卒業論文

データストリームマイニングアルゴリズムの性能評価手法の検討

公立はこだて未来大学
システム情報科学部　複雑系知能学科
複雑系コース　1016149

丸尾 海月

指導教員　新美 礼彦

提出日　2020 年 1 月 28 日

BA Thesis

Performance Evaluation Method in Data Stream Mining Algorithm

by

Mitsuki Maruo

School of Systems Information Science, Future University Hakodate
Complex Systems Course, Department of Complex and Intelligent Systems
Supervisor:　Ayahiko Niimi

Submitted on January 28, 2020

Abstract− In data stream mining, it is necessary to handle continuously generated data streams. Therefore, it is unrealistic to construct a model using all available data like in the case of batch learning, which is why, the model can be developed by applying mini-batch learning or online learning. To evaluate data stream mining algorithms, the static data may be processed and used as a data stream. However, the performance of a data stream mining algorithm cannot be correctly evaluated using the method based on static data, as particular data streams have local characteristics. Moreover, with regard to mini-batch learning or online learning, there is an issue that a learning result may change depending on the order of the learning data. In this paper, to solve the aforementioned problems, we propose a new method to evaluate the performance of a data stream mining algorithm using time series data as a pseudo data stream. To verify the effectiveness of the proposed method, we have conducted the experiments to evaluate the performance of an online algorithm, using one static data set and seven time series data sets. Although the experimental results did not show the problem of using static data, it was confirmed that the model could not be properly constructed if the order of the training data was intentionally biased.

Keywords: Data Stream, Time Series Data, Performance Evaluation, Online Learning, Data Mining

概 要： データストリームマイニングでは，無限に生成されるデータストリームを扱う必要がある．そのため，データストリームマイニングでは，バッチ学習のように全てのデータを使ってモデルを構築することは非現実的であるため，ミニバッチ学習やオンライン学習を用いてモデルの構築を行う．データストリームマイニングのアルゴリズムを評価する場合，静的なデータを加工してデータストリームとして利用することがある．しかしデータストリームには局所的に変化する性質のものもあるため，静的なデータを加工する方法ではデータストリームマイニングアルゴリズムの性能を正しく評価できない問題がある．またミニバッチ学習やオンライン学習では，学習データの順序によって学習結果が変化する問題がある．本論文ではこれらの問題の解決を図るため，時系列データを擬似データストリームとして扱い，データストリームマイニングアルゴリズムの性能評価を行う手法を提案する．提案手法の有効性を検証するため，１つの静的データセットと７つの時系列データセットを用いて，データストリームマイニングアルゴリズムの一種である，オンラインアルゴリズムの性能を評価する実験を行った．実験の結果から，静的なデータを利用することの問題点は示せなかったが，学習データの順序を意図的に偏らせた場合は，うまくモデルが構築できない場合がある

ことを確認した.

キーワード： データストリーム, 時系列データ, 性能評価, オンライン学習，データマイニング

目次

第 1 章

序論

本章では，本研究の背景と目的について述べる．

1.1 背景

昨今の情報過多に伴い，得られる全てのデータをデータベースに入力し，それら全てを用いて解析するのは困難となっている．IDC の調査によると，2017 年時点で，世界のデータ量の 15% はデータストリームであり，2025 年には 30% のデータがデータストリームになるであろうという予測 [1] がされている．

データストリームは動的な大規模データであり，従来の静的なデータを対象としたデータマイニング手法をそのまま適用することは困難である．そのため，動的なデータを対象としたアルゴリズムが古くから研究されている．今後更に増加するであろうデータストリームを分析するために，データストリームマイニングに関する研究が盛んになると考えられる．

データストリームマイニングでは，次々にやってくるデータストリームを処理しなければならない．そのため，従来の静的なデータを対象としたデータマイニングのように，十分な時間とリソースを掛けることができない．そこで，データストリームマイニングでは，厳密解を求めることを諦めて，近似解を求めるという手法が取られている．

データストリームを対象としたアルゴリズムを評価する場合，最も望ましいのはデータストリームを使用することである．しかし，データストリームを用いる場合は前処理に十分な時間を割くことが出来ないという問題や，データストリームを調達してくるのが難しいという問題が存在する．アルゴリズムを評価する際，重要なのはアルゴリズムであってデータではない．そのため，データは軽視されがちであり，データストリームを対象としたアルゴリズムであったとしても，静的なデータを用いるか，自作のデータストリームを使うことがしばしば存在する．静的なデータを用いる場合，アルゴリズムにはデータストリームを用いる場合と同じように，逐次的にデータが入力される．静的なデータを対象とするデータマイニングアルゴリズムでは，全てのデータを使用してパラメータを更新するため，データの順序が結果に影響することがない．しかし，データストリームマイニングアルゴリズムでは全て

のデータを使用せずにパラメータを更新するため，データの順序によって結果が異なってしまうという問題点が生じる．

1.1.1 データストリームとは

有村らによると，データストリームとは，「膨大な量のデータが，高速なストリームを通じて，時間的に変化しながら，終わりなく到着しつづけるもの」[2] と定義している．

データストリームの例としては，センサからリアルタイムに取得するデータや，株価，気象データ，POS データ，Twitter のツイート，クレジットカードの利用情報などが挙げられる．

データストリームには，局所的に変化するという性質が存在する．例えばバースト出現のように，データの突発的な大量出現が起こった時，それまでにデータストリームを生成していた状態から変化し，データが大量に生成される状態に推移したと考えられる．

1.1.2 時系列データとは

時系列データとは，その名の通り，時系列を含むデータのことである．つまり，得られたデータに順序が存在する場合，時系列データとみなす．データストリームに対して，いつ到着したかという情報を付加してデータベースへ保存することで，データストリームを時系列データとして保存することができる．

1.1.3 静的なデータとは

静的なデータとは，データに流れが存在しないようなデータを指す．本論文では，時系列情報は流れとみなすため，時系列データは静的なデータではないと定義する．

1.1.4 データストリームと時系列データと静的なデータの違い

データストリームと時系列データと静的なデータの違いを表 1.1 に示す．データストリームには，時間的に変化するという性質があるため，時系列データも時間的に変化するという性質を保持していると考える．データストリームには終わりなく到着しつづけるという性質があるため，無限に生成されていくと考えられるが，時系列データでは，データベース上に保存されているデータが全てだと考えるため，データストリームが保有していた，終わりなく到着しつづけるという性質は失われる．そのため，時系列データは静的なデータと同様に有限となる．時系列データは，前項で述べたように，データストリームをデータベース上に保存したものである．データストリームには，時間的に変化するという性質があるため，時系列データも時間的に変化するという性質を保持するため，時系列データの性質は動的となる．

表 1.1 データストリームと時系列データと静的なデータの特徴

	データストリーム	時系列データ	静的なデータ
データの性質	動的	動的	静的
有限か	無限	有限	有限

1.1.5 データストリームマイニングとは

データマイニングとは，データから有益な情報を掘り起こすことを指す．データストリームマイニングとは，データストリームを対象としたマイニングを指す．

データストリームマイニングでは，データを素早く処理することが重要である．例として，10 分後の株価の予想に，1 時間掛けて予測しても意味がない．データストリームマイニングでは，高速に処理を行うため，データストリームを主記憶装置上に展開し，必要な情報のみを取り出すという手法が取られる．

また，データストリームは無限に生成されるが，生成される全てのデータを保存していると，ストレージがすぐに溢れてしまう可能性も考えられる．そこで，データストリームから必要な情報のみを取り出し，残りの情報を破棄するといった処理が行われる場合もある．例として，センサを用いて何らかの異常を検知する場合を想定すると，正常であることを確認すると，正常な状態の時にセンサから送られてくるデータには価値がなくなり，保存する必要が無くなる．

1.1.6 従来のデータマイニングとデータストリームマイニングの差

従来の静的なデータを対象としたデータマイニングと，データストリームを対象としたデータストリームマイニングの差を表 1.2 に示す．従来のデータマイニングでは，データマイニングを行う時点で，データマイニングに使用する全てのデータが揃っていて，かつ前処理に時間的な制約が無く，理論上無制限のリソースを使うことが出来る．

一方，データストリームマイニングでは，データマイニングを行う時点では，データマイニングに使用するデータが全て揃っておらず，逐次データが到着する．そして，到着するデータをリアルタイムに処理する必要があるため，必然的に前処理に対して時間的な制約が課せられる．また，無限に生成されるデータストリームに対して，有限の計算資源だけを用いて，処理を行う必要が生じる．

1.2 目的

データストリームマイニングのアルゴリズムを性能評価する場合，最も望ましいのはデータストリームを使用することである．しかし，データストリームを使用する場合，検討すべ

表 1.2 従来のデータマイニングとデータストリームマイニングの特徴

	従来のデータマイニング	データストリームマイニング
データ	全て揃っている	逐次到着
前処理	時間的な制約無し	リアルタイム処理が必要
リソース	理論上無限	制限あり

き事項が増える．例えば，データストリームが到着するまでの経路によっては，データに欠損が起こる可能性が考えられる．欠損したデータに対して，リアルタイムに前処理を行う必要がある．一方，静的なデータを使用する場合，擬似的にデータストリームとしてアルゴリズムに渡すため，経路によってデータに欠損が起こる可能性は考えられない．また，欠損しているデータを使用する場合では，前処理に時間的な制約は存在しない．このように，静的なデータを使用するほうが，検討すべき事項が少ないため，一般的にデータストリームマイニングアルゴリズムを評価する場合，静的なデータを用いて性能評価を行うことが多い．しかし，静的なデータにはデータストリームが持つ，局所的に変化するという性質が含まれていない．局所的に変化する性質を持つ，データストリームを対象としたデータストリームマイニングアルゴリズムの性能を評価する際に，局所的に変化するという性質が含まれていない静的なデータを使用すると，正しく性能評価が行えない可能性が考えられる．また，静的なデータには順序が存在しないため，データストリームマイニングアルゴリズムに対して，疑似データストリームとして与える場合，与える順序をどのように決定するのかという問題が存在する．そこで，局所的に変化するという性質を持ち，データに順序が存在する時系列データを用いることで，これらの問題を解決できると考える．そこで，本研究では時系列データを用いることで，静的なデータを使用した場合に比べて，データストリームマイニングアルゴリズムの性能をより正しく評価できるかを検討する．

1.3 論文構成

第 1 章では本研究についての背景と目的について述べた. 第 2 章では, 関連研究・技術について述べる. 第 3 章では, 本研究で用いるデータストリームを対象としたマイニング手法の一種であるオンライン学習について述べる. 第 4 章では, 本研究で提案する手法とその概要について述べる. 第 5 章では, 実験と結果について述べる. 第 6 章では, 実験結果についての考察を述べる．第 7 章では本論文のまとめと今後の展望について述べる．

第 2 章

既存研究・技術

本研究に関連のある研究・技術を示す．背景ではデータストリームマイニングについて述べたが，本章では機械学習における学習手法について述べる．データストリームマイニングでは，逐次的に到着するデータを素早く処理する必要がある．従来の静的なデータを対象としたデータマイニングで用いられる学習手法と，データストリームマイニングで用いられる学習手法について述べた後，データストリームを扱った関連研究を踏まえ，本研究を位置づける．

2.1 バッチ学習

バッチ学習とは，学習の対象であるデータを全て一括で処理する手法である．バッチ学習では．モデルを更新する際に，学習データを全て使用する．学習データの順序による影響を受けないという特徴を持つ．しかし，学習データを追加する場合，一からモデルを構築し直す必要が生じるため，モデル更新のコストが高くなる．学習データを全て使用してモデルを更新するため，メモリデータストリームに対してバッチ学習を行う場合，次々に到着するデータに対して対応するためには膨大なリソースが必要となるため，データストリームマイニングで使用されることは無い．

2.2 ミニバッチ学習

ミニバッチ学習とは，バッチ学習と，後述するオンライン学習の中間的な手法である．ミニバッチ学習では，学習する際に，一度に入力するデータ数を指定して学習を行う．そのため，ミニバッチ学習ではモデルを更新する際に，指定したデータ数を使用する．学習データを分割して学習を行うため，学習過程を並列化することが可能となる．バッチ学習では，学習するデータの順序による影響を受けないが，ミニバッチ学習では，学習データの順序によって構築されるモデルが変化する．データストリームに対してミニバッチ学習を行う場合，指定したデータ数になるまでデータを貯めておき，指定したデータ数に達すると学習を

行い，貯めたデータを破棄し，新たなデータの到着を待つという一連の流れを繰り返して学習する．

2.3 オンライン学習

オンライン学習とは，データが 1 つずつ逐次的に与えられる状況下において，データが与えられる度にモデルを更新する学習手法である．具体的なオンライン学習は 3 章で詳しく述べる．オンライン学習では，新たに追加したデータと，既存のモデルを比較してモデルを更新するため，モデル更新のコストが低く，素早く更新することが出来るという特徴を持つ．また，学習中にメモリ上に展開するデータは基本的に入力データのみとなるため，少ないメモリ使用量で学習を進めることが出来る．しかし，学習データ 1 つに対してモデルを更新するため，ノイズに弱いという欠点と，学習データの順序によって構築されるモデルが変化するという欠点が存在するデータストリームに対してオンライン学習を行う場合，データが到着する度に学習を行い，到着したデータを破棄し，新たなデータの到着を待つという一連の流れを繰り返して学習する．

2.4 関連研究

データストリームを扱った関連研究について述べる．

2.4.1 交通シュミレータを用いて，ベンチマーク用のデータストリームを作成

アルスらが 2004 年に提案した手法 [3] を示す．交通シミュレータから，擬似的に高速道路上のデータを生成し，車両の位置や料金，事故情報といったデータとデータストリームとして生成し，データストリーム管理システムを評価する手法を提案している．交通シミュレータからデータを生成する際に，指定しなければならないため，データストリームを得るまでにコストが掛かる．

2.4.2 データストリームマイニングアルゴリズムの評価例

データストリーム中の頻出アイテムを近似的に抽出するオンライン型アルゴリズムである Lossy Counting Algorithm に先頭系列頻度を用いて，頻出部分系列を高速近似抽出するオンラインアルゴリズムを提案しているが，評価実験では人工的に作成したデータを用いている [4].

逆単調性を満たす全体頻度なる出現頻度を用いて，大規模時系列データを対象とした頻出パターンのオンライン型高速抽出アルゴリズムを提案しているが，評価実験では乱数で作成したアイテム集合を用いている [5].

Frequent に，k-reduced bag の考えを用いて改良した KRB というアルゴリズムを提案しているが，評価実験では zipf 則に基づく人工データを用いている [6].

人工的に作成したデータでは，実在するデータストリームと乖離している可能性が考えられる．

第 3 章

オンライン学習の各手法

本節では，具体的なオンライン学習手法を紹介する．

3.1 単純パーセプトロン

単純パーセプトロン [7] はもっとも単純な線形分離器である．本来の単純パーセプトロンは全データの識別が成功するまで学習を繰り返すが，確率勾配降下法によりパーセプトロンを学習させることで，1 データごとにパラメータの更新するオンライン学習に拡張させることができる．教師データの値の正負によって 2 値分類を行い，教師データの値と識別結果の正負が間違っている場合にのみ重みベクトルの更新を行う．重みベクトルの更新式は以下のようになる．

$$\mathbf{w}_{i+1} = \begin{cases} \mathbf{w}_i + (y_i \mathbf{x}_i) & (-y_i \mathbf{w}_{i\top} \mathbf{x}_i > 0) \\ \mathbf{w}_i & (otherwise) \end{cases}$$

この方法は単純であるが，予測結果が間違っている場合にのみ一定の更新幅で重みの更新を行うため，収束までに時間がかかる問題が存在する．そこで，正しく予測出来た場合であっても，十分なマージンが取れていない場合は重みを更新する Passive-Aggerresive という手法が登場した．

3.2 Online Passive-Aggressive

Online Passive-Aggressive[7][8] はオンライン学習における教師あり学習の一手法である．Online Passive-Aggressive では，ヒンジ損失関数を用いて最適化問題を逐次的に解いていき，更新する重みベクトルを求める．

入力には，データ点集合 $\mathbf{x}_i$ と正解ラベル y_i が与えられる．データが入力された時点での重みベクトル $\mathbf{w}_i$ を用いて，入力されたデータ点 d_i が正解であるか，不正解であるかを予測し，正解ラベル y_i と比較して重みベクトル $\mathbf{w}_{i+1}$ を更新する．ヒンジ損失関数は以下の通り

である．

$$\ell_{\text{hinge}}(\mathbf{x}_i, y_i, \mathbf{w}_i) = \max\left(0, 1 - y_i \mathbf{w}_i^\top \mathbf{x}_i\right)$$

重みベクトルの更新するための最適化問題は以下の通りである．

$$\mathbf{w}_{i+1} = \arg\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w} - \mathbf{w}_i\|^2$$

subject to

$$\ell_{hinge}(\mathbf{x}_i, y_i, \mathbf{w}_i) = 0$$

この最適化問題から，重みの更新式は以下のようになる．

$$\mathbf{w}_{i+1} = \begin{cases} \mathbf{w}_i + \frac{\ell_{\text{hinge}}(\mathbf{x}_i, y_i, \mathbf{w}_i)}{\|\mathbf{x}_i\|^2} y_i \mathbf{x}_i & (\ell_{hinge}(\mathbf{x}_i, y_i, \mathbf{w}_i) > 0) \\ \mathbf{w}_i & (otherwise) \end{cases}$$

3.3 Online Passive-Aggressive-I

Online Passive-Aggressive では，ヒンジ損失関数が 0 となる重みに更新する．そのため，ノイズに対して敏感であり，それまで学習した結果を破棄して，ノイズに合わせて大きく学習してしまうという問題点があった．そこで，ある程度の誤りを許容する，Online Passive-Aggressive-I という方法が提案された．どの程度の誤りを許容するかというパラメータ $C > 0$ を指定する必要がある．重みベクトルの更新するための最適化問題は以下の通りである．

$$\mathbf{w}_{i+1} = \arg\min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w} - \mathbf{w}_i\|^2 + C\xi$$

subject to

$$\ell_{hinge}(\mathbf{x}_i, y_i, \mathbf{w}) \leq \xi \quad and \quad \xi \geq 0$$

この最適化問題から，重みの更新式は以下のようになる．

$$\mathbf{w}_{i+1} = \begin{cases} \mathbf{w}_i + \frac{\min(C, \ell_{\text{hinge}}(\mathbf{x}_i, y_i, \mathbf{w}_i))}{\|\mathbf{x}_i\|^2} y_i \mathbf{x}_i & (\ell_{hinge}(\mathbf{x}_i, y_i, \mathbf{w}_i) > 0) \\ \mathbf{w}_i & (otherwise) \end{cases}$$

3.4 Online Passive-Aggressive-Ⅱ

Online Passive-Aggressive-I では，ヒンジ損失関数が 0 でなかった場合のペナルティを線形に考えていたが，Online Passive-Aggressive-Ⅱ では，ペナルティを 2 乗で考える．重みベクトルの更新するための最適化問題は以下の通りである．

$$\mathbf{w}_{i+1} = \arg\min_{\mathbf{w}} \frac{1}{2} \left\|\mathbf{w} - \mathbf{w}_{(i)}\right\|^2 + C\xi^2$$

subject to

$$\ell_{hihge}\left(\mathbf{x}_i, y_i, \mathbf{w}\right) \leq \xi$$

この最適化問題から，重みの更新式は以下のようになる．

$$\mathbf{w}_{i+1} = \begin{cases} \mathbf{w}_i + \frac{\ell_{\text{hinge}}(\mathbf{x}_i, y_i, \mathbf{w}_i)}{\|\mathbf{x}_i\|^2 + \frac{1}{2C}} y_i \mathbf{x}_i & (\ell_{hinge}(\mathbf{x}_i, y_i, \mathbf{w}_i) > 0) \\ \mathbf{w}_i & (otherwise) \end{cases}$$

第 4 章

提案手法

本章では，提案する手法について述べる．

4.1 提案手法

本論文では，時系列データはデータストリームをデータベース上に保存したものと考える．局所的に変化するという性質を持つ時系列データを用いて，データストリームマイニングアルゴリズムの性能評価を行う方法を提案する．時系列順にデータストリームとすることで，実在するデータストリームに近い疑似データストリームをアルゴリズムに与えることが出来るのではないかと考える．時系列データを使用することで，よりデータストリームに近いデータを用いてアルゴリズムを評価することが出来るかどうかを検証する．

4.2 アルゴリズム

提案手法を評価するためのアルゴリズムとして，データストリームマイニングアルゴリズムの一種である，オンラインアルゴリズムの Online Passive-Aggressive[8] を用いる．

Online Passive-Aggressive は実装が容易であり， オンラインアルゴリズムの特徴であるノイズに弱いという特徴を持っている．ノイズに弱いということは，データに対して機敏に反応するため，本実験に適していると判断した．

4.2.1 オンラインアルゴリズムのモデル更新

オンラインアルゴリズムでは，次々に到着する未知のデータに対して，高い予測精度の実現を目指してモデルの構築を行う．そのため，オンラインアルゴリズムではデータが到着するたびにモデルの更新を行う．モデルの更新を行う際には，到着したデータを更新前のモデルでうまく予測できるかどうかによって，更新する値が大きく変わる．

例として，Online Passive-Aggressive では，境界面から十分なマージンを取って分類が出

来る場合，モデルの更新を行わない．つまり，正例だけが極端に集中して流れてきた場合，正例は十分なマージンを取って分類が出来るため，モデルの更新が行われず，負例が到着した時に大きくモデルの更新が行われるということも考えられる．この時，負例をうまく予測できるようにモデルの更新が行われるため，正例がうまく分類出来ないモデルに更新される可能性が考えられる．

4.3 時系列データを使用する利点と欠点

本論文では，時系列データはデータストリームをデータベース上に保存したものと考える．そのため，時系列データを時系列順にオンラインアルゴリズムに与える場合，オンラインアルゴリズムには時系列データが時系列順に１つずつ入力される．これは，ある期間のデータストリームをオンラインアルゴリズムに与えることに等しいと考える．

次に，静的なデータセットを用いてオンラインアルゴリズムを評価する場合を想定する．オンラインアルゴリズムでは，データを 1 つずつ逐次的に与えて学習を行う．そのため，オンラインアルゴリズムにデータを与える時，アルゴリズムに与えるデータの順番を決める必要がある．特に順番を考えずに与える場合，データセットのインデックスに基づいてデータが与えられるが，データセットによっては，ラベルでソートされているものも存在する．このような，極端に偏ったデータを使用する場合は，4.2.1 で述べたようにうまくモデルが構築できない可能性も考えられる．データのインデックスを振り直すことで，この問題は解決出来るが，どのようにしてインデックスを決定するのかという問題が発生する．インデックスをランダムに決める場合，偏ったデータにならないという保証はない．また，そもそもにおいて，静的なデータセットの中にはデータの順序は存在していない．バッチ学習ではデータの順序を考慮せずにモデル構築が行われるが，オンライン学習ではデータの順序を考慮してモデル構築が行われる．データの順序が含まれていないようなデータセットを使用してオンラインアルゴリズムを評価しても正しく性能評価が出来ない可能性が考えられる．

一方，時系列データであれば時系列情報自体がデータの順序となるため，静的なデータセットを使用する場合に考えられる問題は解消出来ると考える．

欠点として，時系列データセットは静的なデータセットに比べて入手が困難であるという点が挙げられる．

4.4 検証方法

局所的に変化するという性質を持つ時系列データを用いることで，よりデータストリームに近いデータを用いてアルゴリズムを評価することが出来るかどうかを検証する．従来のバッチ学習で用いられる，時系列情報を含まない静的なデータと，時系列情報を含む時系列データをそれぞれ用いてアルゴリズムの評価を行う．学習するデータの順番を変化させて，精度がどの程度変化するかを検証する．データの順番は以下のように変化させる．

- 正例と負例を極端に配置
- データの順序をランダマイズ

4.5 評価方法

評価指標として，正解率 (accuracy)，適合率 (precision)，再現率 (recall) と混合行列 (Confusion Matrix) を使用する．評価には，学習に使用するデータとは別の評価用データを用いる．学習する毎に評価データを使って正解率の算出を行い，評価時にはモデルの更新は行わない．

第 5 章

実験と結果

本研究で行った実験手法と使用したデータ，評価手法について述べる．

5.1 実験概要

本実験では，2 クラス分類問題を想定して行う．静的なデータセットを用いて，データストリームマイニングアルゴリズムを学習し，評価を行った場合と，局所的に変化するという性質を持つ時系列データセットを用いてデータストリームマイニングアルゴリズムを学習し，評価を行った場合との，正解率の推移を比較する．

静的なデータを用いてデータストリームマイニングアルゴリズムを実行する時，データをアルゴリズムに与える順序を決める必要がある．仮に順序をランダムに決める場合，正例と負例が極端に配置された順序になる可能性は否定できない．正例と負例が極端に配置された順序の場合でも，アルゴリズムが正しく評価できているかを検証するために与える順序がどの程度影響するのかを検証するために，静的データに下記のような加工を施したデータを用いて実験を行う．

- ターゲットラベルに対して昇順でソート (先に負例)
- ターゲットラベルに対して降順でソート (先に正例)
- データの順序をランダマイズ

時系列データは，時系列自体がデータを与える順序を担うと考える．本稿の定義では，時系列データから時系列情報を削除することで静的データとなるので，時系列を無視して静的データとして扱った場合．どの程度正解率に影響するのかを検証する．時系列データに行う加工は以下の通りである．

- 時系列を無視し， ターゲットラベルに対して昇順でソート (先に負例)
- 時系列を無視し，ターゲットラベルに対して降順でソート (先に正例)
- 時系列を無視し，データの順序をランダマイズ

5.2 使用したツール

本実験で使用したツールを表 5.1 に示す.

表 5.1 使用したツール

ツール名	バージョン
Python	3.7.4
Gnuplot	5.2
pandas	0.25.1
scikit-learn	0.21.3
NumPy	1.16.5

Numpy とは，Python で主に配列数値計算を行うためのライブラリであり，Python で標準で搭載されている配列機能に比較して，効率的に多次元配列を扱うことができる.

pandas とは，Python でデータ分析を支援するライブラリであり，R 言語に類似するデータフレームを提供する．本研究では，使用するデータセットを読み込むために使用した.

scikit-learn とは，Python で機械学習を行うためのライブラリであり，本研究では，混合行列 (Confusion Matrix) を測定するために使用した.

Gnuplot とは，コマンドライン上で動作するグラフを作成するためのソフトウェアである.

5.3 対象のデータストリームマイニングアルゴリズム

本稿では，オンラインアルゴリズムを対象に実験を行う．オンライン学習とは，データが１つずつ逐次的に与えられる状況下において，データが与えられる度にパラメータを更新する学習方法である．実験で使用するアルゴリズムは，以下の通りである.

- Online Passive-Aggressive
- Online Passive-Aggressive-I
- Online Passive-Aggressive-Ⅱ

今回は，重みの初期値を 1 と設定する．実装は Python で行った.

5.4 使用するデータセット

本実験では，静的なデータとして，Fisher の Iris Flower Data Set，時系列データとして，The UEA & UCR Time Series Classification Repository[9] にある，クラス数が 2 のデータセットを用いて実験を行う．静的なデータとして用いたのは，表 5.2，時系列データとして

用いたのは，表 5.3 の通りである．

表 5.2 使用した静的なデータセット

	概要	訓練	評価	属性
Iris Data Set 1	訓練データとして 80 個，残りを評価データに分割	80	20	4
Iris Data Set 2	訓練データとして 50 個，残りを評価データに分割	50	50	4

表 5.3 使用した時系列データセット

	概要	訓練	評価	属性
Earthquakes	北カルフォルニアの地震測定データ	322	139	512
FordA	エンジンノイズからサブシステムの故障を分類	3601	1320	500
FordB	評価データをノイズの多い条件下で収集	3636	810	500
FreezerRegular_Train	キッチンとガレージの冷蔵庫の消費電力測定データ	150	2850	301
FreezerSmall_Train	上記のデータと同じデータセットから作成	28	2850	301
ItalyPowerDemand	イタリアの 12 ヶ月の電力需要時系列データセット	67	1029	24
Wafer	シリコンウェーハが正常か異常かを分類する	1000	6164	152

次に，使用したデータセットについて，説明する．

Iris Flower Data Set は 1936 年に Fisher が発表した論文上で取り上げたデータである [10]．多変量データとして，統計学では有名なデータセットである．インスタンス数は 150，属性数は 4 である．

Earthquakes は北カルフォルニアで 1067 年 12 月 1 日から 2003 年に渡って，1 時間毎に測定されたセンサデータである．512 時間分のデータを 1 インスタンスとして扱い，512 時間の間に地震が発生したかどうかを分類する．訓練データは 322 インスタンス，評価データは 139 インスタンス，属性数は 512 である．

FordA は 2008 年の IEEE World Congress on Computational Intelligence でのコンペティションに使用されたデータセットである．各インスタンスはエンジンノイズの 500 回の測定データで構成されており，自動車のサブシステムに特定の症状が存在するかどうかを分類する．訓練データは 3601 インスタンス，評価データは 1320 インスタンス，属性数は 500 である．FordA はノイズを最小限に抑えた状態で、訓練データとテストデータを収集されている。

FordB も，同じく 2008 年のコンペティションで使用されたデータセットで，各インスタンスも FordA と同様である．訓練データは 3636 インスタンス，評価データは 810 インスタンスである．FordB は，訓練データは FordA と同じ条件で収集しているが，テストデータはノイズの多い条件下で収集されている．

FreezerRegularTrain は，2013 年から 2014 年の間に，イギリスの 20 世帯に置かれているキッチンの冷凍庫と，あまり使用されないガレージの冷凍庫の消費電力を測定したデータセットである．消費電力から，どちらの冷凍庫かを分類する．訓練データは 150 インスタンス，評価データは 2850 インスタンス，属性数は 301 である．

FreezerSmallTrain は，FreezerRegularTrain と同じデータセットから作られているが，訓練データが 28 インスタンスという違いがある．評価データ，属性数は FreezerRegularTrain と同じである．

ItalyPowerDemand は，イタリアの 12 ヶ月の電力需要時系列データである．訓練データは 67 インスタンス，評価データは 1029 インスタンス，属性数は 24 である．4 月から 9 月と，10 月から 3 月までにそれぞれラベルが付けられている．

Wafer は，2001 年に R. Oiszewski によって発表された論文 [11] で作成された，半導体マイクロエレクトロニクス製造に関してのデータセットである．半導体製造用のシリコンウェーハの処理中に，様々なセンサから測定されたデータが含まれており，シリコンウェーハが正常か異常かを分類する．正常と異常の間に大きな不均衡が存在する．訓練データが 1000 インスタンス，評価データが 6164 インスタンス，属性数は 152 である．

5.5 実験に当たっての前処理

Fisher の Iris Flower Data Set は 3 クラスのデータであるが，Online Passive-aggressve でクラス分類を簡潔に行うため，3 クラスのデータを 2 クラスの組に変更した．今回の実験で使用したのは setosa と versicolor である．Iris Flower Data Set は訓練データと評価データに分かれていないため，事前に訓練データとして 80 インスタンス，訓練データ以外を評価データとして分割を行った場合と，訓練データとして 50 インスタンス，訓練データ以外を評価データとして分割した場合の，2 組の訓練データと評価データの組を作成した．データを分割するために，乱数を用いてサンプリングを行ったが，実行する度に結果が変化しないように，乱数のシードを固定した．今回は，setosa を負例，versicolor を正例として扱った．

The UEA & UCR Time Series Classification Repository で配布されていたデータでは，目的変数がデータセットによって値が異なっていたため，目的変数の値が大きい方を正例, 小さい方を負例として扱った．また，arff 形式のデータを読み込んで実験を行ったが，arff の中に含まれるデータセットのメタ情報部分に 2 バイト文字が含まれており，読み込みに支障をきたすため，メタ情報部分の削除を行った．

学習データの順序を変えるため，学習データに対して，ターゲットラベルで昇順ソートと降順ソート，乱数のシードを固定して，学習データの数だけ非復元抽出を行った．

5.6 提案手法のパラメータ設定

今回の実験では，データを擬似データストリームとして扱う．そのため，エポック数は 1 とした．Online Passive-Aggressive-I 並びに，Online Passive-Aggressive-II では， 誤りの許容量 $C > 0$ をパラメータとして指定するが，今回は $C = 0.1$ と設定した．

5.7 評価方法

評価指標として，学習する毎の正解率 (accuracy) と学習が終了した時点での正解率 (accuracy)，適合率 (precision)，再現率 (recall)，混合行列 (Confusion Matrix) を測定する．評価には，学習に使用したデータとは別の評価用データを用いた．学習する毎に評価データを使って正解率の算出を行い，評価時にはモデルの更新を行わない．

5.8 結果

まず，静的なデータである Iris Flower Data Set の結果を述べる．80 インスタンスを訓練データ，残りを評価データとした場合，特にデータに加工を行わなかった場合は，Passive-Aggressive では 1 個目を学習した時点で精度が 1 となり，Passive-Aggressive-I では 7 個目を学習した時点で精度が 1，Passive-Aggressive-II では，3 個目を学習した時点で精度が 1 となり，それ以降精度が変わることはなかった．次に，昇順ソートをした場合，3 個目を学習した時点で全てのアルゴリズムの精度が 1 となった．降順でソートをした場合，40 個強のデータを学習するまでは精度が 0.5 で，それ以降は 1 となった．学習データをランダマイズした場合，10 個弱のデータを学習した時点で．全てのアルゴリズムで精度は 1 となった．

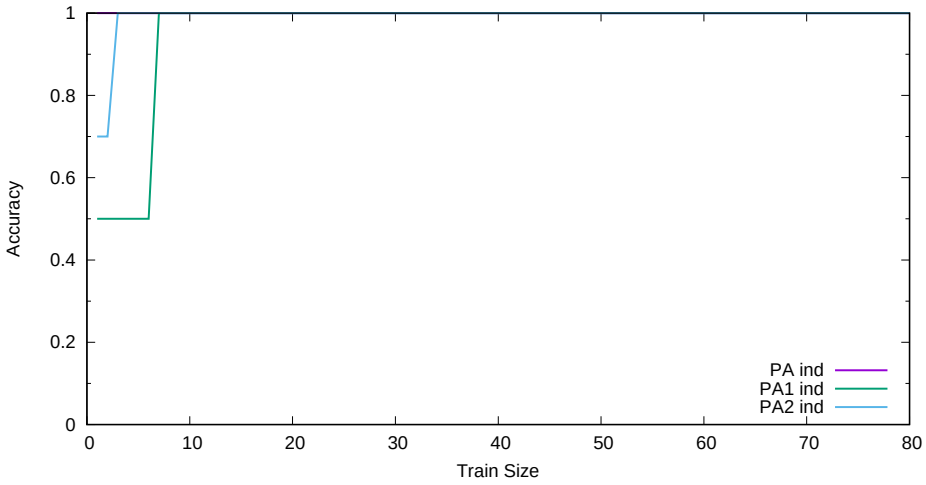

図 5.1 Iris Train80 加工なし

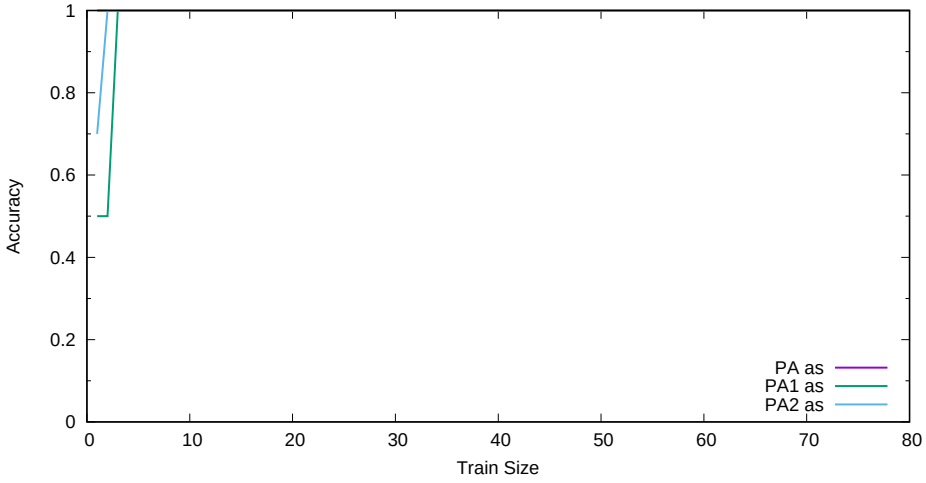

図 5.2 Iris Train80 昇順ソート

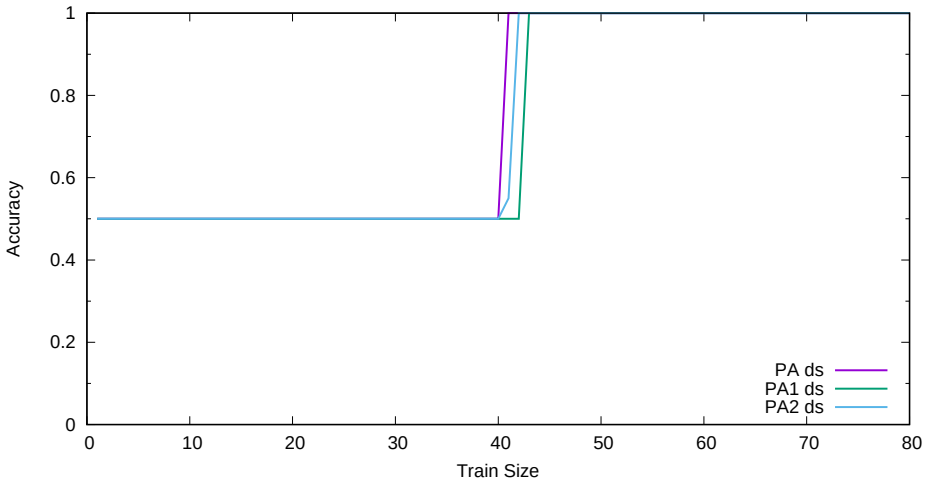

図 5.3 Iris Train80 降順ソート

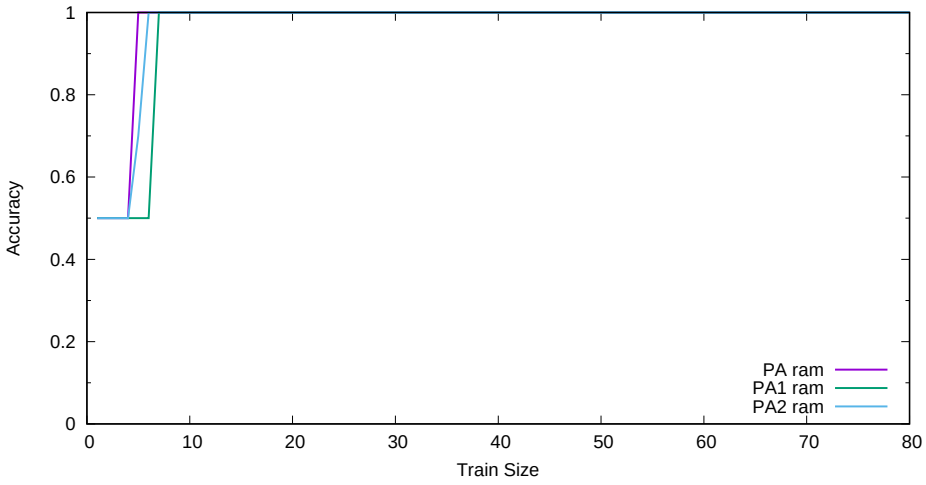

図 5.4 Iris Train80 ランダマイズ

表 5.4 Iris Train80 に対する各モデルの予測結果

モデル	TP	TN	FP	FN
PA 無加工	10	10	0	0
PA 昇順	10	10	0	0
PA 降順	10	10	0	0
PA ランダム	10	10	0	0
PA-I 無加工	10	10	0	0
PA-I 昇順	10	10	0	0
PA-I 降順	10	10	0	0
PA-I ランダム	10	10	0	0
PA-II 無加工	10	10	0	0
PA-II 昇順	10	10	0	0
PA-II 降順	10	10	0	0
PA-II ランダム	10	10	0	0

次に，50 インスタンスを訓練データとした場合は，降順でソートした時，25 個強のデータを学習するまでは精度は 0.5 で，それ以降は 1 となった．その他は学習データが 80 インスタンスの場合と大きく差がなかった．

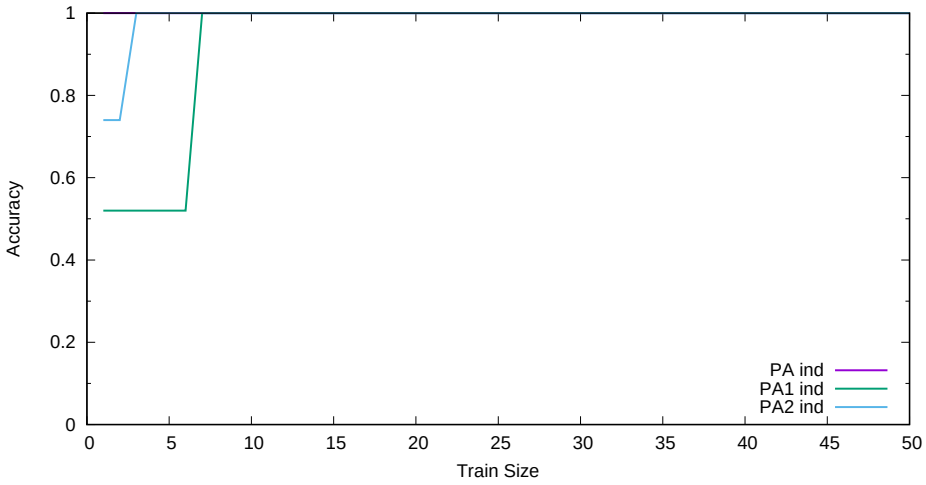

図 5.5 Iris train50 加工なし

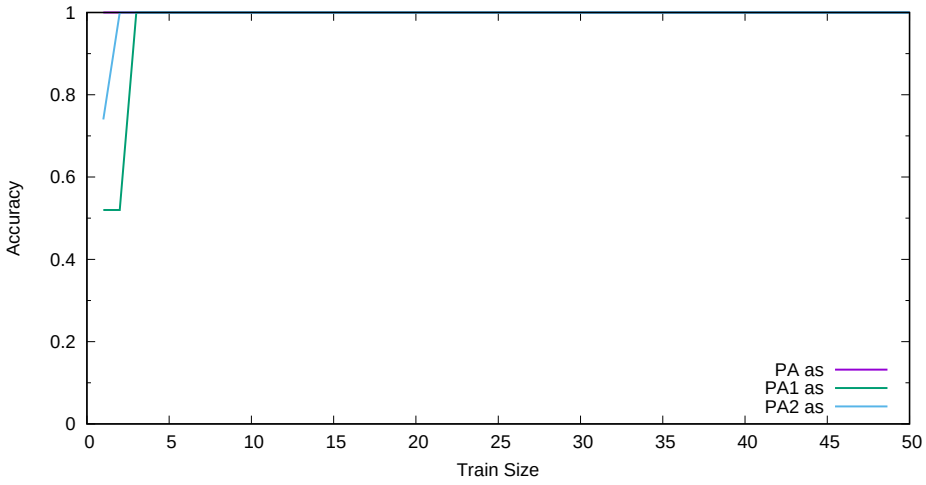

図 5.6 Iris train50 昇順ソート

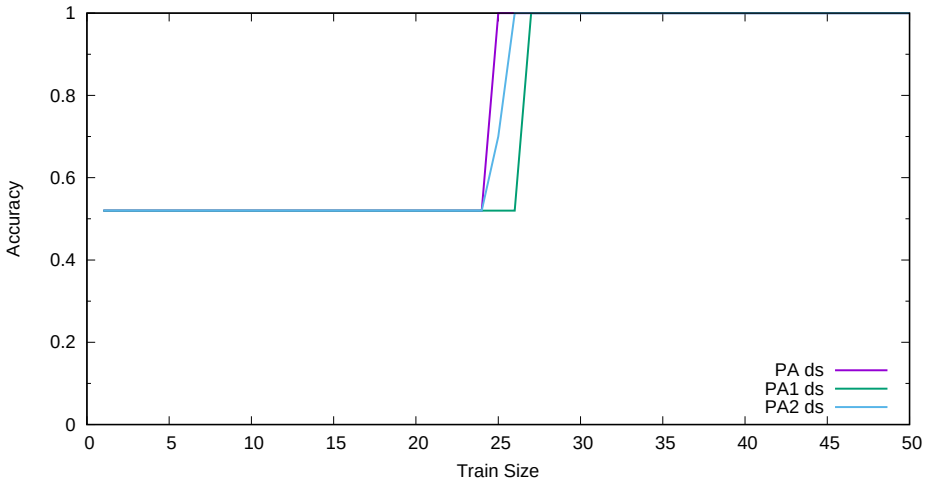

図 5.7　Iris train50 降順ソート

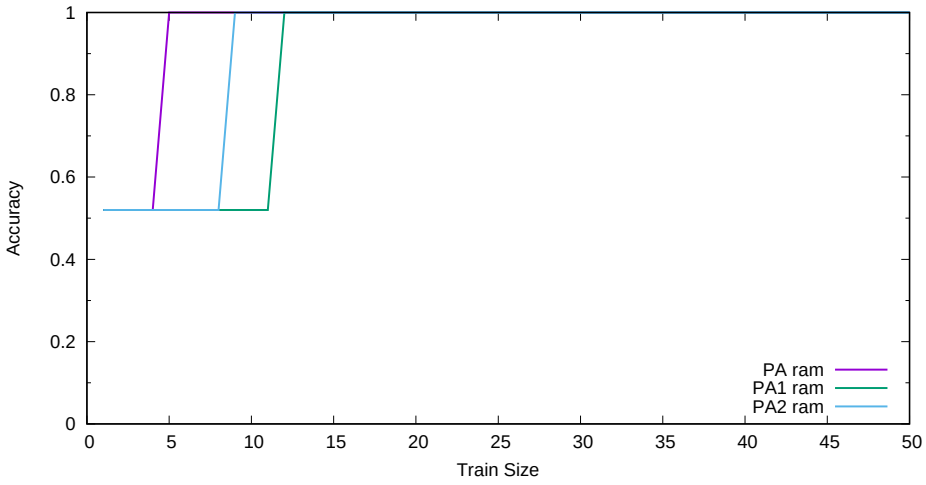

図 5.8　Iris train50 ランダマイズ

表 5.5 Iris Train50 に対する各モデルの予測結果

モデル	TP	TN	FP	FN
PA 無加工	26	24	0	0
PA 昇順	26	24	0	0
PA 降順	26	24	0	0
PA ランダム	26	24	0	0
PA-I 無加工	26	24	0	0
PA-I 昇順	26	24	0	0
PA-I 降順	26	24	0	0
PA-I ランダム	26	24	0	0
PA-II 無加工	26	24	0	0
PA-II 昇順	26	24	0	0
PA-II 降順	26	24	0	0
PA-II ランダム	26	24	0	0

次に時系列データの場合の結果について述べる．Earthquakes，FordA，FordB では，いずれの条件でも精度は 0.5 程度で大きな変化は生じなかった．

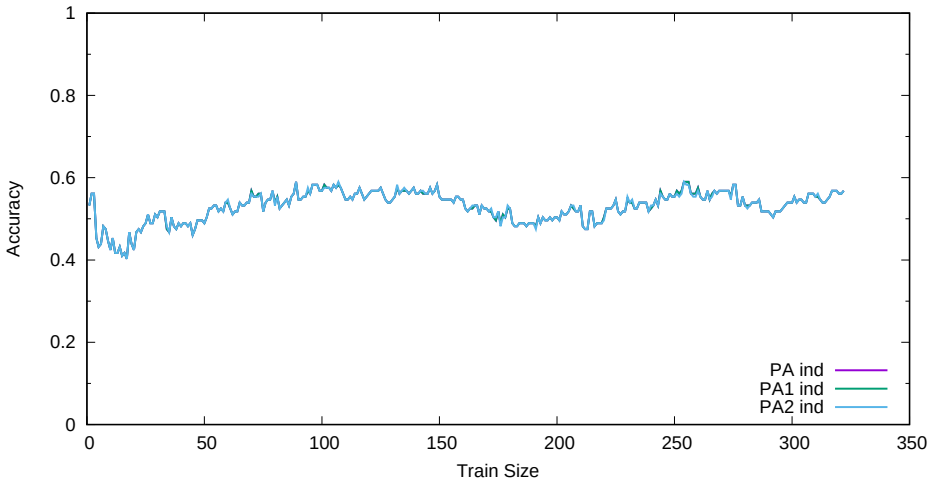

図 5.9 Earthquakes 加工なし

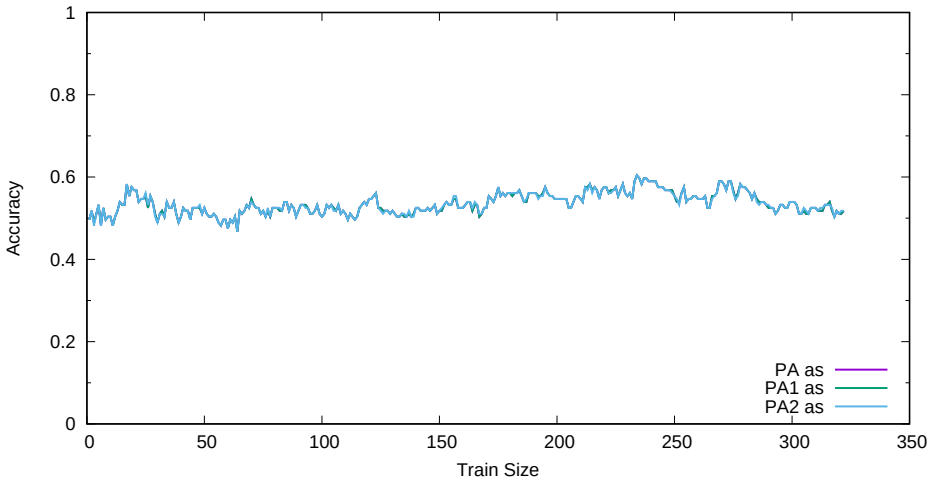

図 5.10 Earthquakes 昇順ソート

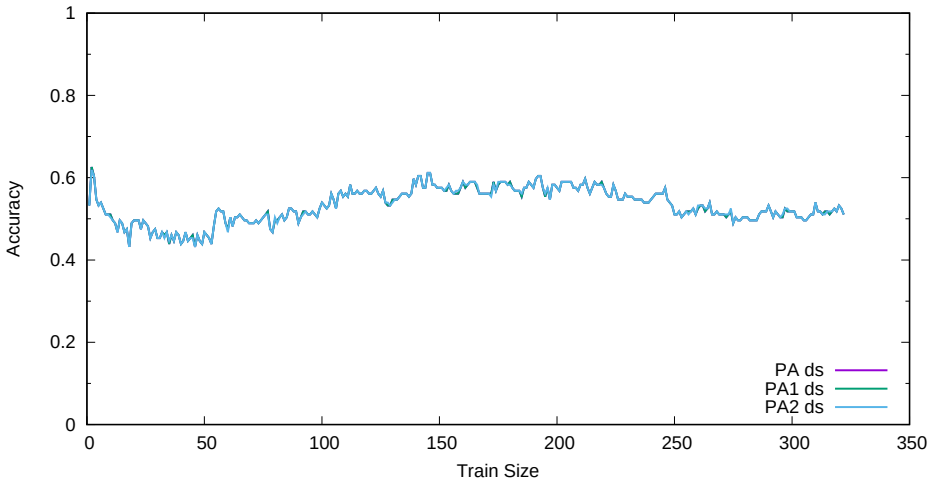

図 5.11 Earthquakes 降順ソート

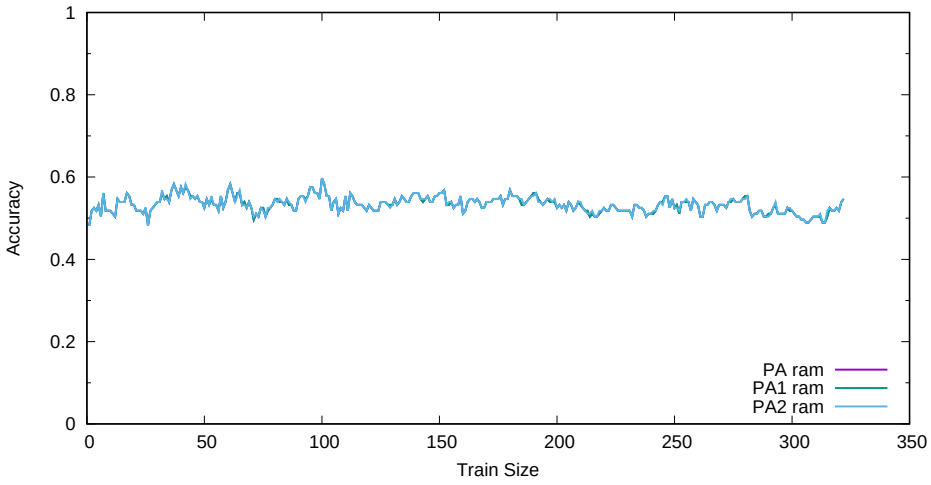

図 5.12 Earthquakes ランダマイズ

表 5.6 Earthquakes に対する各モデルの予測結果

モデル	TP	TN	FP	FN
PA 無加工	18	61	43	17
PA 昇順	18	54	50	17
PA 降順	17	54	50	18
PA ランダム	18	58	46	17
PA-I 無加工	18	61	43	17
PA-I 昇順	18	54	50	17
PA-I 降順	17	54	50	18
PA-I ランダム	18	58	46	17
PA-Ⅱ 無加工	18	61	43	17
PA-Ⅱ 昇順	18	54	50	17
PA-Ⅱ 降順	17	54	50	18
PA-Ⅱ ランダム	18	58	46	17

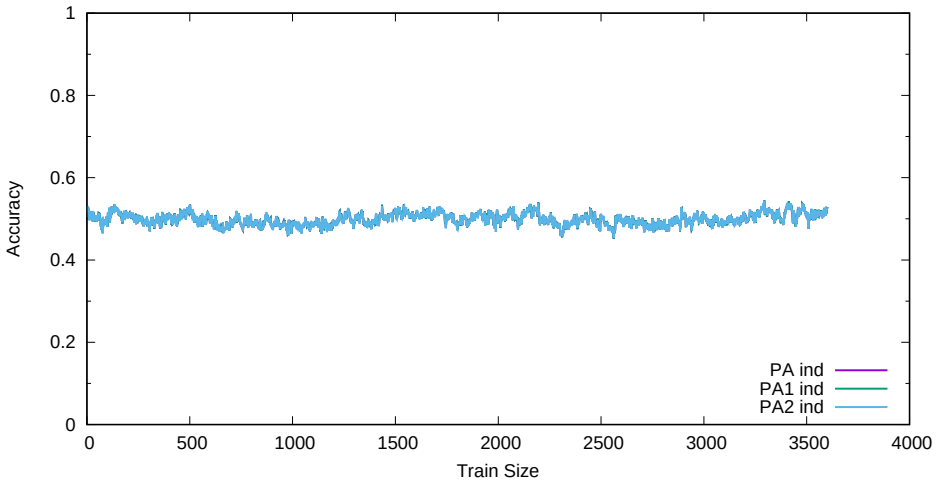

図 5.13　FordA 加工なし

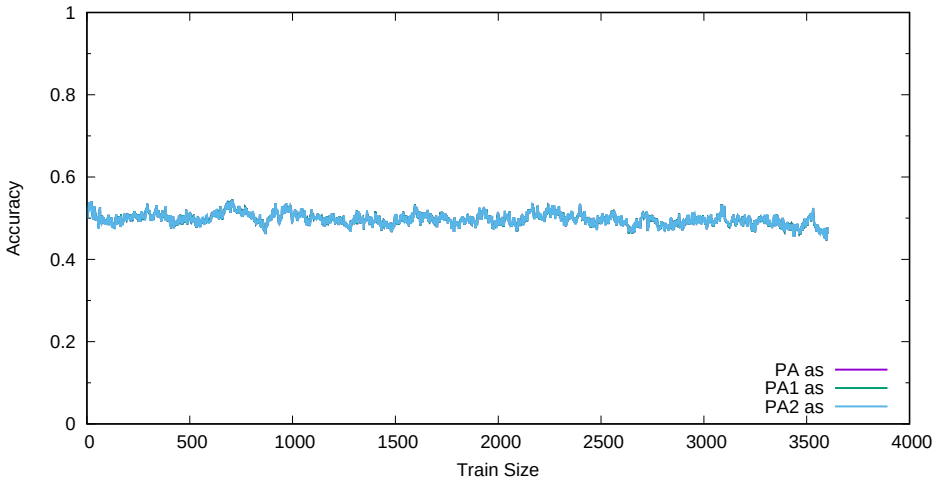

図 5.14　FordA 昇順ソート

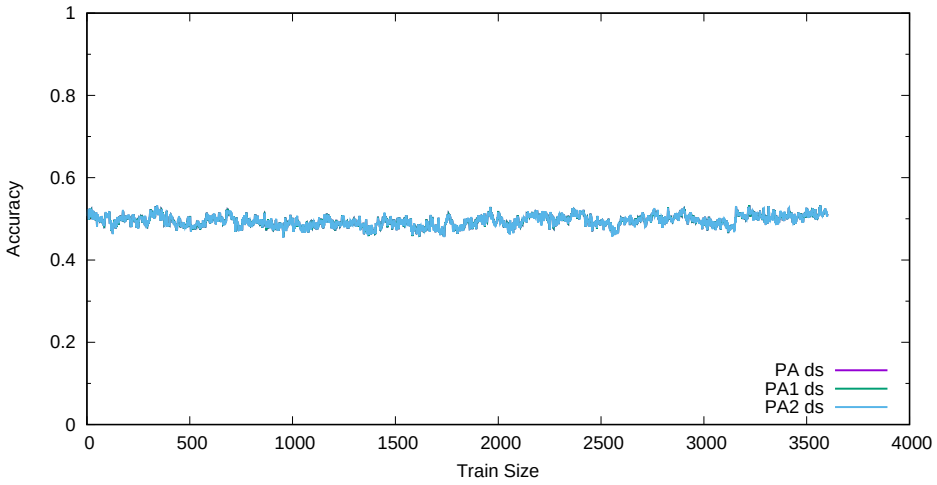

図 5.15　FordA 降順ソート

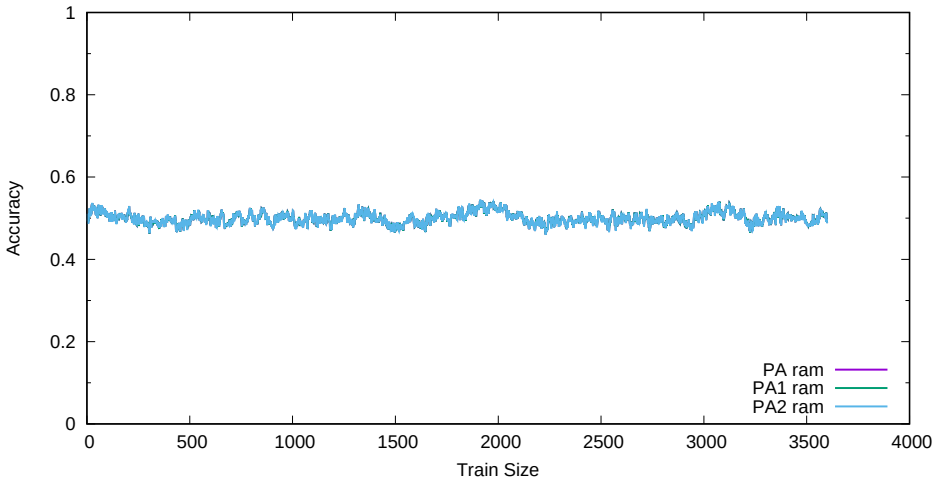

図 5.16　FordA ランダマイズ

表 5.7 FordA に対する各モデルの予測結果

モデル	TP	TN	FP	FN
PA 無加工	336	359	322	303
PA 昇順	304	327	354	335
PA 降順	322	350	331	317
PA ランダム	323	324	357	316
PA-I 無加工	336	359	322	303
PA-I 昇順	304	327	354	335
PA-I 降順	322	350	331	317
PA-I ランダム	323	324	357	316
PA-I 無加工	336	360	321	303
PA-Ⅱ 昇順	304	327	354	335
PA-Ⅱ 降順	324	349	332	315
PA-Ⅱ ランダム	324	324	357	315

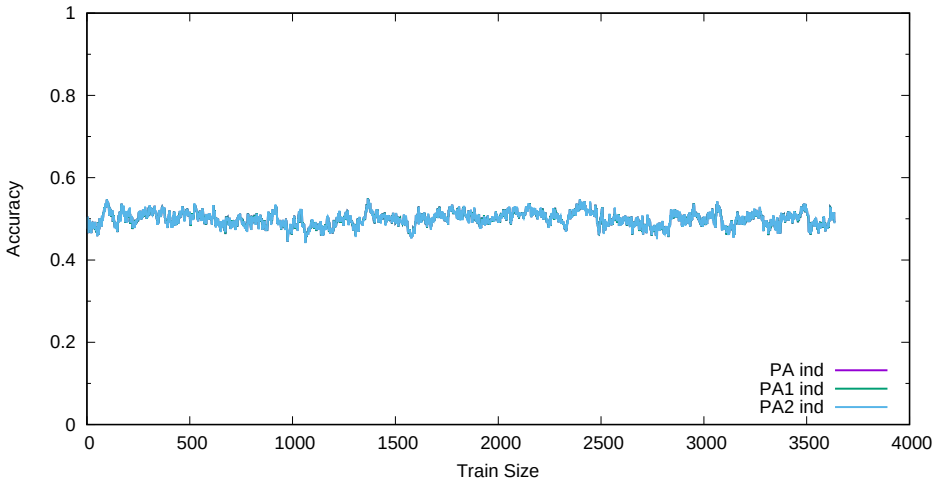

図 5.17　FordB 加工なし

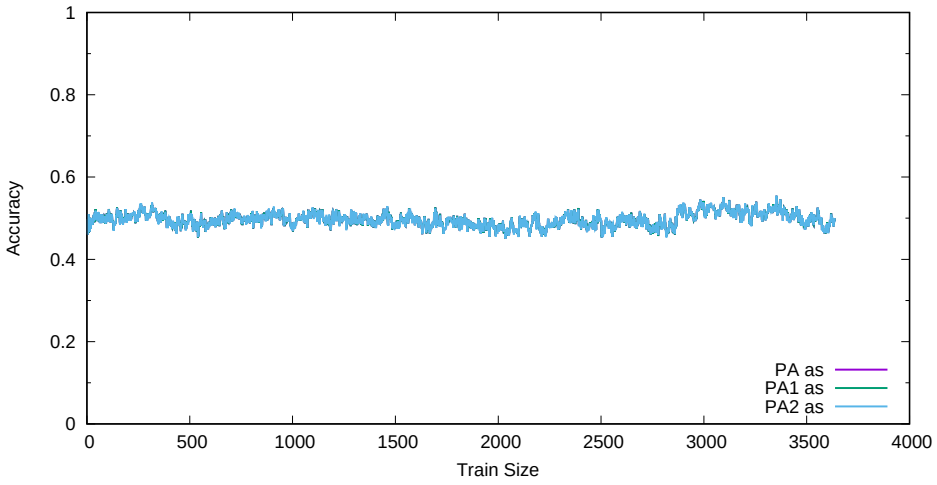

図 5.18　FordB 昇順ソート

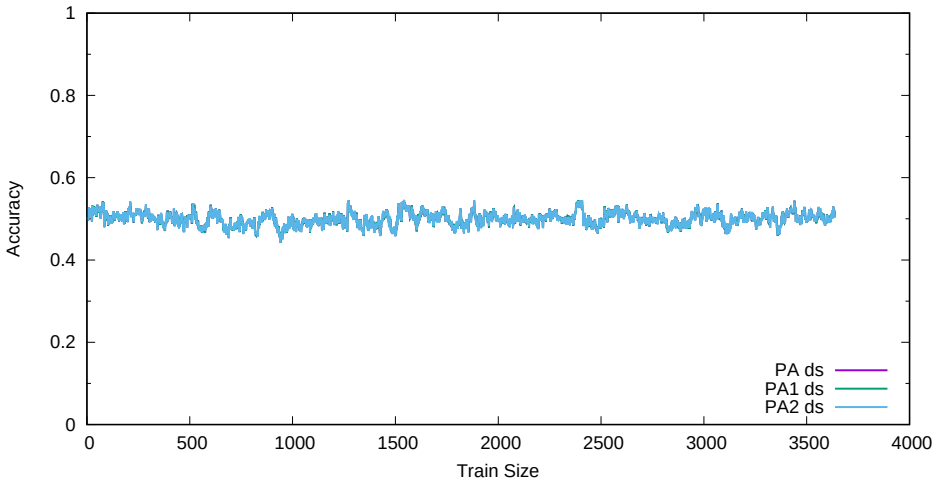

図 5.19　FordB 降順ソート

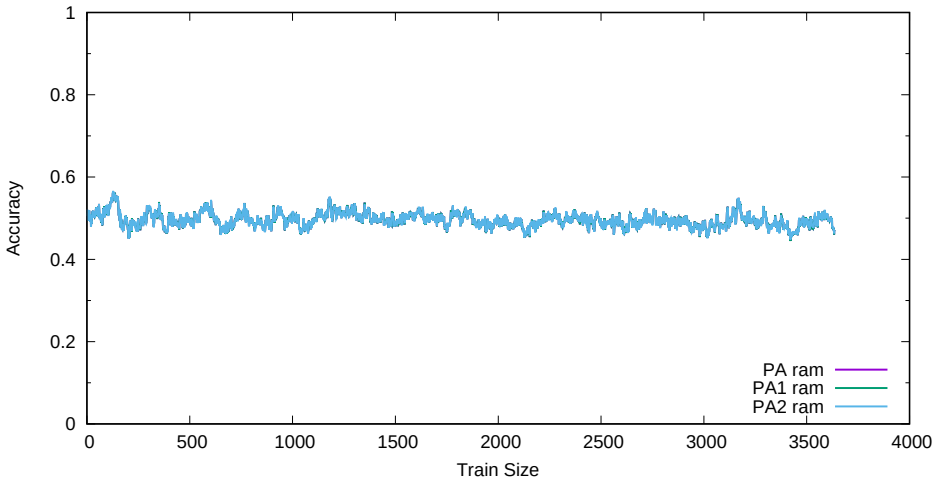

図 5.20　FordB ランダマイズ

表 5.8 FordB に対する各モデルの予測結果

モデル	TP	TN	FP	FN
PA 無加工	206	192	209	203
PA 昇順	199	200	201	210
PA 降順	221	200	201	188
PA ランダム	191	188	213	218
PA-I 無加工	206	192	209	203
PA-I 昇順	199	200	201	210
PA-I 降順	221	200	201	188
PA-I ランダム	191	188	213	218
PA-II 無加工	206	190	211	203
PA-II 昇順	198	199	202	211
PA-II 降順	223	200	201	186
PA-II ランダム	190	189	212	219

FreezerRegular_train では，時系列順に学習させた場合と，昇順でソートした場合の精度は 0.5 であった．また，全ての事例を正例と予測していた．降順でソートした場合，ほぼ全ての事例を負例と予測しており，精度は 0.5 よりもほんの少し高い程度であった．ランダマイズした場合，精度は 0.5 から 0.8 の間で激しく変動していた．

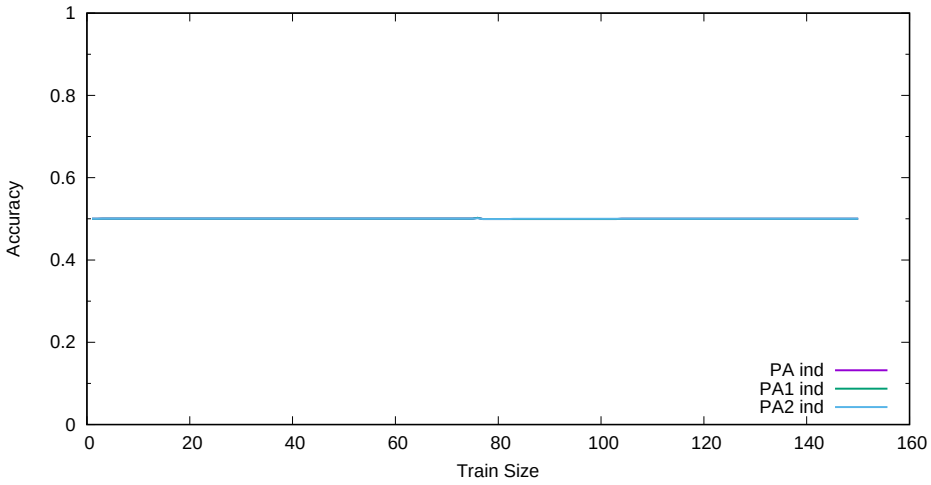

図 5.21 FreezerRegular_train 加工なし

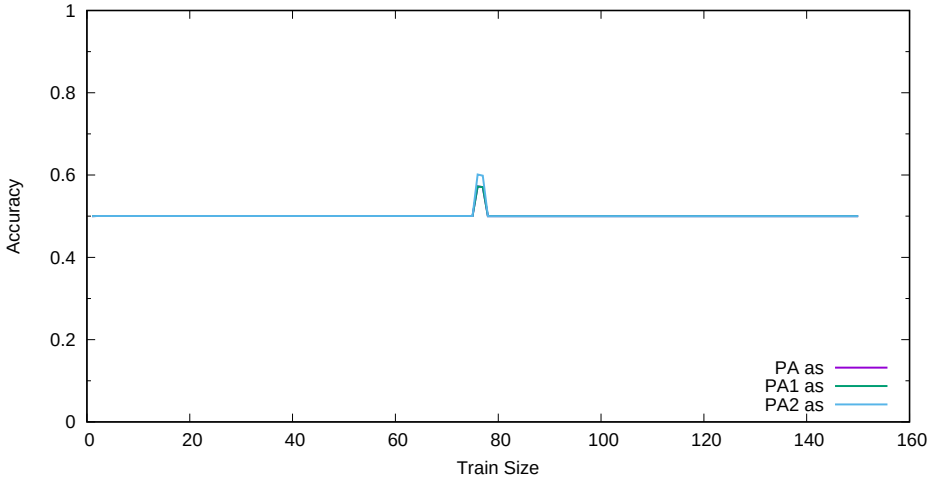

図 5.22 FreezerRegular_train 昇順ソート

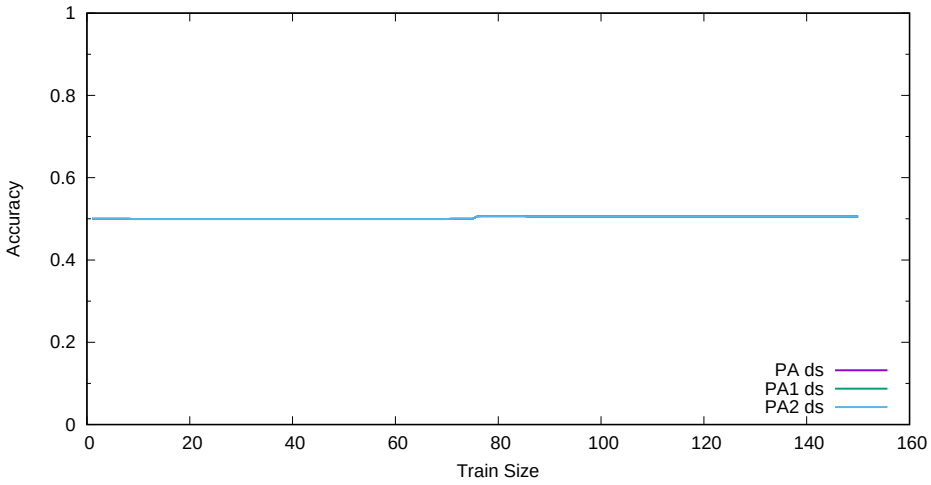

図 5.23 FreezerRegular_train 降順ソート

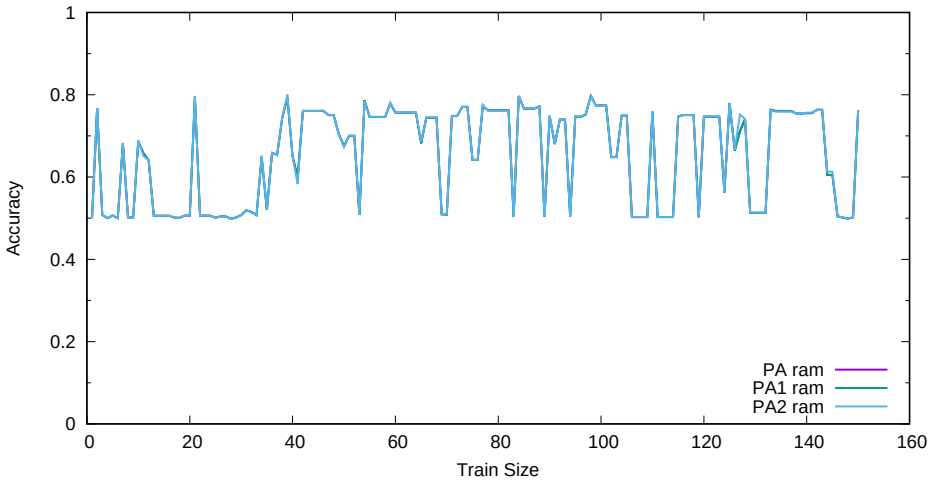

図 5.24 FreezerRegular_train ランダマイズ

表 5.9 FreezerRegular_train に対する各モデルの予測結果

モデル	TP	TN	FP	FN
PA 無加工	1425	0	1425	0
PA 昇順	1425	0	1425	0
PA 降順	16	1425	0	1409
PA ランダム	1192	979	446	233
PA-I 無加工	1425	0	1425	0
PA-I 昇順	1425	0	1425	0
PA-I 降順	16	1425	0	1409
PA-I ランダム	1192	979	446	233
PA-II 無加工	1425	0	1425	0
PA-II 昇順	1425	0	1425	0
PA-II 降順	16	1425	0	1409
PA-II ランダム	1192	977	448	233

Freezersmall_train では，時系列順に学習させた場合と，昇順でソートした場合の精度は 0.5 であった．また，1 つの事例以外を全て正例と予測していた．降順でソートした場合，ほぼ全ての事例を負例と予測しており，精度は 0.5 よりもほんの少し高い程度であった．ランダマイズした場合，精度は 0.3 から 0.8 の間で変動しているが，学習が終わった時点の精度は 0.76 であり，ある程度の精度を示している．

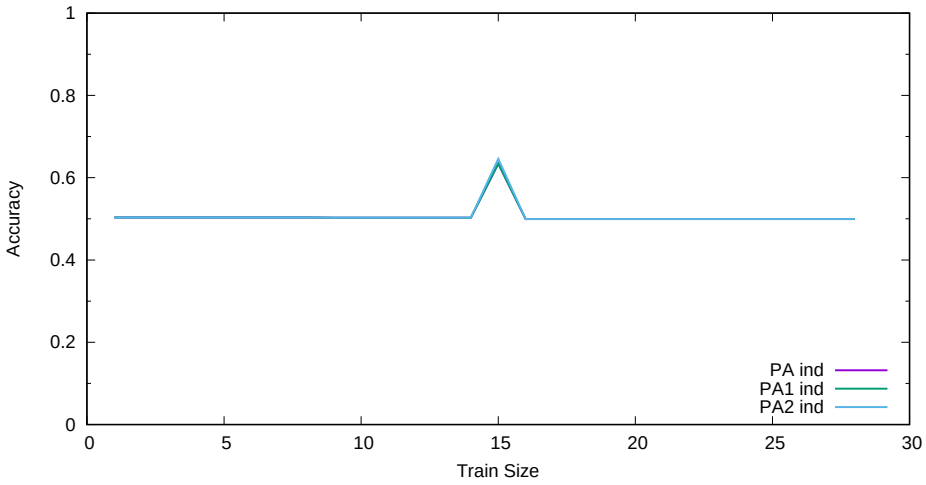

図 5.25　FreezerSmall_train 加工なし

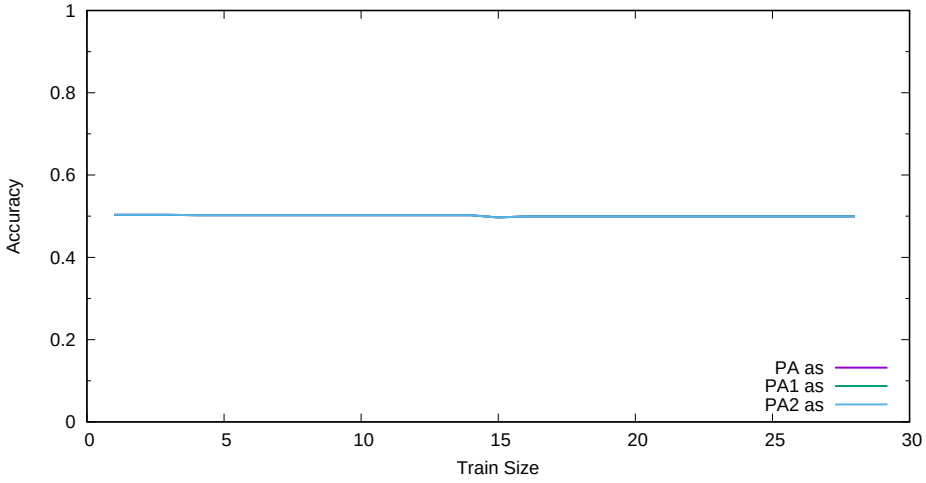

図 5.26　FreezerSmall_train 昇順ソート

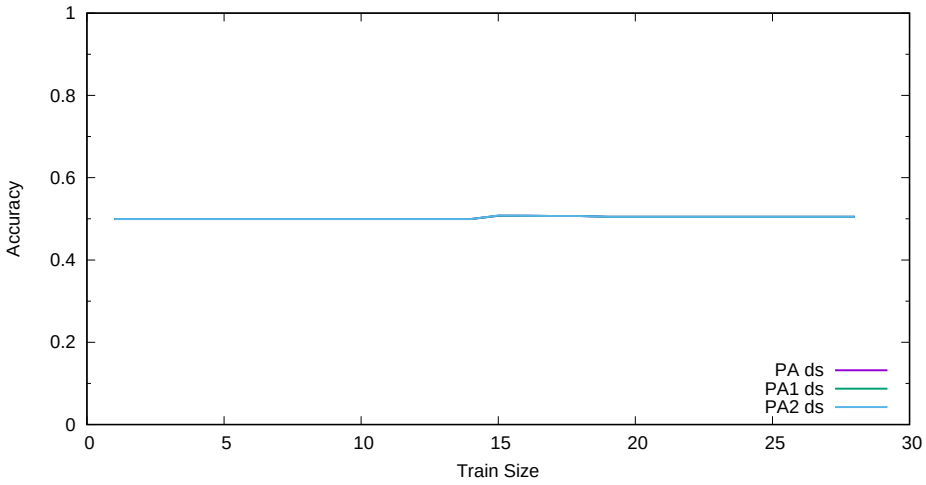

図 5.27　FreezerSmall_train 降順ソート

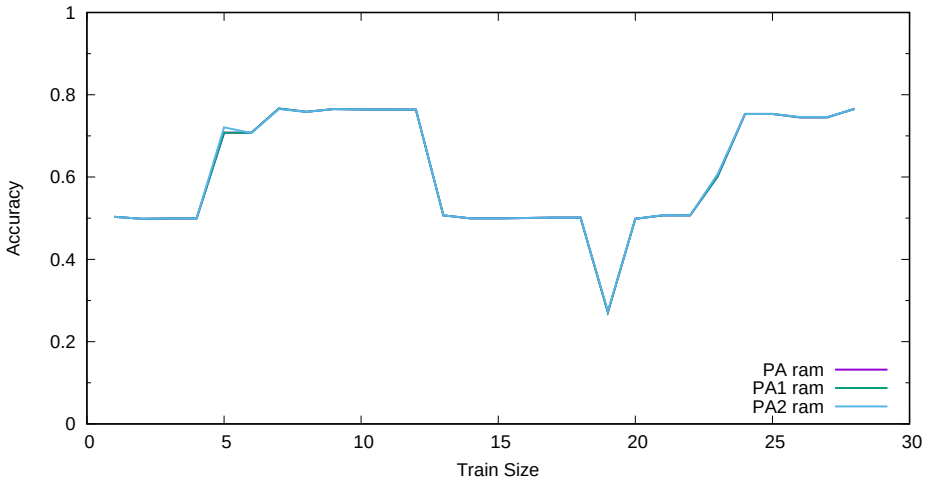

図 5.28　FreezerSmall_train ランダマイズ

表 5.10 FreezerSmall_train に対する各モデルの予測結果

モデル	TP	TN	FP	FN
PA 無加工	1424	0	1425	1
PA 昇順	1424	0	1425	1
PA 降順	14	1425	0	1411
PA ランダム	1046	1137	288	379
PA-I 無加工	1424	0	1425	1
PA-I 昇順	1424	0	1425	1
PA-I 降順	14	1425	0	1411
PA-I ランダム	1046	1137	288	379
PA-II 無加工	1424	0	1425	1
PA-II 昇順	1424	0	1425	1
PA-II 降順	14	1425	0	1411
PA-II ランダム	1044	1137	288	381

ItalyPowerDemand では，昇順でソートした場合と降順でソートした場合の精度は 0.5 程度であった．一方，時系列順に学習させた場合や，ランダマイズした場合は，学習数が増えるに従って 1 に向かって漸近している．

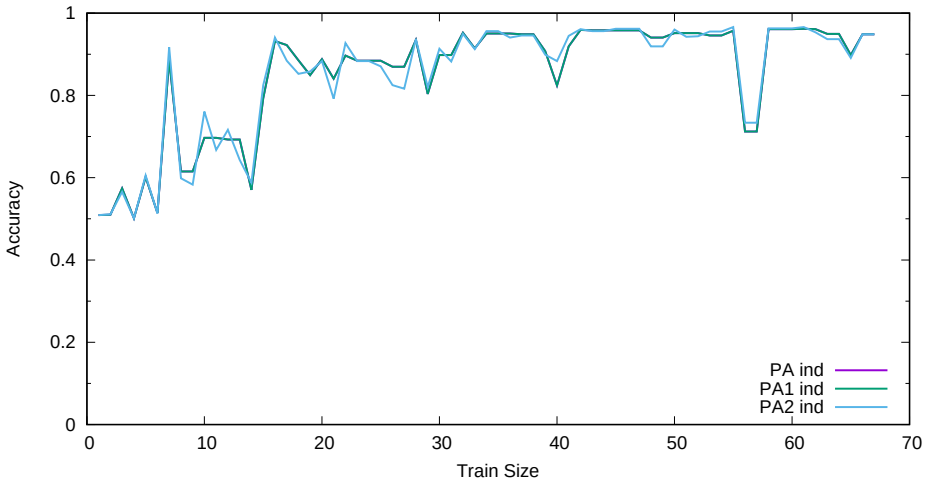

図 5.29 ItalyPowerDemand 加工なし

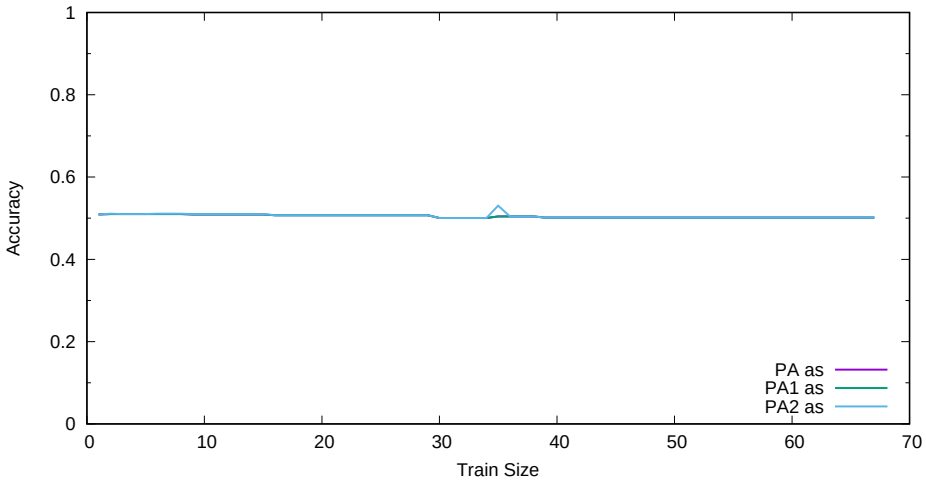

図 5.30 ItalyPowerDemand 昇順ソート

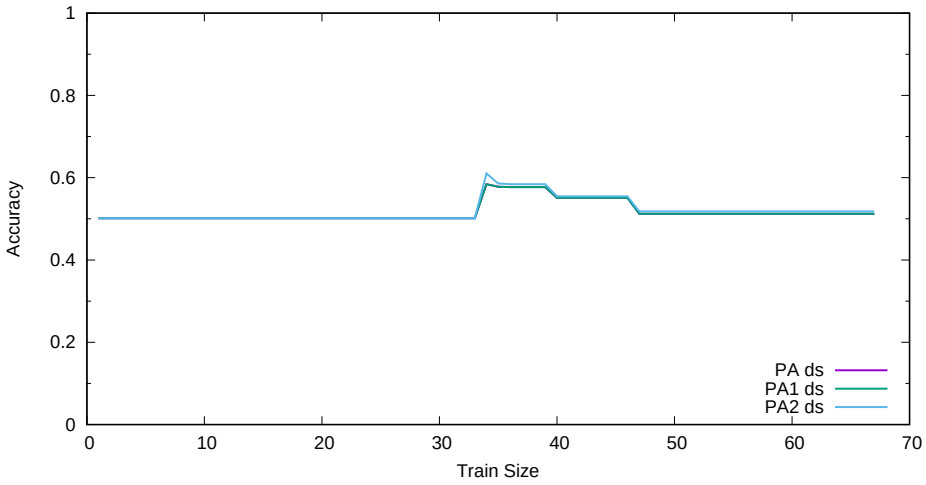

図 5.31 ItalyPowerDemand 降順ソート

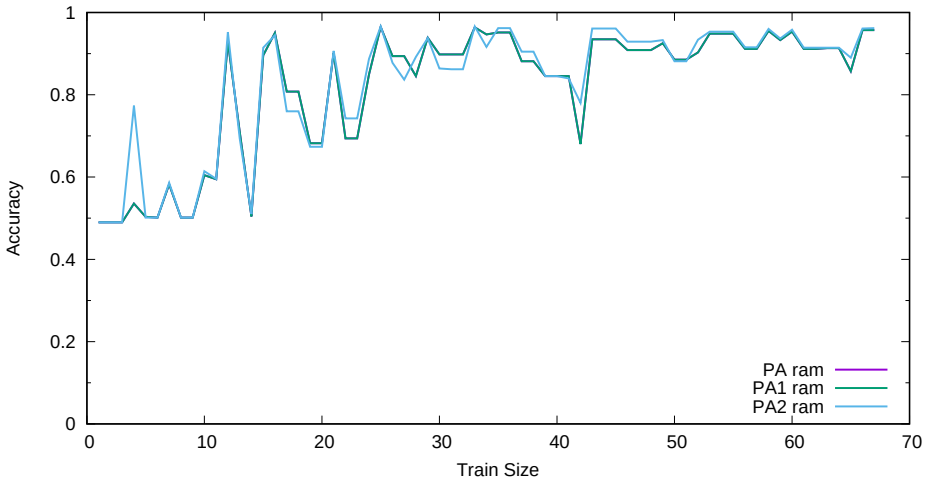

図 5.32 ItalyPowerDemand ランダマイズ

表 5.11 ItalyPowerDemand に対する各モデルの予測結果

モデル	TP	TN	FP	FN
PA 無加工	473	503	10	43
PA 昇順	516	0	513	0
PA 降順	14	513	0	502
PA ランダム	492	493	20	24
PA-I 無加工	473	503	10	43
PA-I 昇順	516	0	513	0
PA-I 降順	14	513	0	502
PA-I ランダム	492	493	20	24
PA-II 無加工	473	503	10	43
PA-II 昇順	516	0	513	0
PA-II 降順	20	513	0	496
PA-II ランダム	493	497	16	23

Wafer では，昇順でソートした場合のみ，100 個程度学習するまでの精度が 0.4 前後で，学習が終了したときの精度は 0.8 程度であった．それ以外では，精度が 0.8 前後を推移していた．

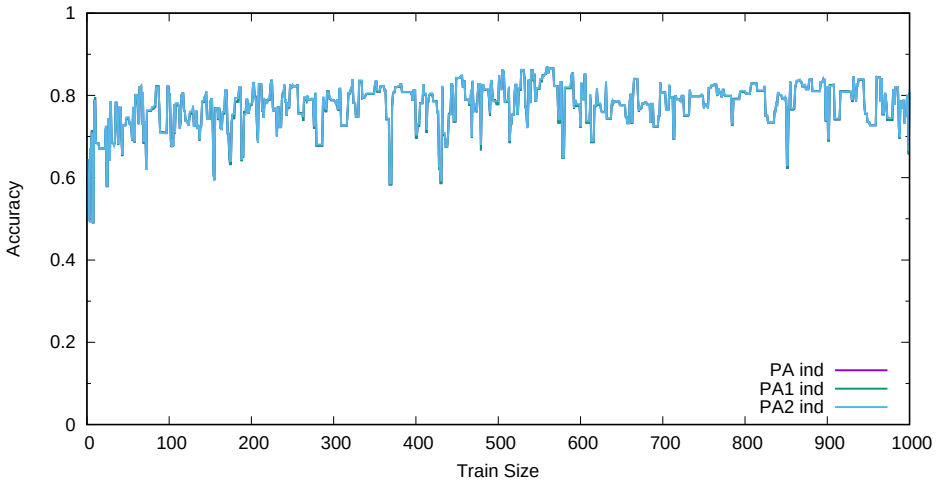

図 5.33 Wafer 加工なし

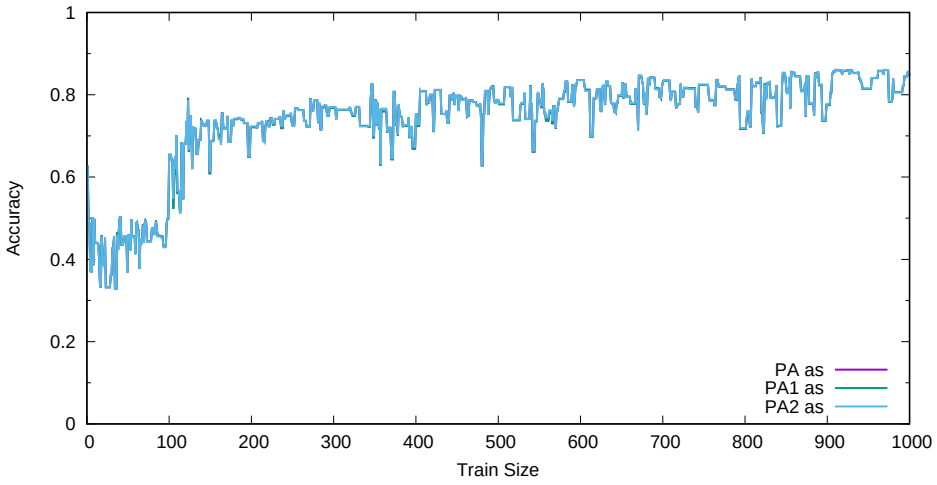

図 5.34 Wafer 昇順ソート

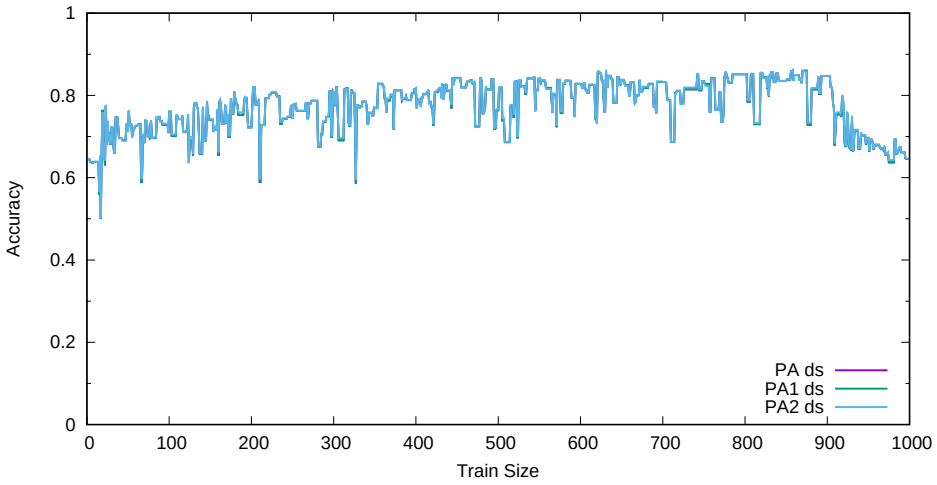

図 5.35　Wafer 降順ソート

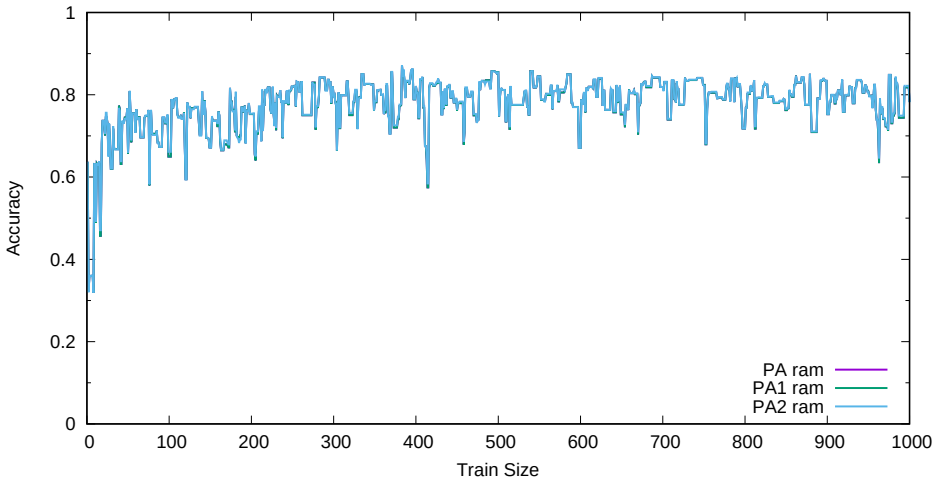

図 5.36　Wafer ランダマイズ

表 5.12 Wafer に対する各モデルの予測結果

モデル	TP	TN	FP	FN
PA 無加工	4534	446	219	965
PA 昇順	5040	173	492	459
PA 降順	3499	480	185	2000
PA ランダム	4422	402	263	1077
PA-I 無加工	4534	446	219	965
PA-I 昇順	5040	173	492	459
PA-I 降順	3499	480	185	2000
PA-I ランダム	4422	402	263	1077
PA-Ⅱ 無加工	4573	451	214	926
PA-Ⅱ 昇順	5036	174	491	463
PA-Ⅱ 降順	3509	478	187	1990
PA-Ⅱ ランダム	4417	400	265	1082

第 6 章

考察

実験の結果を受けての考察について述べる．

6.1 結果の考察

まず，静的なデータとして使用した Iris Flower Data Set の結果について考察する．データ数が少なく，線形分離可能なデータであったため，比較的早い段階で学習が終了したと考えられる．降順ソートをした場合，40 強のデータを学習するまでは正解率が 0.5 であったのは，初期の重みで正例を上手く分類ができたため，重みの更新が行われず，加えて初期の重みでは負例を上手く分類出来なかったからではないかと考えられる．

次に，Passive-Aggressive-I の正解率が安定するまでに時間が掛かった理由を考察する．Passive-Aggressive-I では，重みの更新幅が C を超える際には，C にクリップされてしまう．つまり，重みを大きく更新して，より早く学習を終えることが出来なかったため，このような結果になったのではないかと考えられる．

Iris Flower Data Set は線形分離可能で分類が容易であったため，正解率の変動があまり見られなかった．分類が容易な場合は，モデルがすぐに構築可能なため，学習するデータの順序は大きく影響を及ぼさないのかもしれない．

次に，上手く学習が行えなかった，Earthquakes，FordA，FordB についての考察を行う．Earthquakes は 512 時間分のデータを 1 インスタンスとして扱い，1 時間分のデータが 1 属性となっている．つまり，1 時間ずつのデータが 512 個の属性として並んでいる．同じ属性には，512 時間毎のデータになるため，このような属性に対して，重みを決定するのはほぼ不可能である．よって，学習を行っても正解率が 0.5 前後を推移していたのは，このような属性が含まれるデータセットでは，Passive-Aggressive を用いて学習するのが困難なため，うまくモデルを構築することができなかったからだと考えられる．同様の理由で，FordA，FordB に対して，Passive-Aggressive を用いて学習するのは困難だと考えられる．

FreezerRegular，FreezerSmall では，時系列順の場合，昇順でソートした場合，降順でソートした場合で正解率が 0.5 であった．訓練データが少ないため，時系列順の場合でもう

まくモデルが構築できなかった可能性が考えられる．実際，時系列順と昇順でソートした場合は，全ての事例を正例と判断しており，降順でソートした場合は，ほぼ全ての事例を負例と判断している．一方，ランダマイズしたデータで学習した場合は，偏った判定をするモデルにはなっておらず，ある程度の正解率を得られている．しかしながら，正解率の推移が激しく，偶然うまくモデルの構築ができたという可能性を否定することはできない．

ItalyPowerDemand では，昇順でソートした場合，降順でソートした場合においては，正解率はほぼ 0.5 であった．昇順の場合は全ての事例を正例，降順の場合はほぼ全ての事例を負例と判断している．一方，時系列順もしくはランダマイズした場合，学習数が増えるに連れて正解率は 1 に近づいている．このデータセットでも，偏ったデータで学習するとうまくモデルが構築できないことが示唆された．

Wafer では，昇順でソートした場合のみ，100 個程度学習するまでは正解率が 0.5 前後であった．それ以外では，始めから正解率は 7 割前後であった．Wafer は不均衡なデータであり，負例の数が正例に比べて少ないため，負例に合わせた学習を行うために正解率が下がるのだと考えられる．

次に，Passive-Aggressive，Passive-Aggressive-I，Passive-Aggressive-Ⅱ のそれぞれで同じ訓練データを用いた場合の結果について述べる．今回使用したデータセットにはノイズがほとんど含まれておらず，モデルを構築する際の重みの収束に差があったのみであった．実際，Passive-Aggressive-I では，重みが収束するまでの学習回数が Passive-Aggressive や Passive-Aggressive-Ⅱ に比べてより必要な場合も存在した．Passive-Aggressive-I や Passive-Aggressive-Ⅱ はノイズに対して機敏に反応してしまう弱点を考慮したアルゴリズムであるため，ノイズを含むデータを使用しないと差は顕著に出ないのだと考えられる．

今回は，静的なデータとして使用したデータは線形分離可能な小規模なデータであった．そのため，学習が容易であったため，顕著な差が出なかった可能性が考えられる．また，時系列データとして使用したデータセットでは，訓練データが著しく少ないデータや，Online Passive-Aggressive でモデルを構築するのには向いていないデータセットがあった．より，Online Passive-Aggressive に向いたデータや，時間が経過するとデータの性質が極端に変わるようなデータセットであれば，より良い結果を得られたかもしれない．また，今回はノイズがほぼ含まれていない理想的なデータセットを用いて実験を行っているため，Passive-Aggressive，Passive-Aggressive-I，Passive-Aggressive-Ⅱ のそれぞれの特徴を確認することができなかった．それぞれのアルゴリズムの特徴を確認するためには，ノイズが含まれているようなデータを用いて実験を行う必要がある．

6.2 提案手法に対する考察

本研究では，局所的に変化するという性質を持つ時系列データを用いて性能評価を行うという手法を提案した．実験では，局所的に変化するという性質を持たない静的なデータを用いた場合との比較を行ったが，静的なデータとして使用したデータセットが線形分離可能な

小規模なデータであったため，局所的に変化するという性質を持たないデータを使用して性能評価を行った場合の問題点を明らかにすることはできなかった．

しかし，学習データの順序が著しく偏っている場合，分類モデルがうまく構築されない場合があることが判明した．静的なデータを用いてデータストリームマイニングアルゴリズムの性能評価を行う場合，アルゴリズムにデータを与える順序を決めなければならない．静的なデータによっては，インデックス順で流し込むと偏った順序になる場合が考えられる．また，静的なデータに対して，ランダマイズを行う場合，偏った順序にならない保証はない．時系列データであれば，時系列自体がデータストリームとして与える順序を担うため，静的なデータを使用する場合に比べ，時系列データを使用する利点を示せたと考える．

第 7 章

結論と今後の課題

データストリームマイニングアルゴリズムの性能評価を行う場合，最も望ましいのはデータストリームを使用することである．しかし，データストリームを使用するのはコストが掛かるため，多くの場合は静的なデータが使用される．しかし，静的なデータには局所的に変化するという性質が無く，データに順序が存在しないため，データストリームマイニングアルゴリズムに与える順序をどのように決定するのかという問題が存在する．そこで本研究では，局所的に変化するという性質を持ち，データに順序が存在する時系列データを用いることで，これらの問題を解決することができると考えた．

提案手法を評価するために，データストリームマイニングアルゴリズムとして，オンラインアルゴリズムの Online Passive-Aggressive を対象に，静的なデータと時系列データを用いて実験を行った．実験の結果から学習データの順序が著しく偏っている場合，分類モデルがうまく構築されない場合があることが判明した．しかし，静的なデータとして使用したのは線形分離可能な小規模なデータセットであったため，局所的に変化するという性質を持たない静的なデータを使用する問題点を明らかにすることはできなかった．結論として，学習データの順序が著しく偏っている場合，正しく性能評価が出来ない場合があるため，静的なデータを使用する場合，アルゴリズムに与える順序を留意して決定する必要があることが判明した．時系列データであれば，時系列情報自体がアルゴリズムに与える順序を担うため，アルゴリズムに与える順序を考慮する必要がない．このことから，データストリームマイニングアルゴリズムを評価する上では，静的なデータを使用するよりも，時系列データを使用するほうが望ましいことが示唆された．

今後の課題としては，静的なデータとして，より規模の大きいデータセットや，線形分離不可能なデータセットを用いて実験を行い，静的なデータを使用する問題点を明らかにしていく必要がある．また，ノイズが含まれているようなデータで実験した場合，Passive-Aggressive の改良版である Passive-Aggressive-I や Passive-Aggressive-II が Passive-Aggressive に比べてノイズに強いことが確認できるかどうかを検証する．

謝辞

本研究を進めるにあたって，熱心にご指導頂いた新美礼彦准教授に深く感謝いたします．
また，研究について活発に議論しあった新美研の皆様にも感謝いたします．

参考文献

[1] Reinsel David, Gantz John, and Rydning John. The digitization of the world from edge to core, 2018. An IDC White Paper.

[2] 博紀有村, 拓也喜田. データストリームのためのマイニング技術. 情報処理, Vol. 46, No. 1, pp. 4–11, jan 2005.

[3] Arvind Arasu, Mitch Cherniack, Eduardo Galvez, David Maier, Anurag Maskey, Esther Ryvkina, Michael Stonebraker, and Richard Tibbetts. Linear road: A stream data management benchmark. 10 2004.

[4] 順平村田, 宏治岩沼, 龍一石原, 英知鍋島. F-043 精度保証付きオンライン型高速近似系列マイニング (人工知能・ゲーム, 一般論文). 情報科学技術フォーラム講演論文集, Vol. 8, No. 2, pp. 499–503, aug 2009.

[5] 龍一石原, 宏治岩沼, 英知鍋島. Lf-002 大規模データ系列中に頻出する部分系列のオンライン抽出アルゴリズム (f 分野:人工知能・ゲーム). 情報科学技術レターズ, No. 4, pp. 89–92, aug 2005.

[6] 直弥鳥谷部, 拓也喜田. データストリームに対する効率良い頻出アイテム発見アルゴリズム. DEIM Forum 2019 D4-1 6 ページ, 2019.

[7] 海野 裕也, 岡野原 大輔, 得居 誠也, 徳永 拓之. オンライン機械学習. 講談社, 東京, April 2015.

[8] Koby Crammer, Ofer Dekel, Joseph Keshet, Shai Shalev-Shwartz, and Yoram Singer. Online passive-aggressive algorithms. *Journal of Machine Learning Research*, Vol. 7, pp. 551–585, 03 2006.

[9] Anthony, Bagnall and Eamon, Keogh and Jason, Lines and Aaron, Bostrom and James, Large. The UEA & UCR Time Series Classification Repository. `https://timeseriesclassification.com/`. (Accessed on 2020/01/28).

[10] R. A. Fisher. The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, Vol. 7, No. 7, pp. 179–188, 1936.

[11] Robert Thomas Olszewski, Roy Maxion, and Dan Siewiorek. *Generalized Feature Extraction for Structural Pattern Recognition in Time-Series Data.* PhD thesis, USA, 2001. AAI3040489.