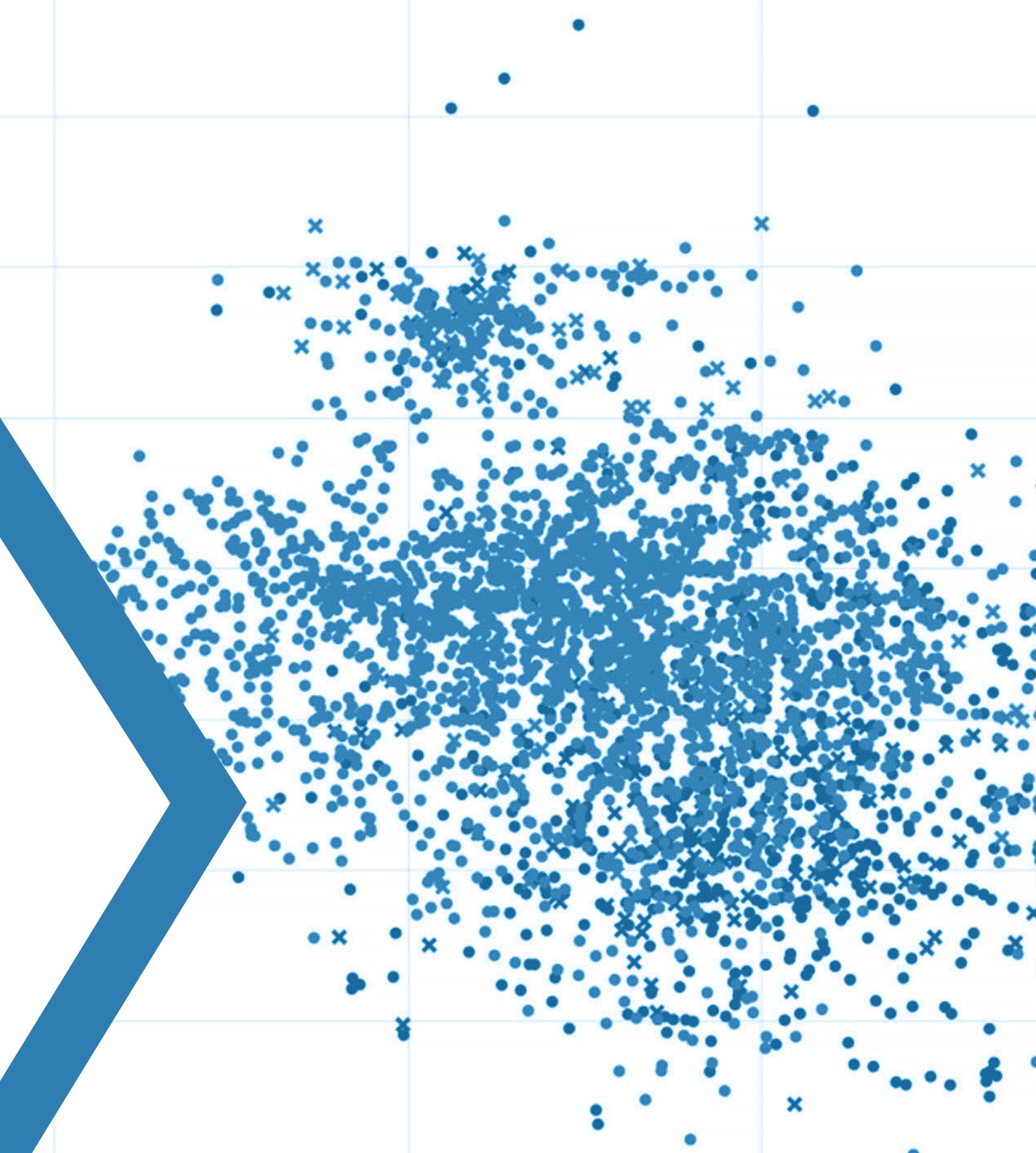




機械学習をマスターする

MATLAB ステップ・バイ・ステップ ガイド



はじめに

この ebook は、機械学習の基礎を概説し、教師ありの手法と教師なしの手法を紹介した [MATLAB による機械学習 の続編](#)です。

心音分類器を例として使用し、データの読み込みから学習済みモデルの配布まで、実際の機械学習アプリケーションを開発するワークフロー全体について説明します。各学習段階で正確なモデルを導くために不可欠な手法を示し、アルゴリズムの選択、モデルパラメーターの最適化、過適合の回避など、より難しい学習タスクの習得を支援します。

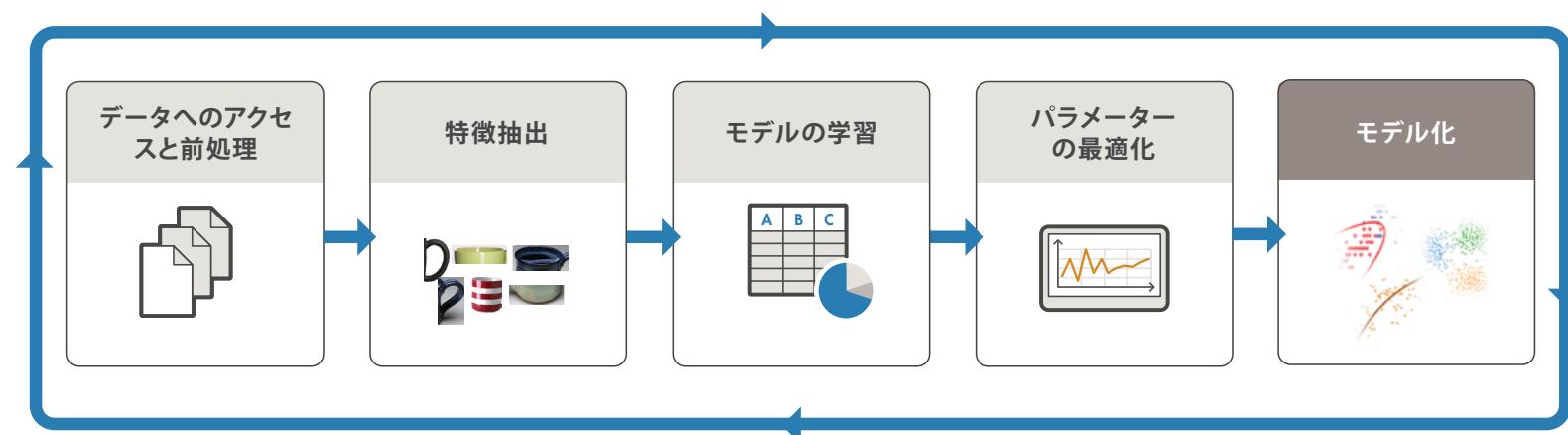
この ebook では、新しいデータをモデルに学習させて特徴を抽出し、組み込みデバイスで配布するコードを生成することにより、モデルを予測ツールに変換する方法についても学習します。

基礎の確認

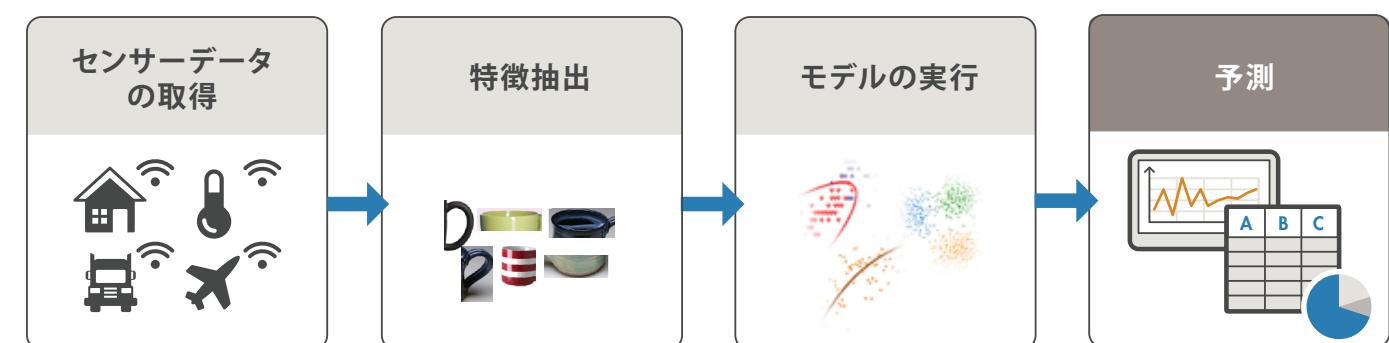
[機械学習とは](#) (2:39)

[MATLAB による機械学習 \(ebook シリーズ\)](#)

学習: 満足な性能が得られるまで反復を繰り返します。



予測: 学習済みのモデルをアプリケーションに統合します。



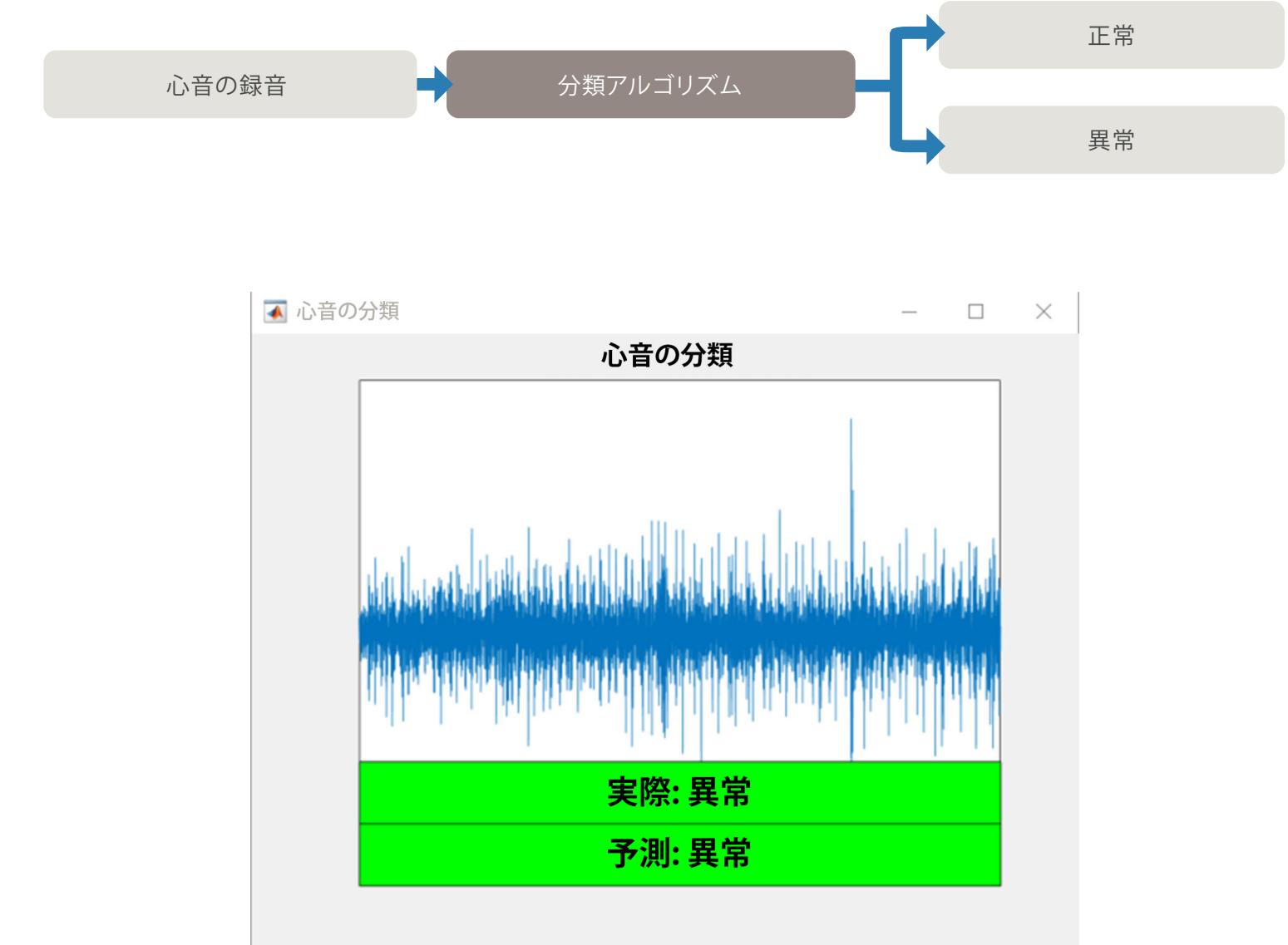
MATLABによる心音分類アプリケーション開発

心音は、心臓病変の早期診断のための貴重な情報源です。正常な心音と異常な心音とを区別するには、特別な訓練を受けた臨床医が必要です。ここでの目標は、異常な心音を識別できる機械学習アプリケーションを開発することです。心音の監視アプリケーションを使用すれば、専門医が不在の場合でも通常の医療スタッフが心臓病をスクリーニングすることや、患者が自身をモニターすることもできます。

このアプリケーションの開発では、以下の手順に従います。

1. データへのアクセスと探索を行う
2. データを前処理して特徴を抽出する
3. 予測モデルを開発する
4. モデルを最適化する
5. 運用システムに解析を配布する

提供される例を実際に自分で試してみることをお勧めします。以下の MATLAB® コードをダウンロードし、ebook 内の演習内容に従ってください。



心音分類器とプロトタイプのアプリの概略図

演習に必要なツール

[機械学習向け MATLAB 30 日間無料評価版をダウンロードする](#)
[演習用に心音診断アプリケーションの MATLAB コードをダウンロードする](#)

手順 1. データにアクセスして探索する

この例では、5 秒から 120 秒までの長さで録音された数千もの心音で構成される PhysioNet/Computing in Cardiology Challenge 2016 のデータセットを使用します。

演習

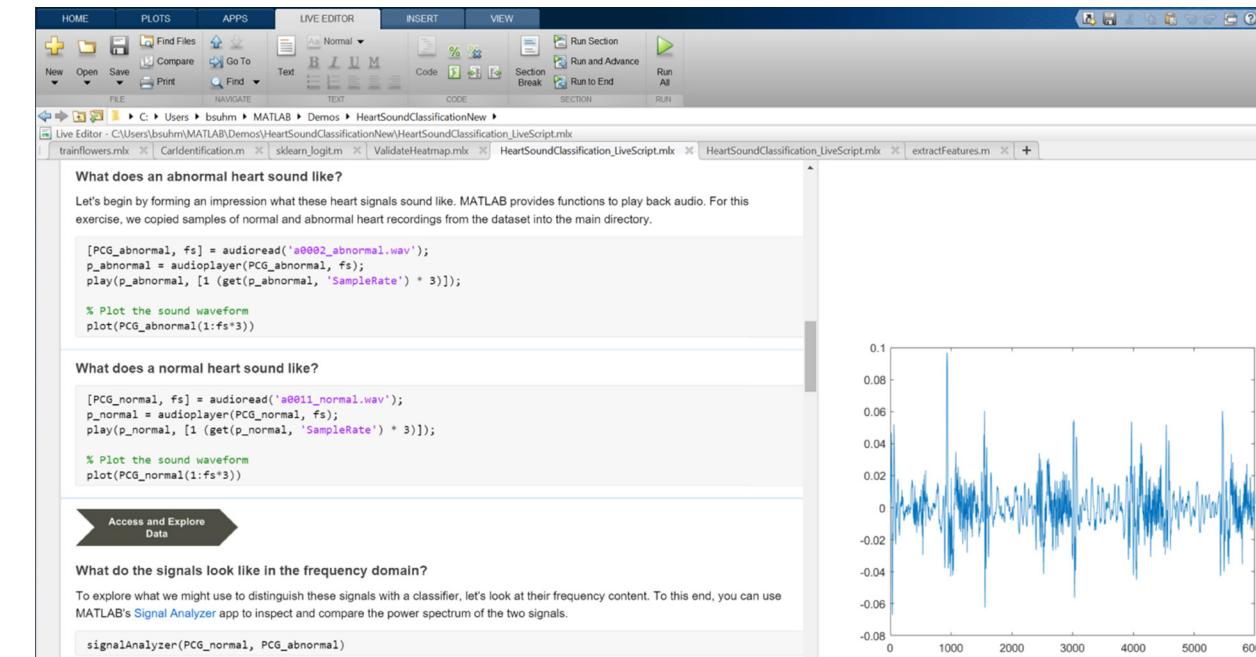
Live Editor スクリプトの最初のセクションを実行します。このスクリプトにより PhysioNet/Computing in Cardiology Challenge 2016 から心音データセットがローカル ワークスペースにダウンロードされます。

このデータセットには、モデルの学習用に 3240 の録音データとモデルの検証用に 301 の録音データが含まれています。データをダウンロードした後、学習セットと検証セットを個別のフォルダーに保存します。これは機械学習の標準的な手順です。

データの調査

どのような機械学習プロジェクトにおいても、最初の手順は処理するデータの種類を理解することです。データを探索する一般的な方法には、いくつかのデータの検証、可視化の作成、パターンを識別する（より高度な）信号処理手法またはクラスタリング手法の適用が含まれます。

正常な心音と異常な心音とを区別するのに何が必要なのかを理解するため、いくつかのサンプルを聴くことから始めます。



MATLAB Live Editor で生成された異常な心音のプロット

演習

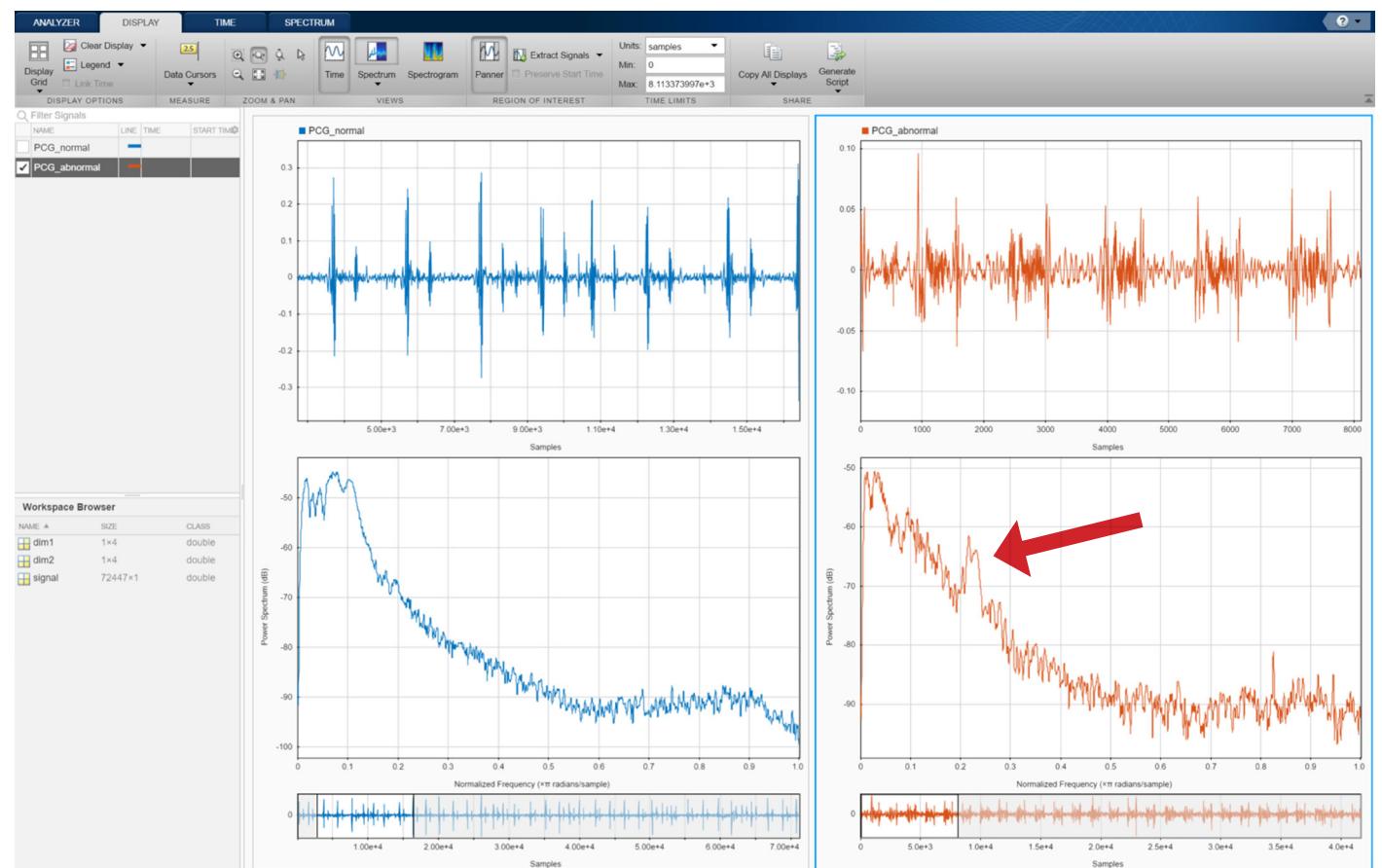
コード例のメイン ディレクトリに提供された異常な心音と正常な心音の例を再生します。MATLAB にはオーディオファイルを再生するためのオーディオリーダーとオーディオプレーヤーが用意されています。これらのファイルはスクリプト例の “What does an (ab)normal heart sound like?” セクションで使用されます。

異常な心音は心拍間にノイズがあり、周波数が高いことがわかります。正常な心音はより規則的で、心拍間は無音です。

手順 1. データにアクセスして探索する – 続き

信号の解析

Signal Processing Toolbox™ の信号アナライザーアプリを使用し、周波数領域に信号を並べて表示することにより、コードを記述しなくてもこれらの信号間の相違点をより詳しく理解できます。



信号アナライザーアプリに表示される正常な心音(左)と異常な心音(右)
赤い矢印は 200 Hz 前後の周波数成分における異常な心音に対するスパイクを示しています。

この最初のデータ調査と分類タスクの後、前処理を行う全データをメモリに読み込みます。MATLAB を使用すると、複数のディレクトリにまたがって存在する大規模なデータセットの読み込みを簡略化できます。つまり、まず完全なデータセットとしてのハンドルを作成してから、ひとつか複数の塊ごとにデータをメモリに読み込むことができます。大規模なデータセットの場合には複数の計算リソース間で実行を分散することもできます。

各録音データは異常または正常としてラベル付けされたオーディオファイルです。このデータセットにおいて、私たちは各ファイルの真のカテゴリー ("グラウンドトゥルース") を知っているため、既知の入力データのセットとそのデータに対する既知の応答(出力)を使用して、新しいデータに対する応答を適切に予測するようにモデルを学習させる[教師ありの機械学習](#)を適用できます。

演習

学習データと対応する真のカテゴリーをメモリに読み込むために、スクリプト例の "Prepare to read the data into memory" セクションと "Create a table with filenames and labels" セクションを実行します。

手順 2. データを前処理して特徴量を抽出する

ほとんどのデータセットでは、特徴量を抽出する前にいくつかの前処理が必要です。一般的なタスクには、外れ値とトレンドの削除、欠損データの補完、データの正規化などがあります。ここで使用するデータセットは PhysioNet Challenge の主催者によって前処理済みなので、これらのタスクは必要ありません。

特徴量の抽出と選択を行うと正確な結果を生成する可能性が最も高いデータに焦点を当てるようになるため、機械学習アルゴリズムの性能が向上します。

特徴量の抽出

特徴抽出は生のデータを機械学習アルゴリズムに適した情報に変換することから機械学習の最も重要な機能の一つであると言えるでしょう。特徴抽出によりさまざまなタイプの測定データに存在する冗長性が削除され、学習段階の汎化が容易になります。汎化はモデルを特定のデータに過適合するのを回避するために不可欠なものとなります。

関連情報

[MATLABによる機械学習 \(ebookシリーズ\)](#)では、センサー、画像、動画、トランザクションデータの一般的な特徴抽出手法について説明しています。

特徴量の選択

特徴抽出は最初の手順ですが、使用する特徴量の数が多すぎないようにしなければなりません。たくさんの特徴量をもつモデルは学習段階で必要な計算リソースが増えるだけでなく、あまりにも多くの特徴量を使用することで結果的に過適合をひきおこします。

ここでの課題は、データの本質的なパターンを捉え得る最小限の数の特徴量を見つけることです。

特徴選択は、特定のモデリングタスクに関連性の高い特徴量を選択し、不要な特徴量または冗長な特徴量を削除するプロセスです。特徴量を選択する一般的な方法には、ステップワイズ回帰、逐次特徴選択、正則化などがあります。

大規模なデータセットにおいて大量の特徴量がある場合、特徴抽出に時間がかかる場合があります。こうしたプロセスを高速化するために、[Parallel Computing Toolbox™](#)で `parfor` ループの構成を使用し、使用可能な複数のコアに計算を分散（または一つのクラスターにスケーリング）させることができます。

手順 2. データを前処理して特徴量を抽出する – 続き

心音の例では、信号分類の知識を基にして次のタイプの特徴量を抽出します。

- 要約統計量: 平均、中央値および標準偏差
- 周波数領域: 優位周波数、スペクトルエントロピー、メル周波数ケプス トラム係数 (MFCC)

上記のタイプの特徴を抽出すると、オーディオ信号から 26 の特徴が得られます。

演習

スクリプト例の “Preprocess Data” セクションは心音ファイルからこれらの特徴量を生成しますが、この処理は数分かかるため、既定では事前に生成された特徴量をファイル `FeatureTable.mat` から読み込むだけになります。特徴量を抽出するには、このセクションを実行する前にこのファイルを削除（または名前を変更）しなければなりません。

モデルを開発する場合、特徴量は表または行列形式で取得しなければなりません。この例では、各列が特徴量を示す表を作成します。次の図は、メル周波数ケプストラムと音源のラベル（“正常” または “異常” な心音の真のカテゴリー）をいくつか示しています。

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

(部分的な) 特徴量の表

手順 3. 予測モデルを開発する

予測モデルの開発は、以下の手順の反復処理です。

- i. 学習データと検証データを選択する
- ii. 分類アルゴリズムを選択する
- iii. 分類モデルの学習と評価を反復処理する

i. 学習データと検証データの選択

実際の分類器に学習させる前に、学習セットと検証セットにデータを分ける必要があります。検証セットはモデル構築時に精度を測るために使用します。心音データなどの大きなデータセットの場合、データの特定の割合をホールドアウトすることが適切です。小さいデータセットの場合は、モデルの学習に使用するデータ量が最大化されるため交差検定が推奨され、通常、高い精度で汎化を行うモデルが得られます。

[検定なし] を選択すると、モデルはデータセット全体で学習および評価され、データはホールドアウト検定されません。特に非常に限られたデータセットの場合では、データセット全体でモデルを再学習させることは、学習器の性能に重大な影響を与えます。また、学習セットでモデルがどれくらいの精度で動作するかを知ることで、さらにモデルを改良するための指針が得られます。

手順 3. 予測モデルを開発する – 続き

ii. 分類アルゴリズムの選択

一つの機械学習のアルゴリズムがすべての問題に有効であることはなく、適切なアルゴリズムを見つけ出す作業は、しばしば試行錯誤を伴うものとなります。しかし、さまざまなアルゴリズムの主な特徴を知っていると、最初に試すアルゴリズムを選択したり、アルゴリズムの持つトレードオフについて理解したりするのに役立ちます。以下の表に、一般的な分類アルゴリズムの特徴を一覧表示します。

心音の例には、[分類学習器アプリ](#)を使用して分類器を高速に比較します。

演習

分類学習器アプリは [アプリ] タブから起動することも、コマンド ウィンドウに `ClassificationLearner` と入力してプログラムで起動することもできます。新しいセッションを開始したら、処理するデータとして `feature_table` を選択し、前の手順で抽出した 26 の全特徴を使用して (現時点では)、25% のデータをホールドアウトした [ホールドアウト検定] を検証法として選択します。

| アルゴリズム | 予測速度 | 学習速度 | メモリ使用量 | 必要な調整 | 一般的な評価 |
|--------------------------|------|------|-----------|-------|--------------------------------|
| ロジスティック回帰 (および線形 SVM) | 高速 | 高速 | 小 | 最小 | 線形判別境界のある小さい問題に良好 |
| 決定木 | 高速 | 高速 | 小 | ある程度 | 全般で良好だが過適合しやすい |
| (非線形) SVM (およびロジスティック回帰) | 低速 | 低速 | 中 | ある程度 | 多くのバイナリ問題に良好で、高次元のデータを効果的に処理する |
| 最近傍法 | 中程度 | 最小 | 中 | 最小 | 精度は低下するが、使用と解釈が簡単 |
| 単純ベイズ | 高速 | 高速 | 中 | ある程度 | スパムのフィルター処理など、テキストで広く使用される |
| アンサンブル | 中程度 | 低速 | 場合によって異なる | ある程度 | 小～中サイズのデータセットに対して高精度で良好な性能 |
| ニューラル ネットワーク | 中程度 | 低速 | 中から大 | 多数 | 分類、圧縮、認識および予測で一般的 |

手順 3. 予測モデルを開発する – 続き

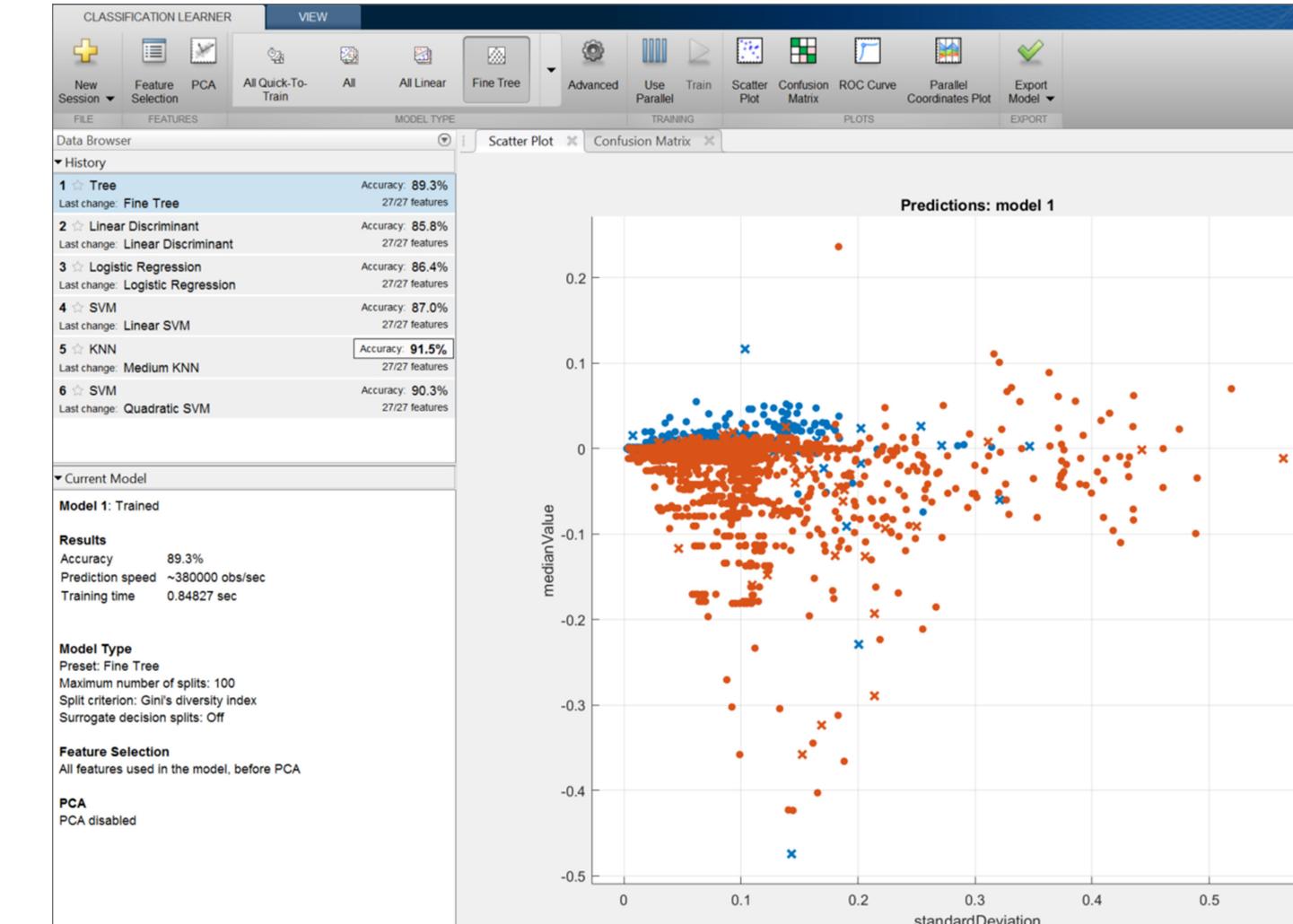
iii. 分類モデルの学習と評価の反復処理

ここまでで、モデルの学習と評価を反復処理する準備ができました。総当たりですべてのアルゴリズムを実行（分類学習器アプリで簡単に実行可能）するか、特定の分類タスクに最も適していると思われるアルゴリズムから開始することもできます。

個別の分類器または“すべてのサポートベクター マシン”など複数の分類器を同時に選択できます。次に、分類器を並列で学習させ、データでそれぞれの性能を比較できます。学習させる各分類器の精度は、ホールドアウトされた検定データか交差検定で推定されます。これは、一つ前の手順でどのデータを検定に使うかを選択したかに依存して決まります。

ここで使用する心音診断アプリケーションの場合、初期結果から、“細かい” k 最近傍 (KNN) 分類器が適切に機能し、その後に 2 次サポートベクター マシン (SVM) と(細かい) 決定木が続くことが考えられます。

初期の心音分類器では、90% を超える精度が得られます。これは良好な精度に見えますが、心臓のスクリーニング アプリケーションとしては不十分です。モデルの性能をさらに向上するには、どうしたらよいでしょうか。



複数の分類アルゴリズムの最初の比較

関連情報

学習ワークフローの概要については、
[分類学習器アプリを使用したデータの分類 \(5:12\)](#) をご覧ください。

手順 4. モデルを最適化する

モデルの性能を向上するには、その他の（より複雑な）アルゴリズムを試すのでなければ、そのプロセスに変更を加えなければなりません。一般的なアプローチでは、モデル構築の次のいずれかの側面に焦点を当てます。

- **モデルパラメーターの調整** 主要なモデルのパラメーターをデフォルトの設定から変更することで、ほぼ常に性能を向上できます。
- **学習データの追加または変更** 過適合点（誤り率が増加し始める点）に到達するまでであれば、学習データの追加が助けになることがあります。また、追加の前処理により、破損したデータ、外れ値、欠損値などの見落としがちなデータ自体の問題を修正することができます。
- **特徴量の抽出または変換** 現在の特徴量のセットがデータに固有の動きをすべて捉えていない場合、追加の特徴量を抽出すると役立つ可能性があります。一方、過適合の兆候が見られた場合は、主成分分析 (PCA)、線形判別分析 (LDA)、特異値分解 (SVD) などの低次元化手法を適用し、さらに特徴量の数を減らすことができます。特徴量のスケールが大きく異なる場合は、正規化などの変換が役立つ可能性があります。
- **タスク固有のトレードオフ** 他よりも望ましくない誤判別がある場合、特定の予測クラスに異なる重みを代入するコスト行列を適用できます（実際の心臓病を正常とする誤判別など）。

この例のモデルではテストデータにおいてほぼ学習データと同じレベルの精度レベルが得られたため、学習データをさらに追加しても精度はこれ以上に向上しそうにはありません。この心音分類器の精度をさらに高めるには、まずはより複雑なアルゴリズムを試し、バイアスを導入した後で、更にモデルパラメーターの調整を行います。

演習

ホールドアウトしたテストデータでさまざまな最適化手法の影響を評価できるように、スクリプト例の“Split data into training and testing sets”セクションを実行します。

より複雑な分類器の試行

個別の分類木を使用すると学習データが過適合になる傾向があります。この傾向を解決するために、分類木のアンサンブル（一般的に“ランダムフォレスト”と呼ばれる）を試します。心音データでは、バギングされた決定木アンサンブルで 93% の精度が得られます。この場合、最適な個別の分類器に改善はみられません。

手順 4. モデルを最適化する – 続き

バイアスの導入

ここまででは、分類誤差がすべて同等に望ましくないと仮定していますが、必ずしもそうであるとは限りません。心音アプリケーションでは、たとえば偽陰性（実際の心臓病の検出に失敗すること）は偽陽性（正常な心音を異常と誤って分類すること）よりもはるかに深刻な結果をもたらします。

さまざまな誤判別のトレードオフを探るために、混合行列を使用します（分類学習器アプリで散布図ビューを切り替えると、混合行列を簡単に表示できます）。心音分類器の混合行列は、この仮のモデルで正常な心音の5%のみが異常と誤判別され、異常な心音の12%が正常と分類されていることを示しています。言い換えれば、誤ってフラグが立てられる健康な心臓は5%であるのに対し、検出に失敗する実際の心臓病は10%以上であり、このような状況は医療現場で明らかに受け入れられません。

この例は、全体的な精度（このモデルの場合はおよそ94%）のみを基準にして性能を解析すると、誤りを招く可能性がいかに高くなるかを示しています。こうした状況はクラス毎のデータ数が不均衡な場合によく発生しますが、いまの例では正常な心音の数が異常な心音の数の約4倍もデータセットに含まれています。

タスク固有の性能を向上するために、分類器にバイアスを導入し、実際の心臓病の誤判別を最小化します。このトレードオフにおいてはモデルはより正常な心音を異常と誤判別しやすくなりますが、これは許容することができるものと言えるでしょう。

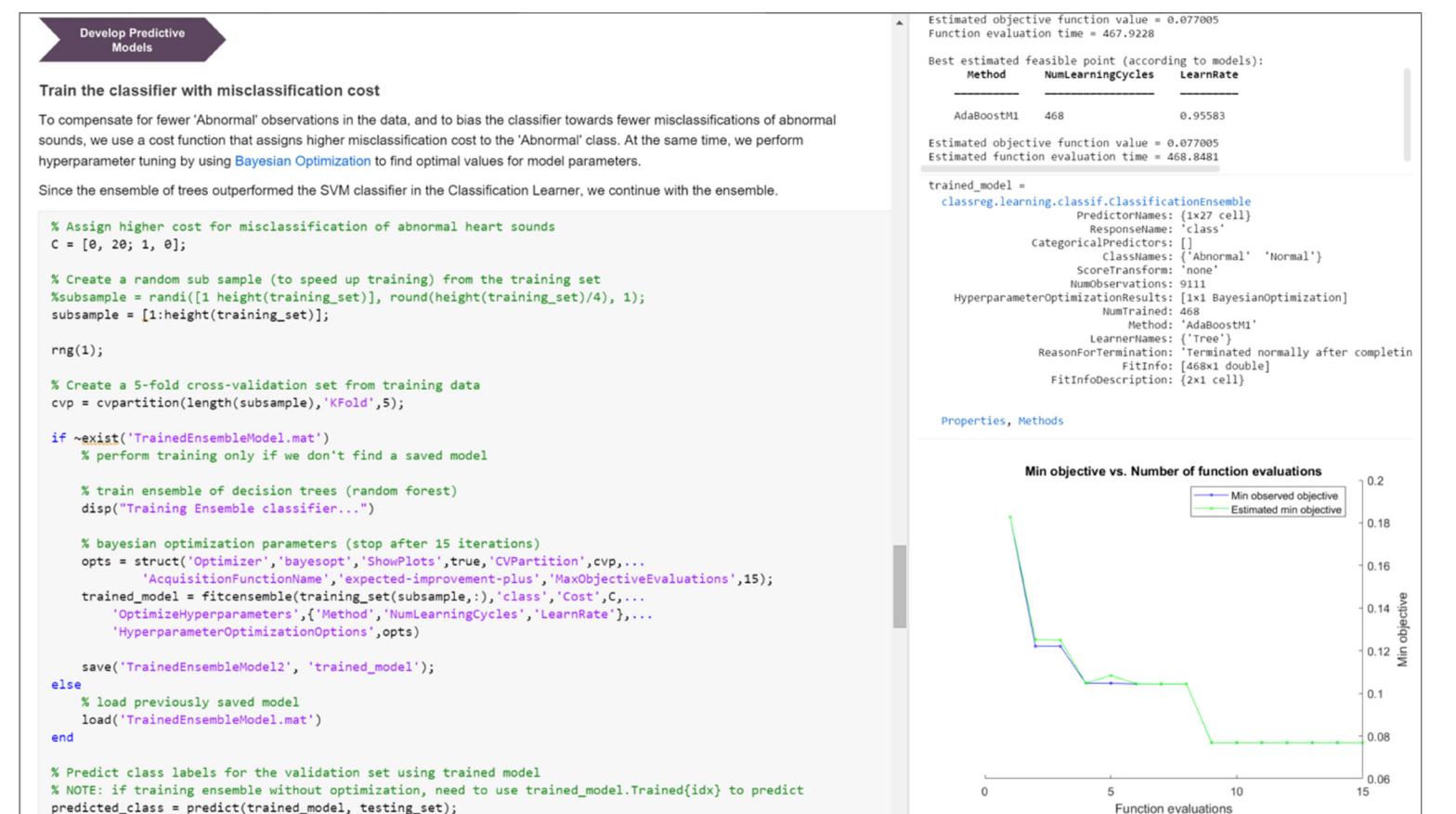


分類学習器アプリによる分類器のバイアス評価

手順 4. モデルを最適化する – 続き

分類器にバイアスを導入する標準的な方法は、望ましくない誤判別に高いペナルティを割り当てるコスト関数の導入です。

次のコードは、異常な心音の誤判別に20倍のペナルティを与えるコスト関数を導入します。



The screenshot shows a MATLAB script titled "Train the classifier with misclassification cost". It includes comments explaining the purpose of the script: to compensate for fewer 'Abnormal' observations and bias the classifier towards fewer misclassifications of abnormal sounds. The script uses Bayesian Optimization to find optimal values for model parameters. It includes code for creating a random subsample, performing 5-fold cross-validation, and training an AdaBoostM1 ensemble. The script also handles existing saved models and saves the trained ensemble to a file. A plot titled "Min objective vs. Number of function evaluations" shows the optimization progress, with the minimum observed objective value decreasing from approximately 0.18 to 0.08 over 15 function evaluations.

```
Develop Predictive Models

Train the classifier with misclassification cost

To compensate for fewer 'Abnormal' observations in the data, and to bias the classifier towards fewer misclassifications of abnormal sounds, we use a cost function that assigns higher misclassification cost to the 'Abnormal' class. At the same time, we perform hyperparameter tuning by using Bayesian Optimization to find optimal values for model parameters.

Since the ensemble of trees outperformed the SVM classifier in the Classification Learner, we continue with the ensemble.

% Assign higher cost for misclassification of abnormal heart sounds
C = [0, 20; 1, 0];

% Create a random sub sample (to speed up training) from the training set
%subsample = randi([1 height(training_set)], round(height(training_set)/4), 1);
subsample = [1:height(training_set)];

rng(1);

% Create a 5-fold cross-validation set from training data
cvp = cvpartition(length(subsample),'KFold',5);

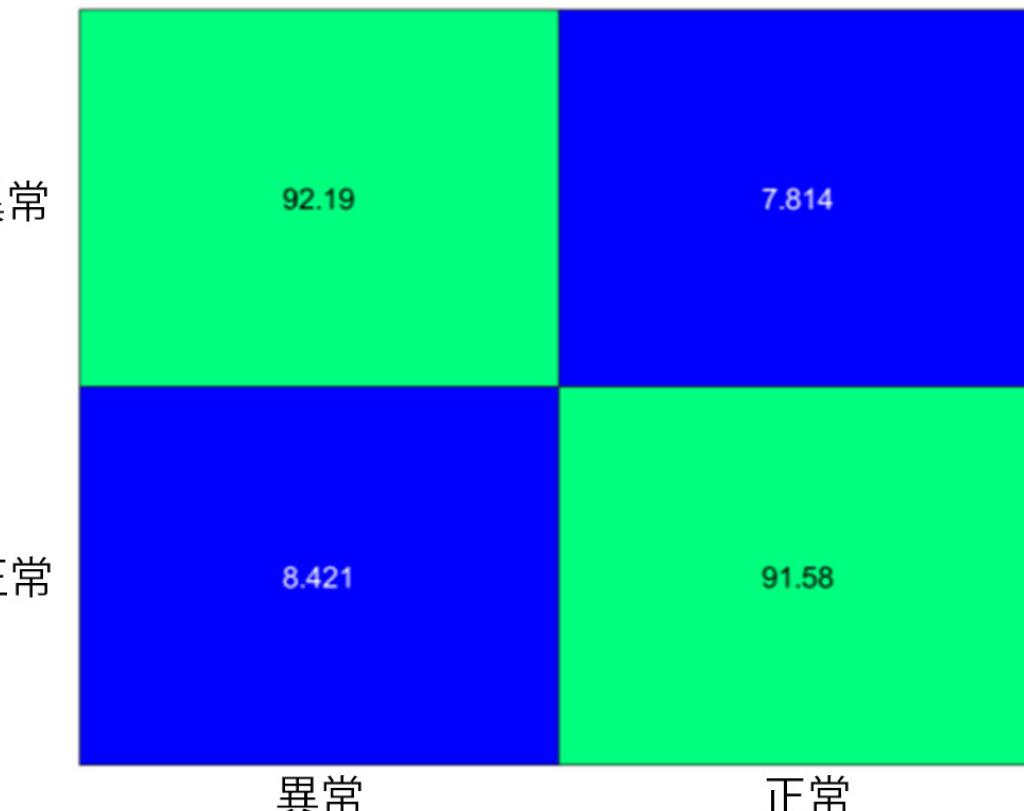
if ~exist('TrainedEnsembleModel.mat')
    % perform training only if we don't find a saved model
    % train ensemble of decision trees (random forest)
    disp("Training Ensemble classifier...")
    % bayesian optimization parameters (stop after 15 iterations)
    opts = struct('Optimizer','bayesopt','ShowPlots',true,'CVPartition',cvp,...
        'AcquisitionFunctionName','expected-improvement-plus','MaxObjectiveEvaluations',15);
    trained_model = fitcensemble(training_set(subsample,:),'Class','Cost',C,...
        'OptimizeHyperparameters',{ 'Method','NumLearningCycles','LearnRate'},...
        'HyperparameterOptimizationOptions',opts)

    save('TrainedEnsembleModel2', 'trained_model');
else
    % load previously saved model
    load('TrainedEnsembleModel.mat')
end

% Predict class labels for the validation set using trained model
% NOTE: if training ensemble without optimization, need to use trained_model.Trained{idx} to predict
predicted_class = predict(trained_model, testing_set);
```

コスト関数を導入して精度と偽陰性の許容誤差間で異なるバランスを取る

この混合行列は、この結果得られたモデルで正常な心音を異常と誤判別するのがやや多くなったのに対し(バイアスなしのモデルでは5%であるのに比べて8%)、異常な心音の検出に失敗するのは8%未満になったことを示しています。このモデルの全体的な精度は92%の高さを保っています。



コスト関数による異常な心音の誤判別の低減

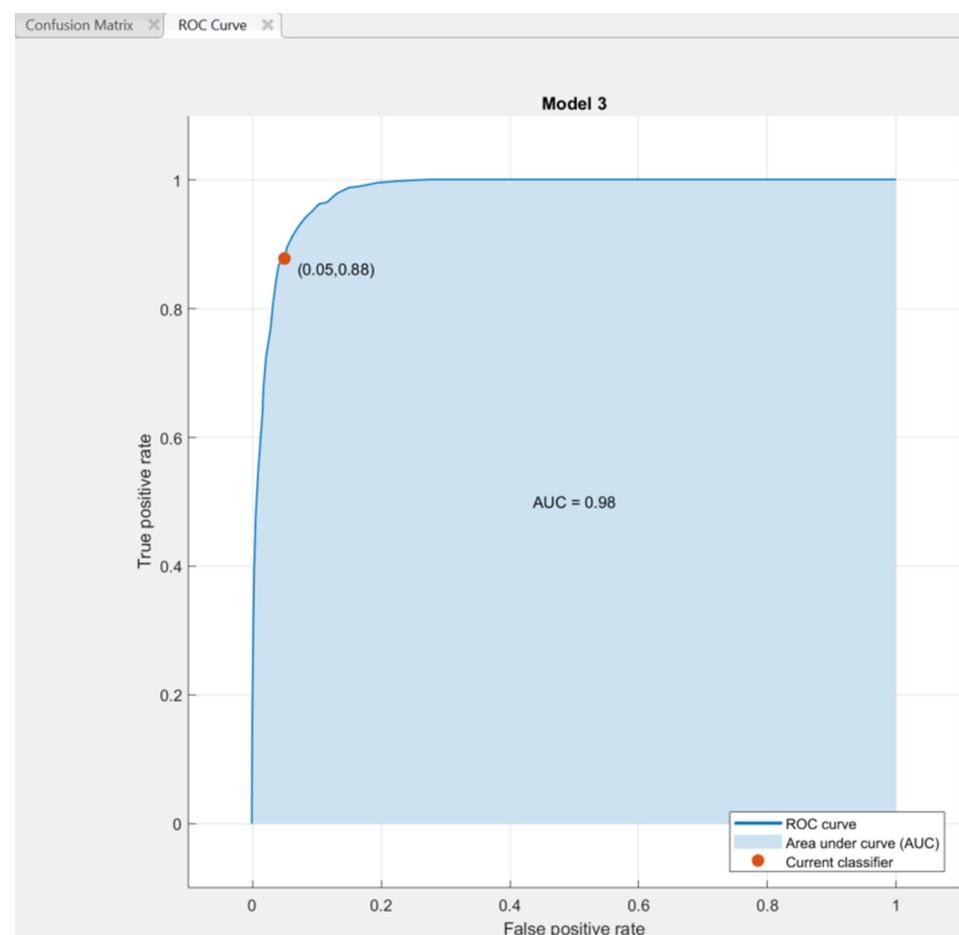
演習

スクリプト例の "Train the classifier with misclassification cost" セクションを実行します。このスクリプトはコスト行列を適用し、同時にハイパーパラメーターのベイズ最適化を実行します。

手順 4. モデルを最適化する – 続き

モデル パラメーターの調整

真陽性と偽陽性間のトレードオフを探るために、散布図と混合行列の代わりに受信者動作特性 (ROC) 曲線を使用できます。ROC 曲線は真陽性と偽陽性間のトレードオフを視覚的に探る場合に便利なツールです。ROC 曲線を使用して、コスト関数で定義する全体的な“コスト”を分類器が最小化する最適な動作点またはカットオフを選択できます。



バギングされた決定木アンサンブル分類器の ROC 曲線

これまで確認してきたように、パラメーターを調整することで、機械学習のアルゴリズムはよりうまくデータに適合するようになります。最適なモデルを実現するパラメーターセットを見つけ出すプロセスは、しばしば“ハイパーパラメーター調整”とも呼ばれています。ハイパーパラメーター調整をより効率的に行い、最適なパラメーター値を見つける可能性を高めるために、MATLAB では自動化されたグリッド探索とベイズ最適化を使用できます。

グリッド探索は決まった数のパラメーター値の組み合わせを網羅的に検索しますが、時間がかかる場合があります。

ベイズ最適化はハイパーパラメーター空間の統計モデルを開発し、最適な値を見つけるのに必要な実験回数を最小限に抑えることを目的としています。

このスクリプト例は、コスト関数を導入すると同時にベイズ最適化を使用してハイパーパラメーター調整を実行します。

関連情報

[ベイズ最適化の特徴](#)

手順 4. モデルを最適化する – 続き

特徴選択による誤判別と過適合の修正

これまで、モデルの学習時に抽出した 26 の特徴量を使用しました。性能を最適化する最後の段階では、冗長な特徴量または有用な情報をもたない特徴量を除去する特徴選択が必要となってきます。このステップにより、計算コストとストレージへの要求水準が低くなり、過適合しにくいシンプルなモデルが得られます。

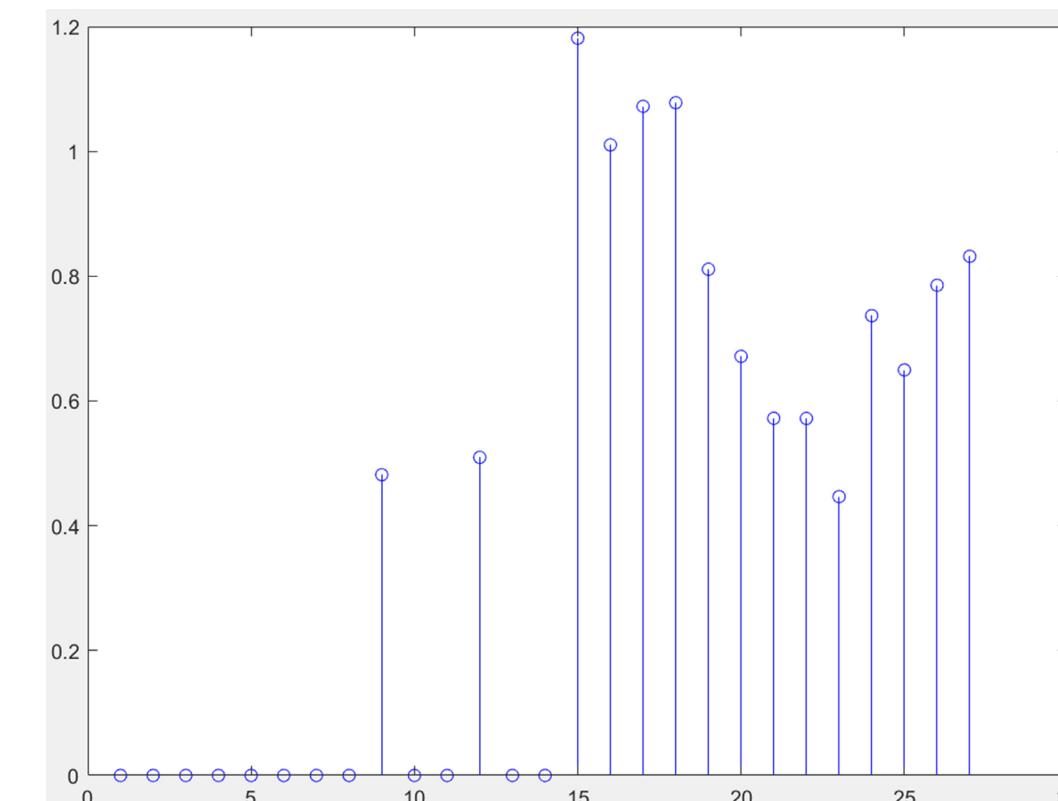
特徴量の削減は、この心音診断アプリケーションにとっては特に重要となります。なぜならば、特徴選択によりモデルのサイズが縮小され、組み込みデバイスへの配布が容易になるためです。

特徴選択の手法には、システムティックに特徴量のサブセットを評価する方法（計算コストは非常に高くなります）とそれぞれの特徴量に重みを導入することで、モデルを構築する過程に特徴選択を組み込んでしまう方法があります（より少ない特徴量を使うことで学習の過程で利用される目的関数の最小化はより簡単になります）。

この心音分類器では、非常に高次元のデータセットを処理できる強力な特徴選択の手法である近傍成分解析（NCA）を使用します。NCAにより、特徴量の約半分がモデルにあまり寄与しないことがわかります。したがって、特徴量の数を 26 から 15 に減らすことができます。

演習

スクリプト例の “Perform feature selection using Neighborhood Component Analysis” セクション、“Train model with selected features” セクションの順に実行します。



近傍成分解析を使用して最も関連性の高い特徴量を識別する、特徴選択を自動化した結果

関連情報

[高次元データを分類するための特徴選択](#)

手順 4. モデルを最適化する – 続き

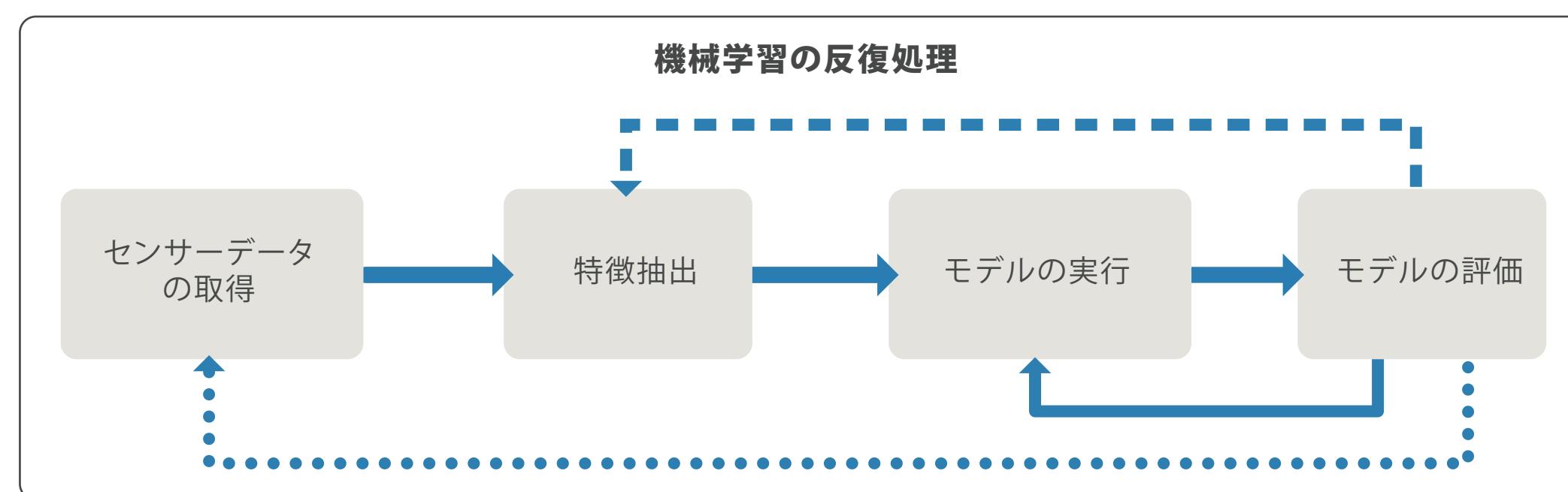
15の特徴量のみを選択した場合の性能への影響を評価するために、ハイパーパラメーター調整とコスト関数を有効にしてモデルを再学習させます。更新されたモデルで異常な心臓状態の検出に失敗したのは6%のみで、以前よりもわずかに改善しましたが、正常な状態の15%を異常と誤判別しており、全体的な精度はわずかに低下しています。医療現場では、陽性結果はすべて追加のテストで追跡され、初期のスクリーニングから15%偽陽性のほとんどが排除されます。

他のアルゴリズム試行の繰り返し

アルゴリズムから得られる向上の度合いは最適化手法によって異なるため、モデルをさらに向上するために、異なるアルゴリズムで同じ一連の

最適化手順を試すことができます。前のセクションで説明したように、最初に良好に機能したKNNアルゴリズムを再確認するなど、反復処理を繰り返すことができます。特徴抽出の段階に戻り、追加の特徴量を探すこともできます。最適なモデルを識別するには、機械学習ワークフローのさまざまな段階を繰り返すことが常に必要になります。現在のモデルを評価して、次に何を試せばよいかわかるようになれば、機械学習をマスターしたと言えるでしょう。

この心音分類の例題では、ようやく分類器を他のユーザーに展開する最終段階に進む準備が整いました。



手順 5. 運用システムに解析ロジックを配布する

機械学習アプリケーションは、デスクトップ上の運用システム、エンタープライズ IT システム（オンサイトまたはクラウド上）および組み込みシステムに配布できます。デスクトップとエンタープライズ IT システムの場合、スタンダロン アプリケーション、あるいは C/C++、Python®、Java®、.NET といった別の言語で記述されたアプリケーションとの統合用のソフトウェアコンポーネントとしてコードをコンパイルし、サーバーに MATLAB アプリケーションを配布できます。多くの組み込みアプリケーションでは、モデルを C コードで配布しなければなりません。

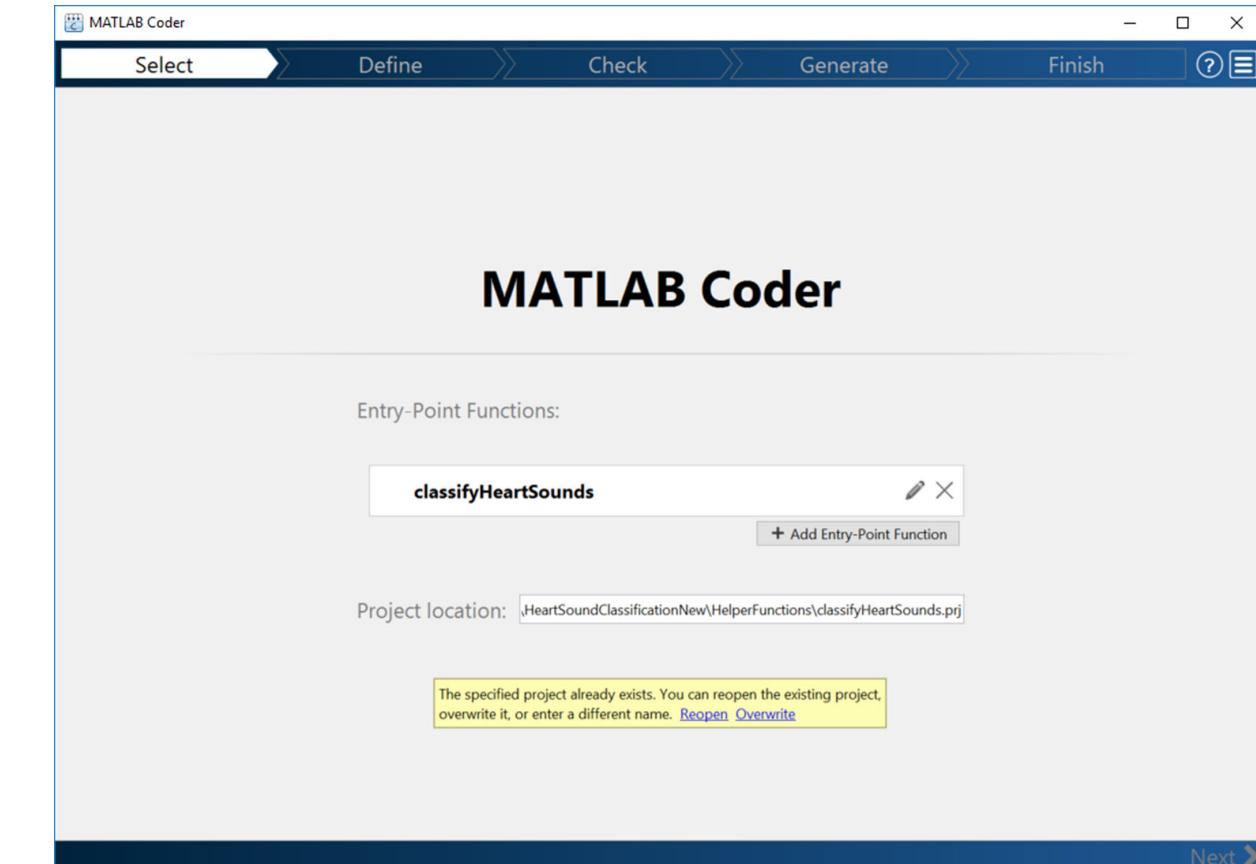
MATLAB Coder™ を使えば、MATLAB を C コードに自動変換することにより、組み込みシステムへのアプリケーション配布はより簡単なものになります。

C コードの生成

この心音診断アプリケーションはウェアラブル心臓モニターやモバイルアプリなどの医療機器で実行することになります。アプリケーションを配布するための準備として、次の手順を実行してモデルから C コードを生成します。

1. 学習済みのモデルをコンパクト モデルとして保存します。
2. MATLAB Coder を起動します。
3. raw センサー データを入力として使用し、患者の心音を正常または異常のいずれかとして分類するエントリ ポイント関数を作成します。

MATLAB Coder は対応する C コードを自動的に生成します。



MATLAB Coder による C コードの生成

必要なツール

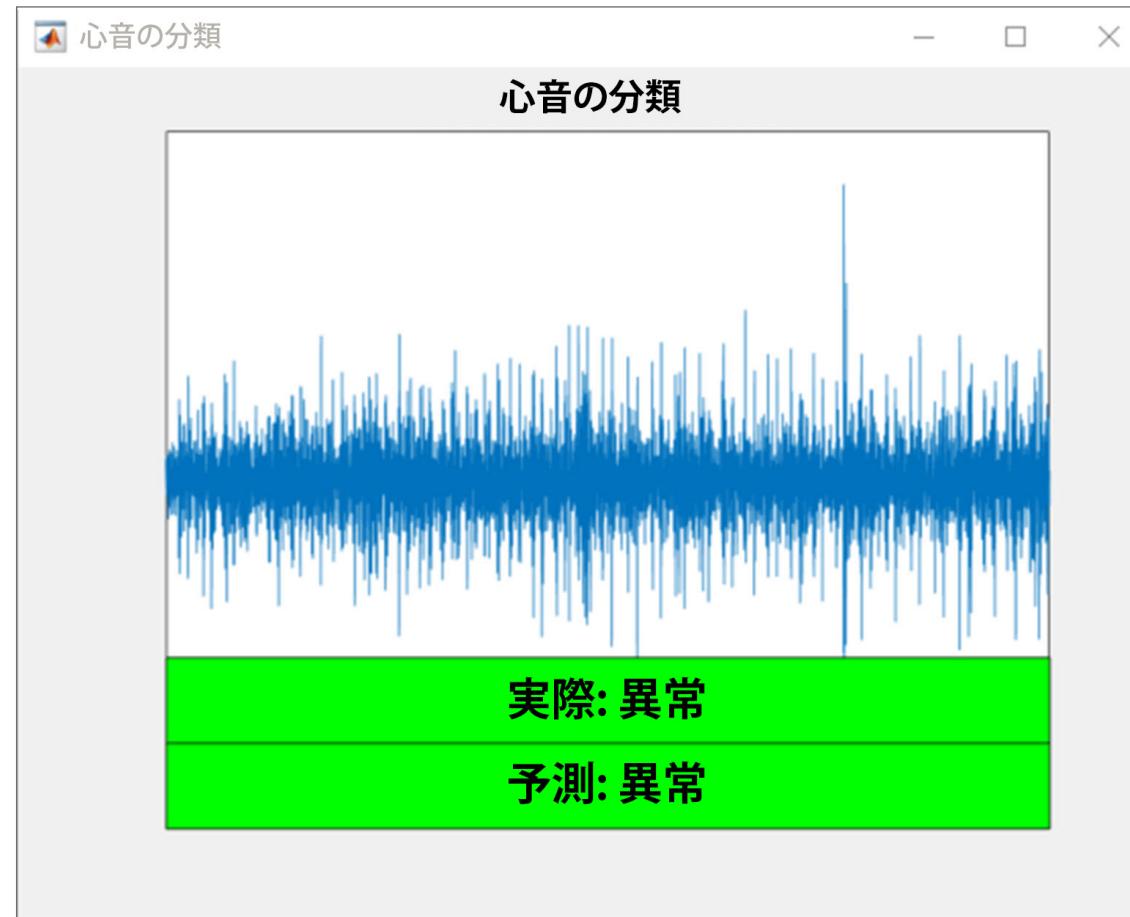
MATLAB Coder 評価版をお申し込みください。

演習

スクリプト例の “Validate final model” セクションを実行します。アプリケーションによって、開始当初に学習データと一緒にダウンロードされた “検証” データセットの数百の心音に対する真のクラスと共に予測クラスが表示されます。

手順 5. 運用システムに解析ロジックを配布する - 続き

生成された C または C++ コードを検証するために、検証データセットから対話形式でファイルを分類する単純なプロトタイプアプリケーションを実装します。ここで説明する 3 つの手順により、MATLAB 内から呼び出すことができる実行可能バージョンが生成されます。このバージョンがプロトタイプ (C ではなく MATLAB で記述された) で使用されます。



MATLAB による分類器の評価

この時点で、携帯用デバイスにアプリケーションを実装する準備ができました。

関連情報

- [組み込みコードの生成](#)
- [MATLAB からのCコード生成 \(38:14\)](#)

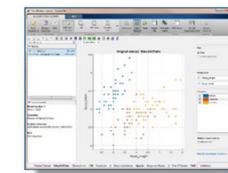
機械学習に不可欠なツール

MATLAB と関連するツールボックスを使えば、機械学習やデータサイエンスについての幅広い知識がなくても予測解析を構築したり、調整したり、配布したりすることができます。MATLAB および関連製品には、解析ロジックの構築と配布に必要なすべてのツールが用意されています。

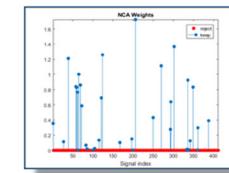
- **データ処理機能:** 欠損データまたは外れ値の処理、ノイズの削除、サンプルレートが異なるデータのサンプルレート合わせなど、時間がかかるタスクを簡略化します。
- **機械学習専門のアプリ:** ワークフローを高速化し、アルゴリズムをすばやく比較および選択することができます。
- **計画的なワークフロー:** 不要な特徴量を削減し、モデルのパラメータを調整することで、モデルがロバストに性能を発揮させるためのワークフローがあります。
- **機械学習ワークフローをスケールさせるためのツール: ビッグ データおよび計算クラスターで利用することができます。**
- **自動コード生成ツール:** 組み込みターゲットに解析ロジックをすばやく配布することができます。

あらかじめ組み込まれている関数とアルゴリズムは、ディープラーニング、コンピューター ビジョン、金融、画像処理、テキスト解析、自律エージェントなどのさまざまな専門的な用途のアプリケーションをサポートしています。

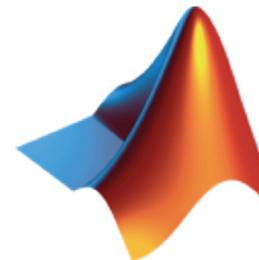
機械学習用 MATLAB



アプリを使用してすばやくモデルを学習し、アルゴリズムを評価



特徴抽出によってモデルのサイズを減らし、過適合を回避



ハイパーパラメーター調整とコスト行列を適用して精度を向上



自動コード生成ツールを用いて、解析ロジックを組み込みターゲットにすばやく配布

機械学習に不可欠なツール - 続き

データへのアクセスと探索

さまざまなタイプのデータのサポート:

- 信号、音、画像、金融データ、テキスト、地理空間およびその他の形式の処理
- 信号の処理と解析

データ アクセスおよびデータ探索用の関連製品:

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

データの前処理と特徴量の抽出

高品質なライブラリと各分野のツール:

- 金融、統計、信号処理、画像処理、テキスト解析、コンピューター ビジョンの特徴抽出に業界標準のアルゴリズムを使用
- フィルター処理、特徴選択、特徴変換でモデルを向上

専門分野向けの関連製品:

- *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™*

予測モデルの開発と最適化

反復的なアプリ駆動型のワークフロー:

- すばやくモデルを学習させて比較
- プログラミングではなく機械学習に焦点を当てる
- 最適なモデルの選択とパラメーターの調整
- マルチコアやクラスターに合わせた計算のスケーリング

モデルを改良・調整するための専門のアプリと製品:

- [分類学習器アプリ](#)
- [Neural Network Toolbox™](#)
- [Parallel Computing Toolbox™](#)
- [回帰学習器アプリ](#)
- [Statistics and Machine Learning Toolbox™](#)

運用システムに解析を配布する

高品質なライブラリと各分野のツール:

- 解析ロジックを運用システムに変換するツールの取得
- 組み込みターゲットに配布するコードの生成
- 幅広いターゲット プラットフォームおよびエンタープライズ システムへの配布

専門分野向けの関連製品:

- [HDL Coder™](#)
- [MATLAB Coder™](#)
- [MATLAB Compiler™](#)
- [MATLAB Compiler SDK™](#)
- [MATLAB Production Server™](#)

さらに詳しく知るには?

ダウンロード

心音診断アプリケーション用の MATLAB コード

見る

MATLABによる機械学習の基礎～特徴抽出・分類器・交差検定～ (38:31)

データを読み解くための機械学習 - MATLABでデータ解析の問題

に立ち向かう (55:18)

Machine Learning Using Heart Sound Classification Example (22:03)

読む

ディープラーニングとは?

MATLAB で始めるディープラーニング

機械学習とは?

MATLAB を活用したビッグ データ

MATLAB を使用したクラウド上での並列計算