

P1-

quelques exemples de tâches :

1. Détecter les spams dans un ensemble d'emails
2. Trouver dans le catalogue d'une bibliothèque de ouvrages qui traitent de l'art macabre au XVème siècle
3. Trouver un site internet qui vend des ordinateurs portables sans système d'exploitation
4. Déterminer le nom du héros et des principaux protagonistes d'un texte narratif
5. Découper la transcription d'un discours en parties thématiques
6. Déterminer les symptômes présentés par un patient à partir d'un rapport clinique
7. Repérer dans un ensemble de tweets ceux qui sont sarcastiques ou ironiques

Les ressources :

Listes → mots, caractères...

Corpus

Bases de connaissances → bases de données, ontologies, lexiques, wordnets...

Programmes de pré-traitement → segmentation, normalisation...

Programmes réalisant des traitements linguistiques poussés → lemmatisation, étiquetage morphosyntaxique...

Ressources linguistiques :

Le Lexique Extensionnel des Formes Fléchies du Français (Lefff) 2. Wikidata 3. UDPipe 4. Le French Treebank 5. Universal dependencies 6. JeuxDeMots

Quelques exemples de programmes issus d'apprentissage artificiel :

1. SEM (POS tagging, chunking, détection d'entités nommées)
2. UDPipe (segmenteur/étiqueteur/parser)
3. Google translate (traduction)
4. HMTL (multi-tâches de compréhension)
5. @picdescbot (description d'images en langage naturel2)

P2-

有三个集：

Jeu d'entraînement (train) 训练集: utilisé pour apprendre le modèle.

Jeu de test (test) 测试集: utilisé pour évaluer ses performances.

Dev set 开发集--En plus des jeux d'entraînement et de test, on introduit un **jeu de développement (ou validation)**.

有三个评估方法(通常评估的是模型的 预测输出 vs 正确答案 之间的差距)：

-**Train/Test 分割** 是最基础的方法 Séparation Entraînement/Test : éviter le biais de mémorisation

-**Le jeu de développement** sert à ajuster les paramètres et choisir le meilleur modèle sans toucher au jeu de test.

- Validation croisée (Cross-validation) 1 : On découpe le corpus en **n parts égales** (souvent 5 ou 10). 2 : À chaque itération : on utilise n-1 parties pour entraîner et 1 partie pour tester. 3 : On répète n fois, puis on **moyenne les résultats**. Avantages : Exploite mieux les données

F-mesure

$$F_1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

• $\text{precision} = \frac{TP}{TP+FP}$, 表示真正例占预测为正例的比例

• $\text{recall} = \frac{TP}{TP+FN}$, 表示真正例占所有实际正例的比例

F-mesure pondérée 加权 F-测度

P5 - 05-representation_donnees

Heaps law :描述了语料中不同词汇（类型，types）随已处理总词数（标记，tokens）增长而增加的经验关系。

直观含义 1 :随着你读越多文本，新词出现的速度会逐渐放缓，类型数不是线性增长，而是次幂增长。

2 : $\beta < 1$ 意味着“边际新词率”在不断下降：再增加同样数量的词，新增词汇数越来越少。

$$V(N) = K N^\beta$$

- $V(N)$: 在已扫描 N 个词后，累计出现过的不同词数（词典大小、类型数）。
- N : 已处理的总词数（token 数）。
- K, β : 经验常数，通常 $K \approx 10 \sim 100$, $\beta \approx 0.4 \sim 0.6$ 。

- 随机或依照原始顺序，逐段（或等量词块）统计：

$$(N_1, V_1), (N_2, V_2), \dots, (N_m, V_m)$$

例如：每 10 000 词记录一次不同词数。

对数-对数线性回归

- 取对数： $\log V = \log K + \beta \log N$ 。
- 用最小二乘拟合直线，
 - 斜率即 β ,
 - 截距的指数即 $K = e^{\log K}$ 。

例子

N (tokens)	V (types)	$\log N$	$\log V$
10 000	4 500	9.21	8.41
20 000	6 800	9.90	8.83
50 000	10 200	10.82	9.23

{: .small-table}

然后在 $(\log N, \log V)$ 平面上做线性回归即可得 β 。

线性回归 : $y = a + bx$

zipf 定律

将语料中所有词按照出现频率从高到低排序，记排名为 r 的词频率为 $f(r)$ 。Zipf 定律指出：

$$f(r) \approx \frac{C}{r^s}$$

- r : 词的频率排名（最常见的词排名 $r = 1$ ）。
- $f(r)$: 该词出现的相对频率或绝对频数。
- C : 归一化常数，以保证所有频率之和为 1（若用相对频率）。
- s : 幂律指数，语言中通常 $s \approx 1$ 。

当 $s = 1$ 时，最简单形式就是

$$f(r) \approx \frac{C}{r}$$

即排名第二的词频约为第一的 $1/2$ ，第三的词频约为第一的 $1/3$ ，依此类推。

在自然语言中，“哪些词”出现得多、“哪些词”出现得少，遵循一个简单的幂律：词频与排名成反比。这一规律在文本预处理、特征选择和语言建模中都大有用武之地。

Zipf 描述的是频率–排名关系，

Heaps 描述的是类型–规模关系。理论上，如果 Zipf 的指数为 s ，则 Heaps 的指数 $\beta \approx 1/s$ 。

P6----

Hypothese D'independance : 独立性假设

Bien entendu, l'hypothèse d'indépendance est fautive en général → Si un texte contient Quixote, il est plus probable qu'il contienne également Don

→ Si un texte contient Don, il est plus probable qu'il contienne

également Quixote ou Giovanni. Et représenter les documents par des points a ses limites

→ Que signifie le fait de sommer les attributs de deux documents ? Mais ces approximations permettent d'utiliser une machinerie mathématique qui marche suffisamment bien en pratique.

独立性假设通常不成立

词与词之间其实是有强关联的。

例如，若一篇文章出现了“Quixote”（堂吉珂德中的人物名），“Don”出现的概率就会很高；同样出现“Don”时，也很可能找到“Quixote”或“Giovanni”（相关的名字）。

也就是说， a_k 和 a_l 并非真正独立。

把文档当成点/向量表示也有局限

当我们把两篇文档的向量相加（“把属性相加”），语义上其实并没有很明确的含义：“把 A 文档和 B 文档的词频向量相加”——这在现实中代表什么？向量空间模型在数学上方便，但忽略了词序、上下文和深层语义。

为什么还要这样做？

尽管是近似和简化，这套“向量+线性代数”的工具链（内积、距离、投影、降维、分类器等）在实践中往往已经足够好用，能高效地完成检索、分类、聚类任务。这也正体现了**文本挖掘（fouille de textes）**的一个核心思路：接受若干简化假设，用成熟的数学方法快速挖掘大规模文本的结构与规律。

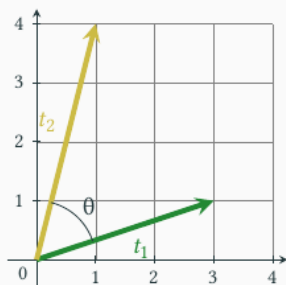
向量（vecteur） 内积 / 点乘（produit scalaire） 角度差 / 余弦相似度（écarts angulaires）

Les maths en cinq secondes : écarts angulaires

En pratique, une propriété qui nous intéresse plus c'est que

$$\langle t_1 | t_2 \rangle = \|t_1\|_2 \times \|t_2\|_2 \times \cos(\theta)$$

Où $\|t_i\|_2$ est la norme euclidienne de t_i — sa longueur pour la distance euclidienne — et $\theta = \widehat{(t_1, t_2)}$ est l'écart angulaire entre les deux vecteurs.



20

在那张幻灯片中，我们看到的是向量内积与夹角之间的关系：

$$\langle t_1 | t_2 \rangle = \|t_1\|_2 \times \|t_2\|_2 \times \cos(\theta)$$

下面逐项解释公式中每个符号的含义：

符号	含义
t_1, t_2	两个向量（在文本处理中通常是两个文档的词频/TF-IDF 向量）
$\langle t_1 t_2 \rangle$	内积（Dot Product）： $\sum_{i=1}^n t_{1,i} t_{2,i}$ ，即把两个向量对应坐标相乘后求和。
$\ t_1\ _2, \ t_2\ _2$	欧几里得范数（Euclidean Norm）： $\ t_1\ _2 = \sqrt{\sum_i t_{1,i}^2}$ ，同理 $\ t_2\ _2$ 。它表示向量的“长度”或“大小”。
θ	夹角（Angle）：两向量在几何空间中从公共原点出发所成的角度。
$\cos(\theta)$	余弦值：度量两向量方向一致程度的数值（范围 $[-1, 1]$ ；词频/TF-IDF 向量为非负，一般在 $[0, 1]$ 之间）。

公式两边的意义

1. 左边 $\langle t_1 | t_2 \rangle$:

- 直接计算坐标相乘求和, 越大说明两个文档在同一维度 (同一词汇) 上同时出现的权重越高。

2. 右边 $\|t_1\| \|t_2\| \cos \theta$:

- $\|t_1\| \times \|t_2\|$ 是如果两个向量“完全同向”($\theta = 0$), 那么它们的内积就等于长度乘积。
- 乘上 $\cos(\theta)$ 后, 当方向不完全一致时, 内积会相应缩小: 夹角越大 (方向越不相同), $\cos(\theta)$ 越小, 内积也越小。

余弦相似度 (Cosine Similarity)

实际中我们往往关心规范化后的“方向相似度”:

$$\text{cosine_sim}(t_1, t_2) = \frac{\langle t_1 | t_2 \rangle}{\|t_1\| \|t_2\|} = \cos(\theta).$$

- 分子:** 内积, 反映“同现”强度。
- 分母:** 两个向量长度的乘积, 把内积归一化, 去掉文档长度的影响, 得到纯粹的方向一致性。

当 $\cos(\theta)$ 越接近 1, 说明两篇文档的词分布方向越一致; 越接近 0, 则说明主题/词汇几乎不重叠。

欧几里得范数 (也称 L_2 范数) 就是把向量看成几何空间中的一条从原点到点 (v_1, v_2, \dots, v_n) 的“箭头”, 它的长度就由勾股定理得到——各分量平方和再开方:

$$\|v\|_2 = \sqrt{v_1^2 + v_2^2 + \dots + v_n^2}.$$

步骤

1. 取各坐标的平方

如果向量是 $v = (v_1, v_2, \dots, v_n)$, 先计算 $v_1^2, v_2^2, \dots, v_n^2$ 。

2. 求和

把这些平方值加起来: $S = v_1^2 + v_2^2 + \dots + v_n^2$ 。

3. 开方

最后 $\|v\|_2 = \sqrt{S}$ 。

示例

- 二维向量** $v = (2, 3)$:

$$\|v\|_2 = \sqrt{2^2 + 3^2} = \sqrt{4 + 9} = \sqrt{13} \approx 3.606.$$

- 三维向量** $w = (1, -2, 2)$:

$$\|w\|_2 = \sqrt{1^2 + (-2)^2 + 2^2} = \sqrt{1 + 4 + 4} = \sqrt{9} = 3.$$

Autres normalisations : versions booléennes

Si les coordonnées des vecteurs indiquent la présence d'un mot plutôt que sa fréquence, on a

- Mesure de Sørensen–Dice

$$2 \times \frac{\#(t_1 \cap t_2)}{\#t_1 + \#t_2}$$

- Mesure de Jaccard

$$\frac{\#(t_1 \cap t_2)}{\#(t_1 \cup t_2)}$$

- Coefficient de recouvrement

$$\frac{\#(t_1 \cap t_2)}{\min(\#t_1, \#t_2)}$$

Dans ce cas, ces mesures sont bien comprises entre 0 et 1.

1. Sørensen–Dice 系数

$$\text{Dice}(t_1, t_2) = \frac{2 |t_1 \cap t_2|}{|t_1| + |t_2|}$$

- $|t_i|$ 表示文档 t_i 中出现的不同词数（即向量中 1 的个数）。
- $|t_1 \cap t_2|$ 是两篇文档都出现过的词数。
- 乘以 2 是为了平衡分母，使得如果两篇文档的词集合完全相同，结果恰好是 1；完全不相交时是 0。

2. Jaccard 系数

$$\text{Jaccard}(t_1, t_2) = \frac{|t_1 \cap t_2|}{|t_1 \cup t_2|}$$

- 分母 $|t_1 \cup t_2|$ 是两篇文档出现过的所有不重复词的总数。
- 当两个集合完全相同时， $|t_1 \cap t_2| = |t_1 \cup t_2|$ ，结果为 1；不相交时为 0。

3. 覆盖系数（Overlap Coefficient）

$$\text{Overlap}(t_1, t_2) = \frac{|t_1 \cap t_2|}{\min(|t_1|, |t_2|)}$$

- 分母取两个文档中较小的词数。
- 如果较小集合完全包含于较大集合，则分子 = 分母，结果为 1；否则小于 1。

为什么用这些“布尔”版本？

- 当只关心“有没有出现”而不是“出现了几次”时，用二值化更简单。
- 这些指标都在 $[0, 1]$ 之间，直观易懂：0 表示完全不相似，1 表示完全相同（或完全包含）。
- 不同指标对文档长度和集合大小的敏感度略有差别，可根据任务选择最合适的度量。

各种距离：

1. 曼哈顿距离 (Manhattan Distance, L_1)

定义

对于两个 n 维向量

$$\mathbf{t}_1 = (v_{1,1}, v_{1,2}, \dots, v_{1,n}), \quad \mathbf{t}_2 = (v_{2,1}, v_{2,2}, \dots, v_{2,n}),$$

它们的曼哈顿距离（也称出租车距离、Taxicab distance）定义为

$$d_1(\mathbf{t}_1, \mathbf{t}_2) = \sum_{k=1}^n |v_{2,k} - v_{1,k}|$$

- 含义：在每个坐标轴上分别计算绝对差，再求和。

- 符号解释：


- n ：向量维度。
- k ：坐标索引（从 1 到 n ）。
- $v_{i,k}$ ：向量 \mathbf{t}_i 在第 k 维的数值。
- $|v_{2,k} - v_{1,k}|$ ：第 k 维的绝对差。

- 示例（二维）

令 $\mathbf{t}_1 = (3, 1)$, $\mathbf{t}_2 = (1, 2)$ ，则

$$d_1(\mathbf{t}_1, \mathbf{t}_2) = |1 - 3| + |2 - 1| = 2 + 1 = 3.$$

- 应用场景：

- 网格城市（如出租车、机器人）行驶路径长度估算。
- 离散特征（如整数计数、分类指标）的相似度量。
- 在 k -NN、聚类算法中对稀疏向量（如词频）常用 。

2. 欧氏距离 (Euclidean Distance, L_2)

定义

同样对于 $\mathbf{t}_1, \mathbf{t}_2$ ，欧氏距离为

$$d_2(\mathbf{t}_1, \mathbf{t}_2) = \sqrt{\sum_{k=1}^n (v_{2,k} - v_{1,k})^2}.$$

- 含义：空间中两点的“直线”最短距离。


- 符号解释：

- $(v_{2,k} - v_{1,k})^2$ ：第 k 维差的平方。
- 外层开根号 $\sqrt{\dots}$ 还原到“长度”量纲。

- 示例（二维）

$$d_2(\mathbf{t}_1, \mathbf{t}_2) = \sqrt{(1 - 3)^2 + (2 - 1)^2} = \sqrt{4 + 1} = \sqrt{5} \approx 2.236.$$

- 应用场景：

- 经典的空间几何场合（点、向量长度计算）。
- k -means 聚类、PCA、MDS 等需度量真实空间距离的算法。
- 在连续特征空间中衡量相似度时最直观 .

3. 切比雪夫距离 (Chebyshev Distance, L_∞)

定义

$$d_\infty(\mathbf{t}_1, \mathbf{t}_2) = \max_{1 \leq k \leq n} |v_{2,k} - v_{1,k}|.$$

- 含义：两点在每个坐标轴上差值的最大者，也称棋盘距离 (King's move distance)。
- 符号解释：
 - $\max\{\dots\}$ ：取所有坐标绝对差中的最大值。
- 示例 (二维)

$$d_\infty(\mathbf{t}_1, \mathbf{t}_2) = \max(|1 - 3|, |2 - 1|) = \max(2, 1) = 2.$$

- 应用场景：
 - 当“最差情况”决定整体相似度（或距离）时使用。
 - 棋盘游戏中“王”行走步数——在一步内可纵横或斜向移动。
 - 在质量控制、容差分析等需关注**最大偏差**的领域。

4. Minkowski 距离 (Minkowski Distance)

定义

对任意实数 $p \geq 1$, Minkowski 距离定义为

$$d_p(\mathbf{t}_1, \mathbf{t}_2) = \left(\sum_{k=1}^n |v_{2,k} - v_{1,k}|^p \right)^{1/p}.$$

- 含义：一个统一的距离族，调整参数 p 可以在 L_1 (曼哈顿) 和 L_2 (欧氏) 之间平滑过渡，并在 $p \rightarrow \infty$ 时收敛至 L_∞ (切比雪夫)。
- 特殊情形：
 - $p = 1 \rightarrow L_1$;
 - $p = 2 \rightarrow L_2$;
 - $p \rightarrow \infty \rightarrow L_\infty$ 。
- 示例 (二维, $p = 3$)

$$d_3(\mathbf{t}_1, \mathbf{t}_2) = (|1 - 3|^3 + |2 - 1|^3)^{1/3} = (2^3 + 1^3)^{1/3} = (8 + 1)^{1/3} \approx 9^{1/3} \approx 2.08.$$

- 应用场景：
 - 需要灵活平衡“总差”和“最大差”影响时可调节 p 。
 - 在某些模式识别、信号处理与机器学习文献中可选用特定 p 值以满足对不同噪声敏感度的需求。

何时选用哪种距离？

距离类型	特点	典型应用
L_1 (曼哈顿)	强调各维度差异的绝对和，对离群值不如 L_2 敏感	网格路径、稀疏高维 (文本、计数特征)
L_2 (欧氏)	几何直观，强调大差异平方放大	连续空间聚类 (k-means)、降维 (PCA/MDS)
L_∞ (切比雪夫)	只看最“坏”那一维的差异	容差分析、棋盘距离、最大偏差决策
L_p (Minkowski)	可调参数 p 跨越 L_1 - L_2 - L_∞ 族	需要对差异敏感度做平滑调节的场景

P7 --- classification et apprentissage

Classification (分类)

定义与任务

分类是将给定数据划分到预先定义好的若干类别中的任务。每个输入实例必须恰好属于一个类别。典型应用包括垃圾邮件检测、情感极性分类、文本体裁识别等。

工作流程

1. **输入**：一组带标签的训练数据。
2. **训练**：构建分类模型。
3. **预测**：使用模型对未知标签的新实例进行类别预测。

2. Classification par règles expertes (基于规则的分类)

思路

专家手动编写规则，将实例映射到类别。例如，为情感分析制定词典：每个词赋予 +1（正面）或 -1（负面），文本得分为词极性之和，得分为正则判为正面，反之为负面。

优缺点

- **优点**：解释性强，规则透明。
- **缺点**：难以穷尽所有用例，规则维护困难，规则间可能冲突，需要领域专家耗时编写。

3. Classification par apprentissage artificiel (基于学习的分类)

概念

利用已标注样本自动学习模型，将机器“训练”成能够对新样本做出分类决策的系统。

流程示意

- 样本集 → 学习程序 → 模型 → 分类程序 → 预测结果

关键要素

1. **模型空间**：候选函数族（如线性函数、多项式）。
2. **评价度量**：衡量模型与数据拟合优度（如损失函数）。
3. **优化算法**：求解最优参数（如梯度下降、最小二乘）。

4. Algorithme de la classe majoritaire (ZeroR) (多数类算法)

原理

总是预测训练集中出现频率最高的类别，不考虑输入特征。

- 在 Weka 中对应 rules > ZeroR。

特点

- **训练/预测极快**，无超参数。
- **性能差**，仅作为基线（baseline），若数据类别高度不平衡，准确率可略优于随机。

5. k-plus proches voisins (k-NN) (k 最近邻算法)

原理

对新样本，找训练集中距离最近的 k 个实例，输出它们的多数类标签。

- 距离度量可用欧氏距离或重叠系数（布尔向量）。
- 无显式训练阶段，延迟学习（lazy learning）。

超参数

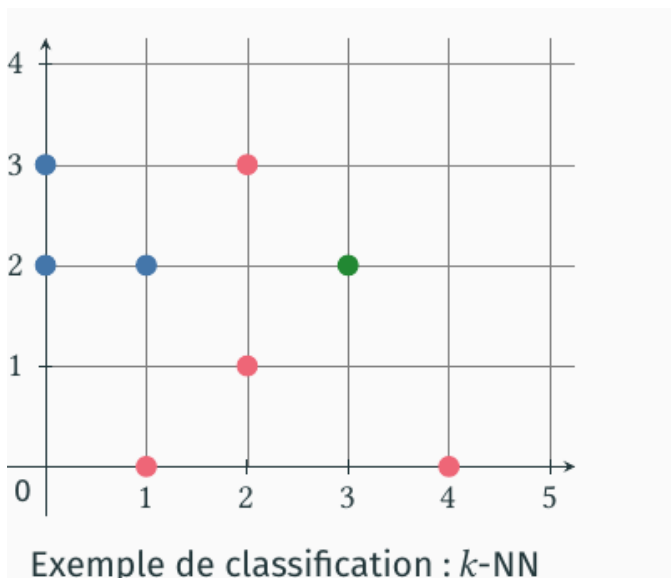
- **k**：邻居数量，通常小于训练集大小，且非类别数的倍数。

优缺点

- **优点**：实现简单，无模型训练；
- **缺点**：预测阶段代价高（需计算所有距离），对稀疏、高维数据效果差。

2. 二维示例 (第二张图)

- 坐标系中的点：
 - 蓝点 (●) 和 红点 (●) 是训练好的已知类别样本。
 - 绿色点 (●) 是我们需要分类的新输入 x 。
- 距离计算
 - 例如，对绿色点 (1,1)，计算它到每个现有点的欧氏距离：
 - 到蓝点 (1,2) 距离 = 1
 - 到红点 (2,1) 距离 = 1
 - 到蓝点 (0,2) 距离 = $\sqrt{2} \approx 1.414$
 - ...
- 取最近的 k 个邻居
 - 假如 $k=3$ ，那么与 (1,1) 最近的三个点分别是
 - (1,2) → 蓝
 - (2,1) → 红
 - (0,2) → 蓝
 - 其中有 2 个蓝、1 个红，**多数投票得“蓝”**，于是把绿色点分类为 蓝。



6. Arbres de décision (决策树)

模型结构

以树形结构表示分类规则：

- 内部节点：测试属性及其取值；
- 分支：对应属性的不同取值（或阈值分支）；
- 叶节点：类别标签。

决策过程

从根节点出发，根据实例属性值沿相应分支递归，直到到达叶节点并输出类别。

7. Indice de diversité de Gini (基尼指数)

定义

对二分类情况， $Gini(S) = \sum_{i=1}^2 p_i(1-p_i)$ ，其中 p_i 为类 i 在 S 中的比例。衡量划分纯度：值越大表示越能均匀分割两类。

用途

在属性选择时，计算划分前后 Gini 的减少量（Gini 增益），选择增益最大的属性。

8. Entropie (熵)

定义

$H(S) = -\sum_{1 \leq i \leq n} p_i \log_2(p_i)$, p_i 为类 i 在 S 中的比例。熵越高, 数据分布越不确定。

用途

衡量划分后的不确定性, 用信息增益 (Entropy 减去条件熵) 进行属性选择。

9. Choix des attributs – Gain d'information/Gini 增益 (属性选择)

计算公式

- Gini 增益 $g(S,a) = \text{Gini}(S) - \sum_{v \in \text{Vals}(a)} (|S\{a=v\}|/|S|) \cdot \text{Gini}(S_{\{a=v\}})$
 - 信息增益 $IG(S,a) = H(S) - \sum_{v \in \text{Vals}(a)} (|S\{a=v\}|/|S|) \cdot H(S_{\{a=v\}})$
- 选择增益最大的属性分裂, 提高树的判别能力。

10. Attributs à valeurs scalaires – Seuils (数值属性: 阈值划分)

方法

将连续属性 a 转化为若干布尔属性 $a \leq r$, 通过尝试不同阈值 r , 计算对应增益, 选择最佳阈值进行分裂。

示意

在属性轴上测试各候选点作为分界, 评估分裂后的纯度提升。

11. Propriétés des arbres de décision (决策树的性质)

- 可解释性强 (White box): 易于理解和可视化;
- 小模型: 对小规模数据表现良好;
- 易组合: 可与 Bagging、Boosting 结合 (随机森林等);
- 缺点: 对数据微小波动敏感, 易过拟合, 树结构不稳定。

12. Hyperplans (超平面)

定义

在 n 维空间中, 维度为 $n-1$ 的几何体称为超平面。例如:

- 1D: 点; 2D: 直线; 3D: 平面。

分割空间

超平面将空间分为两个半空间, 用于线性分类判别。

13. Classifieurs linéaires (线性分类器)

原理

寻找超平面将训练样本划分为两个类别, 使各类别样本落入超平面对应的半空间。

问题

1. 是否线性可分?
2. 若可分, 选择哪一个分割超平面?

14. Notion de marge (间隔)

定义

对于超平面 H , 两类样本到 H 的最小距离之和称为间隔 $M(H)$ 。通常选择最大化间隔的超平面以提高泛化能力。

15. Vecteurs de support (支持向量)

定义

使两类样本到超平面距离达到最小值的那些样本点称为支持向量。模型解仅依赖这些点的位置。

16. Support Vector Machines (SVM) (支持向量机)

概念

通过二次规划或梯度方法，在所有超平面中寻找最大间隔分割超平面。Weka 中对应 functions > SMO。

17. Marge douce (Soft margin) (软间隔)

思路

在非线性可分情况下，引入惩罚项 $C(H)$ 对误分类样本罚分，使优化目标 $L(H)=M(H)-C(H)$ ，平衡间隔大小与分类错误。

18. Kernel trick (核技巧)

思路

通过核函数将输入映射到高维空间，使非线性可分问题在映射后变为线性可分，避免显式计算高维映射。常用核：多项式核、高斯核等。

19. Propriétés des SVM (SVM 的性质)

- 优点：泛化性能优，一旦训练完成，预测速度快，内存占用低；
- 缺点：模型不易解释，核选择需经验，传统 SVM 不支持增量学习。

P11 -1 chaines_de_markov

Idée générale : 马尔可夫链表示一个离散时间中随机变量 une chaîne de Markov2 représente l'évolution dans un temps discret d'une variable aléatoire (ici : caractères, mots, etc..), où l'état suivant se prédit uniquement sachant l'état qui le précède.其中下一个状态的预测仅依赖于前一个状态

Chaîne de Markov : processus de Markov, ici à temps discret.

Processus de Markov 马尔可夫过程 : processus stochastique qui vérifie la propriété de Markov.

Processus stochastique 随机过程: représente l'évolution dans le temps d'une variable aléatoire. Il vérifie la propriété de Markov si la prédiction d'un état futur ne se fait que selon l'état qui le précède.表示一个随机变量随时间演变的过程。如果它满足马尔可夫性质，即对未来状态的预测仅依赖于当前状态，而与过去的状态无关，那么它就是一个马尔可夫过程。

Un modèle de langue est un modèle mathématique qui modélise la probabilité qu'un mot apparaisse étant donné un contexte. Traditionnellement, on cherche à modéliser le mot suivant étant donné les mots précédents (ex : GPT-34), même si des modèles peuvent utiliser des contextes étendus comme l'ensemble des mots alentours (ex : BERT5). Les chaînes de Markov prennent en contexte n mots, on les appelle également des modèles de langue **n-grammes**.

N-grammes : 文本中连续的 n 个词或字符

Un n-gramme est une sous-séquence de n éléments construite à partir d'une séquence donnée. L'idée semble provenir des travaux de [Claude Shannon](#) en [théorie de l'information](#).

Plus formellement, un modèle de langue basé sur les n-grammes estime la probabilité conditionnelle :

$$P(w_n \mid w_{n-1}, w_{n-2}, \dots, w_1)$$

en supposant souvent une hypothèse de Markov d'ordre (n-1), c'est-à-dire que :

$$P(w_n \mid w_{n-1}, \dots, w_1) \approx P(w_n \mid w_{n-1}, \dots, w_{n-k})$$

pour un k fixé (souvent k = n-1).

我们只看 "Bob aime les" 后面出现的词: chiens: 出现了 2 次 glaces: 出现了 1 次 Schtroumpfs: 出现了 1 次. 所以总共 4 次, 我们就可以计算条件概率: $P(X = \text{chiens} \mid \text{"Bob aime les"}) = 2 / 4 = 0.5$ $P(X = \text{glaces} \mid \text{"Bob aime les"}) = 1 / 4 = 0.25$ $P(X = \text{Schtroumpfs} \mid \text{"Bob aime les"}) = 1 / 4 = 0.25$ Quand on génère un texte, faudrait-il plutôt que dire : • "Alice n'en veut point", ou "Alice n'en veut pas" ? On peut comparer les probabilités des n-grammes pour décider.

句子: "Bob aime les chiens"

我们用 4-gram (= 三阶马尔可夫模型):

- 我们要预测下一个词, 比如给定 "Bob aime les", 我们想知道下一个词是 "chiens" 的概率是多少。
- 那就去查所有以 "Bob aime les" 开头的句子, 数后面跟什么词。
- 最后得出条件概率:

$$P(X \mid \text{"Bob aime les"}) = \frac{\text{出现次数 (Bob aime les X)}}{\text{"Bob aime les" 出现的总次数}}$$

所以它是个基于统计的马尔可夫链语言模型。

p2-11-réseaux_neurones

感知机 (Perceptron) : 笔记里有, 4, 4>

二分类任务 如垃圾邮件识别 (垃圾 vs. 非垃圾)、病症诊断 (病人 vs. 健康)、图像中白天/夜晚检测等。

2. 特征预处理 3. 归一化: 将各维特征缩放到同一量级 (如 [0,1] 或服从标准正态), 有助于算法收敛

4. 决策边界可视化 (二维示例)

对于二维输入 (x1, x2), 分界线方程: $w_1x_1 + w_2x_2 + b = 0 \implies x_2 = -(w_1/w_2) \cdot x_1 - b/w_2$.

扩展: 多层感知机 (MLP) : 单层 (原始) 感知机只能解决线性可分问题; 当数据线性不可分时, 可堆叠多层神经元并改用可微激活函数 (如 Sigmoid、ReLU), 构建深度神经网络。

La fonction de Heaviside H est l'activation classique pour faire du perceptron un classifieur binaire : elle transforme les nombres négatifs en 0 et les nombres positifs en 1.

Heaviside 函数 $H(z)$ 的数学表达式为:

$$H(z) = \begin{cases} 1, & z \geq 0 \\ 0, & z < 0 \end{cases}$$

1. 计算加权和 $z = w^T x = 2 \cdot (-2) + (-1) \cdot (-1) + 0 \cdot 5 + 1 \cdot 4 = -4 + 1 + 0 + 4 = 1$.

- 输入: 加权和 $z = w^T x + b$.

2. 应用激活函数

对于“简单感知机”通常用阶跃 (Heaviside) 函数:

- 输出: 二值信号, 0 表示“不开火”(不激活), 1 表示

$$y = H(z) = \begin{cases} 1, & z \geq 0 \\ 0 \text{ (或 } -1), & z < 0 \end{cases}$$

注: 在感知机中, 为了对称, 有时会将输出定义为 $\text{sign}(z) = +1$ 当 $z \geq 0$, -1 当 $z < 0$.

这里 $z = 1 \geq 0$, 所以输出 $y = 1$.

结论: 这个题目就是让你把给定的权重向量 w 和输入向量 x 做点积, 然后用阶跃激活判断正负, 最后给出二分类的输出值——本例中答案是

$$y = 1$$

On considère un perceptron simple dont les poids sont

$$w = \begin{pmatrix} 2 \\ -1 \\ 0 \\ 1 \end{pmatrix}$$

Donner la sortie y du neurone pour l'entrée

$$x = \begin{pmatrix} -2 \\ -1 \\ 5 \\ 4 \end{pmatrix}.$$

翻译成中文就是：

“我们有一个最简单的感知机（Perceptron），它的权重向量是 $w = (2, -1, 0, 1)^T$ 。将所有权重 w_i 设为 0

求当输入向量为 $x = (-2, -1, 5, 4)^T$ 时，神经元的输出 y 是多少？”

感知机的训练算法：Le perceptron s'entraîne en parcourant l'ensemble d'entraînement point par point.

1. On commence en initialisant tous les poids w_i à 0 (ou à un petit nombre aléatoire).

2. Pour chaque exemple x dont la classe est

y

Calculer la classe observée 对每一个带标签的样本 (x, y) （这里 y 是它的真实类别，通常取值为 0/1 或 +1/-1）：

1. Calculer la classe observée $\hat{y} = H(\sum w_i x_i)$
2. • Si $y = \hat{y}$, on ne change rien
• Sinon, on met à jour les poids w_i selon la règle

$$w_i \leftarrow w_i + \alpha(y - \hat{y})x_i$$

1. 计算感知机的输出（预测类别

如果预测 \hat{y} 和真实 y 相同，就不做任何操作。

如果预测错误，就按下面的规则更新每个权重：这里 $\alpha > 0$ 是学习率（taux d'apprentissage），控制每次更新的步长

当把整个训练集都“扫”完一次，就完成了—一个训练周期（epoch）可多次重复直到模型稳定。

“画布”是二维平面 横轴是第一个特征 x_1 ，纵轴是第二个特征 x_2 。每个点 (x_1, x_2) 就是一个样本。

2. 红色和蓝色区域：模型“判”出的类别 **红色区域**：感知机当前把它们都判作“类 0”。**蓝色区域**：感知机当前把它们都判作“类 1”。这两块“颜色区”由一条直线分开，这条直线就是感知机的**决策边界**（方程是 $w_1 x_1 + w_2 x_2 + b = 0$ ）。

3 算法里的两个向量： **w （黑色粗箭头）**：当前模型的权重向量，指向“蓝色”（正类）那一侧，方向垂直于决策边界。 **x （细黑箭头）**：从原点指向某个样本点的箭头，就是我们要看的训练样本。

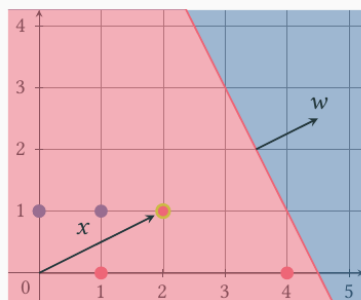
4. 已知的训练样本（散点）：**紫色实心点**：真实标签是 1（正类）的样本。**红色实心点**：真实标签是 0（负类）的样本。

5. 找到一个“误分类”的点：图中用**黄色圈**高亮的那个 **空心紫点**（大约在 (2,1)）——它真实是“类 1”，但因为它落在红色区域，模型当前却判成“类 0”，属于**错误分类**。

6. 更新权重的“借力点”感知机算法：每次挑一个被错分的样本 x ，用它来“纠正”权重。更新公式（学习率 $\alpha = 0.5$ ）： $w \leftarrow w + \alpha(t - y)x$ 这里 $t = 1$ （真实类）， $y = 0$ （模型判的类），所以就是： $w \leftarrow w$

$+ 0.5 * 1 * x$ 也就是说，新的 w 会往这个样本 x 的方向“偏移”一点，决策边界也随之向右/向上移动，下一轮就能把这个点(2,1)划进蓝色区了。

En deux dimensions, avec les classes 1 et 0, avec un taux d'apprentissage $\alpha = 0.5$.



Algorithme du perceptron

序列标注：给定一个词序列 $X=(x_1,x_2,...,x_n)$ ，为每个词 x_t 预测一个标签 y_t （如词性、命名实体类型等）。词性标注（POS Tagging）、命名实体识别（NER）、分块（Chunking）等。

CoNLL 格式：

```
Le    DET
petit  ADJ
chat   N
est    V
content ADJ

Vive   V
le     DET
TAL    N
```

- Un mot par ligne
- Colonnes séparées par des tabulateurs
- Séquences séparées par des lignes vides

Est-ce un format tabulaire au sens habituel ?

从学习（机器学习）角度看，“考虑上下文”就是把上下文中的词当作特征（features）来使用。

在表格（tabulaire）格式里，我们把上下文依赖组织成这样的表格：

[-1]	[0]	[+1]	标签 (label)
<s>	Le	petit	DET
Le	petit	chat	ADJ

On a vu qu'en utilisant des patrons, on pouvait ramener la structure des entrée (les mots) à des features.

	[0, 0]	[-1, 0]	[1, 0]	label
•	petit	Le	chat	ADJ
•	Le	<s>	petit	DET
•	content	chat	</s>	ADJ
	chat	petit	est	N

À partir de là, on peut simplement prédire les étiquettes avec un classifieur.

[0]就是当前位置, [-1] 是[0]之前的词语, 标签是[0]的标签

偏移 0: 就是当前行, 也就是“要打标签的那个词”本身

偏移 -1: 往上移一行, 取前一个词。

[-1, 0] 取第 0 列+偏移 -1 → “Le”

偏移 +1: 往下移一行, 取下一个词。

这样, 就能把当前词和它左右两边的上下文都当作特征, 一起输入到分类器里去预测 “petit”这个词的词性标签。

朴素贝叶斯分类器 = 用贝叶斯定理来做分类的算法

朴素贝叶斯公式（简化）： $P(\text{类} | \text{文本}) \propto P(\text{类}) \times P(\text{词} 1 | \text{类}) \times P(\text{词} 2 | \text{类}) \times \dots \times P(\text{词} n | \text{类})$

所以朴素贝叶斯就是把所有概率最大化：关于 $P(A|B)$ 和 $P(B|A)$ 所表达的意思：

X 事件为选中了红球： $p(X) = 8/20$

Y 事件代表选中 A 容器： $p(Y) = 1/2$

AB 两个容器, 分别装有红球和白球

A 容器中红球的概率： $p(X|Y) = 7/10$ 即: 红球且在 A 容器里

A 容器且包含红球 的概率: $p(Y|X) = p(X|Y) * p(Y)/p(X)$

所以是, 先满足左边条件, 再满足右边条件

拉普拉斯平滑（Laplace smoothing）是一种防止概率为零的技巧。

在 Naive Bayes 分类器中，我们会算某个词 w 在某个类 C 中的条件概率：

$P(w | C)$ =类 C 中包含词 w 的文档数 / 类 C 中的总文档数

问题：如果词没出现过怎么办？

举个例子：你在 "fonds marins" 类的文档里从来没见过词 "lune" 那就： $P(\text{lune} | \text{fonds marins})=0$

拉普拉斯平滑的解决办法

我们给每个词的计数 加 1（即使它没出现过也加 1）：

$P(w | C)=(N_{w,C}+1) / (N_C)+2$

$N_{w,C}$: 词 w 在类 C 中的出现次数 (或出现的文档数)

NC : 类 C 中文档总数 加 2 是因为词的取值只有 0 或 1 (伯努利模型)

---因为朴素贝叶斯忽略了 **标签之间** 的依赖, 比如 “DET 之后通常是 ADJ 或 NC”, HMM 就在模型里加上这条依赖:

HMM 隐马尔可夫模型: 在朴素贝叶斯基础上加上前后标签转移概率, 联合建模整条序列。

在 HMM (隐马尔可夫模型) 里, 我们关心的是「观测序列」 $X_{1:n}=x_1,\dots,x_n$ 和「状态 (标签) 序列」 $Y_{1:n}=y_1,\dots,y_n$ 同时出现的概率, 也就是**联合概率**: **HMM 的联合概率就是 “朴素贝叶斯 + 标签转移” 两部分相乘的结果。**

$$P(X_{1:n}, Y_{1:n}) = P(Y_1) \underbrace{\prod_{i=2}^n P(Y_i | Y_{i-1})}_{\text{转移概率}} \times \prod_{i=1}^n P(X_i | Y_i).$$

初始概率 $P(Y_1)$: 标签序列第一个状态是 y_1 的概率。

转移概率 $P(Y_i | Y_{i-1})$: 标签从 y_{i-1} 变为 y_i 的概率,

刻画标签之间的 Markov 依赖。

发射概率 $P(X_i | Y_i)$: 在标签为 y_i 时, 生成 (观测到) 词 x_i 的概率。

把它们连乘, 就得到了「这个标签序列生成这段词的总体概率」。

怎么计算

参数估计 (训练)

- 从标注语料中统计:

- $P(Y_1 = y) \approx \frac{\text{以 } y \text{ 开头的句子数}}{\text{总句子数}}$
- $P(Y_i = y' | Y_{i-1} = y) \approx \frac{\text{出现 } (y, y') \text{ 转移的次数}}{\text{出现 } y \text{ 的次数}}$
- $P(X_i = x | Y_i = y) \approx \frac{\text{标签 } y \text{ 下生成词 } x \text{ 的次数}}{\text{标签 } y \text{ 的总次数}}$

维特比算法: 单独打印

解码/预测时

- 对某条具体的观测序列 $x_{1:n}$, 我们要找最可能产生它的标签序列:

$$\hat{y}_{1:n} = \arg \max_{y_{1:n}} P(x_{1:n}, y_{1:n}).$$

- 直接枚举太慢, 使用**维特比算法** (Viterbi) 做动态规划, 高效算出这个最大值以及对应的路径。