

# 监督学习方法的应用

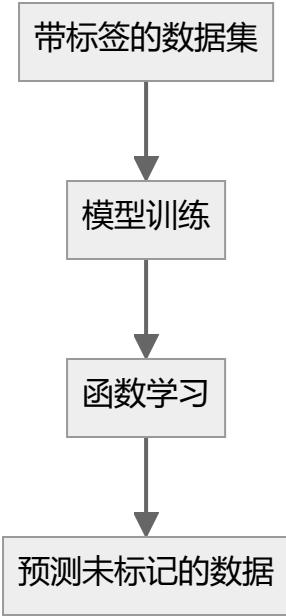
From the basic to the frontier | SPA

Reporter: Jiakai Gu  
Date: Nov 13th, 2023

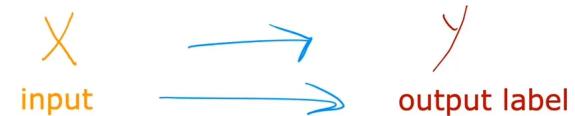
# Table of contents

1. 监督学习的基本要点
2. 分类问题的应用
3. 标注问题（分类问题的推广）的应用
4. 回归问题的应用
5. 监督学习在大语言模型中的应用
6. 总结

# 监督学习的基本要点

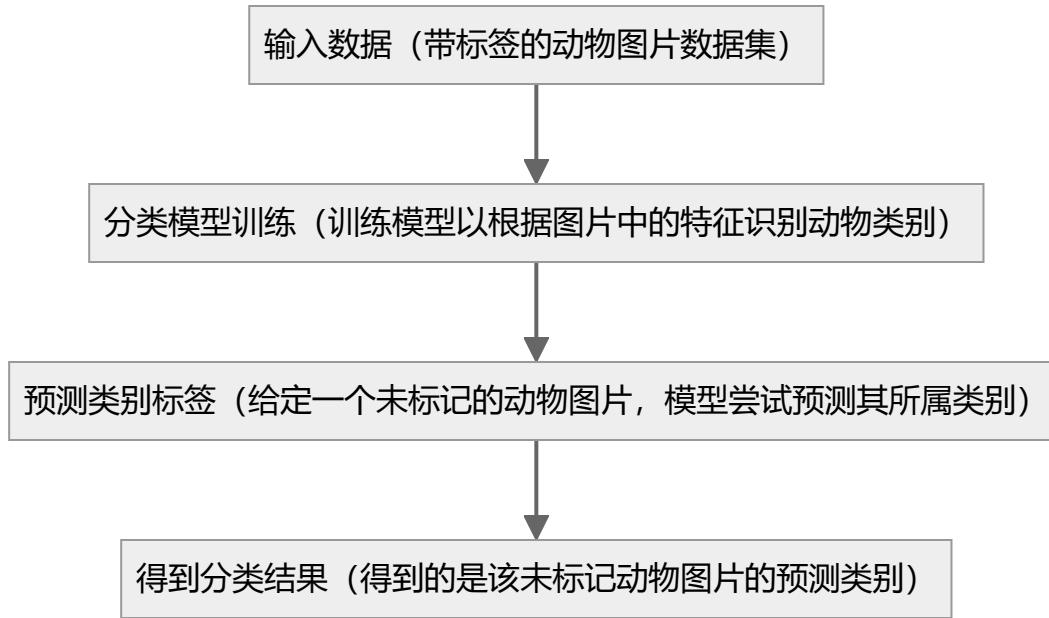


Supervised learning



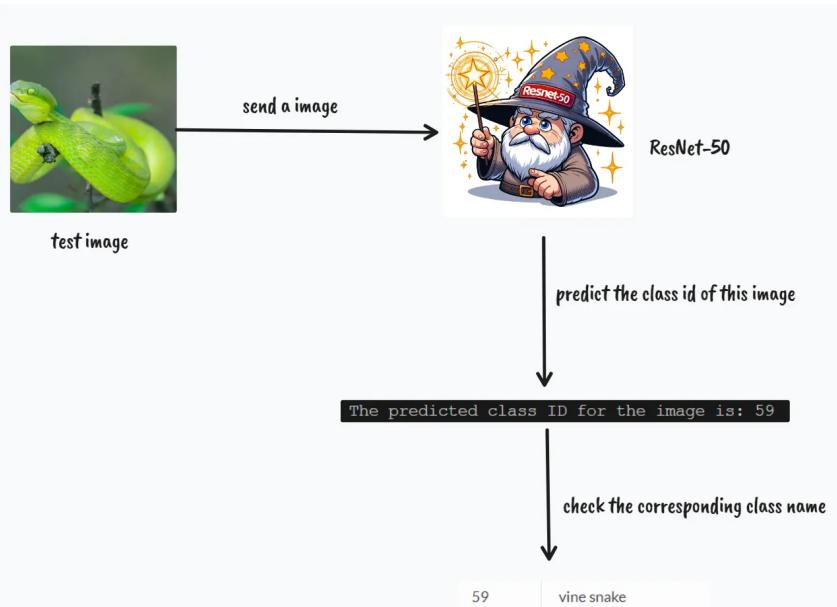
Learns from being given "right answers"

# 分类问题概述（预测离散值/类别标签）



# 分类问题的应用

- 使用预训练的ResNet-50模型（基于ImageNet数据集的子集训练得到）进行图片分类。



代码执行流程

```
# 导入必要的库
import torch
import torchvision.models as models
import torchvision.transforms as transforms
from PIL import Image

# 加载预训练模型
# 初始化ResNet-50模型。'DEFAULT'值使模型使用预训练的权重。
model = models.resnet50(weights='DEFAULT')

# 将模型设置为评估模式。在评估模式下，模型的行为（例如，关闭批处理规范化和丢弃层的训练行为）会有所不同，
# 以确保模型在推理时的准确性和可靠性。
model.eval()

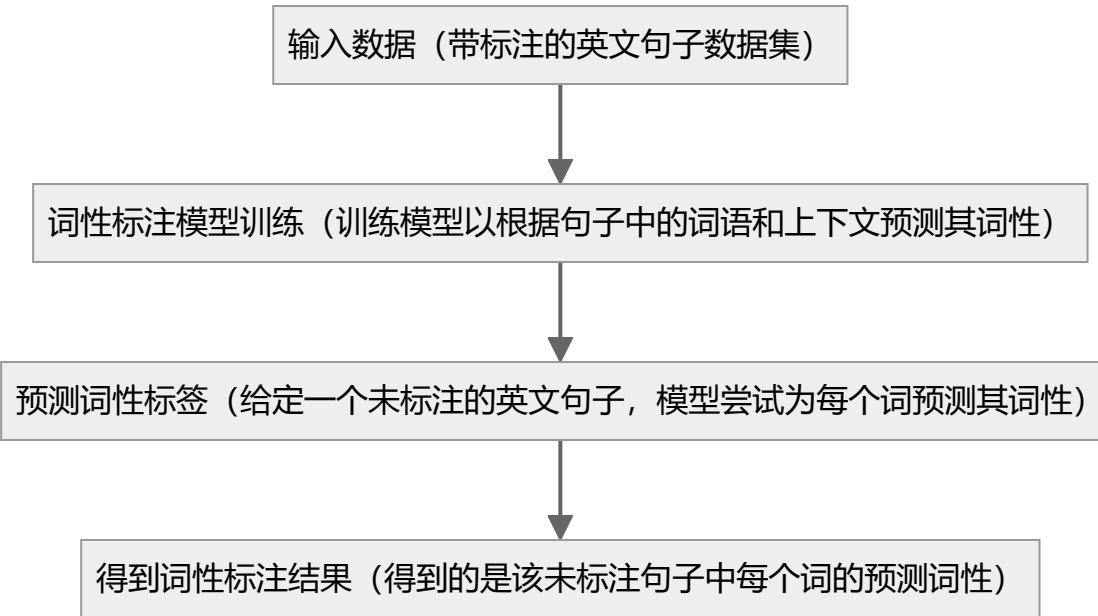
# 定义图像预处理流程
# 这个流程包括：调整图像大小、中心裁剪、将图像转换为张量、以及正则化。
transform = transforms.Compose([
    transforms.Resize(256),           # 调整图像大小为256x256
    transforms.CenterCrop(224),        # 中心裁剪图像为224x224
    transforms.ToTensor(),            # 将图像转换为张量
    transforms.Normalize(             # 正则化张量
        mean=[0.485, 0.456, 0.406],   # 使用Imagenet数据集的均值
        std=[0.229, 0.224, 0.225]     # 使用Imagenet数据集的标准差
    ),
])

# 定义图像类别预测函数
def predict_image_class(image_path):
    # 打开图像
    image = Image.open(image_path)
    # 对图像应用预处理流程
    image = transform(image).unsqueeze(0)
    # 不追踪梯度，以减少内存使用
    with torch.no_grad():
        # 获取模型输出
        outputs = model(image)
        # 获取预测类别的索引
        _, predicted = outputs.max(1)
        # 返回预测类别的索引
        return predicted.item()

# 测试图像类别预测函数
image_path = 'assets/test.jpg' # 替换为您的图片路径
class_id = predict_image_class(image_path)
print("The predicted class ID for the image is: {}".format(class_id))
```

使用ResNet-50模型来分类图像

# 标注问题 (分类问题的扩展，预测离散值/类别标签)



# 标注问题（分类问题的推广）的应用

- 使用NLTK库的预训练模型—平均感知机标注器（averaged\_perceptron\_tagger）和词性标签集—（Penn Treebank），对英文句子进行词性标注。



代码执行流程

```
import nltk

# 下载需要的数据包
# 下载用于分词的数据包: punkt
nltk.download('punkt')
# 下载用于词性标注的预训练模型: averaged_perceptron_tagger
nltk.download('averaged_perceptron_tagger')

# 输入句子
sentence = "The cat sat on the mat."

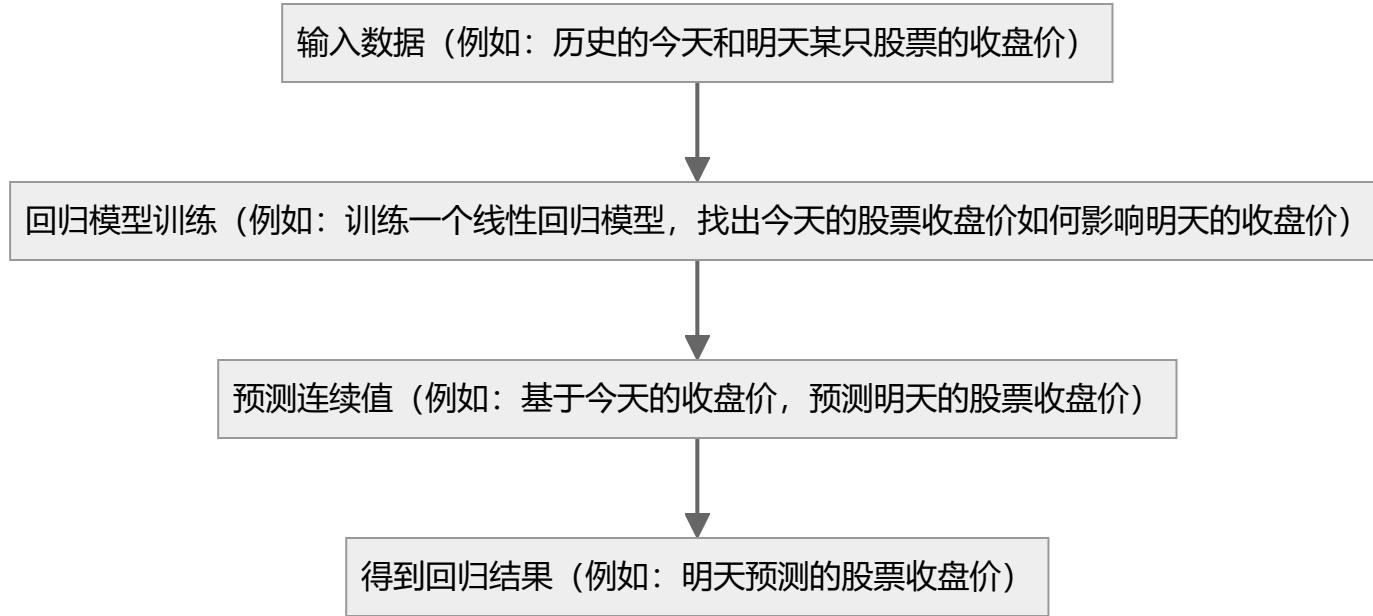
# 分词
tokens = nltk.word_tokenize(sentence)

# 词性标注
tagged_tokens = nltk.pos_tag(tokens)

print("Tagged Tokens: ", tagged_tokens)
```

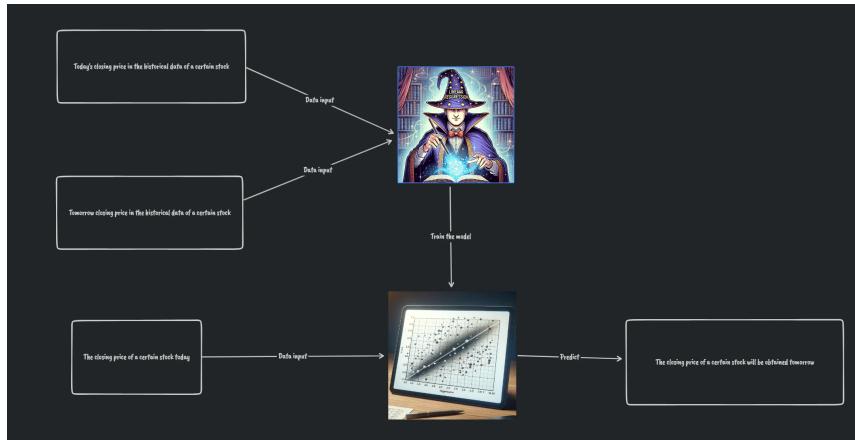
使用平均感知机标注器进行词性标注

# 回归问题（预测连续值）



# 回归问题的应用

- 使用LinearRegression模型，基于NumPy库进行数据准备和模型训练，用于预测股票的明天收盘价。



代码执行流程

```
# 导入所需的库: sklearn中的线性回归模型和numpy
from sklearn.linear_model import LinearRegression
import numpy as np

# 准备输入数据: 今天的收盘价, 单位是美元。
# reshape(-1, 1) 用于将数组转化为二维矩阵, 以满足scikit-learn模型的输入要求。
X = np.array([120, 125, 130, 135]).reshape(-1, 1)

# 准备标签数据: 即我们要预测的明天的收盘价, 单位也是美元。
y = np.array([125, 130, 135, 140])

# 初始化线性回归模型
model = LinearRegression()

# 使用fit方法对模型进行训练。这一步会找到最适合描述X和y之间关系的直线。
model.fit(X, y)

# 准备一个新的输入数据 (今天的收盘价) 以进行预测。
# 同样用reshape(-1, 1)确保它是二维矩阵。
new_X = np.array([140]).reshape(-1, 1)

# 使用模型进行预测
prediction = model.predict(new_X)

# 输出预测结果: 预测的明天的收盘价, 单位是美元。
print(f"预测明天的收盘价为: {prediction[0]:.2f}美元")
```

使用线性回归模型预测明天股票的收盘价

# 监督学习在大语言模型中的应用

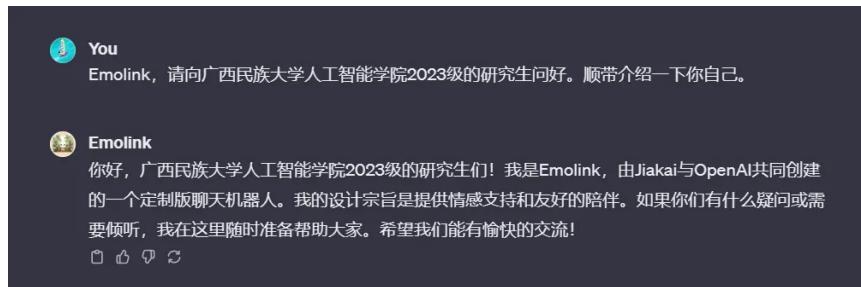
ChatGPT的模型微调过程利用到了基于人类反馈的强化学习(RLHF)，  
基于人类反馈的强化学习可看作是结合了监督学习和强化学习的方法。[via](#)

ChatGPT is based on particular GPT foundation models, namely GPT-3.5 and GPT-4, that were fine-tuned to target conversational usage.<sup>[9]</sup> The fine-tuning process leveraged both supervised learning as well as reinforcement learning in a process called reinforcement learning from human feedback (RLHF).<sup>[10][11]</sup> Both approaches employed human trainers to improve model performance. In the case of supervised learning, the trainers played both sides: the user and the AI assistant. In the reinforcement learning stage, human trainers first ranked responses that the model had created in a previous conversation.<sup>[12]</sup> These rankings were used to create "reward models" that were used to fine-tune the model further by using several iterations of Proximal Policy Optimization (PPO).<sup>[10][13]</sup>

今年9月份，OpenAI新出的GPT-4V，能看，能听，能说。其中的  
听，使用的是OpenAI自己研发的whisper模型，其中利用到了监督学  
习。[via](#)

Whisper is an automatic speech recognition (ASR) system trained on 680,000 hours of multilingual and multitask supervised data collected from the web. We show that the use of such a large and diverse dataset leads to improved robustness to accents, background noise and technical language. Moreover, it enables transcription in multiple languages, as well as translation from those languages into English. We are open-sourcing models and inference code to serve as a foundation for building useful applications and for further research on robust speech processing.

- 研究生小组前段时间使用国内清华大学开源的ChatGLM2-6B大语言模型，附加自己的数据集(train.json和dev.json)对基础大模型进行微调时，用到了监督学习。
- 上周OpenAI首届开发者大会后，推出的Create a GPT功能，允许用户用多轮对话的形式创造出特定领域的GPT，这种多轮对话形式的模型微调，技术中也涉及监督学习的身影。



Fine-Tuning in LLM.

# 总结

- 监督学习应用多样，分类和回归问题方面的应用还有很多。
- 除了分类和回归这两大监督学习的基本问题外，随着技术的进步，可能会出现新的抽象或特定的应用场景，这些场景可能不完全符合传统的分类和回归框架，但它们的核心仍然是基于带标签的数据进行模型训练。
- 监督学习配合其他机器学习分支，为当下大热的大语言模型助力。

Input (X)	Output (Y)	Application
email	spam? (0/1)	spam filtering
audio	text transcripts	speech recognition
English	Spanish	machine translation
ad, user info	click? (0/1)	online advertising
image, radar info	position of other cars	self-driving car
image of phone	defect? (0/1)	visual inspection

Thank you for listening!