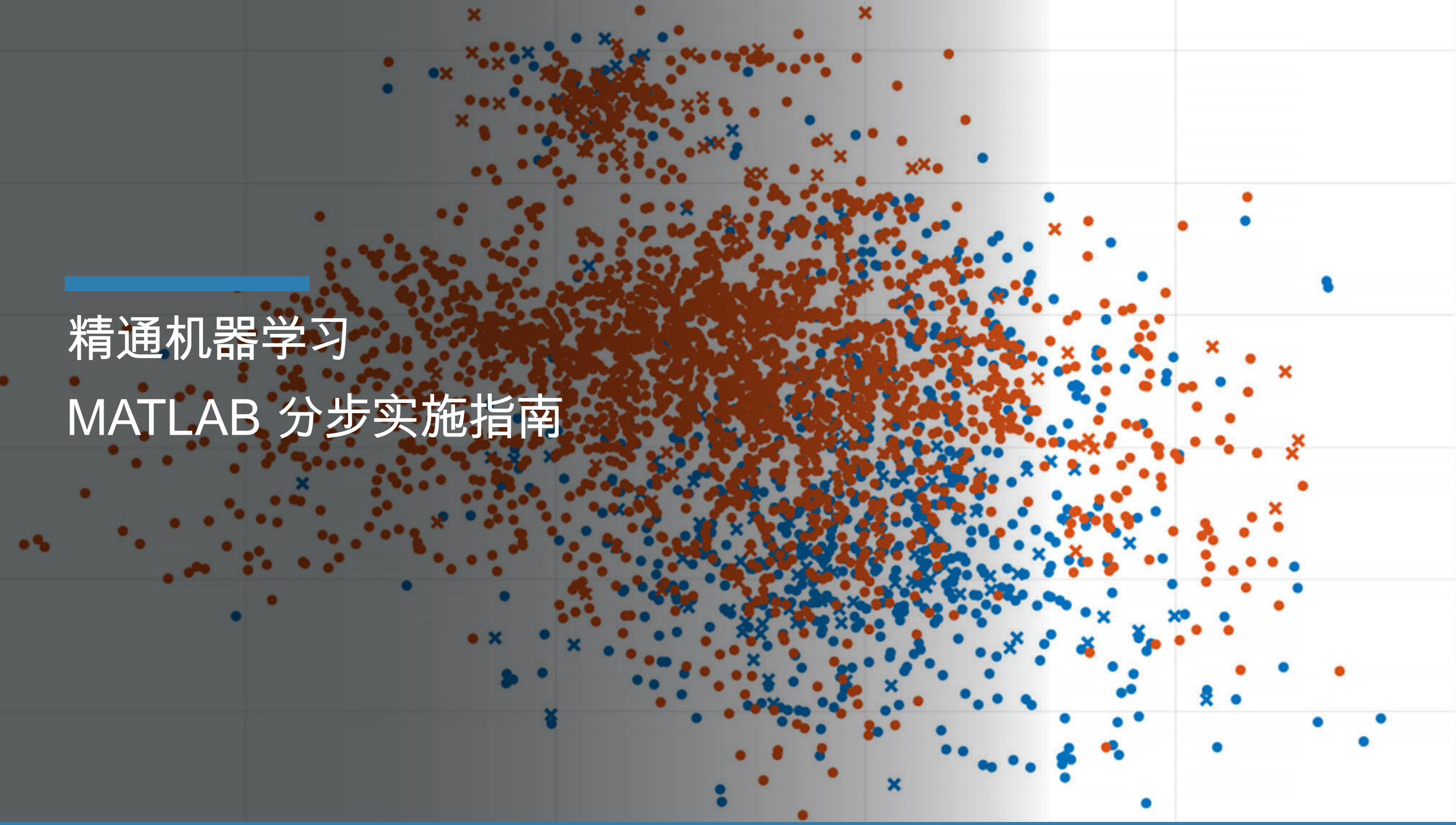
精通机器学习

MATLAB 分步实施指南



MathWorks®

简介

本电子书建立在*使用 MATLAB 进行机器学习* 的基础上，后者回顾了机器学习基础知识，并介绍了监督和无监督学习的技术方法。

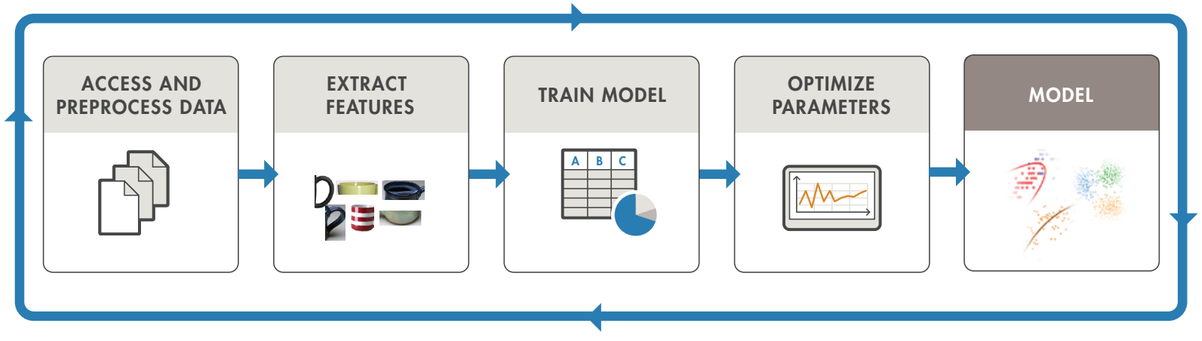
我们使用心音分类器为例，向您介绍真实世界中的机器学习应用程序从加载数据到部署训练模型的整个开发流程。对于每个训练阶段，我们将演示取得精确模型的关键技术，帮助您掌握更具挑战性的训练任务，包括选择算法、优化模型参数和避免过拟合。

在本电子书中，您还将学习如何将模型转变成预测工具，具体包括在新数据上进行训练、提取特征以及生成代码部署到嵌入式设备。

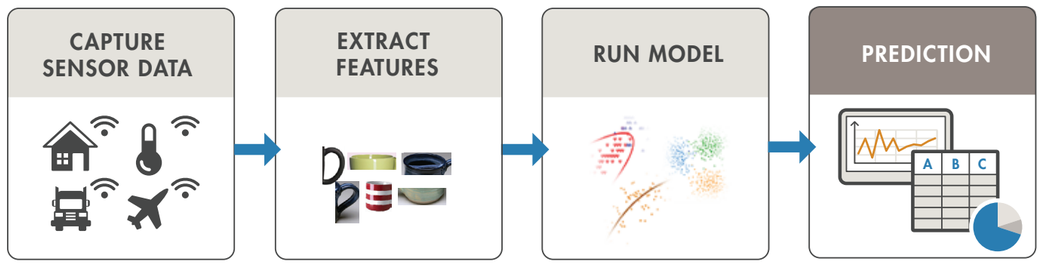
复习基础知识

机器学习概述 3:02
使用 MATLAB 进行机器学习

训练：迭代进行，直到取得令人满意的性能。



预测：将训练好的模型集成到应用程序中。



使用 MATLAB 构建心音分类应用程序

心音是早期诊断心脏疾病的丰富信息源。区分正常与不正常心音需要由经过特别训练的临床医生完成。我们的目标是开发一个能识别不正常心音的机器学习应用程序。使用心音监测应用程序，当专家不在场时，普通医护人员也能够筛查心脏疾病，而患者亦能够进行自我监测。

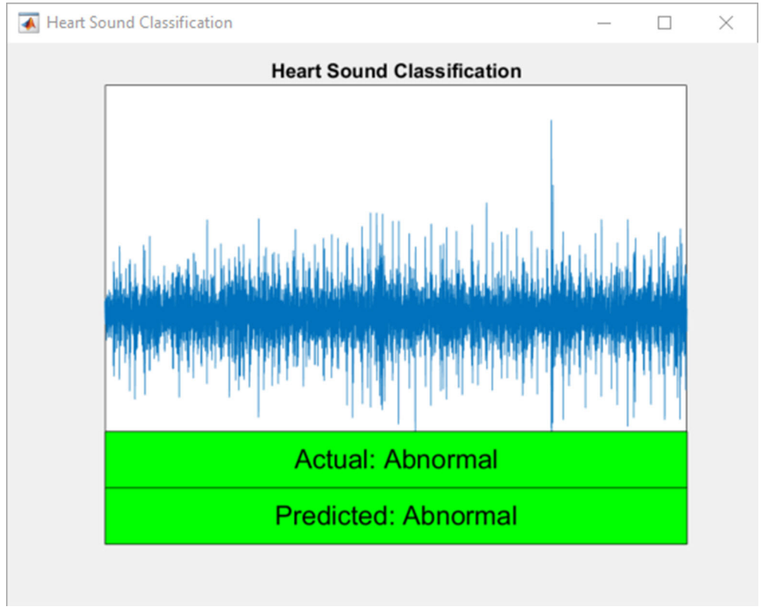
我们遵循以下步骤来开发此应用程序：

1. 访问和探查数据。
2. 预处理数据并提取特征。
3. 开发预测模型。
4. 优化模型。
5. 将分析方法部署到生产系统。

我们建议您亲自完成该示例。只需下载该 MATLAB® 代码，然后依照贯穿本电子书中的“实际操作练习”注解进行操作。

所需工具

下载*免费的 MATLAB 30 天试用版进行机器学习。*



心音分类器和应用程序原型图示。

步骤 1. 访问和探查数据

我们示例使用的数据集来自 2016 年 PhysioNet and Computing in Cardiology 挑战赛，包含数千条记录的心音，长度从 5 秒到 120 秒不

> **实际操作练习**
>
> 运行实时编辑器脚本的第一部分。该脚本将来自 2016 年 PhysioNet and Computing in Cardiology 挑战赛的心音数据集下载到您的本地工作区。

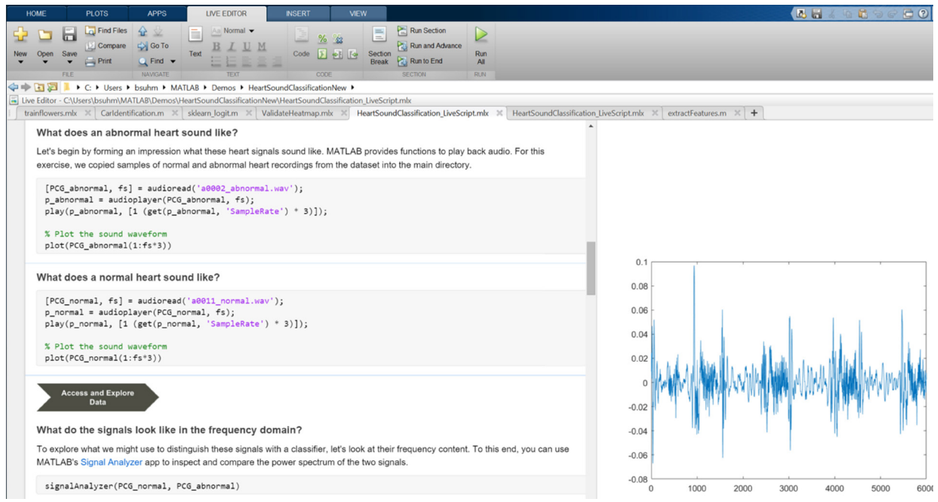
等。

该数据集包括 3240 条用于模型训练的记录和 301 条用于模型验证的记录。下载该数据后，我们将训练和验证数据集分别存放在各自的文件夹中，这是机器学习中的标准步骤。

探查数据

任何机器学习项目的第一步是了解您使用的是什么样的数据。探查数据的常见方法包括检查一些样本、创建可视化以及（更高级的）运用信号处理或聚类技术识别规律。

要了解区分正常与不正常心音涉及什么，让我们从倾听一些样本开始。



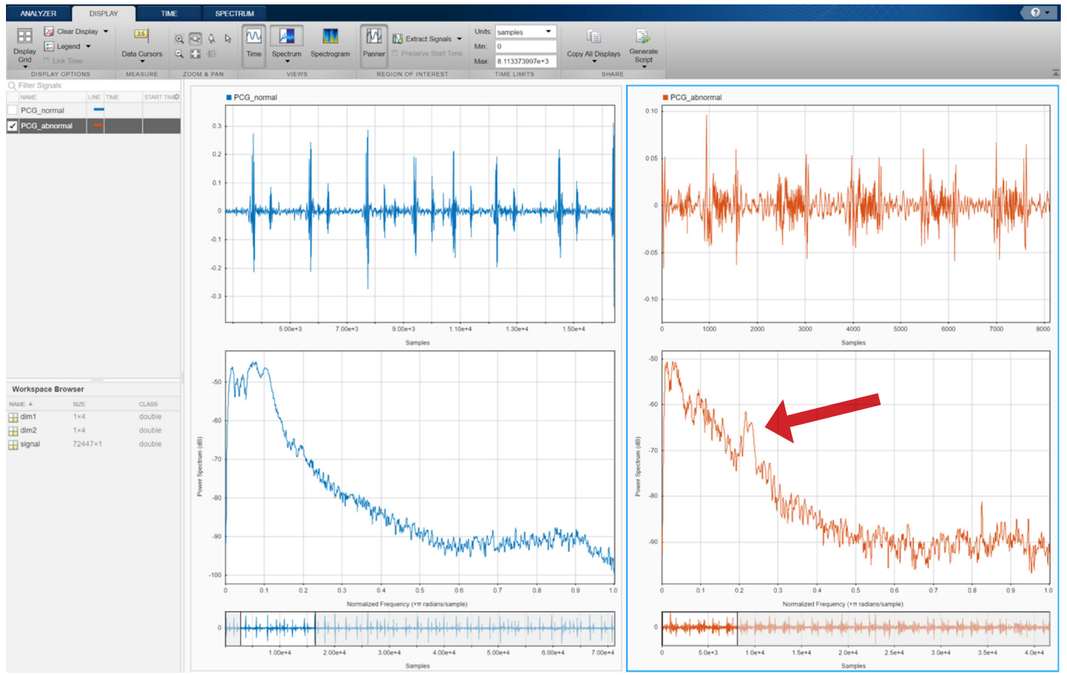
在 MATLAB 实时编辑器中生成的不正常心音的绘图。

> **实际操作练习**
>
> 倾听代码示例的主目录中提供的不正常和正常心音的样本。MATLAB 提供 **Audio Read**（音频读取）和 **Audio Player**（音频播放），用于处理音频文件。这些文件在示例脚本的“What does an (ab) normal heart sound like?”（何为正常/不正常心音？）部分中使用。

您可能注意到不正常心音有较高的频率，在心跳之间有杂音。正常心音

分析信号

使用 *Signal Processing Toolbox™* 中的 Signal Analyzer app 并排查看频域中的信号，无需编写任何代码，即可更深入地理解这些信号间的差异。



Signal Analyzer app 中显示的正常心音（左）和不正常心音（右）。
红色箭头指出不正常心音在频谱中 200 Hz 左右的尖峰。

比较规则，在心跳之间无杂音。
在这种初始探查数据和分类任务之后，我们将所有数据载入内存进行预处理。MATLAB 可以轻松地加载跨多个目录分布的大数据集：它在整个数据集上创建一个句柄，然后可以读取内存中的一个或多个数据块。对于大数据集，MATLAB 能跨多个计算资源分布执行。

每条记录都是一个音频文件，标注为不正常或正常。因为我们知道数据集中每个文件的真正类别（“真实值”），所以我们可以运用*监督式学习*，接受已知的输入数据集和该数据的已知响应（输出），然后训练模型，使模型能够对新数据的响应产生合理的预测。

实际操作练习

要将训练数据和相应的真正类别载入内存，请运行示例中的“准备将数据读入内存”和“创建含有文件名和标签的表”。

步骤 2. 预处理数据和提取特征

在提取特征之前，大多数数据集都需要一些预处理，典型任务包括删除异常值和趋势、填补缺失数据以及对数据进行归一化。我们的数据集不需要执行这些任务，因为 PhysioNet 竞赛的组织者已经进行了预处理。

通过侧重于最可能产生精确结果数据的提取和特征选取，可帮助改进机器学习算法。

提取特征

*特征提取*是机器学习最重要的部分之一，因为它将原始数据转变成适合机器学习算法的信息。特征提取消除了各类测量数据中的冗余现象，有助于学习阶段的泛化。泛化是避免对特定样本模型过拟合的关键。

> 了解更多
>
> 电子书*使用 MATLAB 进行机器学习*介绍了传感器、图像、视频和交易数据的常见特征提取技术。

选择特征

既然特征提取是第一步，我们需要避免使用过多特征。具有大量特征的模型在训练阶段需要更多计算资源，而使用太多特征会导致过拟合。

因而关键就在于找到能获得数据中基本规律的最小数量的特征。

特征选择是选择那些最贴合特定建模问题的特征，剔除不必要或冗余特征的过程。特征选择的常用方法包括逐步回归、序列特征选择和正则化。

如果您有一个大数据集和许多特征，则特征提取可能非常耗时。为加快这一过程，您可以使用 *Parallel Computing Toolbox™* 中的 **`parfor`** 循环

结构，跨核心（或扩展为一个群集）分布计算。

在心音示例中，根据我们的信号分类知识，我们提取以下类型的特征：

- 汇总统计：平均值、中值和标准差
- 频域：主频率、频谱熵和梅尔频率倒谱系数 (MFCC)

实际操作练习

示例脚本的 “Preprocess Data” 部分利用心音文件生成这些特征，但是由于处理过程需要几分钟，所以默认预生成的特征只是从 **`FeatureTable.mat`** 文件中读取的。要进行特征提取，先删除（或重命名）此文件，然后运行这部分脚本。

提取上述这些类型的特征，将从音频信号中产生 26 个特征。
为开发模型，必须以表格或矩阵格式采集特征。在我们的示例中，我们建一个表格，其中每一列代表一个特征。下图显示一些梅尔频率倒谱和声音标签（“正常”或“不正常”心音的真正类别）。

CC8	MFCC9	MFCC10	MFCC11	MFCC12	MFCC13	分类
11315	-0.28488	1.6218	-0.53338	-1.6926	-2.0239	"不正常"
2.394	0.10001	2.9168	-1.3413	-0.90557	-1.4914	"不正常"
.1322	-0.42672	2.3943	1.5946	-2.0933	-1.3693	"不正常"
.8257	0.865	2.4926	-0.91656	-0.55254	-2.2298	"不正常"
.5196	-0.64708	3.923	-0.5634	-1.7582	-0.4827	"不正常"

（部分）特征表。

步骤 3. 开发预测模型

开发预测模型是一个迭代过程，包含以下步骤：

i. 选择训练和验证数据。
ii. 选择分类算法。
iii. 反复训练和评估分类模型。

i.选择训练和验证数据

在训练实际分类器之前，我们需要将数据划分为训练和验证集。验证集用于在模型开发过程中测量精确度。对于大数据集，如心音数据，保留一定百分比的数据是适宜的；对于较小的数据集，建议交叉验证，因为这能最大化使用模型训练的数据量，通常得到泛化更好的模型。

当您选择“无验证”时，将在整个数据集上对模型进行训练和评估，不为验证保留数据。在整个数据集上重新训练模型可能严重影响其性能，尤其是在您的数据集非常有限的情况下。了解模型在训练集上表现的精确程度，可为您提供关于采取哪种方法进一步改进模型的指导。

ii.选择分类算法

没有一种机器算法能适合每个问题，找出正确的算法通常是个试错过程。但是，了解各种算法的主要特点，能让您选择先尝试哪种算法，并了解您所做的权衡。下表列出了流行分类算法的特点。

对于心音示例，我们使用 *Classification Learner app* 来快速比较分类器。

> 实际操作练习
>
> 可从“应用程序”选项卡中启动 Classification Lea ner app，或通过在命令行窗口中键入 **`ClassificationLearner`** 以编程方式启动。在开始新会话后，选择 **`feature_table`** 作为要处理的数据，使用在上一步骤中提取的所有 26 个特征（目前为止），然后选择 “Holdout Validation”（保留验证），保留 25% 的数据作为验证方法。

算法	预测速度	训练速度	内存使用	需要调优	一般评估
逻辑回归（和线性 SVM）	快	快	小	最小	擅长解决有线性决策边界的小问题
决策树	快	快	小	有些	通用性很好，但容易过度拟合
（非线性）SVM（和逻辑回归）	慢	慢	中等	有些	擅长解决许多二进制问题，能很好地处理高维度数据
最近邻	适中	最小	中等	最小	精度较低，但易于使用和解释
朴素贝叶斯	快	快	中等	有些	广泛用于文本，包括垃圾邮件过滤
集成	适中	慢	差异大	有些	对于中小规模的数据集，精度高，性能好
神经网络	适中	慢	中到大	很多	普遍用于分类、压缩、识别和预测

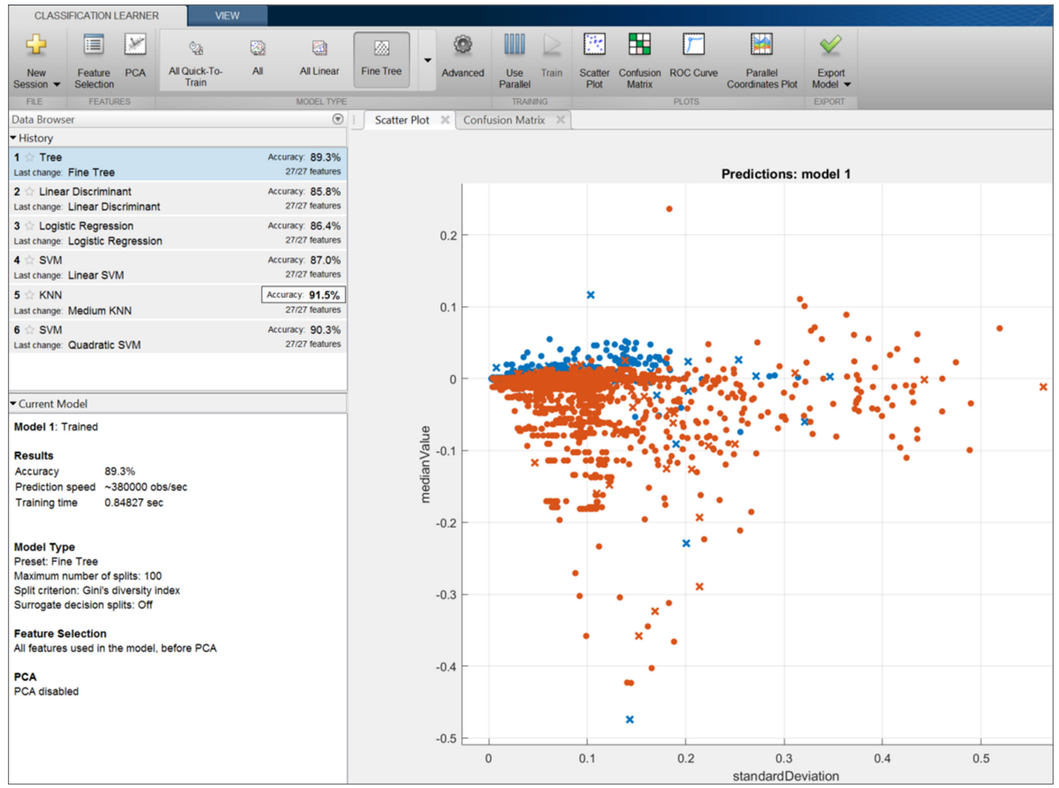
iii.反复训练和评估分类模型

我们现在已做好开始反复训练和评估模型的准备。我们既可以遵循蛮力计算法，运行所有算法（利用 Classification Lea ner app 很容易做到）， 也可以从特定分类任务的看似最佳拟合算法开始。

您可以选择单个分类器，或同时选择多个分类器，例如“All support vector machines”。然后，您可以并行训练分类器，在您的数据上比较它们的性能。对于您训练的每个分类器，以保留验证数据或使用交叉验证对精确度进行评估，具体取决于您在上一步骤中选择用于验证的数据。

对于我们的心音应用程序，最初结果显示（“Fine”）K近邻（KNN）分类器表现良好，随后是二次支持向量机（SVM）和（Fine）决策树。

我们的初始心音分类器可以达到 90% 以上的精确率。听起来不错，但对心脏筛查应用来说，这还不够。我们如何进一步改进模型的性能？



多个分类算法的初始比较。

了解更多

关于训练工作流程的概述，请观看*使用分类学习器应用程序进行数据分类* 5:12。

步骤 4. 优化模型

要提高模型性能，除了尝试其他（更复杂）算法以外，我们还需要更改流程。常用方法侧重于机器学习模型开发的以下某一方面：

- 调节模型参数。 通过更改主要模型参数，不使用其默认设置，我们几
  乎总能提高性能。
- 添加或修改训练数据。 添加训练数据很有用，直至达到过拟合点（当错误率开始增加时）。额外预处理能够利用我们可能忽视的数据本身（如损坏的数据、异常值或缺失值）补救任何问题。
- 变换或提取特征。 如果我们的当前特征集没有捕获数据中固有的所有变异，提取更多特征可能有作用。相比之下，如果我们看到过拟合迹象，可通过运用缩减技术，比如主成分分析（PCA）、线性判别分析（LDA）
  或奇异值分解（SVD），进一步减少特征。如果特征在尺度范围上变化
  很广，则可以借助归一化之类的特征变换。
- 进行特定于任务的权衡。 如果某些误分类不太受欢迎，我们可以运用成本矩阵，对特定预测类（例如，将实际心脏病误分类为正常）分配不

> 实际操作练习
>
> 运行示例脚本中的“Split data into training and testing sets”（将数据拆分为训练和测试数据）部分，以便您能够在保留的测试数据上评估各种优化技术的影响。

同的权重。

既然我们的模型在测试数据上几乎达到与训练数据上相同级别的精度，添加更多训练数据不太可能有所改进。为了进一步改进心音分类器的精确度，我们将先尝试一种更复杂的算法，然后介绍偏差和调节模型参数。

尝试更复杂的分类器

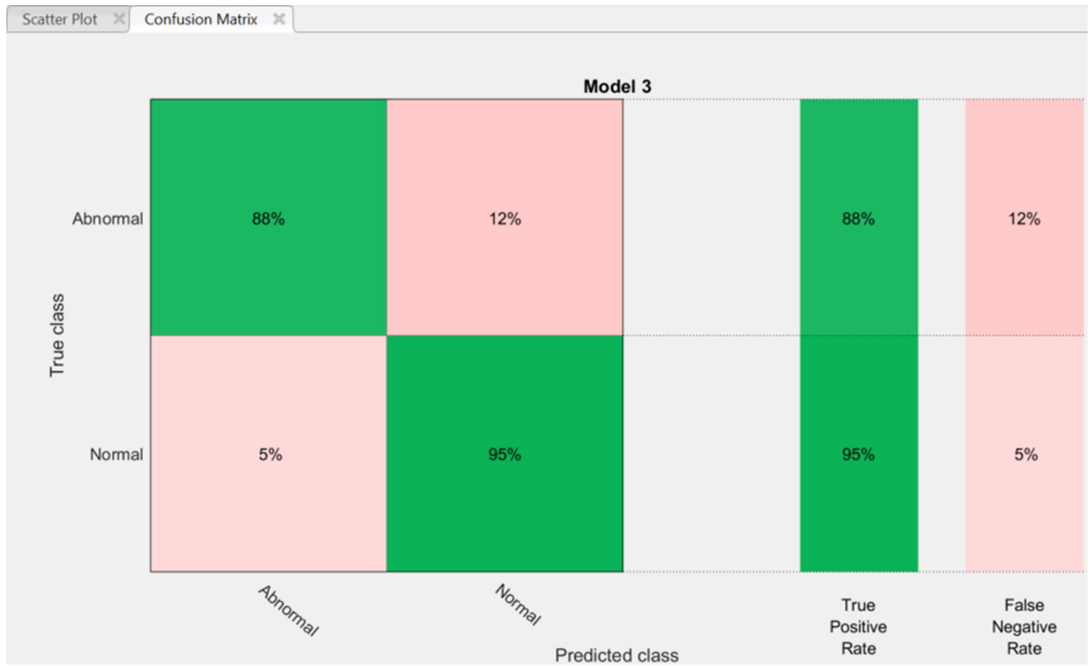
使用单独的分类树往往会过度拟合训练数据。为了克服这种倾向，我们将尝试集成多个分类树（通常称为“随机森林”）。使用心音数据，袋装(bagged) 决策树整体达到 93% 的精确度，在这种情况下，对最好的单独分类器没有改进。

介绍偏差

到目前为止，我们假定所有分类错误都同等不可接受，但情况并不总是这样。在心音应用程序中，例如，漏报（未能检测到实际心脏病）比误报
（错误地将正常心跳声分类为不正常）的后果严重得多。

为了探讨不同误分类之间的权衡，我们使用一种混淆矩阵。（通过在 Classification Lea ner app 内的散点图视图中切换，我们很容易得到一个混淆矩阵。）心音分类器的混淆矩阵显示，我们的初步模型只将 5% 的正常心音误分类为不正常，而将 12% 的不正常心音分类为正常。换言之，
虽然我们只误标了 5% 的健康心脏，但是，我们未能检测出 10% 以上的实际心脏病，这种情况在实际医疗中显然不可接受。

此示例表明，只根据总体精确度分析性能（此模型为 94% 左右）很容易
产生误导。当数据不平衡时会发生这种情况，在本例中，我们的数据集包含的正常心音大概是不正常心音的四倍。



为改进特定任务的性能，我们将调整我们的分类器，将实际心脏病的误分类降至最低。模型会将更多正常心音误分类为不正常，这种权衡是可以接受的。

评估 Classification Lea ner app 评估分类器中的偏差。

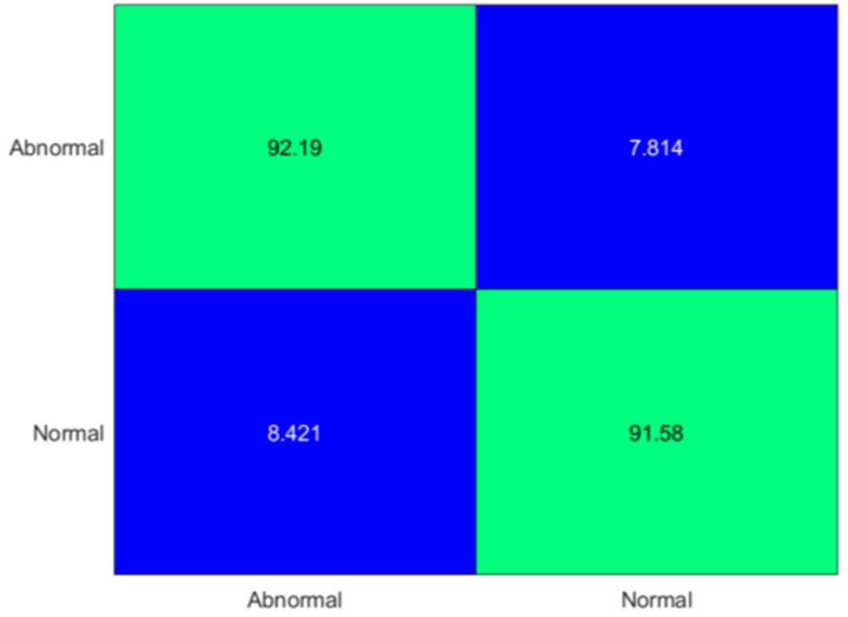
步骤 4. 优化模型 – 续

调整分类器的标准方法是引入成本函数，对不可取的误分类给予更高的罚值。

下列代码引入一个成本函数，按 20 的因数对不正常心音的误分类施加罚值。

通过引入成本函数，在精确度和漏报容差之间达到不同的平衡。

混淆矩阵显示，结果模型未能检测出不到 8% 的不正常心音，而将正常心音误分类为不正常的情况稍多一些（8%，相比之下，未调整模型中的误判率为 5%）。此模型的总体精确度仍高达 92%。



通过使用成本函数，减少不正常心音的误分类。

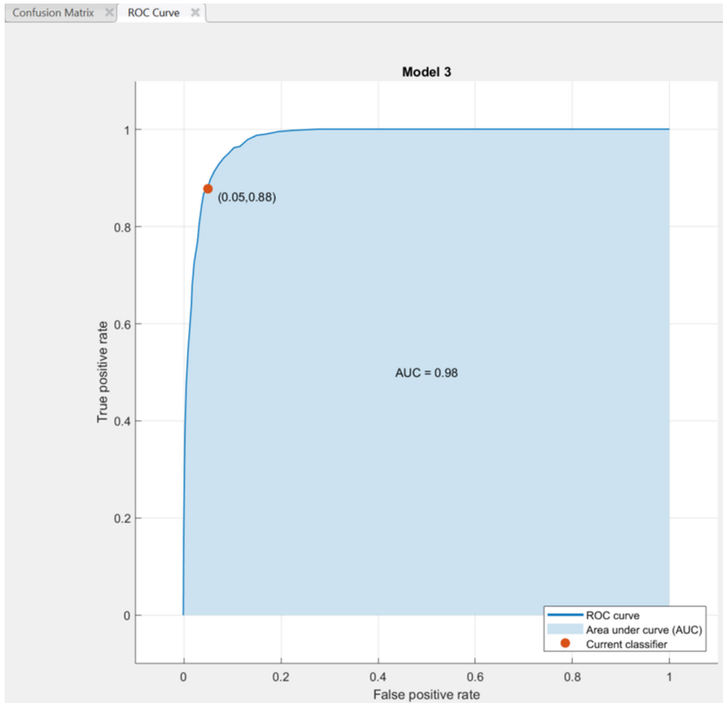
实际操作练习

运行示例脚本中的“Train the classifier with misclassification co ”（使用误分类成本训练分类器）部分。该脚本运用成本矩阵，同时执行超参数的贝叶斯优化。

MathWorks

调节模型参数

除了使用散点图和混淆矩阵探讨真正类与假正类之间的权衡，我们还可以使用受试者工作特征 (ROC) 曲线。ROC 曲线是以直观的方式探讨真正类与假正类之间权衡的有用工具。可以使用 ROC 曲线选择您的分类器的最佳工作点或界值，最大限度地减小由您的成本函数定义的总体“成本”。



袋装 (bagged) 树集成分类器的 ROC 曲线。

如前所述，可以通过调节机器学习算法参数来达到更好的拟合。识别能提供最佳模型的参数集的过程通常称为“超参数调优”。为使超参数调优更加高效，并提高找到最优参数值的机率，我们可以使用 MATLAB 中的自动网格搜索和贝叶斯优化。

网格搜索能彻底搜索参数值组合的有限集，但可能耗时很长。

贝叶斯优化则开发超参数空间的一个统计模型，旨在最大限度地减少找到最优值所需的试验数量。

示例脚本使用贝叶斯优化执行超参数调优，同时还引入成本函数。

了解更多

贝叶斯优化特点

使用特征选择来纠正误分类和过拟合

目前，我们已经使用了在训练模型时提取的所有 26 个特征。优化性能的最后一步留给了特征选择：移除冗余或不承载有用信息的特征。此步骤可以降低计算成本和存储要求，并且产生一个不太可能过拟合的更简单模型。

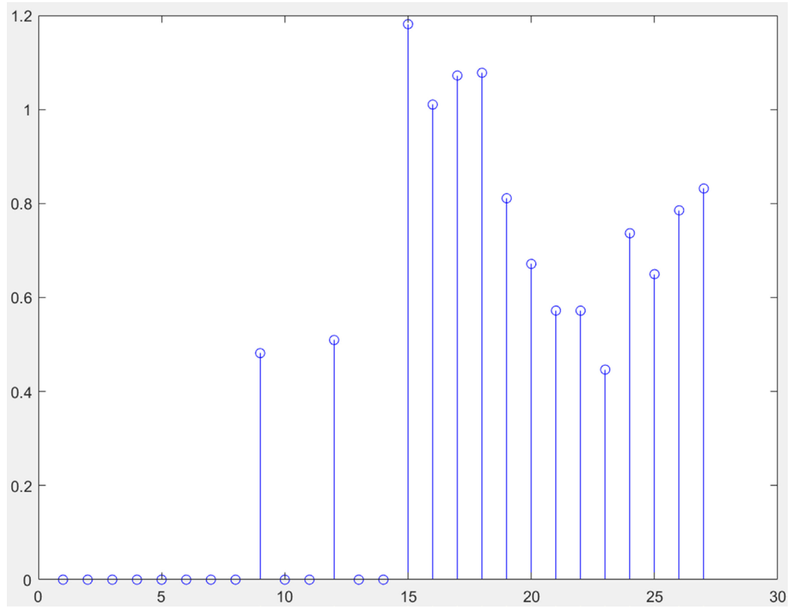
缩减特征集对心音应用程序十分重要，因为它可减小模型的大小，使之更容易在嵌入式设备上部署。

特征选择方法包括系统化地评估特征子集（这在计算上很昂贵），以及通过对每个特征应用权重，将特征选择融入模型建造过程（使用较少的特征有助于最大限度地减少在训练过程中使用的目标函数）。

针对我们的心音分类器，我们运用邻域分量分析（NCA），这是一个强大的、能处理非常高维数据集的特征选择技术。NCA 揭示，大约一半的
特征对模型没有重要作用。因此，我们可以减少特征数量，从 26 个减至 15 个。

实际操作练习

运行示例脚本中的“Perform feature selection using Neighborhood Component Analysis”（使用邻域分量分析进行特征选择）部分，随后运行“Train model with selected features”（使用选择的特征训练模型）。



自动特征选择的结果，使用邻域分量分析识别最相关的特征。

了解更多

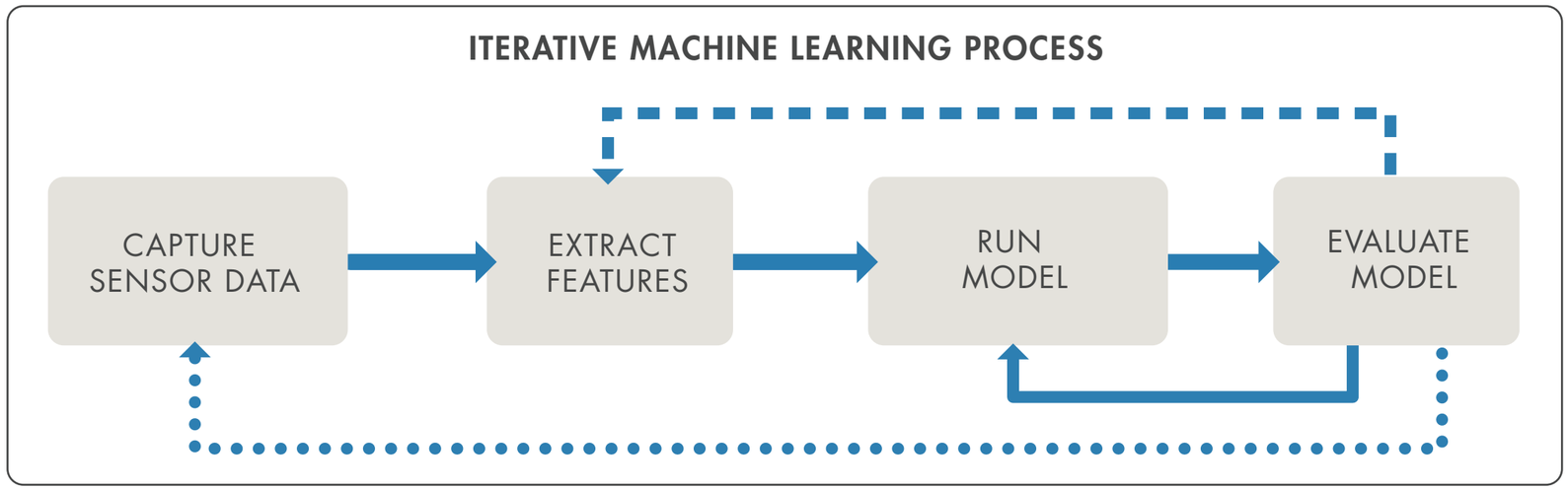
选择特征对高维数据进行分类

为了评估只选择 15 个特征的性能影响，我们在激活超参数调优和成本函数的情况下重新训练模型。更新后的模型未能检测出不正常心脏状况仅占 6%（比以前稍好一点），但将 15% 的正常情况错误地分类为不正常，
稍微降低了总体精确度。在实际医疗中，任何阳性结果随后都会进行额外检验，从初始筛查中排除 15% 误报中的绝大多数。

反复尝试其他算法

为进一步改进模型，我们可以用不同的算法尝试相同系列的优化步骤，因为各种优化技术能有多少改进随算法而异。您可以重复前面部分所述的迭代过程，例如，再次访问 KNN 算法，该算法初始表现很好。您甚至可以后退到特征提取阶段去寻找其他特征。为找出最佳模型，很有必要在机器学习工作流程的不同阶段之间反复试验。当您能够通过评估当前模型推断要尝试什么的时候，您就掌握了机器学习。

在我们的心音示例中，我们准备进行下一步骤，也是最后步骤：部署分类器。



步骤 5. 将分析方法部署到生产系统

机器学习应用程序可以部署到桌面的生产系统、企业 IT 系统（现场或云端）和嵌入式系统。对于桌面和企业 IT 系统，可以在服务器上部署。
MATLAB 应用程序，既可将代码编译为独立应用程序，也可与采用其他语言（如 C/C++、Python®、Java® 或 .NET）编写的应用程序集成。许多
嵌入式应用程序需要模型采用 C 代码部署。

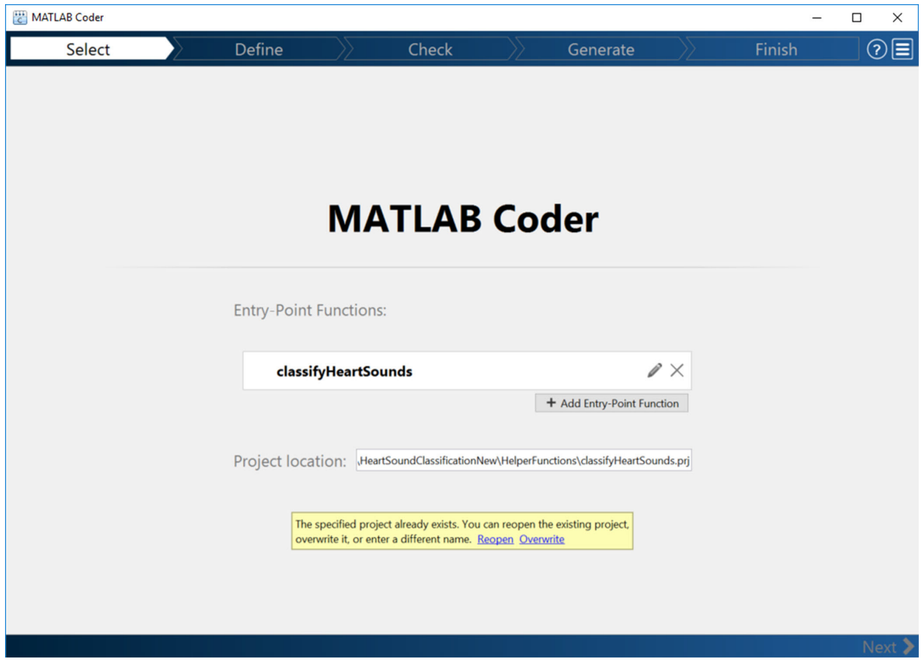
MATLAB Coder™ 通过自动将 MATLAB 转换成 C 代码，轻松实现在嵌入式系统上的部署。

生成 C 代码

我们的心音诊断应用程序将在可穿戴心脏监护器等医疗设备或手机 App 上运行。为了做好应用程序部署准备，我们通过执行以下步骤从该模型生成 C 代码：

1. 将训练模型另存为精简模型。
2. 启动 MATLAB Coder。
3. 创建入口函数，采用原始传感器数据作为输入，将患者的心音分类为正常或不正常。

MATLAB Coder 自动生成相应的 C 代码。



用 MATLAB Coder 生成 C 代码。

所需工具

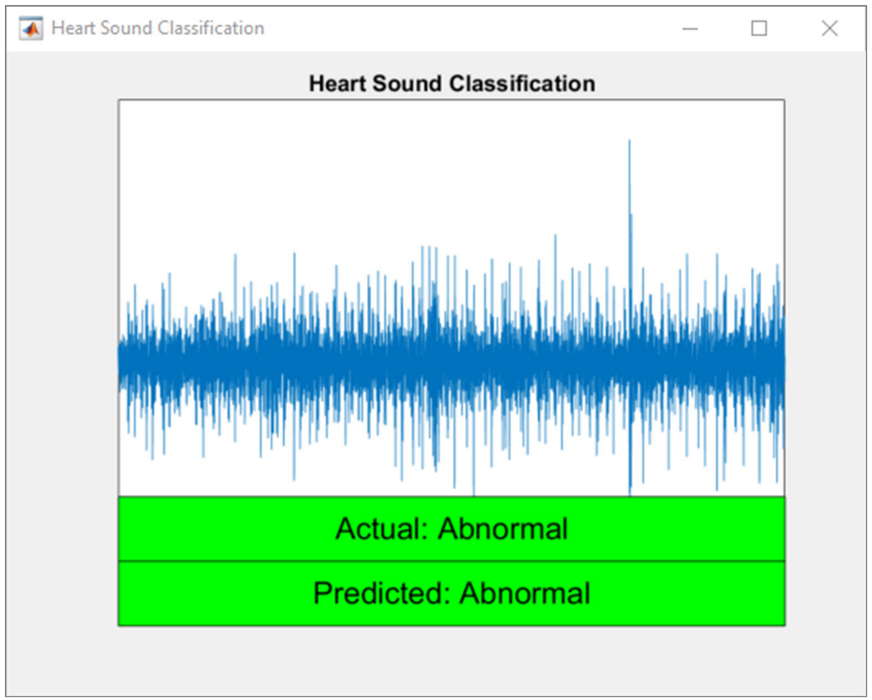
申请 *MATLAB Coder 试用版*。

实际操作练习

运行示例脚本中的“Validate final mode ”（验证最终模型）部分。该应用程序显示 “validation”（验证）数据集里几百个心音的预测类和真正类，该数据集是在刚开始时与训练数据一起下载的。

步骤 5. 将分析方法部署到生产系统 – 续

为了验证生成的 C 或 C++ 代码，我们创建一个简单的原型应用程序，交互式地对来自验证数据集的文件进行分类。以上描述的这个三步过程生成了一个可从 MATLAB 内调用的可执行版本。我们将在我们的原型中使用此版本（用 MATLAB 编写，而不是 C）。



在 MATLAB 中验证分类器。

此时，我们已做好在手持式设备上实现该应用的准备。

了解更多

- *嵌入式代码生成*
- *MATLAB 可轻松转换为 C 语言* (55:15)

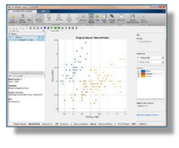
机器学习的基本工具

使用 MATLAB 和专业化的工具箱，无需机器学习或数据科学的广博知识，您也可以开发、调优和部署预测分析应用。MATLAB 和相关产品提供了构建和部署您的分析应用所需的全部工具：

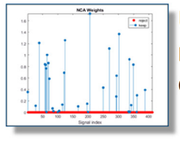
- 数据处理功能简化了耗时的任务，包括处理缺失数据或异常值、消除噪声以及具有不同采样率的时间配准数据
- 专业化的机器学习 App 可加快您的工作流程，让您快速比较和选择算法
- 程序化的工作流程用于剔除不必要的特征和微调模型参数，实现稳健的性能
- 机器学习工作流程扩展工具可扩展到大数据和计算集群
- 自动代码生成工具可将您的分析应用迅速部署到嵌入式目标系统

内置的函数和算法支持专业化应用，如深度学习、计算机视觉、金融、图像处理、文本分析和自主代理。

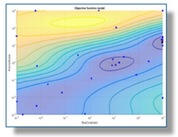
使用 MATLAB 实现机器学习



Use apps to quickly train and evaluate algorithms.



Use feature selection to reduce model size and prevent overfitting.



Use hyperparameter tuning and cost matrix to improve accuracy.

Use automatic code generation to rapidly deploy your analytics to embedded targets.

机器学习的基本工具 – 续

访问和探查数据

广泛的数据支持：

- 可处理信号、声音、图像、金融数据、文本、地理空间和其他格式。
- 处理和分析各种信号。

用于访问和探查数据的相关产品：

- *Database Toolbox™*
- *Datafeed Toolbox™*
- *OPC Toolbox™*
- *Signal Processing Toolbox™*
- *Vehicle Network Toolbox™*

预处理数据和提取特征

高质量的库和域工具：

- 使用行业标准算法，在金融、统计、
  信号处理、图像处理、文本分析和计算机视觉方面实现特征提取。
- 利用过滤、特征选择和变换来改进模型。

用于专业化应用的相关产品：

- *Computer Vision System Toolbox™*
- *Fuzzy Logic Toolbox™*
- *Image Processing Toolbox™*
- *Optimization Toolbox™*
- *Signal Processing Toolbox™*
- *Statistics and Machine Learning Toolbox™*
- *System Identification Toolbox™*
- *Text Analytics Toolbox™*
- *Wavelet Toolbox™*

开发和优化预测模型

交互式、**App** 驱动的工作流程：

- 快速训练和比较模型。
- 专注于机器学习，而不是编程。
- 选择最佳模型和微调模型参数。
- 将计算扩展到多核心或集群。

用于改善和调节模型的专业化 **App** 和产品：

- *Classification Learner App*
- *Deep Learning Toolbox™*
- *Parallel Computing Toolbox™*
- *Regression Learner App*
- *Statistics and Machine Learning Toolbox™*

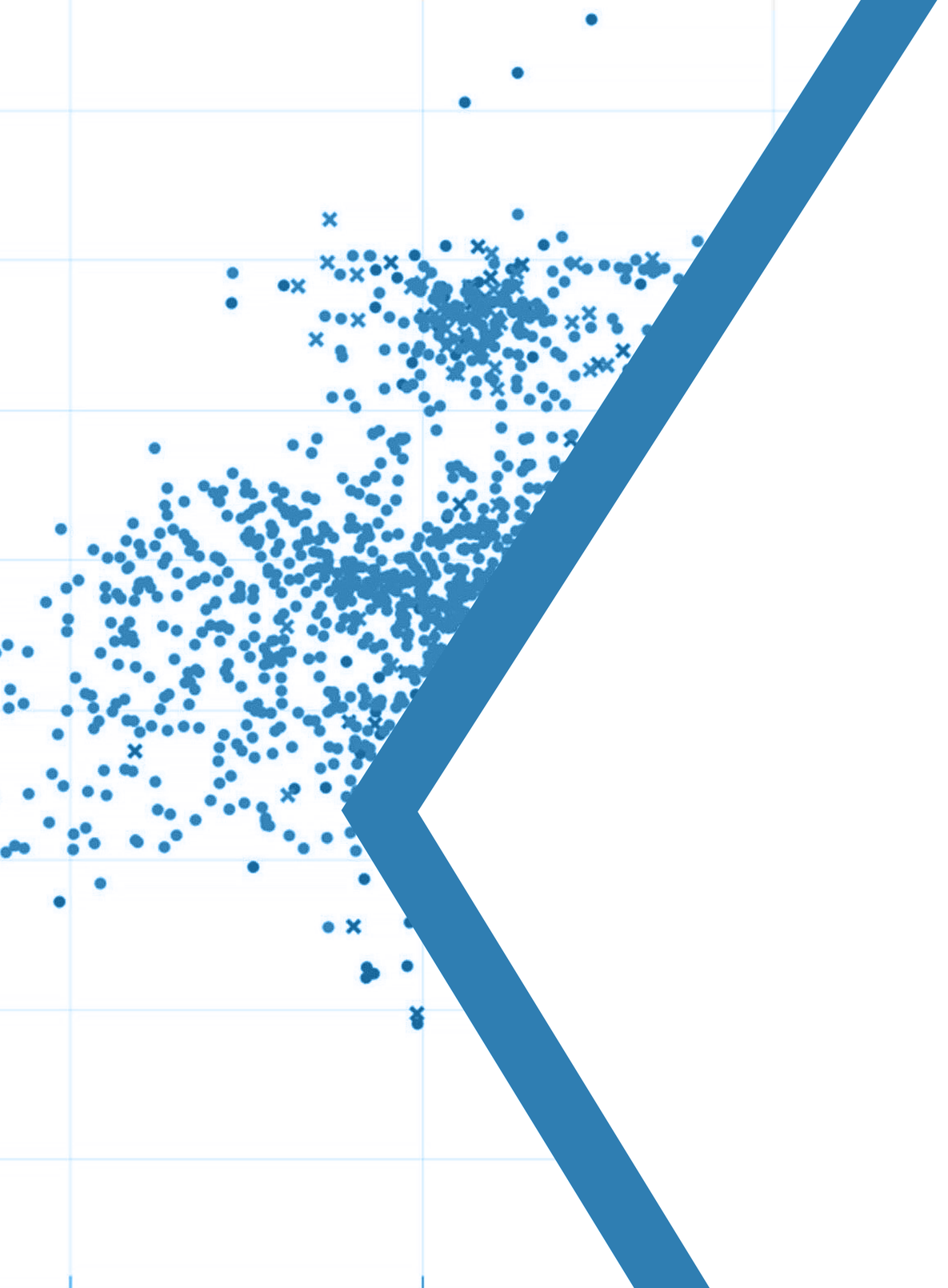
将分析方法部署到生产系统

高质量的库和域工具：

- 获得将分析转变为生产的工具。
- 生成要部署到嵌入式目标系统的代码。
- 部署到范围广泛的目标平台和企业系统。

用于专业化应用的相关产品：

- *HDL Coder™*
- *MATLAB Coder™*
- *MATLAB Compiler™*
- *MATLAB Compiler SDK™*
- *MATLAB Production Server™*



准备更深入地钻研？

下载

心音分类应用程序的 MATLAB 代码

观看

使用心音分类示例的机器学习 (22:03)

阅读

通过 MATLAB 处理大数据
什么是深度学习？
MATLAB 深度学习简介
使用 MATLAB 在云端进行并行计算
预测分析