仕事ではじめる機械学習
1章 機械学習プロジェクトの始め方

M2 倉地亮介

目次

1.1 機械学習はどのように使われるのか

- 教師あり学習
  - 入力データと出力データの関係性を獲得(学習フェーズ)し，獲得したモデルによって未知のデータに対する予測ができるようにプログラムを実現すること(予測フェーズ)
- 教師なし学習
  - 入力データからデータの構造を獲得する
  - 正解がないので算出した特徴量から構造、法則、傾向、分類、定義などを導き出す
  - 傾向分析、未来予測などにも応用できる
- 強化学習
  - 明確な「答え」ではなく、「行動」と「報酬」を与え、どのような行動を取れば報酬が最大でもらえるのかを学習
  - e.g. 囲碁や将棋など

1.2 機械学習プロジェクトの流れ

1. 問題を定式化する
2. 機械学習をしないで良い方法を考える
3. システム設計を考える
4. アルゴリズムを選定する
5. 特徴量，教師データとログの設計をする
6. 前処理をする
7. 学習・パラメータチューニング
8. システムに組み込む

1.2 機械学習プロジェクトの流れ

1. 問題を定式化する
2. 機械学習をしないで良い方法を考える
3. システム設計を考える
4. アルゴリズムを選定する
5. 特徴量，教師データとログの設計をする
6. 前処理をする
7. 学習・パラメータチューニング
8. システムに組み込む

解きたい課題を機械学習で解ける問題設定に落とし込む(1,2)

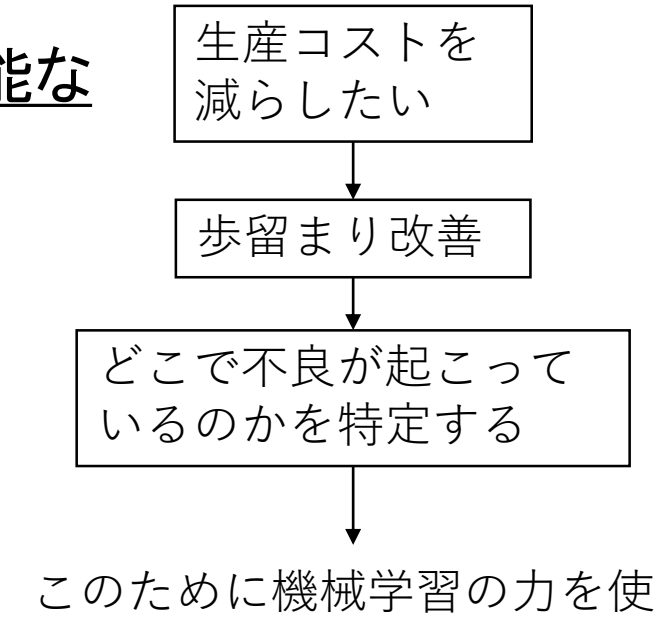
解くための道具選びと前処理(3,4,5,6)

モデルの作成(7)

サービスへの組み込み(8)

1.2.1 問題を定式化する

- どのように問題を解くのかを定式化する
  - 何を目的とするのか、そして解きたい課題について仮説を立て、何をすればいいのかを明確にすること
    - 例:「売上を改善する」,「有料会員数を増やす」,「製品の生産コストを減らす」
  - 大きな目的に対して、より具体的でアクション可能なレベルまでブレイクダウンする
    - KPI (Key Performance Indicator)
    - 業務レベルにおける具体的な目標設定



1.2.2 機械学習しなくても良い方法を考える

- 機械学習を含んだシステムは通常のシステム以上に技術的負債が蓄積しやすい
  - 長期運用しているとトレンドの変化などで入力の傾向が変化する
  - 確率的な処理があるため自動テストがしにくい
  - 処理のパイプラインが複雑になる
  - データの依存関係が複雑になる
  - 実験コードやパラメータが残りやすい
  - 開発と本番の言語/フレームワークがバラバラになりやすい

1.2.2 機械学習しなくても良い方法を考える

- 長期運用しているとトレンドの変化などで入力の傾向が変化する
  - テキストを扱う問題では、単語の用法のトレンドが変化したり、新語が登場したりすると予測精度が下がったり意図しない挙動をする可能性がある

- 確率的な処理があるため自動テストがしにくい
  - 機械学習アルゴリズム内には乱数を用いた確率的な処理があることが多い
  - あらゆるデータに対する挙動を予め確認することは不可能

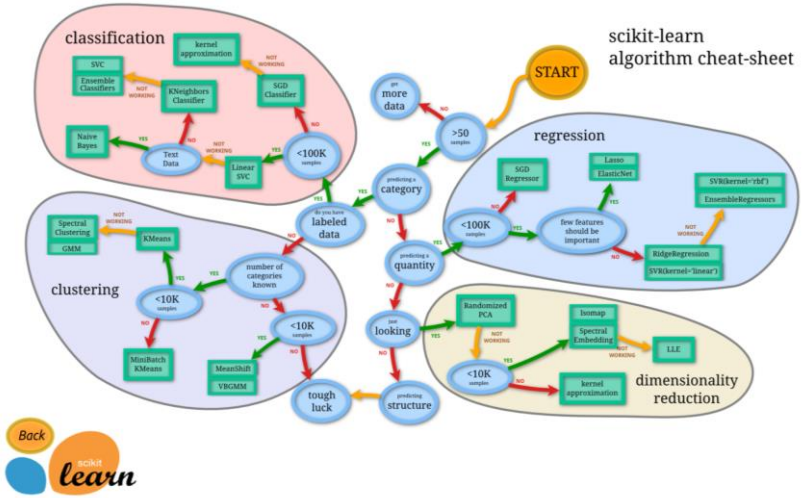
意図しない予測結果が出てしまったときに、誤りをカバーできる仕組みが必要
- 特定ラベルはブラックリストに登録してはじく

1.2.3 システム設計を考える

- 予測結果をどういう形で利用するのか → 4章で詳しくやる
  - e.g. バッチで予測処理をしてその結果をRDBで配布
- 予測誤りをどこで吸収するのか
  - e.g. 予測結果を人手で確認するフェーズを用意する
- ここまできたら撤退ラインを決める

1.2.4 アルゴリズムを選定する

- 分類
- 回帰
- クラスタリング
- 次元削減



1.2.5 特徴量、教師データとログ設計をする

- 特徴量：機械学習の予測モデルに入力をする情報
  - e.g. 今日の気温(1.0℃)，降水量(0.8mm)，風速(0m)，積雪量(2cm)，天気(曇り)
    → 特徴ベクトル[1.0，0.8，0.0，2.0，1]

- 特徴量が、予測に必要な情報を含んでいることを予め確認
  - ビジネスのドメインの知識を持った人と協力して、何が現象に影響与えそうか確認

- 特徴量を決めたら、入力データの正解データを用意
  - 教師データ：正解カテゴリのラベルと元となるデータのセット
    - e.g. 画像認識では、写真に写っている「車」や「犬」といったカテゴリの正解を決めておく
  - 教師あり学習では、質の良い正解ラベルをどのように取得するかが重要

1.2.6 前処理をする

- 不要な情報を削ぎ落すなどデータを機械学習に使える形にするプロセス
- RDBで表現できるような表形式のデータの形にしたい
  - webのログなどの生データはテキスト形式 ← 単語に分割して頻度を数えたり
  - 数値データの一部が取得できていない ← 欠損値処理(値を推定して補間するなど)
  - 異常な値を除外
  - 値の取りうる幅の影響を受けないように正規化

1.2.7 学習・パラメータチューニング

- 機械学習アルゴリズムを調整するパラメータを試行錯誤しながら変更し、より良い結果ができるパラメータを探索
  - ルールベースで決めた正解など、ベースラインの予測性能を決める
  - 最初は、シンプルなアルゴリズムと既存ライブラリを用いて簡単な予測モデルを作る
    - 一部データが正常に取れていないなどデータにバグが潜んでいることが多い
    - 学習時のデータに過剰に適合してしまい、未知のデータを適切に予測できない

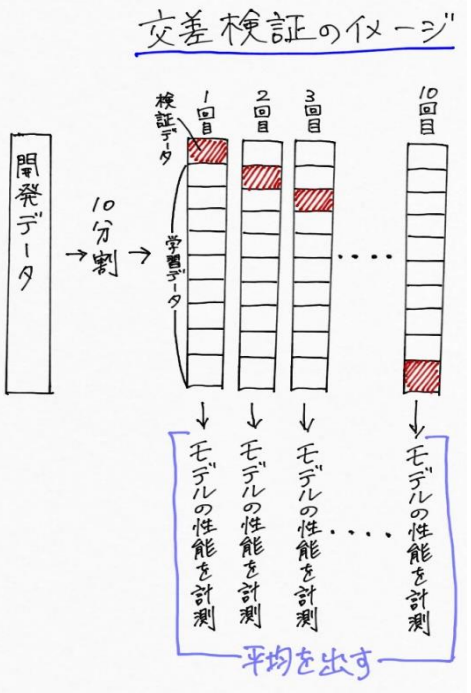
過学習、Data Leakage

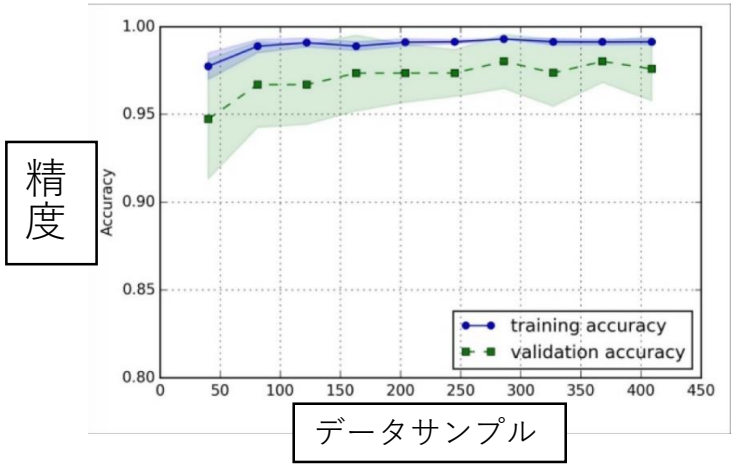
- 過学習
  - 学習に使ったデータに対してはきちんと正解できるけど、知らないデータに対しては全然当たらないこと
- Data Leakage
  - 本来知り得るはずのない正解データの情報が教師データに紛れてモデルの予測性能が不当に高くなってしまうこと

Kaggleの癌予測のためのコンテストのデータに前立腺の手術をしたかどうかというフラグが含まれていたことがありました。このデータを使った予測モデルは非常に高い予測性能の達成しましたが、前立腺がんの人が癌とわかった後で、手術を受けているというだけのデータで、未知のデータには意味のない予測モデルなってしまった

過学習を防ぐには

1. **交差検証**を使ってパラメータを調整
   - 開発データを学習用の訓練データと評価用の検証データを分割して性能を計測することで、特定データによらない汎化性能のあるモデルを得る方法
2. **正則化**を行う
   - 訓練データへの過度な当てはまりを抑制しつつ、より単純なモデルが当てはまるよう促す効果を持つ
   - クラスを分離する境界をなめらかにして、未知のデータに対する対応力をつける
3. **学習曲線**を見る
   - 訓練データに対する精度だけ高かった場合は過学習を疑う





1.2.8 システムに組み込む

- 予測性能とそれに伴うビジネスインパクトをモニタリング
  - e.g. 商品購入のコンバージョン率など
- モニタリングでは、あらかじめ人手で用意したデータと正解ラベルのセットを使って予測性能を計測
  - ゴールドスタンダード
- KPIの改善という目的を見失わない
  - 必要に応じて性能の改善
  - 異常時にはアラートを通知して、アクションをいつでもとれるようにする

実システムにおける機械学習の問題点への対処方法

- システムの変化を前提とした設計を行う
  - 予測モデルをモジュール化してアルゴリズムのA/Bテストができるようにする
    - 複数の予測モデルを並列に検証し、どのモデルがより高い成果を出せるのか調べる
  - モデルバージョンを管理して、いつでも切り戻し可能にする
    - 性能劣化した際に、モデルの更新が原因かどうかを切り分けて考える

1.4 機械学習を含めたシステムを成功させるには

- 機械学習を含めたプロジェクトをビジネスとして成功させる上で重要なチーム構成

1. プロダクトに関するドメイン知識を持った人
2. 統計や機械学習に明るい人
3. データ分析基盤を作れるエンジニアリング能力のある人
4. 失敗しても構わないとリスクを取ってくれる責任者

機械学習がリスクの大きい投資だということを認識した上で、それでも機械学習を使わないとできない価値を生み出すことに背中を押してくれる人

1.5 この章のまとめ

- 解くべき問題の仮説を立て、目的を明確にしコンセプトの検証を最優先する
- 機械学習をしないという選択を恐れない
- 機械学習に適している問題設定か見極める
- 予測性能とKPIの両方のモニタリングし、継続して改善を続ける