大模型之机器学习

# 机器学习的概念

目的：通过计算机系统逐步改善特定任务的性能

方法：海量数据 🡪 训练（函数系数或者参数的选择） 🡪 模型（数据表达或者概率表达） 🡪 预测

## 1.1 人工智能、机器学习、深度学习的关系



## 1.2 发展历史

### 早期探索： 20世纪50-70年代

符号学派尝试赋予机器逻辑推理能力；

图灵提出的“图灵测试”；

### 知识驱动与专家系统：20世纪70-80年代

逻辑推理 + 知识驱动；

专家系统、决策树（ID3算法）、贝叶斯定理的应用

### 数据驱动与统计学：20世纪80年代-21世纪

神经网络（连接主义学派）、统计学、互联网的大量数据；

统计学（支持向量机SVM）：决策树、随机森林

行为与环境交互（行为主义学派）

### 深度学习崛起：21世纪

深度学习：数据规模快速增长、GPU的发展、神经网络层数足够的多、CNN卷积神经网络、GAN生成对抗网络、Transformer

### 大模型与通用人工智能：2020年代

自然语言处理 多模态模型 自监督学习 强化学习和深度学习结合

## 1.3 机器学习的应用

图像处理与计算机视觉、自然语言处理、推荐系统、医疗与健康、金融科技、自动驾驶与智能交通；

## 1.4 基本术语

数据集：训练集（模型的训练数据）、测试集（性能的验证）、验证集（超参数的调整）；

样本、特征、特征向量（x）、标签(y)、参数、超参数、模型（数据表达式）

# 2. 机器学习基本理论

## 2.1 机器学习三要素

模型：总结数据的内在规律，用数学语言描述的参数 ；

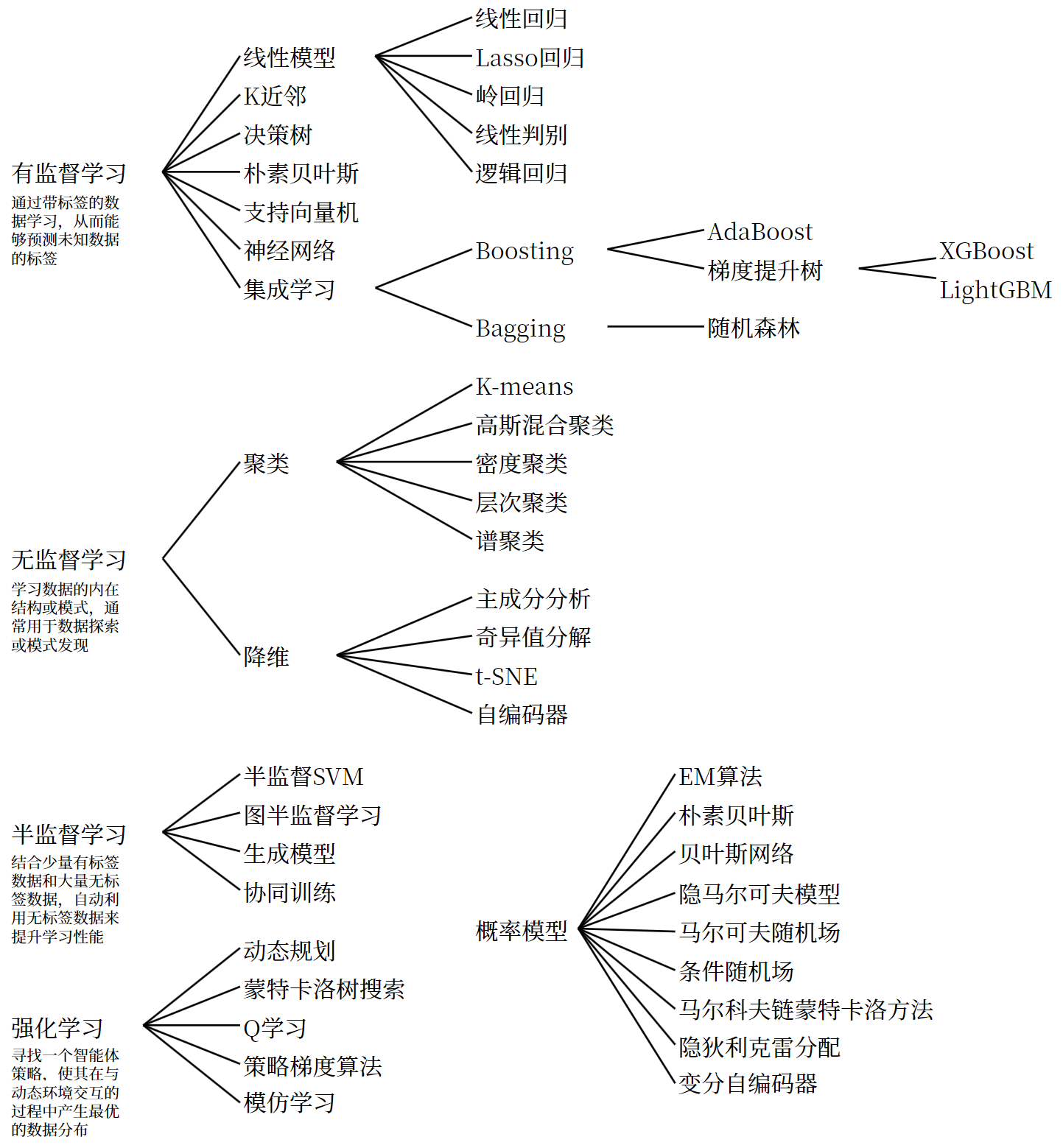
策略：选取最优模型的评价准则；

算法：选取最优模型的具体方法；

## 2.2 机器学习的分类

主要了分类：有监督学习、无监督学习、半监督学习 以及强化学习；

其他的分类：



大模型主要应用的算法是监督学习里的深度神经网络学习（解决问题的大一统的方法），即黑盒中不停地添加神经网络的层级来提高准确度；

## 2.3 建模流程

收集数据、清洗数据、特征工程、选择算法、训练模型、模型评估、模型优化、模型部署

图形用户界面, 应用程序

中度可信度描述已自动生成

## 2.4 特征工程

### 2.4.1 概念

1. 通过对原始数据的处理、转换、和构造，生成新的特征或者选择有效的特征，从而提高模型的性能；

2. 特征工程是一个迭代过程；

3. 本质是数据预处理的阶段；

### 2.4.2 内容

特征选择（减少特征）：

过滤法（低方差），相关系统）、包裹法？、嵌入法？

特征转换：

归一法（缩放到特定范围）、标准化（减去均值并除以方差）、对数变换、类别变量的编码（独热法（二进制编码））；

特征构造：

交互特征、统计特征、日期和时间特征；

特征降维（减少特征）：

低方差过滤法、相关系数法、主成分分析、线性判别分析、t-SNE、自编码器；

### 2.4.3 常用方法

特征选择、特征降维（维度灾难）

#### 低方差过滤

1. 相关代码

F:\MyStudy\LLM\my\_codes\my\_study\_llm\_training\machine\_learning\feature\1\_variance\_filter.ipynb 搜索 “=低方差过滤法=”

1. 该特征方差越小，就越不选择；

#### 相关系数法

##### 皮尔逊相关系数

1. 公式

皮尔逊相关系数（Pearson Correlation）用于衡量两个变量的线性相关性，取值范围。

正相关：值接近1，说明特征随目标变量增加而增加。

负相关：值接近-1，说明特征随目标变量增加而减少。

无关：值接近0，说明特征和目标变量无明显关系。

注：

1. 重点关注的是衡量两个变量的线性相关性
2. 相关代码

F:\MyStudy\LLM\my\_codes\my\_study\_llm\_training\machine\_learning\feature\2\_pearson\_filter.ipynb

##### 斯皮尔曼相关系数

1. 公式

斯皮尔曼相关系数（Spearman’s Rank Correlation Coefficient）的定义是等级变量之间的皮尔逊相关系数。用于衡量两个变量之间的单调关系，即当一个变量增加时，另一个变量是否总是增加或减少（不要求是线性关系）。适用于非线性关系或数据不符合正态分布的情况。

其中：

是两个变量的等级之差

是样本数

斯皮尔曼相关系数的取值范围为：

注：

1. 等级也就是排序;
2. 重点是等级变量之间的皮尔曼祥光系数；

相关代码

F:\MyStudy\LLM\my\_codes\my\_study\_llm\_training\machine\_learning\feature\3\_spearman.ipynb

##### 主成分分析（PCA）

主成分分析（Principal Component Analysis，PCA）是一种常用的降维技术，通过线性变换将高维数据投影到低维空间，同时保留数据的主要变化模式。

图表, 散点图

描述已自动生成

使用sklearn.decomposition.PCA进行主成分分析。参数n\_components若为小数则表示保留多少比例的信息，为整数则表示保留多少个维度。

注：

1. 高维数据投影到低维空间；
2. sklearn.decomposition.PCA分析，一般会用剩余成分的百分比作为降维的结果，这里用参数n\_components作为演示，正太分布方差越小的会被降掉（方差贡献率）

相关代码

F:\MyStudy\LLM\my\_codes\my\_study\_llm\_training\machine\_learning\feature\4\_pca.ipynb

## 模型评估和模型选择（重点）

### 损失函数

概念：是关于模型参数的函数，用来度量预测偏差的程度；

公式（*L(Y,f(X))*）：

常见的损失函数有：

* + - * 1. 0-1损失函数

注：类比向量范数0

* + - * 1. 平方损失函数

注释：类比向量范数2

* + - * 1. 绝对损失函数

注：类比向量范数1

* + - * 1. 对数似然损失函数

注：概率计算是使用似然函数，取log相乘转相加，取负号为了求极小值

### 经验误差（经验风险）

模型f(X)在训练集上的平均误差（损失函数的平均误差）；

公式：

### 泛化误差

上述相同的公式，模型f(X)在测试集数据上的平均误差（（损失函数的平均误差））

### 经验风险最小化

当经验风险最小时，模型最优的策略被称为经验风险最小化（empirical risk minimization ERM）

### 欠拟合与过拟合

#### 概念

**拟合：**

找到数据的规律

**欠拟合：**

高偏差；

原因是：模型复杂度不够、特征不足、训练补充分、过强的正则化；

**过拟合：**

模型过于复杂，高方差；

原因：模型复杂度过高（函数复杂或者参数太多）、特征过多、训练过长；

解决办法：减少模型复杂度、增加训练数据、使用正则化（弱化特征）、交叉验证

**泛化能力：**

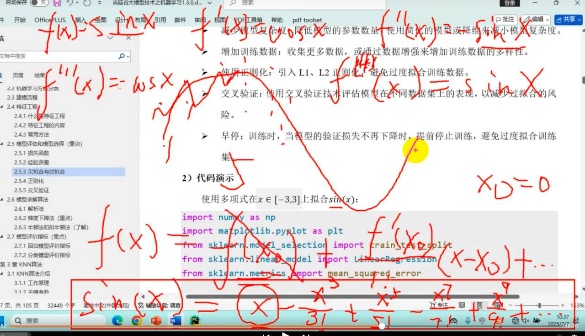
通用能力

图片包含 图示

描述已自动生成

#### 案例：

使用多项式在上拟合，如下使用的是泰勒展开，且用的是5阶的多项式即可



具体代码可看F:\MyStudy\LLM\my\_codes\my\_study\_llm\_training\machine\_learning\1\_fitting\_test.py

#### 注：

1. 泰勒展开 和多项式本质是想通过数学方式表达真是的世界规律，后期重点学习；
2. 核心过程：

数据集划分（训练/测试） 🡪 模型选择 🡪 数据集的特征工程 🡪 模型训练 🡪 模型预测（测试集）🡪 模型误差（测试误差/训练误差）

### 正则化

#### 概念

在损失函数中添加额外项，来惩罚过大的参数，进而限制模型复杂度、避免过拟合，提高模型的泛化能力，主要是针对于过拟合的模型做参数的惩罚；

如在平方损失函数中加入正则化项：

原损失函数 的目的：更好的拟合数据集；

正则化项的目的：减小参数的大小，从而降低模型的复杂度；

： 模型的参数；

是 ：**正则化系数，**用来表示惩罚项的权重。正则化系数不属于模型的参数，需要在模型训练开始之前手动设置，这种参数被称为“**超参数**”；

#### 正则化技术类型

##### L1正则化（Lasso回归）

L1正则化在损失函数中加入参数的绝对值之和

模型会自动“丢弃”一些不重要的特征，L1正则化有助于创建稀疏模型（即许多参数为0）。在解决回归问题时，使用L1正则化也被称为“Lasso回归”；

超参数控制着正则化的强度。较大的 值意味着强烈的正则化，会使模型更简单，可能导致欠拟合；

而较小的 值则会使模型更复杂，可能导致过拟合；

注：模型参数会被直接过滤掉

##### L2正则化（Ridge回归，岭回归）

L2正则化在损失函数中加入参数的平方之和：

L2正则化通过惩罚模型参数的平方，使得所有参数都变得更小，但不会将参数强行压缩为0。它会使得模型尽量平滑，从而防止过拟合。

在解决回归问题时，使用L2正则化也被称为“岭回归”。

注：

会尽量保留模型参数，工程上推荐应用

##### ElasticNet正则化(弹性网络回归)

ElasticNet正则化结合了L1和L2正则化，通过调整两个正则化项的比例来取得平衡，从而同时具备稀疏性和稳定性的优点

，决定L1和L2的权重

#### 案例代码

F:\MyStudy\LLM\my\_codes\my\_study\_llm\_training\machine\_learning\2\_regularization.py

### 交叉验证

#### 概念

是一种评估模型泛化能力的方法，通过将数据集划分为多个子集，反复进行训练和验证；

目的是为了更可靠的估计模型在未知数据上的表现，防止过拟合和欠拟合；

#### 方法

简单交叉验证、K折交叉验证（留一交叉验证的取样K次）、留一交叉验证（所有数据集）

### 总结

模型评估：损失函数、经验误差、泛化误差；

模型选择：正则化项、交叉验证；

最终结果的训练误差很小，不代表测试误差也很小，关注一下训练误差和训练误差与模型复杂度的曲线图；

## 模型的求解

增加了正则化项的损失函数最小，称为结构风险最小化：

这其实就是求解一个 **最优化问题**。代入训练集所有数据，要求最小值的目标函数就是模型中参数的函数；

注：

：损失函数；

：正则化项；

#### 梯度下降法（\*\*\*）

##### 概念

* 是一种迭代算法，主要是沿着梯度方向或者负梯度方向，不停的更新参数，最终取得极小值；

##### 公式

* 因为损失函数是系数的函数，那么如果系数沿着损失函数的负梯度方向变化，此时损失函数减少最快，能够以最快速度下降到极小值。
* θ\_(k+1)=θ\_k- α·▽L(θ\_k)
* ▽L(θ\_k) ：模型参数取值为〖 θ〗\_k 时损失函数 L 的梯；
* α ：每次迭代的“步长”，被称为“学习率”， 是一个常见的超参数；
* 特点
* 梯度下降不一定能够找到全局的最优解，有可能是一个局部最优解；
* 优缺点：计算简单，适用性广，但是收敛速度慢；可能陷入局部最优。

##### 分类

批量梯度下降法（数据两太大）、随机梯度下降法（随机一个样本）、小批量梯度下降法（微批次32或者64推荐）

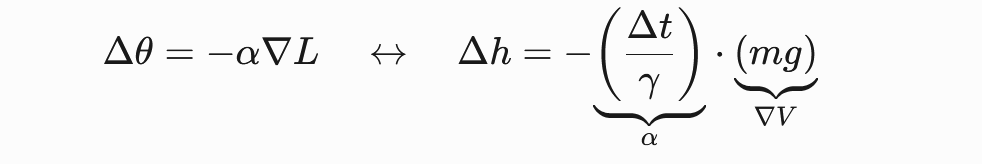
##### 案例代码

##### 总结

1. 梯度下降法最核心的是关于模型参数的损失函数θ的变更，即θ\_(k+1)=θ\_k- α·▽L(θ\_k)
2. θ的变更中减去随时函数在该点位置的导数 \* 学习率，可以类比物理含义理解为如下

想象一个小球在​**​粘稠蜂蜜​**​中下落：

* ​**​势能场​**​：重力势能 V(h) = mgh → 类比损失函数 L(θ)
* ​**​梯度​**​：∇V = mg (向下) → 类比 ∇L
* ​**​粘滞阻力​**​：f = -γv (阻力与速度成正比)
* ​**​终端速度​**​：当 mg = γv 时匀速下降 → ​**​v = (mg)/γ​**​
* ​**​位移更新​**​：Δh = -v·Δt = - (mg/γ) Δt
* 对应梯度下降：



注：α·▽L 可以更加形象的理解为，力作用后的距离 ，▽L是力，α是综合了时间步长 (Δt)、介质粘性 (γ)、质量 (m) 的复合参数；

1. 学习率α（梯度的步长）大小的影响如下

如果太大了，就会无法收敛，如果是太小了，计算的复杂度会大很多；

### 牛顿法和拟牛顿法（了解）

利用目标函数的二阶导数信息，通过迭代逐渐逼近极值点。

θ\_(k+1)=θ\_k- H^(-1) (θ\_k)·▽L(θ\_k)

这里的〖 H〗^(-1) (θ\_k)表示损失函数 L 黑塞矩阵的逆在点 θ\_k 的取值。

由于牛顿法中需要计算黑塞矩阵的逆〖 H〗^(-1) (θ\_k)，这一步比较复杂；所以可以考虑用一个n阶正定矩阵来近似代替它，这种方法称为“拟牛顿法”。

注：黑塞矩阵就是对函数求二阶导数得到的结果；

### 模型评价指标（重点）

主要是对学习的泛化性能进行评估，那么就需要有模型泛化能力的评价指标，也叫做性能度量；

#### 回归模型评价指标

对于回归问题，最常用的性能度量就是“均方误差”（Mean Squared Error MSE）

**平均绝对误差（MAE）**

对异常值不敏感，适用于数据包含异常值的场景

**均方误差（MSE）**

适用于惩罚大误差的场景

**均方根误差（RMSE）**

与MSE一致;

**R²（决定系数）**

衡量模型对目标变量的解释能力，越接近1越好，对异常值敏感。

个人理解：对目标变量的解释能力主要是再说越接近1 那与 ，越能体现目标变量的形态；

#### 分类模型评价指标

最常用的是“准确率”；

##### 混淆矩阵

混淆矩阵（Confusion Matrix）展示了模型预测结果与实际标签的对比情况

表格

描述已自动生成

具体代码

F:\MyStudy\LLM\my\_codes\my\_study\_llm\_training\machine\_learning\metrics\1\_classification\_test.ipynb 搜索“===混淆矩阵===”

##### 准确度 （侧重点：都为真例的，即包含真正例和真负例）

正确预测的比例（结合上述混淆矩阵来看）：真正例和真负例的总和占比

##### 精确度（侧重点：被预测为正例值中为真正例的比例）

被预测为正例值中为真正例的样本中实际为真正例的比例，也叫做查准率

##### 召回率（侧重点：真实值为正例）

真实值为正例的样本中预测值为正例的比例，也叫做查全率

##### F1分数

精确率和召回率的调和平均

具体的工程案例如下：

F:\MyStudy\LLM\my\_codes\my\_study\_llm\_training\machine\_learning\metrics\2\_classification\_report.py

注：逻辑回归底层算法其实就是线性回归+概率判断出来的离散值分布

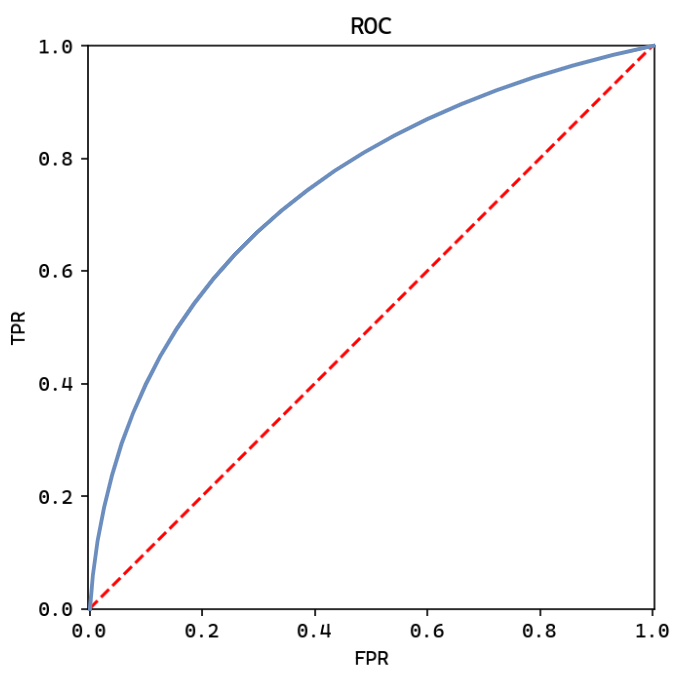
##### ROC曲线

真正例率（TPR）：实际为正例，被预测为正例的比例，即召回率。

假正例率（FPR）：实际为负例，被预测为正例的比例。

阈值（Threshold）：根据阈值将概率转换为类别标签。

ROC曲线（Receiver Operating Characteristic Curve，受试者工作特征）是评估二分类模型性能的工具，以假正例率（FPR）为横轴，以真正例率（TPR）为纵轴，展示不同阈值下模型的表现



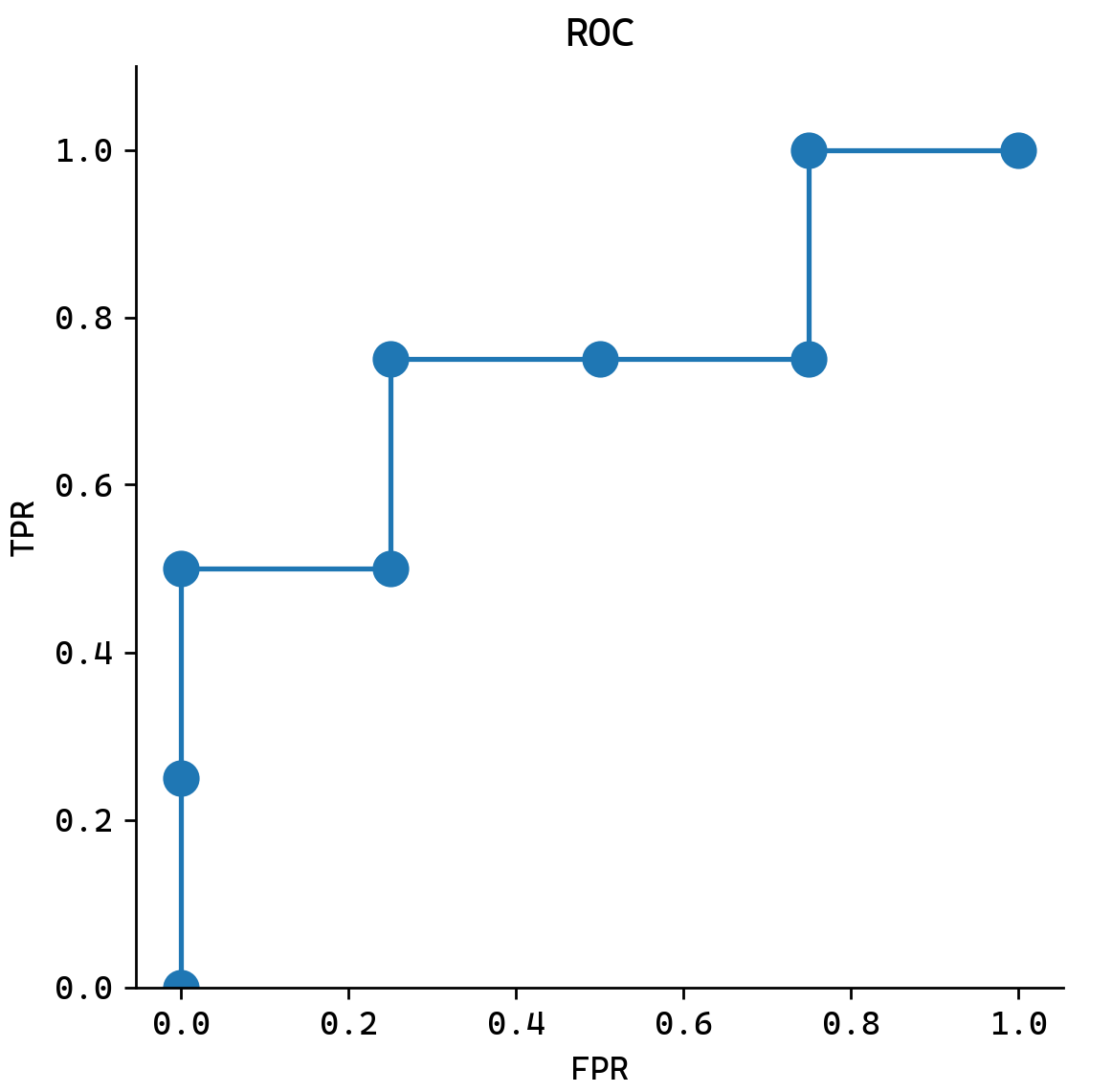
注：阈值是什么呢？看如下案例

案例：

假设一个二分类模型的真实标签和模型输出概率如下：

调整阈值，计算TPR和FPR：

根据坐标点绘制ROC曲线：



具体代码可看和AUC的结合使用

F:\MyStudy\LLM\my\_codes\my\_study\_llm\_training\machine\_learning\metrics\2\_classification\_report.py

##### AUC值

AUC值代表ROC曲线下的面积，用于量化模型性能。AUC值越大，模型区分正负类的能力越强，模型性能越好。AUC值=0.5表示模型接近随机猜测，AUC值=1代表完美模型。

**Roc和auc的案例代码如下**

**F:\MyStudy\LLM\my\_codes\my\_study\_llm\_training\machine\_learning\metrics\2\_classification\_report.py**