**1. 回归问题**

1. pandas读取csv文件
2. 数据的归一化和标准化

将数字转换到0-1的范围

将类别文字转换成one-hot编码

1. 用sklearn.model\_selection.train\_test\_split来生成训练集和测试集
2. 用Dense构建神经网络
3. 保存、加载模型
4. **分类问题（模型训练可以用本地设备）**
5. 读取sklearn.datasets中的make\_circles数据集（二元分类）
6. 用Dense和relu激活函数构建神经网络
7. 用回调函数LearningRateScheduler来获取最好的学习速度
8. 读取tensorflow.keras.datasets中的fashion\_mnist数据集（多元分类）

图片大小为28\*28

归一化灰度数据（除以255）

1. 用Flatten层将二维数组转变成一维向量
2. 模型预测结果为概率数组，用argmax来获取最大概率数字的位置
3. **卷积网络（二元分类问题可以用本地设备训练，如果数据集大则需要google colab）**
4. 用tensorflow.keras.preprocessing.image.ImageDataGenerator来读取图片数据，并自动获取class

归一化：ImageDataGenerator创建时用rescale参数进行数据缩放

数据增强技术Data Augmentation：只用于训练集

图片乱序

尺寸标准化：ImageDataGenerator的flow\_from\_directory函数中用target\_size来 确定最终图片的大小

类别模式：class\_mode设置为binary或categorical

1. 数据增强技术Data Augmentation
2. 用Conv2D、MaxPool2D、Flatten、Dense建立神经网络
3. **迁移学习1（Feature Extraction，可以用本地设备训练，但是可能要半小时）**
4. 使用两种计算机视觉模型：ResNet和EfficientNet

b. 模型为已有模型加上一层输出层（Dense）

**5. 迁移学习2（Fine-tuning）**

a. tensorflow.keras.applications已经包含了一系列计算机视觉模型，可以直接使用EfficientNet（注意：使用 ResNetV250需要进行标准化，将输入值限定为0-1之间的 值。而使用EfficienteNet则不需要）

b. 用GlobalAveragePooling2D或者GlobalMaxPool2D来降维（在4中不需要，因为在这边将include\_top设置为 了False，我们自己去定义输出层）

c. 用10%或者100%的数据来训练，并且使用data augmentation和fine-tuning。与3不同，此处的数据增 强技 术不是在ImageDataGenerator中定义的，而是单独建立一 个层。

d. 先训练模型5轮，然后用fine-tuning再训练5轮（注意：这边需要重新compile模型）

e. 使用ModelCheckpoint回调函数来存储权重值

f. 使用unbatch来拆解test\_data（这里是tensorflow.data.Dataset的形式）来获取测试 集中的label信息

g. 在第6章中，最后一部分为观察衡量指标。

**6. Milestone Project 1**

a. 用TensorFlow Datasets（TFDS）来下载数据（food101）

b. 对dataset使用了batch和prefetch

c. 利用checkpoint来存储最佳权重模型，并可重新载入（需要重新编译）

d. 建立的模型和5中类似，但此处使用了混合精度策略（mixed\_precision），输出层的激活函数需要分离，使得输 出时可以将数值类型从float16转换回float32

e. 使用EarlyStopping回调函数来提前结束训练

f. 使用ReduceLROnPlateau来逐步降低学习速率  
g. Fine-tuning此处对所有的层都设置为trainable，主要是因为训练集非常大。若训练集很小，则只需要对一小部分 的层设置为trainable，否则会出现过拟合

1. **自然语言处理（二元分类问题，推文是否是disaster）**

a. 用Tokenization和Embedding将文本进行token化，并转换为数字

b. help\_functions.py中有一个函数calculate\_results(y\_true, y\_pred)，用来计算accuracy、precision、recall和f1 的值。

c. 模型都很简单，主要的不同在于text\_vectorizer和embedding之后的那一层，有LSTM、GRU、Conv1D等。其 中，迁移学习的效果是最好的。也有将多个模型的预测结果组合起来进行决策的方法，但效果还是不如迁移学习。

d. 也有将多个模型的预测结果组合起来进行决策的

**8. 自然语言处理（多元分类问题，摘要分类，可以用本地设备训练）**

1. 利用tensorflow.data.Dataset.from\_tensor\_slices来建立数据集，从而可以使用batch和prefetch来使数据载入 更快
2. 混合模型的效果最好（有位置信息的最复杂的model\_5）
3. 包括了读取json，并将文本拆分成句子的代码
4. 时间序列
5. help\_functions.py中有一个函数evaluate\_preds(y\_true, y\_pred)，用来计算mae、mse、rmse等标准。
6. 将价格序列输入到make\_windows函数，用来建立全部的窗口和标签。再用make\_train\_test\_splits函数用来将 全窗口划分成训练集和测试集。
7. 模型为全连接模型、Conv1D模型（padding=”casual”，输入值为三维，需要扩展维度）、LSTM（也需要增加维 度）、多参数、N-BEATS算法、Ensemble模型。其中结果最好的是Ensemble、多参数和最简单的全连接网络。