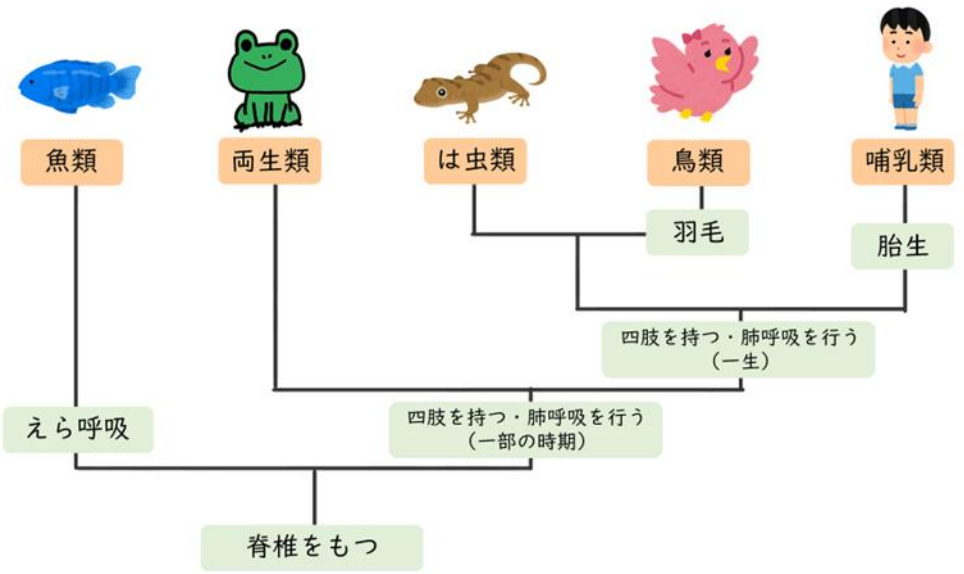
楽曲の人気度予測

- 決定木

勾配ブースティング決定木(GBDT)

dmlc XGBoost

ランダムフォレストや勾配ブーストなどの他の手法を改善したもので、さまざまな最適化の方法を使用することにより、大規模で複雑なデータセットの処理に有効。



XGBoost について

1. **XGBoostとは？**
・**概要**: XGBoost（eXtreme Gradient Boosting）は、効率的で高性能な勾配ブースティングフレームワーク。
2. **使用する理由**
・**高い予測精度**: 他のアルゴリズムと比較して非常に高い精度を誇る。
・**計算効率**: 並列処理とハードウェア最適化により高速に動作。
・**柔軟性**: 回帰、分類、ランク付けなど多様な問題に対応可能。
・**ハンドリング欠損値**: 自動的に欠損値を処理し、前処理が簡単。
3. **仕組み**
・**基本原理**: 複数の弱学習器（決定木）を逐次的に学習し、誤差を補正
・**ブースティングの過程**:
初期モデルの構築
誤差の計算と次の弱学習器の追加
弱学習器の重み付けと最終モデルの構築
4. **パラメータ**
・**主要パラメータ**:
learning_rate: 学習率。小さくすると精度向上だが時間がかかる
n_estimators: 決定木の数。多くすると精度向上だが過学習のリスク
max_depth: 決定木の深さ。大きくするとモデルの複雑さが増す
subsample: サンプリング率。過学習を防ぐために使用
colsample_bytree: 決定木ごとの特徴量のサンプリング率

特徴量：カテゴリカル変数の処理

	artists	album_name	track_name	track_genre
Label Encoding	○	○	○	○
One-hot Encoding				○
Target Encoding	○	○	○	○

・Target Encodingは「○○ごとの人気度の平均」で算出（リークに注意→交差検証）

・アーティスト名のTarget Encodingは分割して行った（効果は微妙）

・元々のカテゴリカル変数のデータ列は削除（使ったモデルの都合）

特徴量：その他の試行錯誤

- 欠損値処理
- explicitをint型に変換（講義資料通り）
- トラックの長さをカテゴリ化→One-hot Encoding
- いくつかの特徴量を追加

  ※既存の特徴量の積、文字列変数の長さ、etc.
- スケーリング