

How to fit a logistic regression model (2)

- Bad news is that there is no analytic solution to the set of equations

$$\nabla \log \mathcal{L}(\boldsymbol{w}) = \boldsymbol{0}$$

- Good news is that the (negative) log likelihood is a convex function: there is a global maximum
- We can use iterative methods such as gradient descent to find parameters that maximize the (log) likelihood
- In practice, it is more common minimize the negative log likelihood

$$J(\boldsymbol{w}) = -\log\mathcal{L}(\boldsymbol{w})$$

$J(\boldsymbol{w})$ is called the loss function, cost function or objective function

- Using gradient descent, we repeat

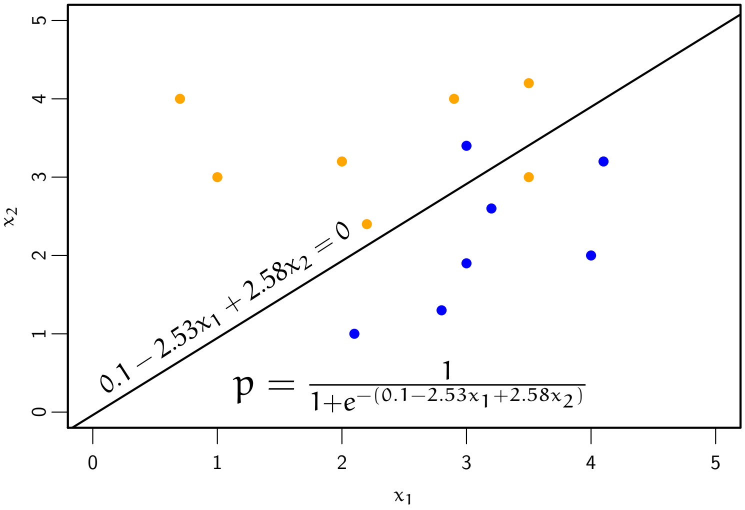
$$\boldsymbol{w} \leftarrow \boldsymbol{w} - \alpha \nabla J(\boldsymbol{w})$$

until convergence. α is called learning rate.

An example with two predictors

```
Call: glm(formula = label ~ x1 + x2, family = binomial, data = d)
Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) 0.09692  4.74728  0.020  0.9837
x1         -2.53416  1.69222 -1.498  0.1343
x1          2.57632  1.36655  1.885  0.0594 .
---
(Dispersion parameter for binomial family taken to be 1)
    Null deviance: 19.408 on 13 degrees of freedom
Residual deviance: 7.987 on 11 degrees of freedom
AIC: 13.987
Number of Fisher Scoring iterations: 6
```

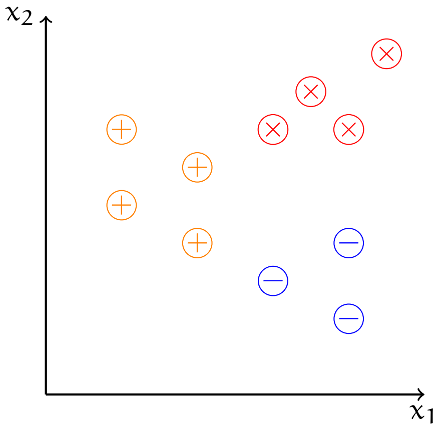
An example with two predictors (2)



More than two classes

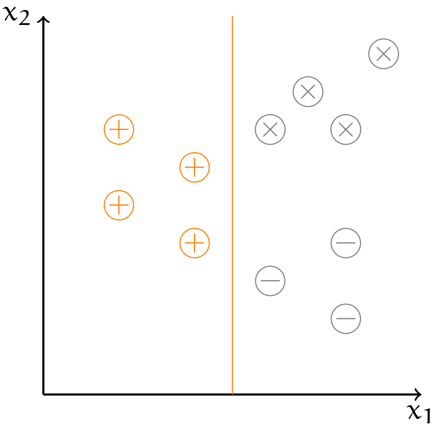
- Some algorithms can naturally be extended to multiple labels
- Others tend to work well in binary classification
- Any binary classifier can be turned into a k-way classifier by
  - training k one-vs.-rest (OvR) or one-vs.-all (OvA) classifiers. Decisions are made based on the class with the highest confidence score. This approach is feasible for classifiers that assign a weight or probability to the individual classes
  - training $\frac{k(k-1)}{2}$ one-vs.-one (OvO) classifiers. Decisions are made based on majority voting

One vs. Rest



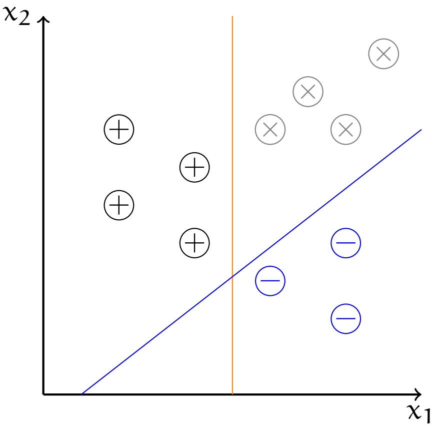
- For 3 classes we fit 3 classifiers separating one class from the rest

One vs. Rest



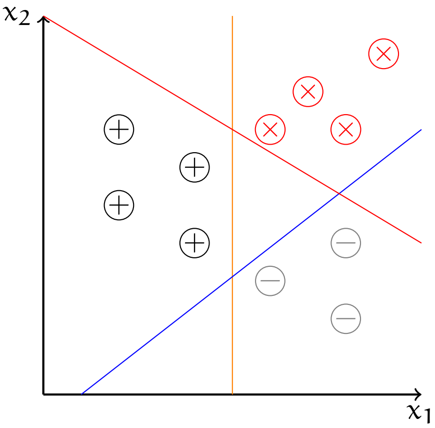
- For 3 classes we fit 3 classifiers separating one class from the rest

One vs. Rest



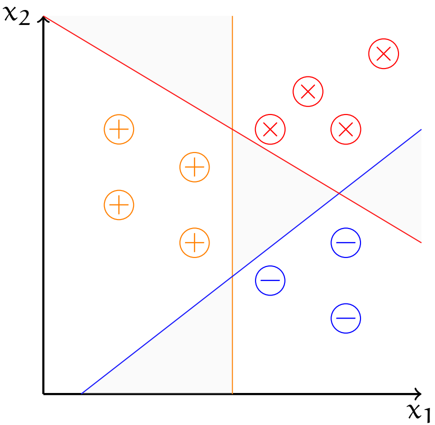
- For 3 classes we fit 3 classifiers separating one class from the rest

One vs. Rest



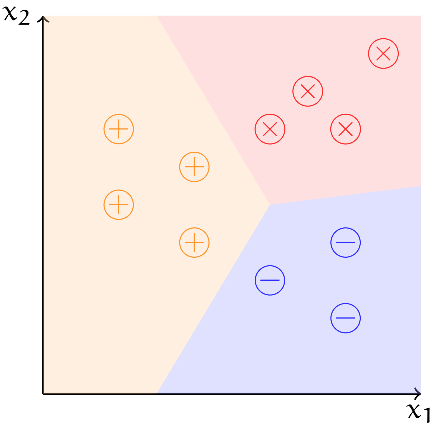
- For 3 classes we fit 3 classifiers separating one class from the rest

One vs. Rest



- For 3 classes we fit 3 classifiers separating one class from the rest
- Some regions of the feature space will be ambiguous

One vs. Rest



- For 3 classes we fit 3 classifiers separating one class from the rest
- Some regions of the feature space will be ambiguous
- We can assign labels based on probability or weight value, if classifier returns one
- One-vs.-one and majority voting is another option

Multi-class logistic regression

- Generalizing logistic regression for more than two classes is straightforward
- We estimate,

$$P(C_k|x) = \frac{e^{\boldsymbol{w}_k \boldsymbol{x}}}{\sum_j e^{\boldsymbol{w}_j \boldsymbol{x}}}$$

  Where C_k is the k^{th} class. j iterates over all classes.
- This model is also known as a log-linear model, Maximum entropy model, Boltzman machine