机器学习

巧用KNN分類模型

——工业轴承故障检测

KNN classification model is used to detect industrial bearing faults.

授课人：黄幪明

目录

CONTENTS

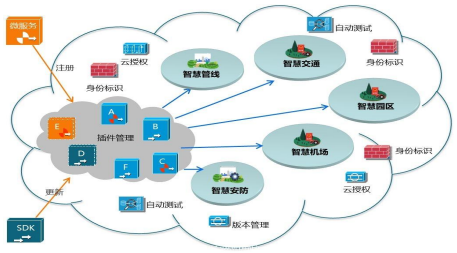
01

課程導讀 COURSE GUIDE

数据—新型生产要素

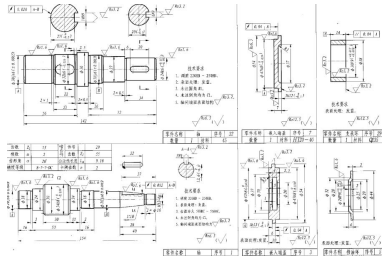
数字经济特征

在数字经济中，数据已经成为关键生产要素，云计算、大数据、人工智能已经成为数字经济的重要生产力，AI对生产力的提升，呈现了指数级效应。数据是数字经济的第一要素。

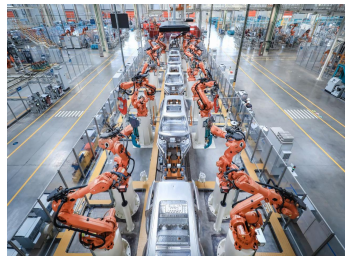


主体数据

行为数据



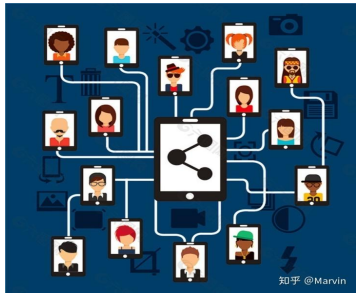
设计



生产

销售

交易数据



交往数据

融资、投资

售后

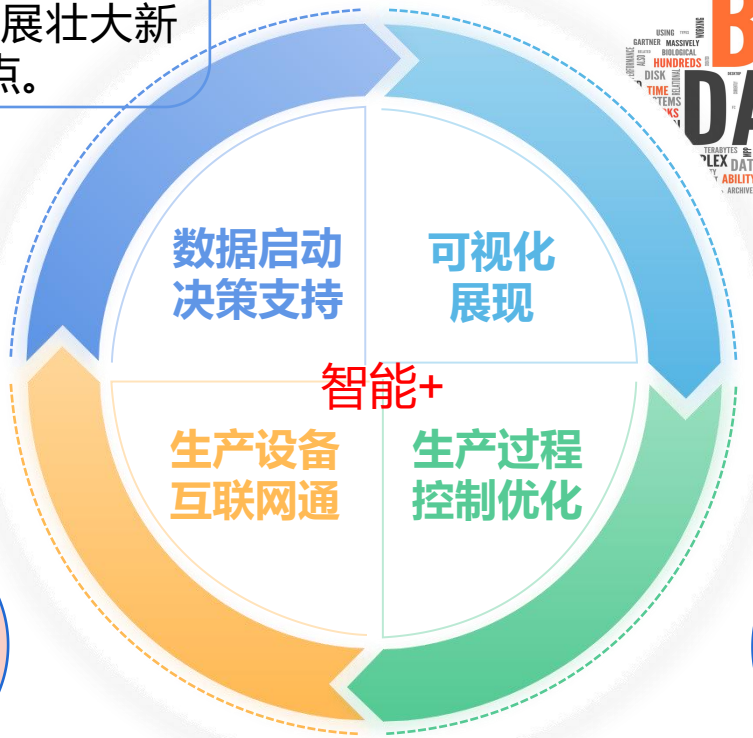
流通

大数据—智能制造新驱动

大数据——新一轮工业革命的地位

中国制造2025

以信息化与工业化深度融合为主线，重点促进云计算、物联网、大数据等新一代信息技术与现代制造业、生产性服务业等的融合创新，发展壮大新兴业态，打造新的产业增长点。



通过智能机器间的连接并最终将人机连接，结合大数据分析，重构全球工业、激发生产力的关键技术。

美国GE工业互联网

移动计算、社会化媒体、物联网大数据、分析和优化/预测是下一次工业革命中的关键技术。

德国工业4.0

AI驱动的智能智造

不断积累数据、知识和经验，产生数据驱动的AI模型工业决策大脑，变成竞争力，通过竞争力提升价值。

数据深度自感知
业务流程自学习
智慧优化自决策
精准控制自执行

四大功能

缩短产品研制周期
降低运营成本
提高生产效率
提升产品质量
降低资源能源小号

五大实施目标

五大活动环节

贯穿设计、生产、物流、运营、服务活动环节。

四大关键特征

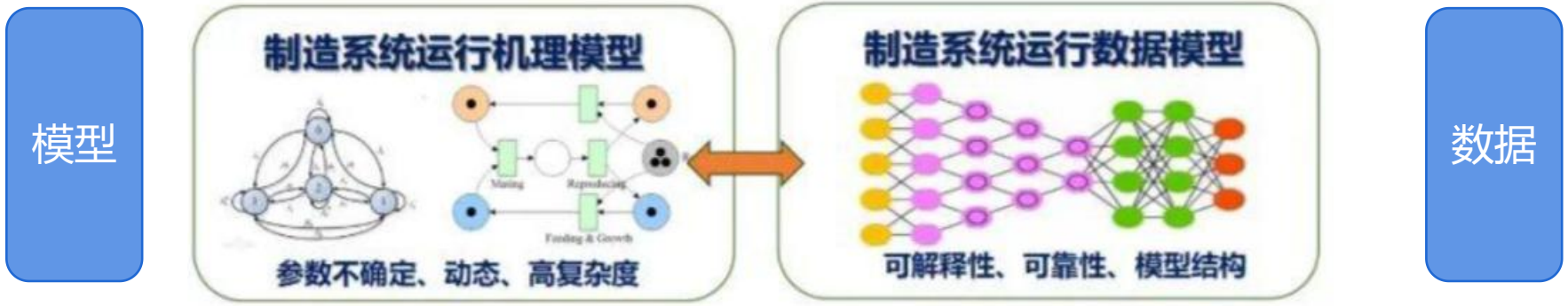
以智能工厂为载体
以关键制造环节为核心
以端到端数据流为基础
以网络互联为支撑

AI驱动的智能智造

机器学习：基于知识的方法→基于数据的方法

传统的问题解决思路——“问题→知识→问题”，即根据问题找“知识”，并用“知识解决“问题”。

机器学习兴起了另一种方法论——“问题→数据→问题”，即根据问题找“数据”，并直接运用数据(不需要把“数据”转换成“知识”的前提下)解决问题。

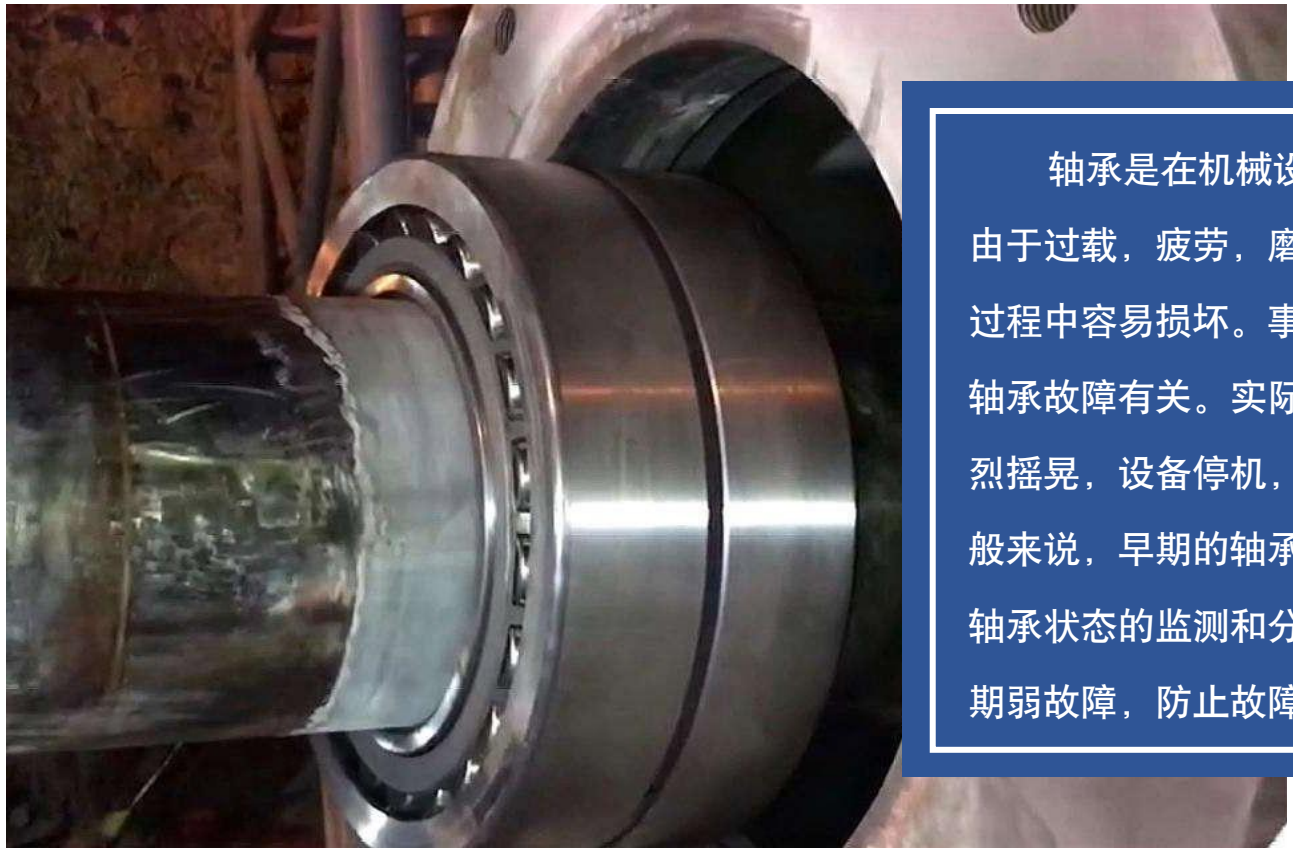


AI驱动的智能智造

机器学习：实现智能制造核心赋能技术

项目背景

工业大数据轴承故障分类



轴承是在机械设备中具有广泛应用的关键部件之一。由于过载，疲劳，磨损，腐蚀等原因，轴承在机器操作过程中容易损坏。事实上，超过50%的旋转机器故障与轴承故障有关。实际上，滚动轴承故障可能导致设备剧烈摇晃，设备停机，停止生产，甚至造成人员伤亡。一般来说，早期的轴承弱故障是复杂的，难以检测。因此，轴承状态的监测和分析非常重要，它可以发现轴承的早期弱故障，防止故障造成损失。

项目背景

工业大数据轴承故障分类

项目要求：利用真实的轴承振动信号数据集，选手需要使用机器学习技术判断轴承的工作状态。

需要设计模型根据轴承运行中的振动信号对轴承的工作状态进行分类。

轴承有3种故障：外圈故障，内圈故障，滚珠故障，外加正常的工作状态。如表1所示，结合轴承的3种直径（直径1,直径2,直径3），轴承的工作状态有10类：

	外圈故障	内圈故障	滚珠故障	正常
直径1	1	2	3	0
直径2	4	5	6	
直径3	7	8	9	

02

本課内容

COURSE CONTENT

课程目标

01

认 知 目 标

理解KNN分类模型的基本原理，了解其优缺点和适用场景。

02

技 能 目 标

掌握sklearn模块中的KNN分类器函数的使用方法。并利用KNN算法解决实际问题。

03

素 质 目 标

培养一丝不苟的工匠精神。
养成精益求精的良好习惯。

任务一：数据预处理

课程导读
本课内容
重点难点
项目总结

任务要求：

1. 导入数据集
2. 打印数据集基本信息，检查是否有缺项
3. 提取出分类标签和特征数据
4. 统计每个故障类别的数量并绘制条形图

课程导读
本课内容
重点难点
项目总结

任务一：数据预处理

本项目提供train和test数据集，其中：

训练集数据——1到6000为按时间序列连续采样的振动信号数值，每行数据是一个样本，共792条数据，第一列id字段为样本编号，最后一列label字段为标签数据，即轴承的工作状态，用数字0到9表示。
测试集数据——共528条数据，除无label字段外，其他字段同训练集。

名称	修改日期	类型	大小
test_data.csv	2023/11/13 17:44	Microsoft Excel ...	62,902 KB
train.csv	2023/11/13 17:45	Microsoft Excel ...	94,277 KB

震动信号值

	A	B	C	D	E	F	G	H	I	J	K	L
1	1	2	3	4	5	6	7	8	9	10	11	12
2	0.56365	1.069229	-0.83776	-1.12202	0.433296	0.770755	-0.47715	-0.58842	0.455224	0.555122	-0.21523	-0.19776
3	0.061333	0.05883	0.056952	0.068634	0.073433	0.07239	0.042975	-0.0073	-0.02629	-0.00605	0.028789	0.026286
4	0.035736	0.010964	-0.16487	-0.16771	-0.12508	-0.10477	-0.01665	0.151471	0.137258	0.045076	0.091776	0.175836
5	-0.0467	0.060913	0.00934	-0.0934	-0.06782	0.022335	0.006091	-0.07675	-0.03289	0.023553	-0.01706	-0.02437
6	0.162922	-0.37766	0.014457	0.565437	-0.20337	-0.51151	0.410961	0.228546	-0.51524	0.013157	0.328606	-0.22497
7	0.115573	0.064462	0.044018	0.064462	0.071555	0.039846	0.00605	-0.01502	0.002503	0.037759	0.049859	0.025034

故障类别

HVU
label
7
0
9
9
7
0

任务一：数据预处理

导入数据集：

```
# 导入相应模块
import ___①___ as pd

# 使用read_csv函数读取数据集文件
df = ___②___.read_csv('./data/train.csv')

# 显示数据的前5行
df.___③___()
```

任务一：数据预处理

导入数据集：

```
# 导入相应模块
import pandas as pd

# 使用read_csv函数读取数据集文件
df = pd.read_csv('./data/train.csv')

# 显示数据的前5行
df.head()
```

任务一：数据预处理

导入数据集：

```
# 导入相应模块
import pandas as pd

# 使用read_csv函数读取数据集文件
df = pd.read_csv('./data/train.csv')

# 显示数据的前5行
df.head()
```

✓ 3.7s

	1	2	3	4	5	6	7	8	9	10	...	5992
0	0.563650	1.069229	-0.837759	-1.122021	0.433296	0.770755	-0.477153	-0.588421	0.455224	0.555122	...	-0.050761
1	0.061333	0.058830	0.056952	0.068634	0.073433	0.072390	0.042975	-0.007302	-0.026286	-0.006050	...	0.061333
2	0.035736	0.010964	-0.164872	-0.167714	-0.125075	-0.104771	-0.016650	0.151471	0.137258	0.045076	...	4.272044
3	-0.046700	0.060913	0.009340	-0.093400	-0.067817	0.022335	0.006091	-0.076751	-0.032893	0.023553	...	0.095025
4	0.162922	-0.377662	0.014457	0.565437	-0.203369	-0.511508	0.410961	0.228546	-0.515244	0.013157	...	-0.093563

5 rows × 6001 columns

任务一：数据预处理

检查数据：

```
# 打印数据集的形状
print('数据集的形状为：\n', df.shape)

# 打印数据集描述性统计信息
df.describe()
```

```
数据集的形状为:
(792, 6001)
```

数据预处理：

```
# 获取分类目标标签
y = df['label']
y.shape
# 获取特征
x = df.iloc[:, :-1]
x.shape
```

```
(792,)
(792, 6000)
```

```
# 导入sklearn中的数据预处理模块
from sklearn.preprocessing import Normalizer

# 进行数据归一化
X_std = Normalizer().fit_transform(x)
```

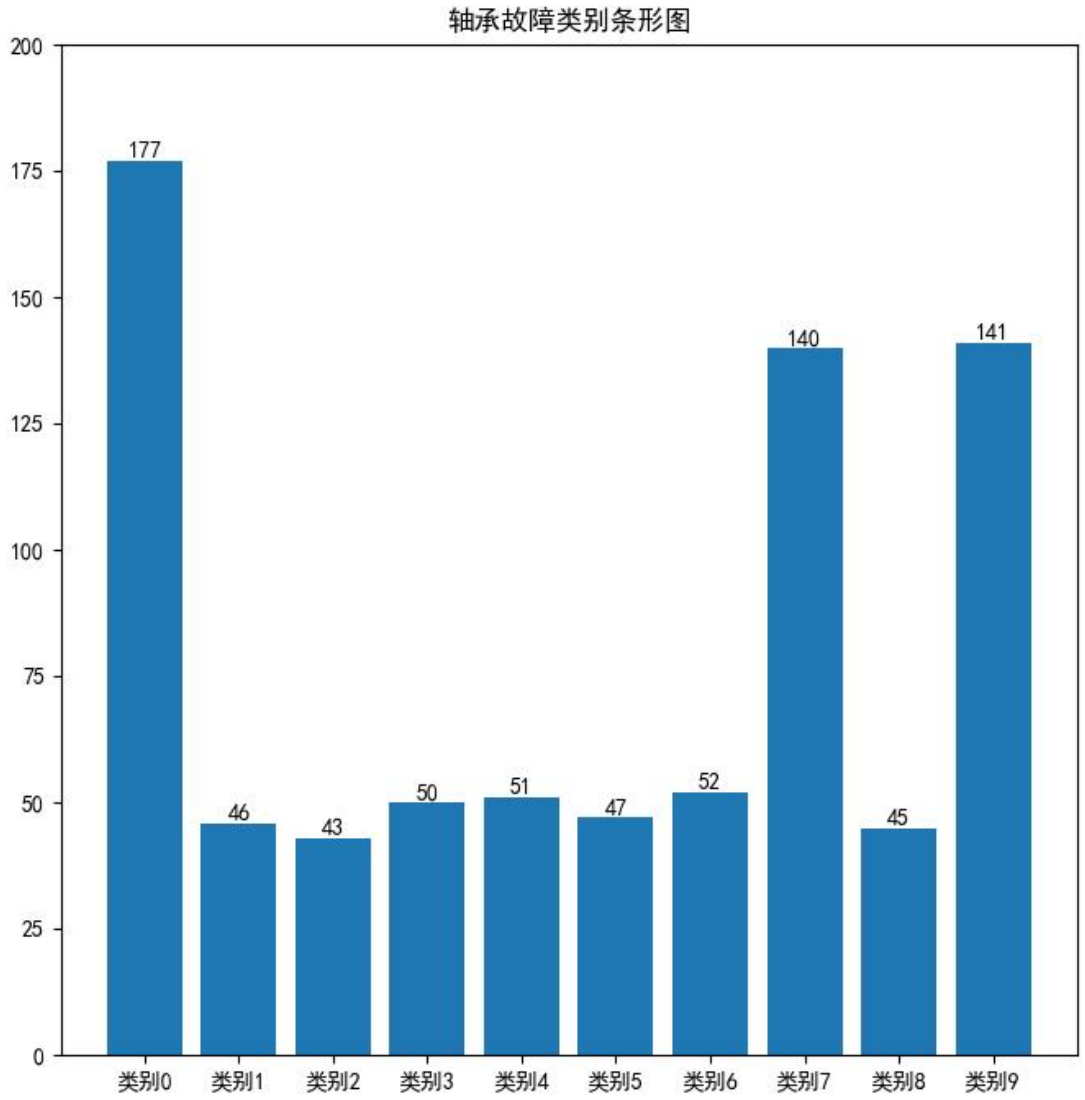
任务一：数据预处理

统计可视化：

```
import matplotlib.pyplot as plt

myHeight = []
for i in range(10):
    myHeight.append(target[target == i].count())
myHeight

x_zhou = range(10)

plt.rcParams['font.sans-serif'] = 'SimHei'
plt.rcParams['axes.unicode_minus'] = False

plt.figure(figsize=(8, 8), dpi=100)
plt.bar(x_zhou, myHeight, width=0.8)

for i in range(len(myHeight)):
    plt.text(i, myHeight[i], '{}'.format(myHeight[i]),va='bottom', ha='center')

plt.xticks(x_zhou, ['类别0', '类别1', '类别2', '类别3', '类别4', '类别5', '类别6', '类别7', '类别8', '类别9'])
plt.ylim([0, 200])
plt.title('轴承故障类别条形图')
plt.show()
```

任务一：数据预处理

统计可视化：



任务二：构建KNN分类器

课程导读
本课内容
重点难点
项目总结

任务要求：

1. 导入KNN分类器所需要的模块
2. 按8：2比例划分数据集为训练集和验证集
3. 创建KNN分类器，使用默认参数
4. 使用测试集数据训练模型

任务二：构建KNN分类器

KNN算法原理：

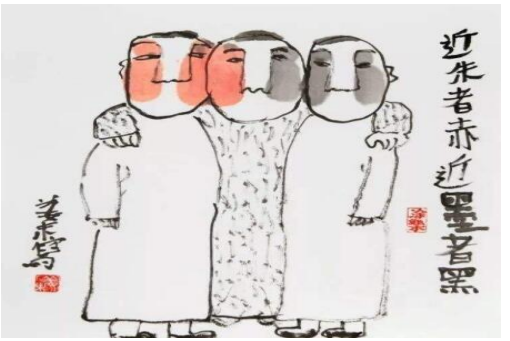
K 近邻算法（K—nearest neighbors，KNN）是一种基本朴实的机器学习方法。KNN在我们日常生活中也有类似的思想应用。比如，我们判断一个人的人品，往往只需要观察他最密切的几个人的人品好坏就能得到结果了。这就是KNN的思想应用——近朱者赤，近墨者黑。

管 宁 割 席

三国时代，管宁与华歆为友，管宁洁身自好，华歆则贪权好利，后来甘当曹操的鹰犬，丝毫没有受管宁的熏陶

孟 母 三 迁

孟轲的母亲为选择良好的环境教育孩子，三次迁居。

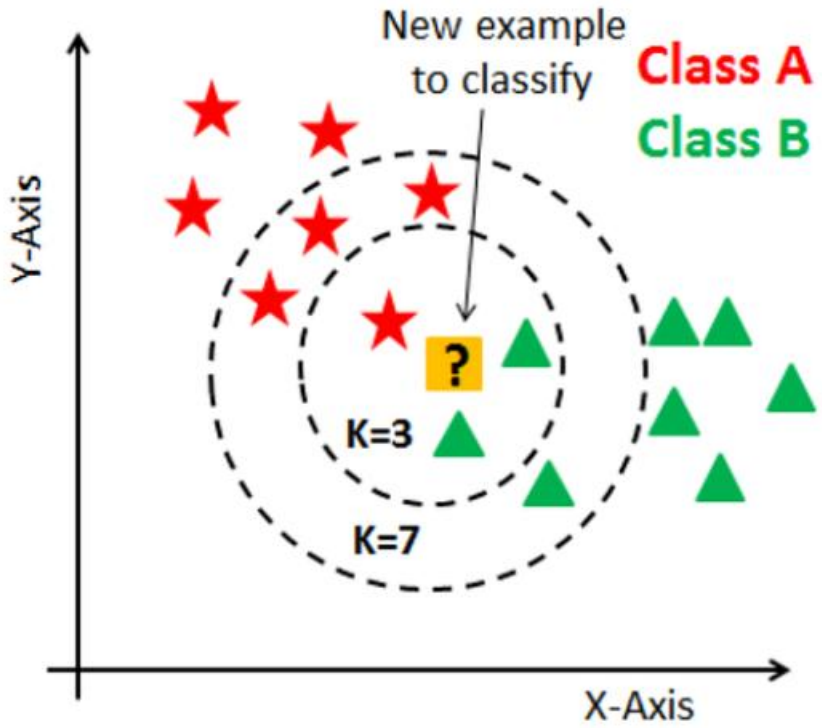


物 以 类 聚

齐宣王喜欢招贤纳士，于是让淳于髡举荐人才。淳于髡一天之内接连向齐宣王推荐了七位贤能之士。

任务二：构建KNN分类器

KNN算法的工作流程：



Setup1：选择K的大小，用于确定邻居数

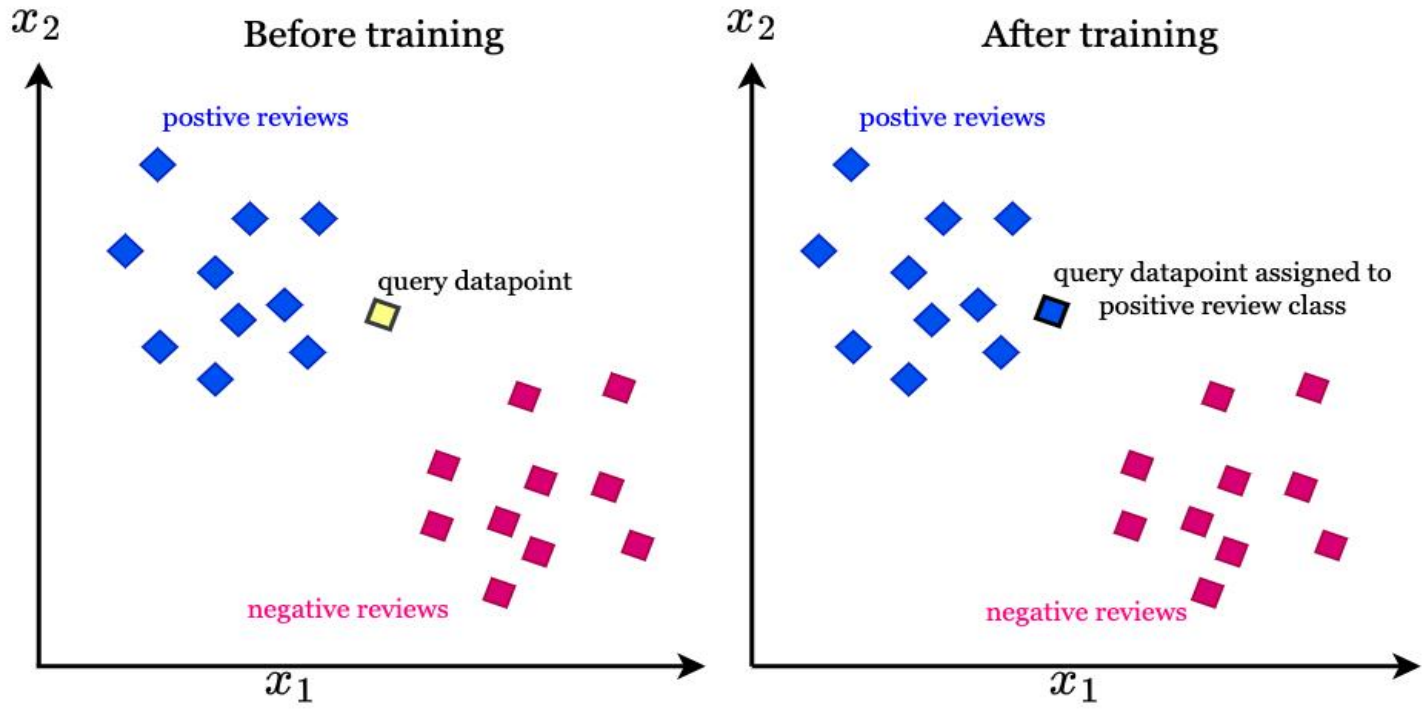
Setup2：计算并找到与目标距离最近的K个样本

Setup3：在K个最近的邻居中，统计每个类别出现的次数

Setup4：把目标划分到K个邻居中出现次数最多的类别中分类任务完成

任务二：构建KNN分类器

KNN算法举例：



任务二：构建KNN分类器

KNNClassifier函数：

```
class sklearn.neighbors.KNeighborsClassifier(n_neighbors=5, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=1, **kwargs)
```

任务二：构建KNN分类器

KNNClassifier函数常用参数及其说明：

参数名称	说明
n_neighbors	接收int。表示近邻点的个数，即K值。默认为5
weights	接收str或callable，可选参数有“uniform”和“distance”。表示近邻点的权重，“uniform”表示所有的邻近点的权重相等；“distance”表示距离近的点比距离远的点的权重大。默认为“uniform”
algorithm	接收str，可选参数有“auto”“ball_tree”“kd_tree”和“brute”。表示搜索近邻点的算法。默认为“auto”，即自动选择
leaf_size	接收int。表示kd树和ball树算法的叶尺寸，它会影响树构建的速度和搜索速度，以及存储树所需的内存大小。默认为30
p	接收int。表示距离度量公式，1是曼哈顿距离公式；2是欧式距离。默认为2
metric	接收str或callable。表示树算法的距离矩阵。默认为“minkowski”
metric_params	接收dict。表示metric参数中接收的自定义函数的参数。默认为None
n_jobs	接收int。表示并行运算的（CPU）数量，默认为1，-1则是使用全部的CPU

任务二：构建KNN分类器

KNNClassifier函数常用方法及其说明：

方法	格式	说明
fit	fit(X, y)	将训练集数据放入模型训练
kneighbors	kneighbors(X=None, n_neighbors=None, return_distance=True)	返回最近邻点及距离
kneighbors_graph	kneighbors_graph(X=None, n_neighbors=None, mode='connectivity')	返回以CSR格式的稀疏矩阵显示的最近邻点
predict	predict(X)	预测类别标签
score	score(X, y, sample_weight=None)	返回测试集的平均准确率
predict_proba	predict_proba(X)	预测概率
get_params	get_params(deep=True)	获取模型参数
set_params	set_params(**params)	设置模型参数

任务二：构建KNN分类器

导入并创建KNN分类器：

```
# 导入数据集划分函数和KNN分类器函数
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier

# 划分数据集，比例8:2
X_train, X_test, y_train, y_test = train_test_split(X_std, target, test_size=___?___,
random_state=1)

# 创建KNN分类器
knn = KNeighborsClassifier()
# 进行模型训练
knn.fit(X_train, y_train)
```

任务二：构建KNN分类器

导入并创建KNN分类器：

```
# 导入数据集划分函数和KNN分类器函数
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier

# 划分数据集，比例8:2
X_train, X_test, y_train, y_test = train_test_split(X_std, target, test_size=0.2,
random_state=1)

# 创建KNN分类器
knn = KNeighborsClassifier()
# 进行模型训练
knn.fit(X_train, y_train)
```

任务三：模型验证

课程导读
本课内容
重点难点
项目总结

任务要求：

1. 利用训练得到的模型，打印验证集前10个样本的分类推理结果
2. 打印验证集在模型上的推理得分
3. 打印验证集前3个样本对应的类别
4. 打印模型的所有参数

任务三：模型验证

利用验证集和KNN分类器方法对模型进行评价：

```
# 打印模型预测的前10个样本分类结果
print('预测前10个结果为：\n', _____①_____)
# print('前10个结果的真实值为：\n', y_test[:10])
# 打印测试集准确率
print('测试集准确率为：', _____②_____)
# 打印测试集前3个对应类别的概率
print('测试集前3个对应类别的概率为：\n', _____③_____)
# 获取模型参数
print('模型的参数为：\n', _____④_____)
```

ⓐknn.score(X_test, y_test)

ⓑknn.predict(X_test[:10])

ⓒknn.get_params()

ⓓknn.predict_proba(X_test[:3])

任务三：模型验证

利用验证集对模型进行评价：

```
# 打印模型预测的前10个样本分类结果
print('预测前10个结果为：\n', knn.predict(X_test[:10]))
# print('前10个结果的真实值为：\n', y_test[:10])
# 打印测试集准确率
print('测试集准确率为：', knn.score(X_test, y_test))
# 打印测试集前3个对应类别的概率
print('测试集前3个对应类别的概率为：\n', knn.predict_proba(X_test[:3]))
# 获取模型参数
print('模型的参数为：\n', knn.get_params())
```

03

重點難點

KEY POINTS
AND DIFFICULTIES

问题思考

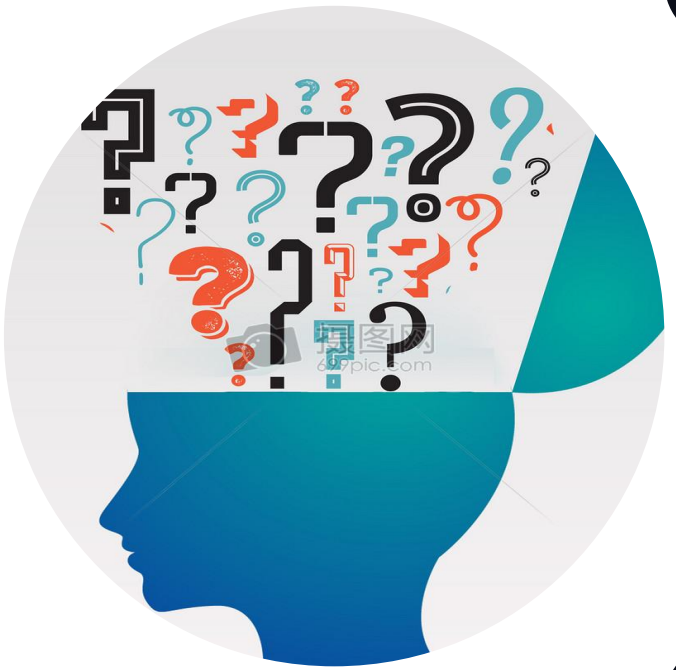
01

当前的模型验证分数过低，如何进一步提升模型性能?

KNN分类器有哪些超参数可调?

02

调参策略



k值

调整近邻点的个数

距离权重weights

有uniform和distance两个可选项，表示近邻点的权重。

距离参数p

距离度量公式，1表示曼哈顿距离，2表示欧式距离。

模型调参

①寻找最佳K值：

```
best_score = 0.0
best_k = -1
for k in range(1, 11):
    knn_clf = KNeighborsClassifier(n_neighbors = k)
    knn_clf.fit(X_train, y_train)
    knn_score = knn_clf.score(X_test, y_test)
    if knn_score > best_score:
        best_score = knn_score
        best_k = k

print("best_k = ", best_k)
print("best_score = ", best_score)
```

模型调参

②寻找最佳K值和权重weights：

```
best_method = ""
best_score = 0.0
best_k = -1
for method in ["uniform", "distance"]:
    for k in range(1, 11):
        knn_clf = KNeighborsClassifier(n_neighbors = k)
        knn_clf.fit(X_train, y_train)
        knn_score = knn_clf.score(X_test, y_test)
        if knn_score > best_score:
            best_score = knn_score
            best_k = k
            best_method = method

print("best_mrthod = ", best_method)
print("best_k = ", best_k)
print("best_score = ", best_score)
```

模型调参

③寻找最佳K值和距离p：

```
best_p = -1
best_score = 0.0
best_k = -1
for k in range(1, 11):
    for p in range(1, 6):
        knn_clf = KNeighborsClassifier(n_neighbors = k,
weights = "distance", p = p)
        knn_clf.fit(X_train, y_train)
        knn_score = knn_clf.score(X_test, y_test)
        if knn_score > best_score:
            best_score = knn_score
            best_k = k
            best_p = p

print("best_p = ", best_p)
print("best_k = ", best_k)
print("best_score = ", best_score)
```

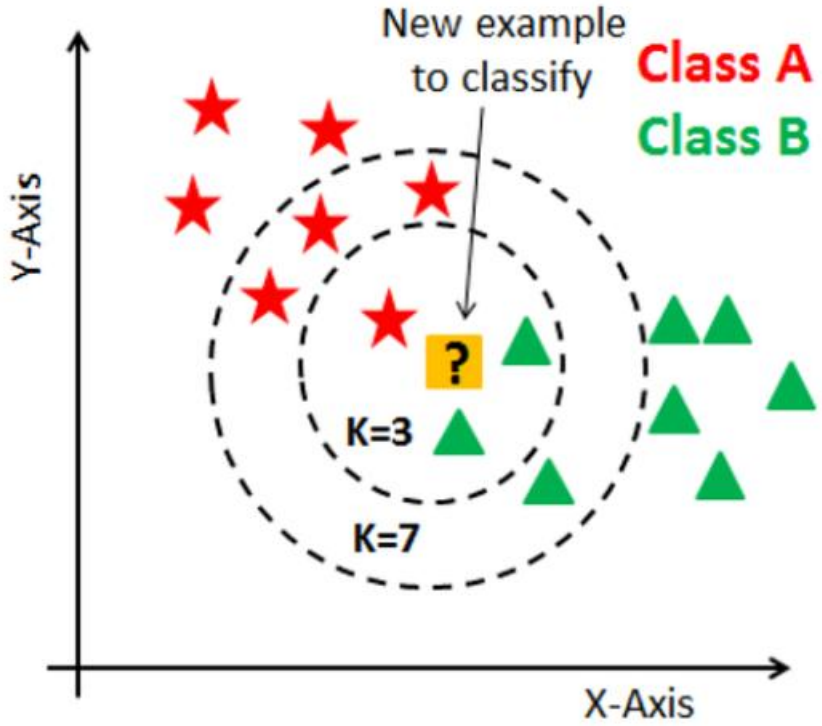
04

項目總結

PROJECT
SUMMARY

项目总结

①KNN算法的工作流程：



Setup1：选择K的大小，用于确定邻居数

Setup2：计算并找到与目标距离最近的K个样本

Setup3：在K个最近的邻居中，统计每个类别出现的次数

Setup4：把目标划分到K个邻居中出现次数最多的类别中

分类任务完成

项目总结

②KNN算法的优缺点：

优点

1.理论成熟，思想简单，既可以用来做分类又可以做回归
2.可以用于非线性分类
3.训练时间复杂度比支持向量机之类的算法低
4.和朴素贝叶斯之类的算法比，对数据没有假设，准确度高，对异常点不敏感
5.比较适用于样本容量比较大的类域的自动分类

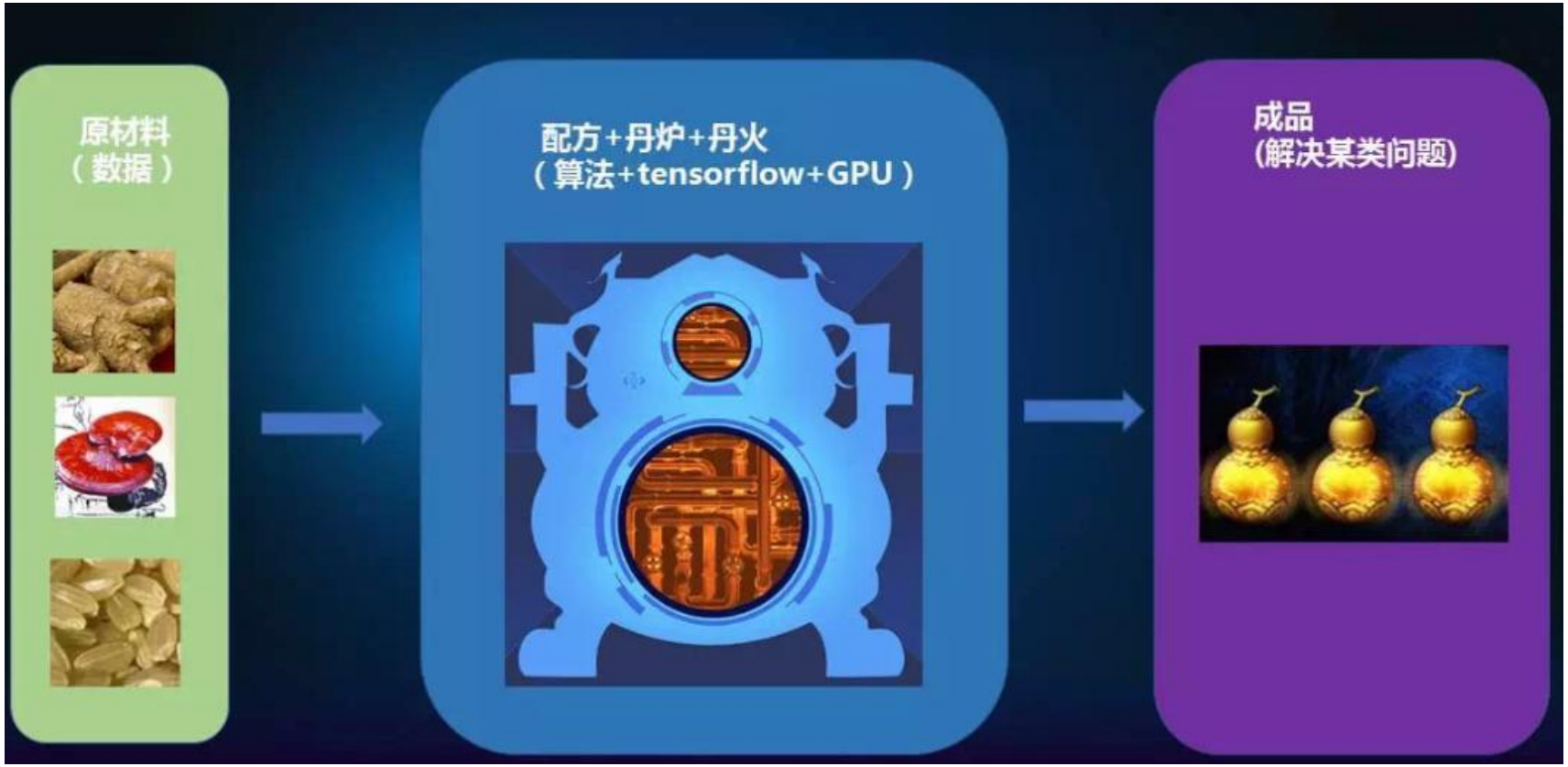
缺点

1.计算量大，尤其是特征数非常多的时候
2.样本不平衡的时候，对稀有类别的预测准确率低
3.是慵懒散学习方法，基本上不学习，导致预测时速度比起逻辑回归之类的算法慢
4.相比决策树模型，KNN模型的可解释性不强

项目总结

③关于调参：

调参就好比选用不同的人做事情，不同的人适合做不同的工作，用此调节不同的参数能更好的适应特定的任务。



稻壳儿PPT模板使用说明（本页为说明页，用户使用模板时可删除本页内容）

01 字体说明

中文 | 字体名称

汉仪综艺体繁

英文文 | 字体名称

汉仪雅酷黑简

中文 | 字体名称

汉仪中黑简

英文文 | 字体名称

汉仪中宋简

【说明】

模板中使用的字体为【汉仪系列】，仅限于个人学习、研究或欣赏目的使用，如需商用请您自行向版权方购买、获取商用版权。

02 素材说明

图片：

【CC0共享协议素材说明】模板中使用的图片来源于【pixabay】，该图片具有CC0共享协议，您可在遵循CC0共享协议的情况下使用。

素材：

【CC0共享协议素材说明】模板中使用的素材来源于【pixabay】，该素材具有CC0共享协议，您可在遵循CC0共享协议的情况下使用。

03 母版说明

使用本套PPT模板时，若在操作界面鼠标不可选取的内容，可以在幻灯片母版中进行查看和编辑，具体操作如下：

1、点击【视图】

2、选择【幻灯片母版】，即可查看/编辑母版内容

3、查看/编辑完成后，点击【关闭】即可