

機械学習エンジニアコース Sprint

－ アンサンブル －



DIVE INTO CODE



今回のモチベーション

目的はなにか アンサンブル学習について理解する

1. **ブレンディング**: 複数のモデルの推定値を使って評価する方法。
2. **バギング**: 訓練データを分割して、複数の決定木モデルの推定値を評価する方法。
3. **スタッキング**: ブレンディングとバギングのハイブリッド手法。複数のモデルを用い、訓練データを分割して評価する方法。



このスライドは?

ここでは、アンサンブル学習の基本的な知識を学びましょう



アンサンブル学習とはなにか

予測精度をあげるため、複数の学習モデルを組み合わせる手法

階層の下のモデルはベースモデル（base / weak learner / genelizer）、
上のモデルはメタモデル（meta learner / stacker）と呼ばれることが多い

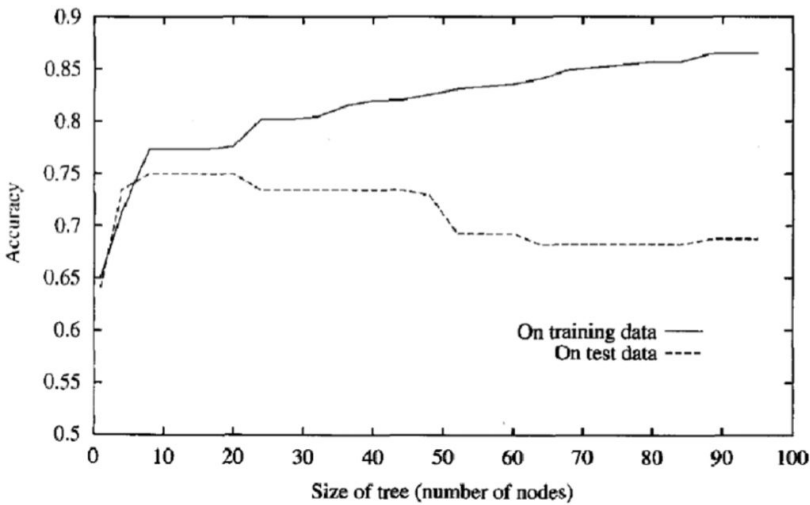


この課題の対象者

① scikit-learnの単一モデルを用いて、学習、推定するコードがかける方



決定木の問題



決定木の問題

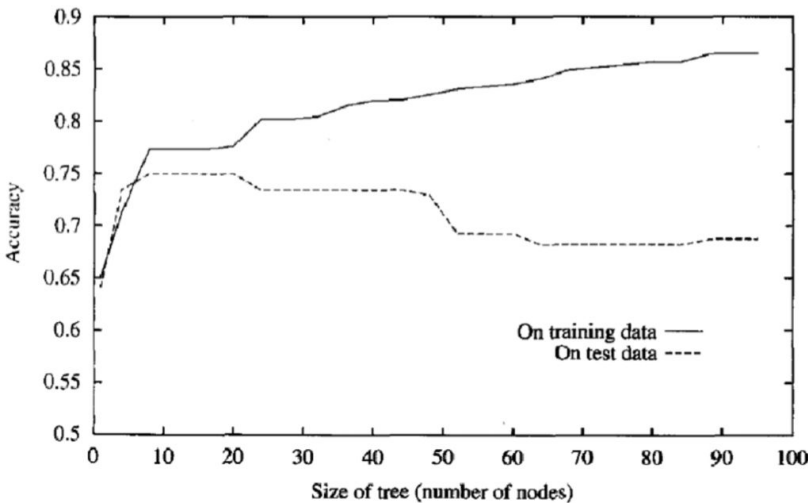
決定木は、バリエーション（予測値の分散）が高くなりやすい。

trainデータが違えば得られるツリー構造が大きく異なる、つまりtrainデータに過剰適合しがちだった。



決定木の問題

汎化誤差を減らしたい



汎化誤差 (generalization error) =

モデルの複雑さ(バリエーション) + 真の関数とモデルのずれ(バイアス)² + ノイズ

そこで、**Bagging**が提案される

単一の決定木だと高くなりがちなバリエーションを下げる
ことができる。



Bagging

Bagging

$$\hat{f} = \frac{1}{B} \sum_{b=1}^B f_b(x')$$

各モデル
標本の数

高分散のモデルにブートストラップ法を適用し、分散を減らす

<https://www.kaggle.com/kashnitsky/topic-5-ensembles-part-1-bagging>

アイデア :

サンプルから離散一様分布に従いランダムな標本再抽出 (ブートストラップサンプリング) を行い、各サブセットに対して決定木を当てはめ、複数の決定木の結果を得る。最後に多数決 (回帰ならば平均) を行う。

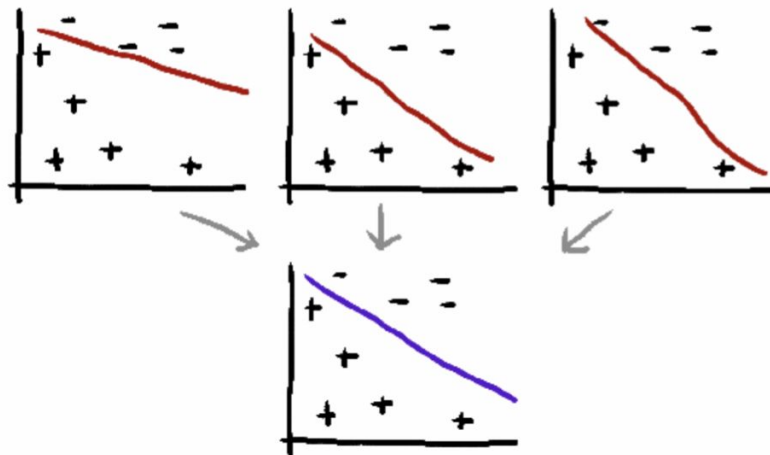
事例 :

<https://www.kaggle.com/mayankkestwal10/ensemble-learning-bagging-and-boosting>

前提 : そもそも。

1994年にレオ・ブレイマンによって提唱された。

<http://statistics.berkeley.edu/sites/default/files/tech-reports/421.pdf>





Bagging

Bagging

&

RandomForest

RandomForestは、Baggingの改良版

RandomForestは、Baggingのアルゴリズムに特徴量のサンプリングも追加した手法。



RandomForest

RandomForest

木の相関を低くすることでBaggingの分散低減の効果をあげる

アイデア：

それぞれの分割の前に、特徴量からm個をランダムに選び、それを分割の候補とする。

事例：

前提：そもそも。

Baggingでは、ブートストラップ法で作成した各々の決定木同士の相関が高いことがある。これに対し、分岐で選ぶ特徴量が異なる決定木をたくさん生成しているRandomForestは決定木間の相関が低い為、分散の式の第一項が小さくなってバリエーション（分散）が下がり、Baggingよりも汎化性能が高くなる。

<http://statistics.berkeley.edu/sites/default/files/tech-reports/421.pdf>

$$\overset{\text{予測値}}{f_{\text{rf}}^B(x)} = \frac{1}{\underset{\text{標本の数}}{B}} \sum_{b=1}^B \overset{\text{ランダムフォレストの木}}{T(x; \Theta_b)}$$

$$\underset{\text{変数間の相関係数}}{\rho} \sigma^2 + \frac{1 - \rho}{\underset{\text{標本の数}}{B}} \sigma^2 \quad \begin{array}{l} \text{分散}\sigma^2\text{の独立同分布に従う} \\ \text{B個の予測値の平均の分散} \end{array}$$



RandomForest

$$\begin{aligned} \text{Var} \left(\frac{1}{B} \sum_{i=1}^B T_i(c) \right) &= \frac{1}{B^2} \sum_{i=1}^B \sum_{j=1}^B \text{Cov}(T_i(x), T_j(x)) \\ &= \frac{1}{B^2} \sum_{i=1}^B \left(\sum_{j \neq i}^B \text{Cov}(T_i(x), T_j(x)) + \text{Var}(T_i(x)) \right) \\ &= \frac{1}{B^2} \sum_{i=1}^B \left((B-1)\sigma^2 \cdot \rho + \sigma^2 \right) \\ &= \frac{B(B-1)\rho\sigma^2 + B\sigma^2}{B^2} \\ &= \frac{(B-1)\rho\sigma^2}{B} + \frac{\sigma^2}{B} \\ &= \rho\sigma^2 - \frac{\rho\sigma^2}{B} + \frac{\sigma^2}{B} \\ &= \rho\sigma^2 + \sigma^2 \frac{1-\rho}{B} \end{aligned}$$

Decreases, if ρ decreases, i.e., if m decreases

Decreases, if number of trees B increases (irrespective of ρ)

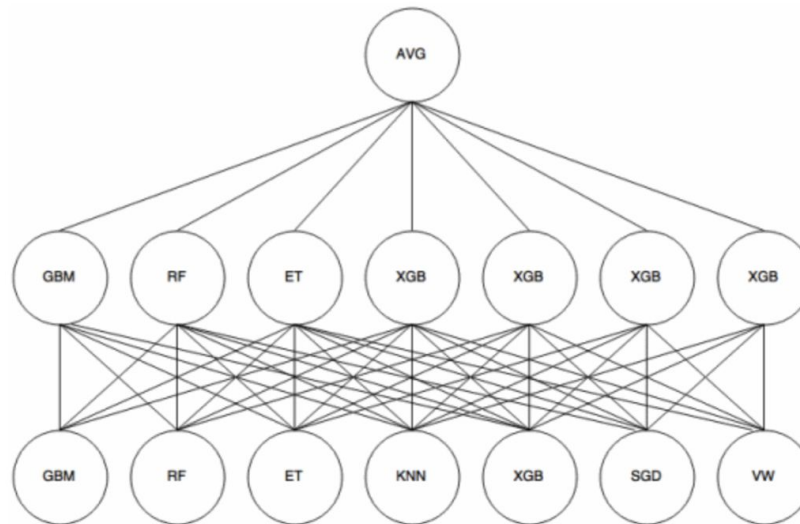


Blending & Stacking

BlendingとStacking :

Bagging/RandomForestと違って、異なる予測モデルを組み合わせる

異なる予測モデル、異なる特徴量、訓練データの異なる部分、異なるパラメータを用いることで十分な多様性を得たい。
組み合わせ方法は多数決、平均、最大値、最小値...など。





Blending

予測を何らかの方法で混合して改善する

<https://www.quora.com/What-is-blending-in-machine-learning>
<https://mlwave.com/kaggle-ensembling-guide/>

アイデア :

後のモデルの入力として10%のholdoutデータのみを用いる。あるいは単に各モデルの加重平均を使用する、...など、多義的に説明されている。

<https://www.datarobot.com/wiki/training-validation-holdout/>
<https://www.kaggle.com/c/porto-seguro-safe-driver-prediction/discussion/44588>

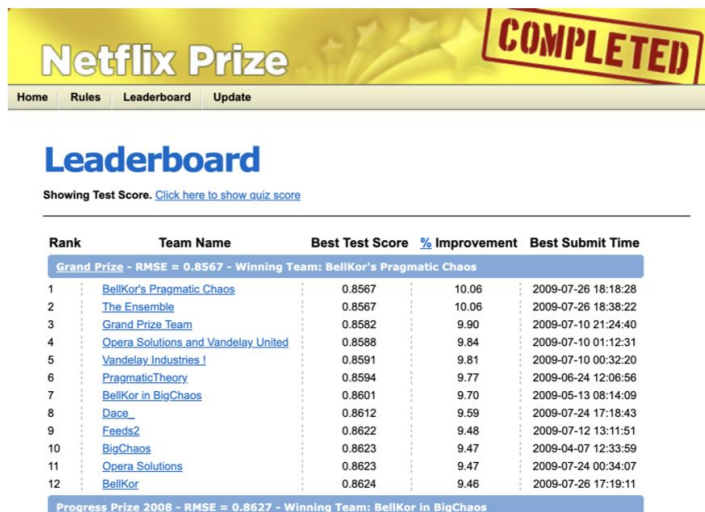
事例 :

<https://github.com/ghmagazine/kagglebook/blob/master/ch01/ch01-01-titanic.p>

前提 : そもそも。

Netflix社が開催していた「Netflix Prize」(アルゴリズムコンテスト)の優勝者が初めて用いた言葉。Stackingとよく比較されるがより単純で、情報リークの恐れが少ない。

<https://www.netflixprize.com/>



The image shows a screenshot of the Netflix Prize Leaderboard. At the top, there is a yellow banner with the text "Netflix Prize" and a red stamp that says "COMPLETED". Below the banner, there is a navigation bar with links: Home, Rules, Leaderboard, and Update. The main section is titled "Leaderboard" and includes a link to "Showing Test Score. Click here to show quiz score". The table below lists the top 12 teams, their best test scores, percentage improvements, and best submit times. The winning team is BellKor's Pragmatic Chaos, with a best test score of 0.8567 and a 10.06% improvement over the baseline.

Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
Grand Prize - RMSE = 0.8567 - Winning Team: BellKor's Pragmatic Chaos				
1	BellKor's Pragmatic Chaos	0.8567	10.06	2009-07-26 18:18:28
2	The Ensemble	0.8567	10.06	2009-07-26 18:38:22
3	Grand Prize Team	0.8582	9.90	2009-07-10 21:24:40
4	Opera Solutions and Vandelay United	0.8588	9.84	2009-07-10 01:12:31
5	Vandelay Industries I	0.8591	9.81	2009-07-10 00:32:20
6	PragmaticTheory	0.8594	9.77	2009-06-24 12:06:56
7	BellKor in BigChaos	0.8601	9.70	2009-05-13 08:14:09
8	Dace	0.8612	9.59	2009-07-24 17:18:43
9	Feeds2	0.8622	9.48	2009-07-12 13:11:51
10	BigChaos	0.8623	9.47	2009-04-07 12:33:59
11	Opera Solutions	0.8623	9.47	2009-07-24 00:34:07
12	BellKor	0.8624	9.46	2009-07-26 17:19:11
Progress Prize 2008 - RMSE = 0.8627 - Winning Team: BellKor in BigChaos				



Stacking

予測を改善するためのモデルの積み重ね

アイデア :

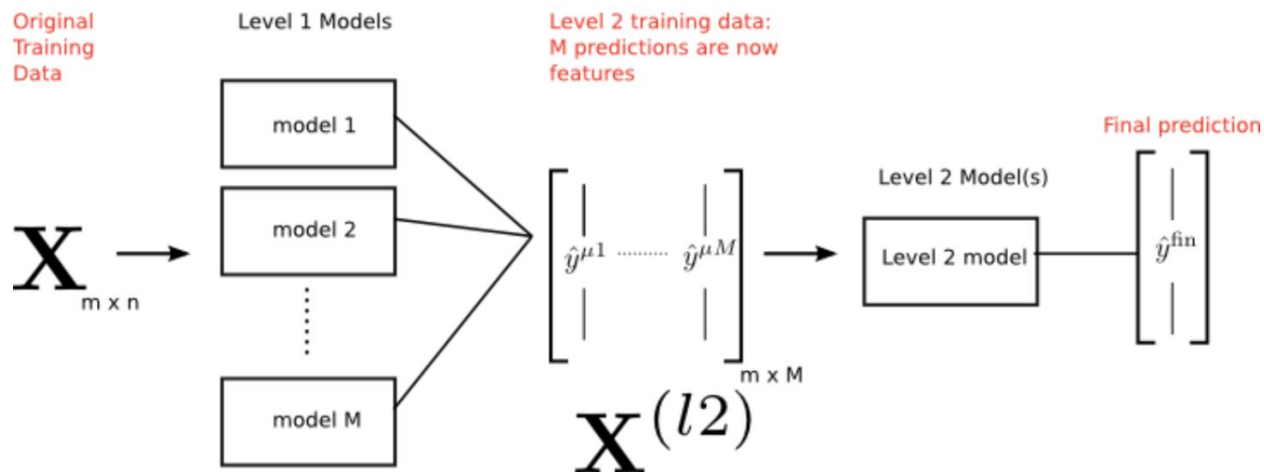
後のモデルの入力として先のモデルの出力を用いる。

事例 :

<https://www.kaggle.com/yekenot/simple-stacker-lb-0-284>

前提 : そもそも。

trainデータを使用してsample内(validation)で予測を行うことはあらかじめ事前に類似問題を見た上で答えることに等しく、過剰適合してしまう恐れがある。
この問題を回避することができる。



アンサンプル 完