
Introduction
to Artificial
Intelligence
for Security
Professionals.
byThe Cylance Data Science Team

セキュリティ技術者のための人工知能入門

サイランスデータサイエンスチーム

発行者：
サイランスデータサイエンスチーム

セキュリティ技術者のための人工知能入門/サイランスデータサイエンスチーム。– 米国カリフォルニア州アーバイン：The Cylance Press、2017年

概要：セキュリティ技術者の方に、人工知能と機械学習の世界を説明と例を交えて紹介します。

ISBN13：978-0-9980169-0-0

1.人工知能. 2.国際セキュリティ.
I.題名.

TA347.A78 C95 2017
006.3—dc23　　2017943790

初版

プロジェクト管理：Jenkins Group, Inc.
www.BookPublishing.com

内部デザイン：Brooke Camfield

米国内にて印刷
21　20　19　18　17　•　5　4　3　2　1

目次

序文 v

はじめに
人工知能：情報セキュリティを高度化する手法 ix

1 クラスタリング
K平均法とDBSCANアルゴリズムの使用 1

2 分類
ロジスティック回帰とディシジョンツリーの使用 37

3 確率 77

4 深層学習 109

おわりに 149

序文

Stuart McClure

私が初めてコンピュータサイエンスというものに触れたのは、米国コロラド州ボルダーのコロラド大学でのことでした。私は、1987年から1991年までの4年間に、心理学、哲学、およびコンピュータサイエンスアプリケーションを学びました。コンピュータサイエンスの一環で、統計学を学び、さらに人間が望む処理をいかにしてコンピュータでプログラミングするかを習得しました。プログラミング言語でマシンを制御したときに非常に高揚し、夢中になったことは今でも忘れません。

このコンピュータサイエンスの講義では、Alan Turingと典型的な「チューリングテスト」を学びました。このテストはいたって単純で、2人の「人物」(一方はコンピュータ)に、一連の文書による質問を投げかけ、それらの質問に対する回答を使用して判定します。コンピュータと人間を区別できない場合、そのコンピュータはテストに「合格」したことになります。このコンセプトにはとても興味を持ちました。コンピュータが、回答、動作、考えなどで人間とまったく同じように自然に振る舞えるでしょうか。私はいつでも「できない理由がない」と確信していました。

そして時が経ち、私が有名なアンチウイルスの企業に入社してから2年後の2010年、私は将来の開発計画と展望について説明する立場にありました。残念ながら、20年間にわたってこの手の会話ではいつも「マルウェアおよびサイバー攻撃の検知のスピードをより速くしていく必要があります」と同じことを述べていました。その会社でも、より速くと言い続けていました。このため、シグネチャと呼ばれる定義ファイルは月ごとに更新していたものを週ごとに更新

するように努めました。さらに、シグネチャの更新を週ごとから毎日行うように画策していました。しかし、数百万ドルのお金をより高速に送金できるような時代になっても、検知を速くするような方法はどこにもないことに気づきました。悪意のある攻撃は、常にその上を行く速さでした。では、悪意のある攻撃を飛び越えて先回りできればどうなるでしょうか。攻撃者が行動する前に、実際に彼らが何をするかを予言できればどうなるでしょうか。

2004年以降、「あなたのコンピュータではどんなアンチウィルスソフトを使っているんですか?」という質問をかなり頻繁に受けてきました。2000年からの10年間はほとんどとあるアンチウイルス企業で上級取締役として仕事をしていたため、人々は「もちろん、自社の製品を使用しています」という答えを期待していたと思います。でも、私は嘘をつけませんでした。私は、自社の製品を使用していなかったのです。それは、セキュリティ製品自体を信用していなかったためです。私は保守的なため、物事の善悪は自分自身の判断しか信じていませんでした。このため、私がセキュリティ会社を辞めるとき、「なぜ、自分が考えるようにコンピュータをトレーニングできないのだろう。善悪が判定できるセキュリティ担当者とまったく同じように」と自問していました。「以前のようにシグネチャの作成を人間に頼るのではなく、過去から適切に学んで、シグネチャを不要にし、最終的にはリアルタイムで攻撃かどうかを予測して防止することはできないだろうか」。

そして、サイランスが誕生しました。

私と、チーフサイエンティストのRyan Permehは、この途方もなく骨の折れる旅に出発しました。それは、あらゆる有意義な手法を採用してもことごとく失敗し続けてきた分野に、数学と科学を用いるという誰もやらなかったアプローチです。そして、優秀なサイランスのデータサイエンスチームによって、ついに目標を達成することができました。すなわち、人工知能を使用してサイバー攻撃を予測し予防することで世界中のすべてのコンピュータ、ユーザー、および関連する設備を保護できるようになったのです。

長年、人工知能と機械学習については多くの本が書かれてきまし

たが、純粋なサイバーセキュリティの観点で、本当に役立つ実用的なガイドはほとんどありませんでした。本書に記載されているサイランスデータサイエンスチームからの提案は、現実的かつ実用的で、サーバーセキュリティに携わる人たちならばだれでも適用できるものです。これにより、日々苦慮しているハッカーがもたらす問題にも機械学習を適用できます。

それでは、あなたもサイバーセキュリティの旅に出発してください。そして過去や常識を信じずに、自分を信じて、自らテストすることをいつまでも忘れないでください。

はじめに
人工知能：情報セキュリティを高度化する手法

人工知能（AI）技術は、瞬く間に学術界やスペキュレイティブフィクションの領域を超えて、産業界の主流になりつつあります。革新的な製品の中でも特にAppleのSiri®デジタルアシスタントやGoogle検索などはAIを利用しており、オンラインでの情報のアクセス方法や活用方法を変えるものです。米国大統領府は、2016年12月に次のように報じています。

> *人工知能（AI）技術と関連分野の進歩は、保健、教育、エネルギー、経済包摂、社会福祉、環境などの重要な分野において、新しい市場と新しい機会を開拓してきました。*[1]

また、AIは、国家の防衛や、他国が支援するサイバー攻撃から金融、エネルギー、情報、通信などの重要なインフラを防御するためにも戦略的に重要な位置付けにあります。米国政府の国家科学技術委員会の技術評議会（NSTCC）が発行した2016年10月のレポート[2]では次のように報じられています。

---

1. 米国大統領行政府、『Artificial Intelligence, Automation, and the Economy（人工知能、自動化、および経済）』、2016年12月20日発行。https://www.whitehouse.gov/sites/whitehouse.gov/files/images/EMBARGOED%20AI%20Economy%20Report.pdfからダウンロードできます。
2. 米国国家科学技術委員会の機械学習と人工知能に関する分科会、『Preparing for the Future of Artificial Intelligence（人工知能の将来のための準備）』、2016年10月発行。https://obamawhitehouse.archives.gov/ sites/default/files/whitehouse_files/microsites/ostp/NSTC/preparing_for_the_ future_of_ai.pdfからダウンロードできます。

サイバーセキュリティは、AIの重要な応用分野であり、サイバー対策の攻撃と防御の両方で重要な役割を担うことが期待されています。(中略)AIを利用することで、急速に拡大する脅威を検知および対処するために必要な、迅速な対応を維持できます。

これらの予測に基づいてNSTCCは、『National Artificial Intelligence Research and Development Strategic Plan(米国人工知能研究開発戦略計画)』[3]を発行し、米国政府が資金提供する研究開発を紹介しています。

過去の重要な新技術がそうであったように、AIは業界の専門家や有力なメディアの間で、興奮と不安の両方を引き起こしています。私たちは、コンピュータがチェスや碁のチャンピオンを負かしたことや、自動運転が近い将来に実現可能な技術であること、さらに一部の倫理学者が、機械が人間に取って代わり、人間の能力が陳腐化することなどの記事を目にしたことがあります。これらの報道の一部は誇張しすぎであり、AIは私達の生活に有意義な役割を果たすものと考えます。AIの研究開発は適切な倫理的原則に従って行われ、今日および将来にわたって構築されるシステムには、完全なる透明性があり、人への報告責任をまっとうするものになるはずです。

短期的には、セキュリティの専門家がAIについての実践的な知識を得ることは非常に重要と考えます。AIとは何であるか、何を実現できるのか、さらになぜセキュリティの専門家の業務や、実際のセキュリティ問題への対応方法において重要な位置を占めるようになったかなどです。この考えが、『セキュリティ技術者のための人工知能入門』を書くきっかけとなりました。

本書で説明しているクラスタリング、分類、確率的モデリングなどの手法、さらに生成モデルや強化学習などの手法の詳細は、さまざまなWebサイトで参照できます。また、技術面に注目している

3. 米国国家科学技術委員会の機械学習と人工知能に関する分科会、『National Artificial Intelligence Research and Development Strategic Plan(米国人工知能研究開発戦略計画)』、2016年10月発行。https://www.nitrd.gov/PUBS/national_ai_rd_strategic_plan.pdfからダウンロード可能。

読者は、これらの手法の基礎となる数学的原理や演算方法についても学習することをお勧めします。本書は、AI分野の初心者に適した入門書となるように、これらの題材については意図的に省略しています。推奨される関連資料については、https://www.cylance.com/intro-to-aiのサイトを参照してください。

本書が、継続的な自己学習を始めるきっかけとなり、セキュリティ技術者としてスキルを向上し、キャリアアップの助けとなり、現在および将来の担当業務の効率化につながることを心から願っています。

AI:認識と現実

AIの分野には、実際には次の3種類の研究分野があります。

- **超人工知能（ASI）**：スペキュレイティブフィクションやマトリクスなどの映画でよく知られている分野です。ASI研究の目標は、事実上すべての能力について人間を超えるコンピュータを制作することであり、作家兼アナリストのWilliam Bryk氏が「完全な記憶と無制限の分析能力」と呼ぶ能力を備えているものです。[4]
- **汎用人工知能（AGI）**：人間と同等の知性を持ち、学習と推論を必要とする幅広い課題について人間と同等の解決能力を持つ機械を指します。AGIの古典的なテスト方法の1つに、「チューリングテスト」[5]と呼ばれるものがあります。このテストでは、検査官が文字による会話を読みますが、その会話は離れて見えないところにいる人と機械との間で交わされたものです。テストは、AGIシステムからの会話と人間側からの会話が検査官によって区別できない場合に合格となります。

4. William Bryk著、『Artificial Intelligence: The Coming Revolution（人工知能：到来する革命）』、Harvard Science Review、2015年秋発行。https://harvardsciencereviewからダウンロード可能。files.wordpress.com/2015/12/hsrfall15invadersanddefenders.pdf

5. A.M.チューリング著、『Computing Machinery and Intelligence（計算機と知能）』、1950年発行、Mind社、59, 433-460。http://www.loebner.net/Prizef/TuringArticle.htmlからダウンロード可能。

AGIの実現は数十年で可能ということにほとんどの専門家が同意しますが、ASIについてはまった実現できないという意見の専門家も少なくありません。NSTCの2016年10月のレポート[6]は、「機械が今後20年で、人間と同等かそれ以上の広範囲に適用可能な知性を持つことは非常に難しい」と報じています。

- **特化型人工知能（ANI）**：コンピュータの優れた性能を利用して、大量のデータを処理して、人間では見つけることが難しいパターンや関係を検知するものです。このデータ中心のシステムは、チェスの対局や、ネットワークトラフィックでの異常性の検知など、特定の作業についてのみ人間よりも高い能力を発揮します。ネットワーク脅威の分析者や、フォレンジックチームなど、より詳細な分析をする場合に利用価値があります。これからのページで重点的に説明するのは、この種類の手法です。

人工知能の分野には、学習、推論、有意義な見識の抽出など、人が持つ能力をコンピュータで実現することを意図したさまざまな技術があります。近年、有益な研究成果や技術的進歩のほとんどが、AIの分野の1つである、機械学習（ML）から得られています。この分野では、データにアルゴリズムを適用することで、機械を学習させることに重点が置かれています。多くの場合、AIとMLという用語は置き換えて使うことができます。これから本書で注目する手法は、機械学習の範疇に入るもののみです。

AIのすべて問題が、機械学習により解決できるとは限りません。機械学習が扱う問題は、データによって解決できるものでなければなりません。十分な量の関連性の高いデータが存在し、入手できる必要があります。さらに、妥当な時間枠内に必要な処理を実行できる、十分な計算能力を備えたシステムが必要です。我々が興味

6. 米国国家科学技術委員会の機械学習と人工知能に関する分科会、『Preparing for the Future of Artificial Intelligence（人工知能の将来のための準備）』、2016年10月発行。https://obamawhitehouse.archives.gov/ sites/default/files/whitehouse_files/microsites/ostp/NSTC/preparing_for_the_ future_of_ai.pdfからダウンロードできます。

を持つ多くのセキュリティの問題が、この要件をとても良く満たしていることが分かるでしょう。

セキュリティ分野における機械学習

組織では、生産性の最大化という明確な目標を追求するために、システム、情報、ネットワーク、および人的なリソースに投資します。したがって、可能性のある攻撃の要素を単純に潰していくだけでは実用的ではなく好ましい方法ともいえません。また、資産の価値や特徴のみに焦点を当てて防御しようとしても侵入を防ぐことはできません。これらの資産がアクセスされる、または利用される状況を考慮する必要があります。たとえば、Webサイトの攻撃について問題とすべきは通信のコンテキストであり、Webサイト上の資産や機能だけではありません。

セキュリティの分野では、コンテキストが非常に重要です。幸いにも、セキュリティの分野では、ログ、ネットワークセンサー、およびエンドポイントエージェントから大量のデータが生成されます。また、分散型ディレクトリや人事管理システムからも、どのユーザーの活動が許可され、どれが許可されないかを示す情報が提供されます。一般的には、これらの大量のデータから脅威を特定し、改善に必要な状況の手がかりを抽出することは可能です。ただし、これはデータを分析できるツールがある場合に限られます。これは、まさにML（機械学習）が得意とする処理です。

MLシステムに、管理下の資産に関係する活動についての幅広い情報を与えることにより、アナリストは、時間軸や、異なるホスト、ユーザー、およびネットワークにわたって広範囲に分散しているイベントがどのように相関しているかを把握できます。MLを適切に用いることで、侵入のリスクを低減するために必要となる状況を知ることができ、さらに「攻撃のコスト」を大幅に増やすことができます。

機械学習の機能

セキュリティ業界全体でMLの導入が大幅に増加しているため、

攻撃者が超えなければならないハードルはすでに高く引き上げられています。システムに侵入することは、ここ2、3年でもかなり難しくなっています。これを受けて、攻撃者も新しい侵入経路を見つけるためにMLの技法を取り入れる可能性があります。同様に、セキュリティの専門家は、ネットワークおよび情報資産を守るために、MLを防御にも利用する必要があります。

サイランスは、プロの囲碁棋士Lee Sedol氏とAlphaGoの間で行われた2016年3月の対局からあるヒントを得ることができました。Lee Sedol氏は18回世界タイトルを獲得した囲碁のチャンピオンであり、AlphaGoはDeepMind社で開発されたコンピュータプログラムです。DeepMind社は、ロンドンに拠点を置くAIラボですが、その後Google社によって買収されました。第2戦では、AlphaGoが今までだれも見たことのない手筋を見せました。試合を観察していた解説者や専門家たちは度肝を抜かれていました。Lee Sedol氏自身も呆然として、次の手を出すまで約15分もかかりました。AlphaGoは、おそらく最高位の5つのタイトル戦に勝つ能力があるでしょう。

セキュリティでの攻撃と防御のやり方は、多くの点で囲碁やチェスのような複雑なゲームでの猛攻とそれをかわすテクニックに似ています。MLの利用により、予想しなかったような複合型の完全に新しい脅威が出現することは間違いないでしょう。これから10年くらい先には、「バトルボッド」によってほぼリアルタイムでネットワークの攻撃と防御が行われるようになるでしょう。防御側でMLが必要になるのは、同等のレベルを維持するためにすぎません。

もちろん、どのような技術でも、労力とリソースを費やせば破られることがあります。ただし、MLベースの防御は、今までの方法よりもはるかに広範囲の脅威の領域に対応するため、突破することが非常に困難です。また、人間の能力である過ちから学ぶ能力を備えているもの大きな強みです。

AIの利用価値

エンタープライズシステムでは、新しいユーザーやビジネスの職務に対応するために、常時更新、修正、および拡張が行われていま

す。このような流動的な環境では、雑音を除去し、異常やその他のフォレンジック値を提供する指標に注目するML対応の「エージェント」があれば役に立ちます。MLは生産性を向上する万能のツールであり、セキュリティの専門家は、方針の立案や対策の実施に集中できます。長い時間をアプリケーションからのログやイベントデータの解析、エンドポイントの管理、および境界防御に費やす必要はありません。MLによって、業務を今までよりも効率的に行うことが可能になります。
新規および既存のセキュリティ製品にMLの機能を組み込むトレンドは、今後も継続することでしょう。2016年4月のGartner社のレポート[7]では、次のように報じられています。

- 2018年までに、検知に使用されるセキュリティ製品の25%に、何らかの形で機械学習機能が組み込まれる。
- 2018年までに、現在ゼロパーセントである、インシデントに自動的に対応する規範的分析が、UEBA製品に10%以上導入される。

これらの製品を適切に導入して管理するには、MLコンポーネントを効果的に活用し、能力を最大限引き出すために、動作原理を理解する必要があり、MLシステムは、全能でもなければ、常に完璧な結果を返すわけでもありません。最善のソリューションは、機械学習システムおよび人間のオペレータの両方を導入することです。したがって、今後3、4年の間に、MLに関する深い知識と能力が、採用条件に入れられるようになるでしょう。

本書について

本書は次の4つの章で構成されています。

1．第1章：クラスタリング　クラスタリングには、サンプルを

7. Gartner Core Security社、『The Fast-Evolving State of Security Analytics, April, 2016（急速に発達するセキュリティ分析、2016年4月）』、レポートID: G00298030。https://hs.coresecurity.com/gartner-reprint-2017よりアクセス可能。

サブグループやクラスタに分割するためのさまざまな技法が含まれます。分割は、主要な特徴量や属性の類似性に基づいて行われます。クラスタリングは、データ探索やフォレンジック分析に特に有効です。これはクラスタリングに、詳細調査が必要な異常値を大量のデータからふるい分ける能力があるためです。この章では、次の点について解説します。

- k平均法とDBSCANクラスタリングアルゴリズムで実行される計算手順。
- 通常の状態でアナリストが実施する必要があるクラスタリングの手順。この手順には、データの選択とサンプリング、特徴の抽出、特徴のエンコードとベクトル化、モデルの計算とグラフ化、モデルの検証とテストなどがあります。
- 正規化、ハイパーパラメーター、および特徴空間の基本概念。
- 連続型とカテゴリ型の両方のデータを組み込む方法。
- 最後は実習のセクションで、パナマ文書関連のデータ侵害と同様の弱点を特定するために、どのようにk平均法とDBSCANモデルを適用できるかを紹介します。パナマ文書問題は、2015年に発覚し、1,150万通の機密文書と、2.6テラバイトの顧客データがパナマ在籍の法律事務所Mossack Fonsecaから流出した問題です。

2．**第2章：分類**　分類では、いくつかの計算手法を使用して、事前定義されたクラスにサンプルが属するかどうかの可能性(尤度)を予測します。たとえば、指定された電子メールがスパムかどうかなどです。この章では、次の点について考察します。

- サンプルをクラスに割り当てるロジスティック回帰アルゴリズムとCARTディシジョンツリーアルゴリズムの計算手順。
- 教師ありと教師なしの学習方法の違い。
- 線形および非線形分類子の違い。

- 通常の教師あり学習の手順には、モデルのトレーニング、検証、テスト、および導入の4つのフェーズがあります。
- ロジスティック回帰：回帰の重み、正則化、ペナルティパラメータ、決定境界、データのフィッティングなどの基本概念。
- ディシジョンツリー：ノードの種類、分割変数、利益スコア、および停止基準に関する基本概念。
- 混同行列や測定値（精度、再現率など）を使用して、2のアルゴリズムから作成されるモデルの精度を評価および検証する方法。
- 最後は実習のセクションで、現在未調査のボットネットのコマンドアンドコントールシステムを検知するために、どのようにロジスティック回帰モデルとディシジョンツリーモデルを適用できるかを紹介します。

3．**第3章：確率**　この章では、サンプルを分類およびクラスタリングする際の予測モデリングの技法である確率について解説します。次のトピックについて説明します。

- ナイーブベイズ（NB）分類子と、ガウス混合モデル（GMM）クラスタリングアルゴリズムの計算手順。
- 試験、結果、イベントなどの基本的な概念、および結合確率と条件付き確率の違い。
- NB：分類の問題を解決する際に、事後確率、クラスの事前確率、予測の事前確率、および尤度が果たす役割。
- GMM：正規分布の特徴、およびそれぞれの分布を平均値と分散パラメータによって一意に識別する方法。また、GMMでは、サンプルをクラスに割り当てるために、どのように2段階の期待値最大化の最適化手法が使用されているかを解説します。
- 最後は実習のセクションで、SMSテキストによって送信されるスパムメッセージを検知するために、NBとGMMモデルをどのように適用できるかを紹介します。

4．**第4章：深層学習**　深層学習という用語は、ニューラルネットワークを中核として使用する幅広い学習方法を指します。ニューラルネットワークはアルゴリズムのクラスであり、このように名付けられているのは、密に相互接続されたニューロンが脳内で相互作用する方法をシミュレートしているためです。この章では、2種類のニューラルネットワークを使って分類の問題を解決するためにどのように適用できるかを解説します。ここでは、次の内容について説明します。

- 長●短期記憶(LSTM)型と畳み込み(CNN)型ニューラルネットワークの計算手順。
- ノード、非表示レイヤー、非表示状態、活性化関数、状況、学習速度、ドロップアウト正則化、抽象レベルを高める方法などの基本概念。
- フィードフォワードおよび再帰型ニューラルネットワークのアーキテクチャの違いと、全結合層および部分的結合層を導入する場合の有意性の違い。
- 最後の実習のセクションでは、テキストのサンプルを難読化するXORキーの長さを決定するために、どのようにLSTMとCNNモデルを適用できるかを紹介します。

実際の経験に勝るものはありません。したがって、実習セクションで示したすべてのスクリプトとデータセットは次の場所からダウンロードできるよう準備されています。

https://www.cylance.com/intro-to-ai

単純化のために、これらのスクリプトはすべて、実用可能な設定によってハードコーディングされています。ただし、これらのスクリプトを変更したり、新しいものを作成したりして実験することで、いかにこれらの方法が柔軟で汎用性が高いかを理解できます。

そして最も重要なのは、職場で最もよく発生するセキュリティの問題に対応するために機械学習をどのように導入できるかを実際に検討することであり、これを行うことを強くお勧めします。

1

クラスタリング K平均法とDBSCANアルゴリズムの使用

クラスタ分析の目的は、主要な特徴量や属性の類似性に基づいて、データを個別のグループやクラスタに分割することです。特定のクラスタ内のデータ項目どうしは、別のクラスタ内のデータ項目よりも類似しています。クラスタは、統計、人工知能、および機械学習のさまざまな技法を使用して作成できます。さらに、データの性質やアナリストの目標に応じて、特定のアルゴリズムを適用するように決定できます。

クラスタ分析は約85年前に社会科学で最初に使用されて以来、データの探索や有効な見識を抽出するための手法として安定性があり、幅広く適用できることが実証されてきました。たとえば、各種の小売業では、顧客を購買行動が類似するグループに分類するためにクラスタ分析が多く使用されており、大規模なデータウェアハウスに保存された数テラバイトの取引記録が分析されています。小売業者は、出力された顧客のセグメンテーションモデルを使用して、顧客ごとに高いグレードの商品の提案(アップセル)と関連製品の提案(クロスセル)を作成します。これらは、とても高い確率で受け入

れられます。また、クラスタリングは、パターン認識、研究データの分析、ドキュメントの分類などさまざまなタスクで、他の分析技法と組み合わせて使用されることがしばしばあります。サイランスではマルウェアが実行可能になる前に検知およびブロックするために使用しています。

ネットワークセキュリティの分野では、通常、クラスタ分析で、適切に定義されたデータの準備および分析の一連の作業が実施されます。本章の最後には、サイランスWebサイトへのリンクが掲載されており、以下の手順を自分で実際に行うために必要なデータや手順にアクセスできます。

手順1：データの選択とサンプリング

どのような機械学習の手法を始めるときにも、あらかじめ何らかのデータが必要になります。一般に、すべてのネットワーク処理とシステムデータを分析対象にして、結果がネットワークとコンピューティング環境を正確に反映することが理想的であると考えられます。ただし多くの場合、これは現実的ではなく、実用的でもありません。データ量は膨大であり、異種のシステムやデータソースに分散したデータを収集および統合することは容易ではありません。したがって、通常は統計的なサンプリングの技法を用います。これによって、管理しやすい分析用のデータのサブセットを作成できます。このサンプルは、データセット全体の特徴をできるだけ厳密に反映している必要があり、そうでなければ結果の正確性が損なわれる可能性があります。たとえば、異なる10台のコンピュータにおけるインターネットの閲覧を分析するように決定した場合、サンプルには10台すべてのシステムの代表的なログエントリを含める必要があります。

手順2：特徴量の抽出

この段階では、サンプル内のどのデータ要素を抽出して、分析の対象とするかを決定します。機械学習では、このデータ要素を「特徴量(属性)」と呼びます。特徴量とは、有意義な見識を生成するた

めに分析する必要があるデータの特性や性質を表します。

たとえば、顔認識分析に関係する特徴量には、目、鼻、口の輪郭、大きさ、形状などがあるでしょう。セキュリティの領域では、関連性のある特徴量として、開いているポート、閉じているポート、またはフィルタリングされたポートの割合、これらの各ポートで実行されているアプリケーション、さらにアプリケーションのバージョン番号などが候補になります。データ流出の可能性を調査している場合は、帯域幅の使用率やログイン時間などの特徴量を含めるべきです。

通常は、数千の特徴量から選択します。ただし、特徴量を追加するごとにプロセッサの負荷が増加し、分析を完了するまでの時間が長くなります。したがって、過去にデータを解読した経験やこの分野の全般的な専門知識に基づいて、無関係であることがわかっているものを除外し、必要な特徴量だけを含めることをお勧めします。また、統計的な手段を使用して、自動的に無用な特徴や重要ではない特徴量を除くこともできます。

手順3：特徴量のエンコードとベクトル化

ほとんどの機械学習アルゴリズムでは、データをエンコードしたり、何らかの数学的手法で表現する必要があります。データをエンコードするための一般的な方法の1つに、サンプルとその特徴量の集合を行と列のグリッドにマッピングする方法があります。この方法で構造化された各サンプルは「ベクトル」と呼ばれます。さらに、行と列の集まり全体は「行列」と呼ばれます。エンコード処理は、各特徴量を表すデータが、連続型、カテゴリ型、またはその他のタイプのいずれであるかによって使い分けます。

連続型のデータには、値の範囲内にある無限の数の値から1つを取ることができます。たとえば、CPU使用率には0から100パーセントの範囲を指定できます。したがって、あるサーバーの1時間あたりの平均CPU使用率は、下に示すような簡単なベクトルの集合で表すことができます。

サンプル(時間)	CPU使用率(%)
午前2時	12
午前9時	76
午後1時	82
午後6時	20

連続データと違って、一般にカテゴリデータは、より狭い範囲にある有効な値の小集合によって表されます。この例としては、ソフトウェア名とリリース番号の2つがあります。カテゴリデータは、その性質上グループを定義する場合に役立ちます。たとえば、カテゴリの特徴量としてオペレーティングシステムやバージョン番号を使用して、特徴量が類似するシステムのグループを指定できます。

このようなカテゴリは、数学的分析を行う前に数値へとエンコードする必要があります。このエンコード方法の1つに、有効なデータ値をすべて格納するために、各ベクトル内にスペースを挿入する方法があります。これにより、各データをカテゴリにマッピングし、さらに値が存在するかどうかを示すフラグをスペース内に付加できます。たとえば、3台のサーバーがあり、Linuxの3種類のバージョンのいずれかが実行されているとします。この場合、オペレーティングシステムの特徴量を次のようにエンコードできます。

ホスト	Ubuntu	Red Hat Enterprise Linux	SUSE Linux Enterprise Server
A	1	0	0
B	0	1	0
C	0	0	1

ご覧のとおり、ホストAではUbuntuを実行しており、さらにホストBとCではそれぞれLinuxのRed HatとSUSEバージョンを実行しています。

この方法の代わりに、値を各オペレーティングシステムに割り当てて、それぞれのホストをベクトル化することもきます。

オペレーティングシステム	割当値	ホスト	ベクトル
Ubuntu	1	A	1
Red Hat Enterprise Linux	2	B	2
SUSE Linux Enterprise Server	3	C	3

ただし、マッピングを無造作に行うことは、クラスタリングアルゴリズムなどの機械学習の処理に悪影響を及ぼす可能性があるため、避けるように注意する必要があります。さもないと、実際には存在しない値の意味が誤って推論される可能性があります。たとえば、上記のマッピングを使用した場合、アルゴリズムが、UbuntuはRed Hatよりも「小さい」と学習する可能性があります。これは、2よりも1が小さいためです。また、値が反転されると、逆の結果になる場合があります。実際には、これよりも多少複雑なエンコード方法が使用され、その方法はしばしば、「ワンホットエンコーディング」と呼ばれます。

多くの場合、連続データとカテゴリデータは組み合わせて使用されます。たとえば、連続的な特徴量（開いているポート、閉じているポート、フィルタリングされたポートなど）の集合と、カテゴリの特徴量（オペレーティングシステム、各ポートで実行されているサービスなど）を組み合わせて取り入れて、リスクプロファイルが類似するノードのグループを識別することができます。このような状況ではしばしば、各ベクトル内の特徴量に等しい重みが与えられるように、「正規化」の処理によって連続するベクトルの値の範囲を圧縮することが必要になります。たとえば、k平均法アルゴリズムでは、類似性によってベクトルをグループ化するために、中心点からの平均距離を使用します。正規化を使用しない場合、k平均法で、カテゴリデータの影響度に過剰に重みが付けられて、結果が不正確になる場合があります。

次の例について考えてみましょう。

サンプル(サーバー)	1秒あたりの要求数	CPU使用率(%)
Alpha	200	67
Bravo	160	69
Charlie	120	60
Delta	240	72

特徴量「1秒あたりの要求数」の値の範囲は、特徴量「CPU使用率(%)」の10倍近くあります。これらの値を正規化しない場合、距離の計算が不正確になり、範囲の不均衡の影響が過度に強調される可能性があります。

たとえば、上の表でサーバーAlphaとサーバーBravoは、「1秒あたりの要求数」については40の差がありますが、「CPU使用率(%)」については2しかないことがわかります。この場合、サーバー間の差の95%を「1秒あたりの要求数」が占めることになり、この不均衡によって以降の距離計算に大きな誤差が出る可能性があります。

この誤差の問題に対応するために、両方の特徴量を0から1の範囲に正規化します。これには、x-xmin/(xmax – xmin)の式を使用します。

サンプル(名前)	1秒あたりの要求数	CPU使用率(%)
Alpha	.66	.58
Bravo	.33	.75
Charlie	0	0
Delta	1	1

正規化した後のサーバーAlphaとBravoの「1秒あたりの要求数」の差は0.33です。一方「CPU使用率(%)」の差は17に減少しています。これで、「1秒あたりの要求数」の差が合計に占める割合が66%になりました。

手順4：計算とグラフ化

特徴量をベクトルに変換したら、その結果を適当な統計分析やデータマイニングアプリケーションにインポートできます。たとえ

ば、IBM SPSS ModelerやSAS Data Mining Solutionなどにインポートできます。また、数百種類あるソフトウェアライブラリの1つを利用して、この分析を実行することもできます。以下の例ではscikit-learnを使用します。scikit-learnは無料のライブラリで、Pythonプログラミング言語で使用できるオープンソースのデータマイニングおよび統計関数です。

データを読み込んだら、最初に、適用するクラスタリングアルゴリズムを選択できます。scikit-learnでは、たとえば、k平均法、Affinity Propagation、平均値シフト、スペクトルクラスタリング、Ward階層的クラスタリング、凝集型クラスタリング、DBSCAN、ガウス混合、Birchなどを選択できます。それでは、最も良く使われているクラスタリングアルゴリズムのk平均法とDBSCANについて考察しましょう。

k平均法によるクラスタリング

人間はこの世界が3つの次元で構成された空間であると体感します。このため、2つの物体の間の距離を、それらを結ぶ最短の直線の長さで測ることができます。これは、ピタグラスの定理を使用して計算できる「ユークリッド距離」です。

クラスタリング分析では、「特徴量空間」の概念が取り入れられており、サンプルセットのすべての特徴量に1つずつ次元を割り当てます。次元数は、数千次元でも定義できます。クラスタリングアルゴリズムは、特徴量空間の特定の座標にベクトルを割り当てて、任意の2つのベクトル間の距離を測定し、それらを同じクラスタへ入れるだけの十分な類似性があるかどうかを判定します。以降の説明で確認するとおり、クラスタリングアルゴリズムでは、さまざまな距離の測定基準を使用して測定できます。ただし、k平均法では、ユークリッド距離のみを使用します。k平均法やその他のほとんどのクラスタリングアルゴリズムでは、ベクトル間のユークリッド距離が短いほど、それらのベクトルが同じクラスタに割り当てられる可能性が高くなります。

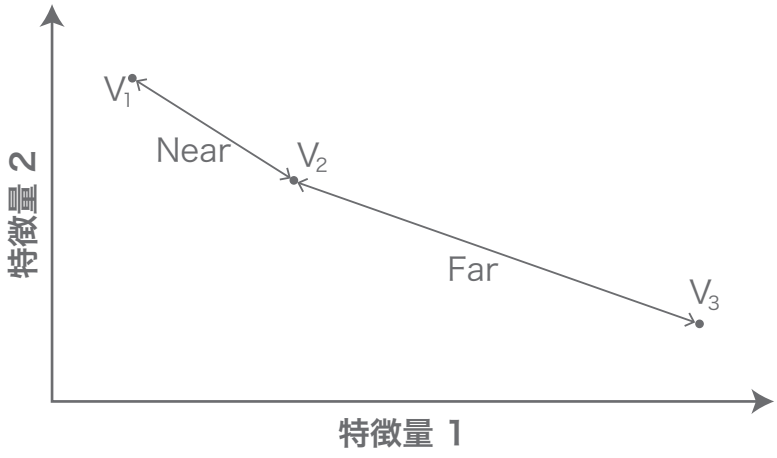


図1.1：特徴量空間内のベクトル

k平均法は計算効率が高く、幅広いデータ分析処理に適用できます。ただし、次のようにいくつかの注意点があります。

- これから説明するk平均法の手法は、連続データのみに適用できます(より高度な手法では、カテゴリーデータにも使用できます)。
- データに内在するパターンのクラスタは、特徴量空間で、直線と平面で囲まれた領域に当てはめて定義することが可能でなければなりません。
- データを有意義な方法でグループ分し、同程度の大きさのクラスタに割り当てることができなければなりません。

これらの条件を満たす場合、次の手順でクラスタリングを進めることができます。

1．データセットをサンプリング、ベクトル化、および正規化して、scikitlearnにインポートします。
2．データアナリストが、k平均法アルゴリズムを起動し、「k」を指定します。これは入力変数または「ハイパーパラメータ」であり、k平均法によって何個のクラスタを作成するかを指定するものです(注：ほとんどのアルゴリズムには、動

作を「チューニング」するためのハイパーパラメータが1つ以上含まれています)。この例では、kを3に設定して、最大3個のクラスタを作成します。

3．k平均法によって、データセットからランダムに3つのベクトルが選択され、特徴量空間内の座標が1つずつ割り当てられます。それぞれの座標は、作成しようとしている3つのクラスタに対応します。これらの座標点は「セントロイド」と呼ばれます。

4．データセットの最初のベクトルに対してk平均法の処理が開始されます。ベクトル座標と、3つのセントロイド座標間のユークリッド距離が計算されます。次に、サンプルが最も近いセントロイドのクラスタに割り当てられます。この方法ですべてのベクトルが割り当てられるまで処理が続けられます。

5．k平均法のアルゴリズムによって、各クラスタを構成するベクトルが確認され、対応するセントロイドからの平均距離が計算されます。セントロイドの現在の位置がこの平均値と一致する場合、セントロイドはその位置にとどまります。一致しない場合は、セントロイドが計算された平均値に一致する新しい座標に移動します。

6．すべてのベクトルに対してk平均法が繰り返され、新しいセントロイドの位置を基準にしてベクトルがクラスタに再度割り当てられます。

7．次のいずれかの状態になるまで、手順5〜6でk平均法が繰り返されます。

- セントロイドの移動が止まり、クラスタを構成するベクトルが変わらなくなった場合。この状態は「収束」と呼ばれます。
- アナリストが事前に指定した、最大繰り返し回数に到達した場合。

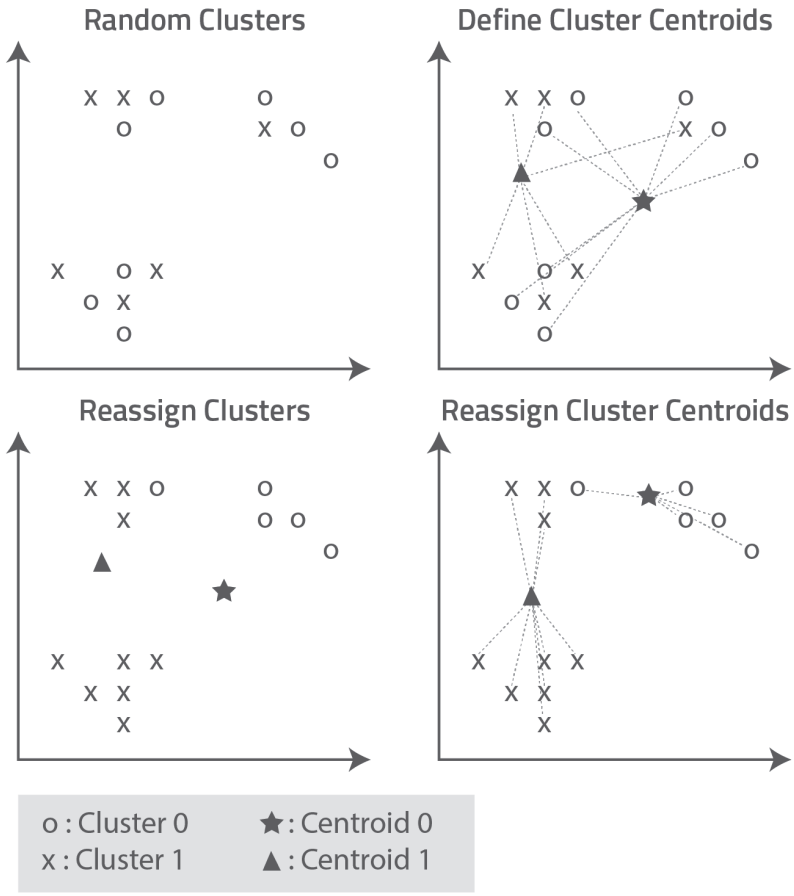


図1.2：k平均法のクラスタリングプロセス

クラスタリングが完了すると、データアナリストは次のことができるようになります。

- さまざまな検証の技法を使用して、結果の精度を評価できます。
- 結果を数学的モデルに変換して、新しいサンプルがどのクラスタに入るかを評価できます。
- 統計および機械学習の技法を使用して、より詳しくクラスタの結果を分析できます。

このプロセスは、数百から数千の多数の次元がある特徴量空間にも同じように適用されます。ただし、毎回の繰り返しにかかる計算時間は、分析する次元数に比例して大きくなります。

k平均の法注意点と制限事項

k平均法は扱いやすく、画期的な結果を導き出すことも可能ですが、ここで説明したk平均法には、誤差やデータの偏りなどいくつかの弱点があります。

アナリストは、最初に作成するクラスタの数について、情報を基に推測する必要があります。これには豊富な経験とこの分野の専門知識が必要になります。実際のところは、最適なクラスタ数を見つけるまでクラスタリングの作業を複数回繰り返すことがしばしば必要になります。

クラスタリングの結果は、セントロイドの最初の場所によって大きく異なる可能性があります。今使用しているk平均法では、セントロイドの場所がランダムに割り当てられるため、アナリストがこの位置を制御することができません。この理由にもより、アナリストはクラスタリングの手順を複数回実行して、最も実用的でデータと一致するクラスタリングの結果を選択することが必要になる場合があります。

次元数が非常に多い特徴量空間では、ユークリッド距離が類似性の尺度として機能しなくなります。これは、機械学習の専門家が包括的に「次元の呪い」と呼ぶ問題の1つです。この状況では、別のアルゴリズムや類似性測定の手法を採用する必要があります。

DBSCANによるクラスタリング

よく使用されるクラスタリングアルゴリズムには他にもDBSCANまたは「Density-Based Spatial Clustering of Applications with Noise（ノイズを伴う用途に対する密度に基づいた空間クラスタリング）」があります。DBSCANは、1996年にハンスピーター・クリーゲルが最初に取り入れました。

DBSCANという名前が表すように、特徴量空間の指定された領

域内にある複数の点の密度を評価することによりクラスタを特定します。DBSCANでは、ベクトルが最も集まった領域にクラスタが作成され、まばらな領域の点はノイズと見なされます。

DBSCANは、k平均法と次のような点が異なります。

- DBSCAN自体によって、作成するクラスタの数が見つけられます。アナリストがハイパーパラメータkを使用して事前にクラスタ数を指定する必要はありません。
- 任意の形状およびサイズのクラスタを作成できます。

DBSCANには2つのハイパーパラメータがあり、クラスタリングプロセスの実行方法を指定できます。

- イプシロン（Eps）は、各点を取り囲む円形領域の半径を指定します。この円形領域は、クラスタ内のベクトルを評価するために使用され、点の「イプシロン近傍」と呼ばれます。半径は、さまざまな距離の測定基準を使用して指定できます。
- 点の最小数（MinPts）は、クラスタに含まれる点について、イプシロン近傍内に存在しなければならない点の最小数を指定します。

DBSCANで実行されるクラスタリングでは、データセット内の各点が評価され、次の3つのカテゴリのいずれかに割り当てられます。

- コア点は点の1種であり、そのイプシロン近傍内にMinPtsの指定数よりも多くの点が含まれているものです。
- 境界点はコア点の近傍内に存在する点で、その点自体には十分な数の近傍が存在しないためコア点になれないものです。
- ノイズ点は、コア点でも境界点でもない点です。

コア点、境界点、ノイズ点の違いを下の図に示します。

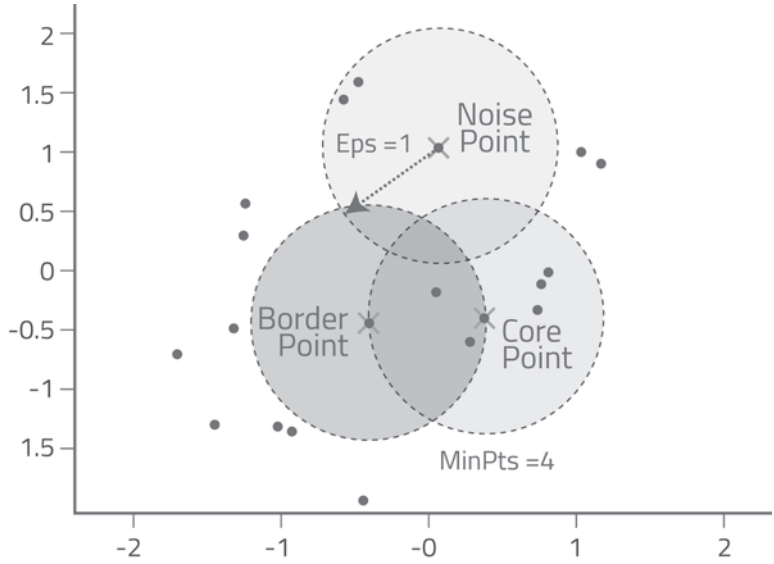


図1.3：DBSCANのクラスタリングプロセス

scikit-learnにおけるDBSCANのクラスタリングセッションは、通常、次のように進められます。

1．データセットがサンプル化、ベクトル化、および正規化されて、scikit-learnにインポートされます。
2．アナリストは、DBSCANオブジェクトを作成して、最初のEpsとMinPts値を指定します。
3．DBSCANによって特徴量空間内の点の1つ、たとえば点Aがランダムに選択されます。次に、点AのEps近傍内にある点の数が数えられます(点Aを含む)。この数がMinPts以上の場合、その点はコア点に分類され、DBSCANによって新しいクラスタに点Aとその近傍が加えられます。既存のクラスタと区別するために、新しいクラスタにはクラスタIDが割り当てられます。
4．DBSCANは、点Aから、近傍の1つのたとえば点Bに移動し、次にその点をコア点と境界点のいずれかに分類します。点Bがコア点と認識された場合、点Bとその近傍は同じクラ

スタに追加され、同じクラスタIDが与えられます。このプロセスは、DBSCANがすべての近傍に移動し、すべてのクラスタのコア点と境界点が検知されるまで続けられます。

5．DBSCANは、前に移動したことのない点に次々と移動し、すべての近傍点およびノイズ点の分類が終わるまで手順3と4を繰り返します。このプロセスが完了すると、すべてのクラスタが特定済みで、クラスタIDが発行されたことになります。

この分析の結果が満足できるものであれば、クラスタリングのセッションは終了です。満足できない場合、アナリストにはいくつかの選択肢があります。EpsおよびMinPtsハイパーパラメータを調整し、結果が要求を満たすまでDBSCANを再実行することができます。また、別の距離の測定基準を適用することで、Eps近傍の定義におけるEpsハイパーパラメータの機能を定義しなおすこともできます。DBSCANでは、次のようにさまざまな測定基準がサポートされています。

- **ユークリッド距離**　これは、以前に説明した「点の間の最短の直線距離」を測定する方法です。
- **マンハッタン距離または都市ブロック距離**　その名のとおり、この方法は、街路と通りが2次元グリッドのようにレイアウトされた大都市で、2つの場所の距離を測定するときに使う方法と似ています。この場合、移動は常に1つの次元に沿うように制限されており、目的地に着くまで角を左または右に曲がり続けながら進みます。たとえば、私たちがマンハッタンで3番街の51丁目から2番街の59町目まで歩く場合、目的地に到達するまでに東へ1ブロック移動し、次に北に8ブロック移動する必要があります。合計のマンハッタン距離は9ブロックになります。特徴量空間を一度に1つの次元しか移動できない多次元グリッドとして扱うことで、DBSCANではほぼ同じ方法で、Eps近傍の数と点の間の距離を計測できます。ここで、点の間の距離は、点AからBに移動

するために通る必要がある経路で、各軸に沿った距離の単位数を合計することで計算されます。

- **コサイン類似性**　クラスタ分析では、特徴量の類似性が、特徴量空間内の相対距離で表現されます。2つのベクトルが互いに近ければ近いほど、同じEps近傍内に存在し、同じクラスタに属する確立が高くなります。ただし、2つのベクトル間の距離は、各ベクトルが三角形の頂点にあり、3番目の頂点がグラフの原点にあるものとして定義することもできます。この場合の距離は、2つのベクトルと原点を結んだ2本の線の間にできる角度のコサインを求めることで計算されます。角度が小さいほど、2つの点の特徴量が類似し、同じEps近傍内に存在する可能性が高くなります。同様に、角度が大きくなると、2点の特徴量はあまり類似しておらず、異なるクラスタに存在する可能性が高くなります。

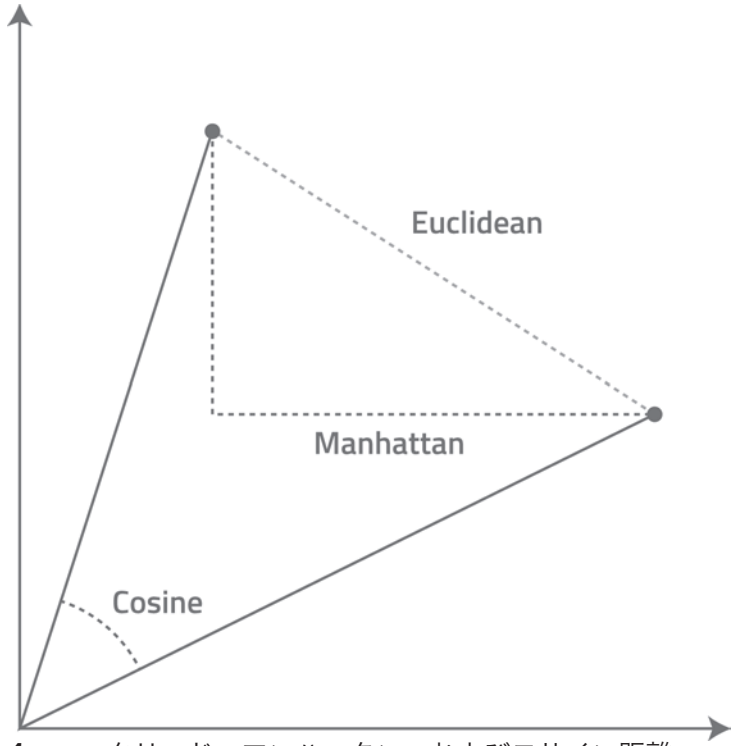


図1.4：ユークリッド、マンハッタン、およびコサイン距離

DBSCANの注意点と制限

DBSCANはk平均法よりも広い範囲のクラスタ形状とサイズを見つけることができますが、次の注意点があります。

- MinPtsとEps設定の小さな変化にも敏感に反応し、適切に定義されたクラスタを小さなクラスタの集合に細分化してしまう可能性があります。
- 次元が増えると計算効率が低下するため、多数の次元がある特徴量空間の場合、許容できないパフォーマンスになる可能性があります。
- MinPtsとEpsには固定値を割り当てなければならないため、さまざまな密度の領域があるデータセットでは、良好な結果が得られない場合があります。

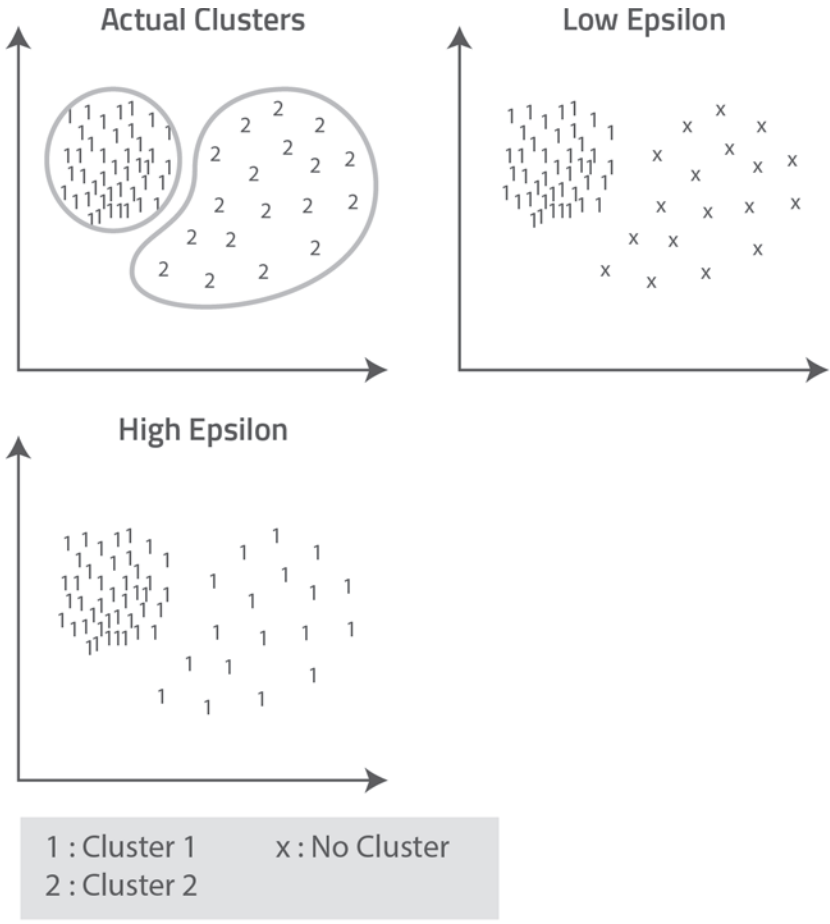


図1.5：DBSCANのクラスタ密度の注意点

クラスタの有効性の評価

クラスタリング手順の説明の最後は、k個のクラスタで構成された結果について考察します。それらのクラスタが元のデータを正確に表現しているかどうかは、どのように評価すればよいでしょうか。このような評価は、問題の解を求めるときに、クラスタリングの処理を異なるアルゴリズムで複数回実行したり、同じアルゴリズムを異なるハイパーパラメータの設定で複数回実行して折り合いを付ける場合に必要になります。

幸いなことに、クラスタの正当性を検証する方法は数多くあります。それらは「インデックス」または「検証基準」と呼ばれます。たとえば、次のような方法を採用できます。

- サンプルセットを外部のモデルを使って実行し、クラスタの割り当て結果が、独自に求めた結果と一致するかどうかを確認します。
- 「未使用のデータ」、つまりクラスタ分析に使用しなかったデータセットのベクトルで結果をテストします。もしクラスタの結果が正しければ、新しいサンプルも、元のデータと同じクラスタに割り当てられることが期待されきます。
- 統計的な手法を使用します。たとえば、k平均法の場合、シルエット係数を計算できます。この手法は、特定のクラスタ内にある点の間の平均距離と、別のクラスタに割り当てられた点との間の平均距離を比較するものです。この係数が小さいほど、クラスタリング結果の精度の信頼性が高くなります。
- 異なるアルゴリズムで作成されたクラスタリングの結果や、異なるハイパーパラメータの設定で同じアルゴリズムから作成された結果を比較します。たとえば、k平均法とDBSCANのシルエット係数を計算して、どちらのアルゴリズムが最適な結果を生成したかを確認したり、Epsに異なる値を使用してDBSCANを実行し、結果を比較したりできます。

クラスタ分析を実際の脅威のシナリオに適用

これまでの説明のとおり、クラスタ分析では、大量のネットワークの処理やシステムデータを調査して、隠れたクラスタメンバー間の関係を検出できます。これはクラスタメンバーの定義に用いられた特徴量の類似性や相違に基づいて行われます。では、現実のネットワーク攻撃を検出および予防するために、この分析能力をどのように適用して、機能させればよいでしょうか。

それでは、パナマ文書問題に対してクラスタ分析がどのように利用できたかを検討しましょう。パナマ文書問題では、1150万通の機

密文書と2.6テラバイトの顧客データがパナマ在籍の法律事務所Mossack Fonseca（MF）から流出してしまいました。

はじめに、次の3つの問題点を挙げます。

- さまざまなメディアやセキュリティ組織から、最も可能性の高い攻撃ベクトルについて説得力のある証拠が提示されていますが、ハッカーの「ジョンドゥ（John Doe）」がMFのWebサーバー、電子メールサーバー、およびクライアントデータベースに1年以上にわたって侵入した方法について明確に説明できた人はいませんでした。この侵害行為の性質や規模を確認するには、MFのネットワークおよびシステムデータを対象にフォレンジック分析を徹底的に行う必要があるでしょう。
- それらのデータは、十分な範囲から取得され、さらに信頼できる品質でなければなりません。攻撃を検知および予防するためによく使用されるデータ主体の技法に必要になるためです。
- また、分析はクラスタリング単独で行われるとはかぎりません。最善を尽くすには、さまざまな機械学習、人工知能、および統計の手法をクラスタリングと併用するはずです。

ただし、ここではクラスタリングのみのシナリオで作業し、信憑性のあるメディアや業界の情報源から入手した証拠を使用します。

ソフトウェアエンジニアリングファームのWordfence[8]によると、たとえば、ハッカーの「John Doe」は、WordPressのRevolution Sliderプラグインの既知の脆弱性を標的に攻撃を始めた可能性があります。この脆弱性は、2014年11月にExploit Database Web

8. Mark Maunder、『Panama Papers: Email Hackable via WordPress, Docs Hackable via Drupal（パナマ文書：WordPressを介してハッキング可能な電子メール、Drupalを介してハッキング可能な文書）』（2016年4月8日）、https://www.wordfence.com/blog/2016/04/panama-papers-wordpress-email-connection/（2016年5月15日参照）、Mark Maunderの『Mossack Fonseca Breach—WordPress Revolution Slider Plugin Possible Cause（Mossack FonsecaのセキュリティTE害—WordPressのRevolution Sliderプラグインが原因として有力）』（2016年4月7日）、https://www.wordfence.com/blog/2016/04/mossack-fonseca-breach-vulnerable-slider-revolution/（2016年5月15日参照）

サイトで報告されています。John Doeは、WordPress Webサーバーに PHPスクリプトをアップロードすることでこの脆弱性を悪用した可能性があります。これにより、シェルアクセスが可能になり、wp-config.phpなどのサーバーファイルにアクセスできます。wp-config.phpは、WordPressのデータベース認証情報がクリアテキストで保存されているファイルです。また、ALO EasyMail Newsletterプラグインを使用して、データベースへアクセスすることにより、クリアテキストで保存されているすべての電子メールアカウントを取得することもできたはずです。MFは、ALO EasyMail Newsletterプラグインの電子メールリスト管理機能を使用していました。結果的に、このメールサーバーやその他のサーバーのハッキングによって、ジョンドゥはMFの大量の電子メールにアクセスし、抜き取ることができたでしょう。

また、フォーブス誌[9]の報道によると、MFが攻撃を受けたときにDrupalバージョン7.23が実行されていました。MFは、これを使用して「セキュリティで保護された」ポータルを管理しており、顧客はこのポータルを使用して個人文書へアクセスしていました。このDrupalのバージョンは、攻撃に対するさまざまな脆弱性があることで広く知られていました。その1つにSQLインジェクションのエクスプロイトがあり、この攻撃のみで大量の文書を取得するための経路を開くことができたでしょう。

サイランスは、これらの情報やその他の情報から、クラスタ分析によってMFのネットワーク活動の異常を検知できること、またJohn Doeの攻撃の性質や規模について重要な手がかりを獲得できる可能性があることに気づきました。これは継続的な検出プロジェクトの一環で発見されました。通常、検出チームのメンバーは、Webサーバーとメールサーバーのログを別々に分析します。次に、いずれかのサーバーで攻撃が検出された場合には、検出チーム

9. Jason Bloomberg,、『Cybersecurity Lessons Learned from 'Panama Papers' Breach（パナマ文書問題から学んだサイバーセキュリティの教訓）』、Forbes.com（2016年4月）、http://www.forbes.com/sites/jasonbloomberg/2016/04/21/cybersecurity-lessons-learned-from-panama-papers-breach/#47c9045d4f7a

がもう一方のサーバーのデータを分析し、同じ攻撃者が両方の攻撃に関係していないか、さらに損害の範囲について示すものはないかを確認します。

メールサーバー側の関連する特徴量で抽出すべきものには、ユーザーのログイン日時、IP アドレス、地理的な場所、電子メールクライアント、管理権限、SMTP サーバーでの活動などがあります。Web サーバー側の関連する特徴量で抽出すべきものには、ユーザーの IP アドレスと場所、ブラウザのバージョン、アクセスしているページのパス、Web サーバーのステータスコード、関連する帯域幅の使用率などがあります。

このクラスタ分析を完了したとき、電子メールと Web のベクトルの大部分が適切に定義された一連のクラスタに割り当てられ、それらのクラスタが通常の処理パターンを表すことが予想されます。そして、密度が希薄な少数のクラスタやノイズ点が、異常なユーザーやネットワークの活動を示しているはずです。次に、それらの異常を詳しく調べることができます。ログデータ全体をグレップ検索して、疑わしい行為と、特定の IP アドレスからアクセスする悪質なアクターの候補を照合します。

この分析により次の検出が可能です。

- **異常な認証パターン：**ロンドンのオフィスを拠点とする MF の役員のクラスタが、今まで使用したことのない電子メールクライアントを使って電子メールアカウントにアクセスを始めた場合、不審な行動として検出できます。また、ロンドンのオフィスを拠点とする従業員のグループが、事務所、顧客、あるいはビジネスパートナーが存在しない場所から定期的に電子メールアカウントにアクセスしている場合には監視できます。
- **異常なユーザーの行動：**顧客のクラスタで、アップロードをまったくしていないにもかかわらず、ログインして長時間大量の文書をダウンロードしている場合は検出が可能です。また、電子メールユーザーのクラスタで、メールの送信はして

いないにもかかわらず、長時間電子メールを読み取っている場合は検出が可能です。

- **異常なネットワークトラフィックのパターン：**パスにDrupalの記述がある顧客のポータルページやURLを宛先とするトラフィックの量については、急激な増加を監視できます。

もちろん、これらの例は仮定にすぎません。パナマ文書問題などの攻撃に対して、どの程度でクラスタリング分析のフラグを立てるかは、実際のネットワークやシステムデータの内容、および検出チームのデータ分析に関する専門知識に従って判断されます。ただし、クラスタ分析よって、セキュリティ侵害に関する重要な手がかりを得られることは明らかです。クラスタ分析がなければ、中規模のネットワークから毎週生成される数千のログエントリを調べて見つけ出すことは容易ではありません。さらに、この手がかりはデータそのものから取得でき、エクスプロイトシグネチャやIDS/IPSシステムからのアラートに依存する必要がありません。

HTTPログデータを利用したクラスタリング

これまで学んだことを利用して、クラスタリングにより、どのように攻撃を検出し、その侵攻状況を追跡できるかを確認しましょう。ここでは、secrepo.comのHTTPサーバーログデータを分析します。これによりパナマ文書流出以前の既知のエクスプロイトに類似する、いくつかのエクスプロイトが明らかになります。この実習を行わない場合は、https://www.cylance.com/intro-to-aiのページを参照してください。このページでは、関連する手順やデータファイルをすべてダウンロードできます。

HTTPサーバーのログには、エンドユーザーについてや、エンドユーザーのインターネットアクセスパターンに関するさまざまな有効なフォレンジックデータが記録されています。これには、IPアドレス、日時のスタンプ、要求内容、サーバーの応答などがあります。この例では、IPアドレスをHTTP動詞（GET、POSTなど）と

HTTP レスポンスコード（200、404 など）を基にクラスタリングします。IP アドレス 70.32.104.50 が WordPress サーバーを狙った攻撃に関係しているという情報を WAF や脅威インテリジェンスフィードから入手した場合、潜在的な侵害の証拠を探索します。特に、WordPress の脆弱性について、Revolution Slider のような深刻なものが報告されていないかどうかを考慮する必要があります。したがって、70.32.104.50 について報告された行動パターンと類似するものを検出するように IP アドレスをクラスタリングします。これが存在する場合は、自社のサーバーがセキュリティ侵害を受けている可能性があることを示します。

この個別のデータセットに使用された、HTTP の応答コードは、以下のとおりです：

200、404、304、301、206、418、416、403、405、503、500

この個別のデータセットに使用された HTTP コマンドは、以下のとおりです：

GET、POST、HEAD、OPTIONS、PUT、TRACE

これから、クラスタリングの手順を k 平均法と DBSCAN によって 1 回ずつ行います。それぞれの手順の最後に、ログファイルを参照して、異常値や疑わしいクラスタに含まれる IP アドレスの振る舞いを詳しく確認します。

k平均法によるクラスタ分析

手順 1：ベクトル化と正規化

まず、分析用のログサンプルを準備します。ここでは、若干手順を省略して、この個別のデータセットをベクトル化および正規化するように明確に記述したスクリプトを適用します。

IP アドレスごとに、HTTP レスポンスコードと動詞の数をカウントします。発生件数を単純に合計するのではなく、これらの特徴量を正規化して連続値として表現します。正規化を行わない場合、ほ

ぼ同一の行動パターンを持つ2つのIPが、別のクラスタに割り当てられる可能性があります。これは単純に、一方のIPで作成されたレポートの数が他方より多いためです。時間とCPUの処理能力が豊富にある場合は、181,332件以上のエントリがあるログファイルで、16,407件のすべてのIPアドレスを確認できるでしょう。

ただし、ここでは先頭から10,000件のIPアドレスに着手し、このサンプルが、攻撃が行われたことを判断するために十分かどうかを確認します。さらに、サンプルは5件以上のログエントリに関連付けられたIPアドレスに限定します。頻度の少ない活動が、WebおよびWordPressサーバーに重大な脅威をもたらす可能性はさほど高くありません。

次のPythonスクリプトは、ベクトル化プロセスを起動します。

`python vectorize_secrepo.py`

```
mldemo@mldemo-virtual-machine:~/mlbook$ python vectorize_secrepo.py
Finished prebuilding samples
```

これにより、「secrepo.h5」というHDF5 (Hierarchical Data Format) ファイルが生成されます。このファイルには、ベクトルとクラスタID、およびそれぞれのベクトルにどのIPアドレスが関係するかを示す「メモ」が記述されています。このファイル内のアドレスは、後から潜在的な悪意のある活動を調べるためにログを再度確認するときに使用します。

手順2：ベクトルのグラフ化

これで、特徴量空間のベクトルを視覚化する準備ができました。

人間は、3次元を超える空間環境を視覚化することができません。このため、高い次元の特徴量空間から得たクラスタリングの結果を判読することは容易ではありません。幸い、特徴量を減らす技法を用いることで、クラスタを3次元のグラフィックス形式で表示することができます。次のスクリプトは、特徴量を減らす技法の1つである、主成分分析を適用しています。これにより、グラフを3つの軸のいずれかを中心に回転してクラスタを調べることができま

す。ただし、回転は計算負荷が大きな処理であるため、表示の更新が遅くなる場合があります。多くの場合、グラフ化の処理では、あらかじめいくつかの表示角度の画像を用意しておくと、高速で、便利になります。後から、あらかじめ用意した表示をすばやく切り替えて、クラスタを異なる角度で表示できます。

次のスクリプトを使用して、ベクトルを視覚化します。

```
`python visualize_vectors.py -i secrepo.h5`
```

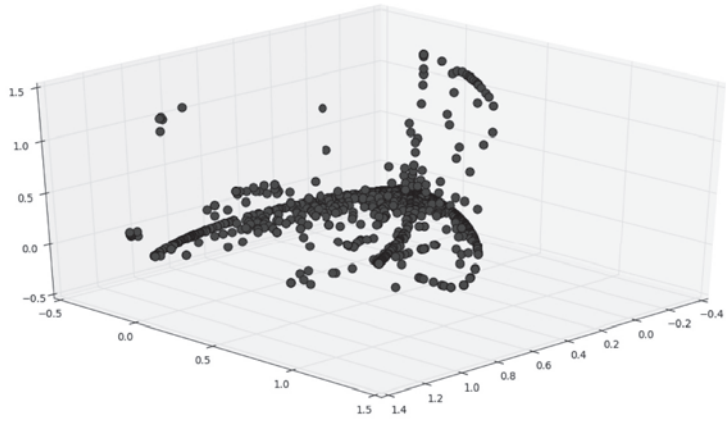


図1.6：ベクトルの投影による視覚化

手順3：k平均法による1回目のクラスタリング

前述のとおり、k平均法で必要になるのはハイパーパラメータkを設定することだけです。このパラメータには作成するクラスタの数を指定します。最初の段階では、kの正しい値はわかりません。したがって、異なるk値を設定して、結果を調べるクラスタリングプロセスを繰り返し行います。これは、データを正確にモデル化して満足できる結果になるまで続けます。最初はkを「2」に設定します。さらに、k平均法では、ベクトル化の際に各クラスタに割り当てたクラスタIDを使用するように指定できます。

```
`python cluster_vectors.py -c kmeans -n 2 -i secrepo.h5 -o
secrepo.h5`
```

次のとおり、k平均法によってサンプルが分析され、ラベルが付けられて、2つのクラスタのそれぞれに割り当てられるベクトルの数が計算されます。

```
mldemo@mldemo-virtual-machine:~/mlbook$ python cluster_vectors.py -c kmeans -n 2 -i secrepo.h5 -o secrepo.h5
Clustered samples into 2 clusters
Label 0 has 662 samples
Label 1 has 3192 samples
```

手順4：クラスタの統計的な検証

クラスタIDが表示されるようになったため、シルエットスコアの計算を適用することにより、サンプルが適切にグループに分けられているどうかを判断できます。このスコアの範囲は-1から+1です。スコアが+1に近いほど、グループ分けの正確さについての信頼性が高くなります。シルエットスコアは次のスクリプトによって生成します。

stats_vectors.py

```
mldemo@mldemo-virtual-machine:~/mlbook$ python stats_vectors.py secrepo.h5
Vectors shape: (3854, 17)
Minimum feature value: 0.0
Mean feature value: 0.128143090123
Maximum feature value: 1.0
Percentage of null values: 83.2259836991%

Minimum distance between vectors: 0.0
Mean distance between vectors: 0.592780335395
Maximum distance between vectors: 2.0

Number of labels: 2
Number of items in label 0: 662  (17.1769590036%) (avg dist: 0.9581536657) (avg silhouette: 0.179235978039)
Number of items in label 1: 3192  (82.8230409964%) (avg dist: 0.32877885612) (avg silhouette: 0.727458032666)

Minimum label centroid distance: 0.899422434302
Mean label centroid distance: 0.899422434302
Maximum label centroid distance: 0.899422434302
Overall Silhouette Score: 0.633290155094
```

ご覧のとおり、クラスタ1は適切にグループ分けされていますが、クラスタ0は適切ではありません。さらにクラスタ1には、クラスタ0よりもかなり多くのサンプルが含まれています。これは、クラスタ1が通常のネットワーク活動を反映し、クラスタ0には標準的ではない活動や、悪意のあるユーザーの行動が含まれていると解釈できます。

手順5：クラスタの検査

ここでは、既知の攻撃者のIPアドレスがどちらのクラスタに含ま

れているかを調べます。次のスクリプトを使用して、ラベルと各ベクトルについてのメモを出力します。

`python label_notes.py -i secrepo.h5 | grep 70.32.104.50`

```
mldemo@mldemo-virtual-machine:~/mlbook$ python label_notes.py -i secrepo.h5 | grep 70.32.104.50
0 70.32.104.50
```

これにより、IP 70.32.104.50がクラスタ0のメンバーであることがわかります。クラスタ0は、疑わしいと考えていたクラスタで、平均のシルエットスコアが低いものでした。この結果を受けて、クラスタ0のすべてのベクトルをフォレンジック分析の対象にするよう検討することもできます。ただし、人的な資源は貴重であり、これらのすべてのIPを調査することは効率的ではありません。したがって、まずクラスタリングの結果を改善することに意識を向けるのが妥当です。これにより、調査するサンプルの数を少なくできます。

手順6：Kの変更によるクラスタの結果の最適化

一般に、k平均法のクラスタリングセッションは、2つ以上のクラスタを作成するようにkを設定して始めることが妥当です。その後に、より高いkの値で繰り返して、クラスタが適切に形成され、サンプルの分布を正確に反映していることが検証されるまで続けます。この例では、手順の3と4を複数回実行し、最終的に12がこのデータセットの最適な数であると判断されました。

次のスクリプトで12個のクラスタを生成します。

`python cluster_vectors.py -c kmeans -n 12 -i secrepo.h5 -o secrepo.h5`

```
mldemo@mldemo-virtual-machine:~/mlbook$ python cluster_vectors.py -c kmeans -n 12 -i secrepo.h5 -o secrepo.h5
Clustered samples into 12 clusters
Label 0 has 16 samples
Label 1 has 1994 samples
Label 2 has 223 samples
Label 3 has 73 samples
Label 4 has 123 samples
Label 5 has 79 samples
Label 6 has 53 samples
Label 7 has 225 samples
Label 8 has 60 samples
Label 9 has 319 samples
Label 10 has 42 samples
Label 11 has 647 samples
```

手順7：検査と検証手順の再実行

さらに、スクリプトを実行して、悪意のあるIPを現在含んでいるクラスタのIDを抽出します。

`python label_notes.py -i secrepo.h5 | grep 70.32.104.50`

```
mldemo@mldemo-virtual-machine:~/mlbook$ python label_notes.py -i secrepo.h5 | grep 70.32.104.50
6 70.32.104.50
```

このとおり、悪意のあるIPは、クラスタ6のメンバーです。このクラスタを、シルエットスコアを使用して検証します。

`python stats_vectors.py secrepo.h5`

```
mldemo@mldemo-virtual-machine:~/mlbook$ python stats_vectors.py secrepo.h5
Vectors shape: (3854, 17)
Minimum feature value: 0.0
Mean feature value: 0.128143090123
Maximum feature value: 1.0
Percentage of null values: 83.2259836991%

Minimum distance between vectors: 0.0
Mean distance between vectors: 0.592780335395
Maximum distance between vectors: 2.0

Number of labels: 12
Number of items in label 0: 16  (0.415153087701%) (avg dist: 0.233847802898) (avg silhouette: 0.797796650703)
Number of items in label 1: 1994  (51.7384535547%) (avg dist: 0.100538559594) (avg silhouette: 0.645049288869)
Number of items in label 2: 223  (5.78619615983%) (avg dist: 0.212557574151) (avg silhouette: 0.635324939352)
Number of items in label 3: 73  (1.89413596264%) (avg dist: 0.424114268165) (avg silhouette: 0.398054166997)
Number of items in label 4: 123  (3.1914893617%) (avg dist: 0.443824030386) (avg silhouette: 0.603191718161)
Number of items in label 5: 79  (2.0498183703%) (avg dist: 0.526204140369) (avg silhouette: 0.614395574605)
Number of items in label 6: 53  (1.37519460301%) (avg dist: 0.038767542786) (avg silhouette: 0.953314629418)
Number of items in label 7: 225  (5.8380902958%) (avg dist: 0.265812329702) (avg silhouette: 0.455516831212)
Number of items in label 8: 60  (1.55682407888%) (avg dist: 0.461322425847) (avg silhouette: 0.382170554132)
Number of items in label 9: 319  (8.27711468604%) (avg dist: 0.184890513566) (avg silhouette: 0.495367654549)
Number of items in label 10: 42  (1.08977685522%) (avg dist: 0.398037851835) (avg silhouette: 0.483237183886)
Number of items in label 11: 647  (16.7877529839%) (avg dist: 0.14448543024) (avg silhouette: 0.403902070145)

Minimum label centroid distance: 0.228328581566
Mean label centroid distance: 1.20139629193
Maximum label centroid distance: 1.87341191894
Overall Silhouette Score: 0.572507223029
```

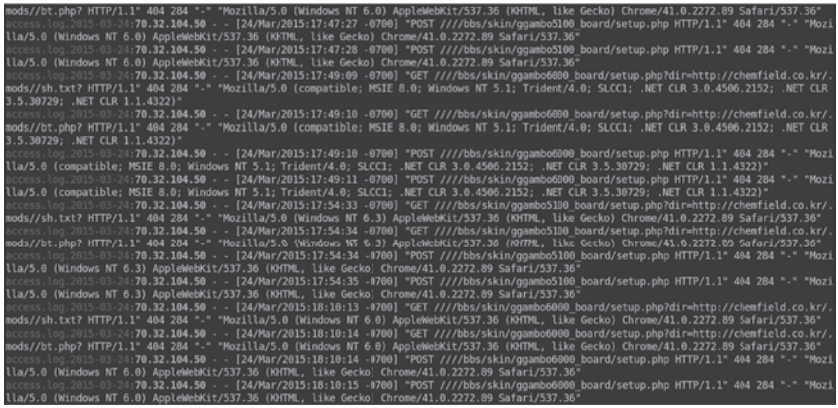
ご覧のとおり、クラスタ6は高いシルエットスコアを持ち、すべてのメンバーと悪意のあるIPアドレスに互いに高い類似性があります。次の作業では、これらのIPアドレスがどのような振る舞いをしていたかを確認する必要があります。Webサーバーログでこれらの活動を追跡します。まず、次のコマンドで、クラスタ6内のすべてのサンプルを出力します。

`python label_notes.py -i secrepo.h5 -l <label>`

```
mldemo@mldemo-virtual-machine:~/mlbook$ python label_notes.py -i secrepo.h5 -l 6
6 95.168.199.227
6 74.208.16.118
6 213.165.70.246
6 181.65.186.34
6 181.65.186.35
6 212.227.18.39
6 192.99.18.191
6 87.106.187.164
6 114.202.2.31
6 184.107.137.250
6 112.218.68.155
6 37.9.169.17
6 180.179.212.185
6 82.98.160.235
6 222.122.56.211
6 180.179.212.214
6 46.252.18.54
6 77.232.91.201
6 211.47.181.38
6 203.58.0.155
6 71.6.150.241
6 41.193.5.54
6 112.175.50.226
6 202.154.23.202
6 70.32.104.50
6 91.142.215.248
6 201.175.20.65
6 185.27.238.194
6 198.144.188.26
6 178.33.226.103
6 212.34.151.164
```

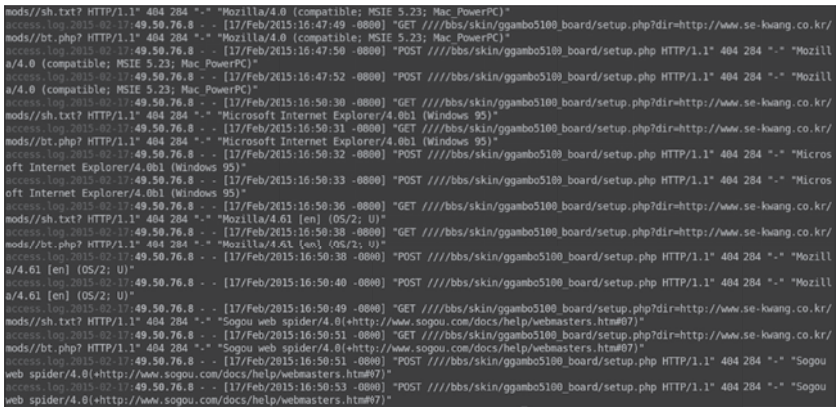
これで、グレップコマンドを使用して、ログ全体を検索し、これらのIPアドレスが現れたエントリを表示できます。最初に、既知の悪意のあるIPで検索します。

```
`grep -ar 70.32.104.50 datasets/http/secrepo/
www.secrepo.com/self.logs/`
```



ご覧のとおり、このIPは、リモートファイルの取り込みに関する脆弱性を利用して、PHPスクリプトペイロードのインストールを試みていました。次に、疑わしいクラスタの別のメンバーで試してみましょう。

```
`grep -ar 49.50.76.8 datasets/http/secrepo/
www.secrepo.com/self.logs/`
```



このIPも悪意のある活動を行っていました。この手順は、クラスタ6のすべてのメンバーを確認してフォレンジック分析を完了するまで繰り返す必要があります。

DBSCANによるクラスタ分析

k平均法の例で、secrepo.h5を既に作成しているため、最初から手順3までは省略し、DBSCANによる1回目のクラスタリングセッションを開始します。初めに、EpsとMinPtsハイパーパラメータをそれぞれ0.5と5に設定します。前述のとおり、DBSCANの場合、正しいクラスタ数をあらかじめ予測する必要がありません。DBSCAN自体でクラスタの数が計算されます。この計算は、特徴量空間でのベクトルの密度と、最小クラスタサイズに基づいて行われます。

クラスタを生成するために次のスクリプトを実行します。

```
`python cluster_vectors.py -c dbscan -e 0.5 -m 2 -i
secrepo.h5 -o secrepo.h5`
```

```
Label -1 has 854 samples
Label 0 has 1782 samples
Label 1 has 117 samples
Label 2 has 5 samples
Label 3 has 15 samples
Label 4 has 38 samples
Label 5 has 57 samples
Label 6 has 38 samples
Label 7 has 29 samples
Label 8 has 40 samples
Label 9 has 30 samples
Label 10 has 25 samples
Label 11 has 26 samples
Label 12 has 16 samples
Label 13 has 12 samples
Label 14 has 6 samples
Label 15 has 15 samples
Label 16 has 16 samples
Label 17 has 10 samples
Label 18 has 25 samples
Label 19 has 54 samples
Label 20 has 25 samples
Label 21 has 8 samples
Label 22 has 34 samples
Label 23 has 42 samples
Label 24 has 21 samples
Label 25 has 9 samples
Label 26 has 17 samples
Label 27 has 11 samples
Label 28 has 44 samples
Label 29 has 8 samples
Label 30 has 7 samples
```

ご覧のとおり、DBSCANは62個のクラスタを作成し、854個のサンプルを割り当てないまま残しました。これら854個のノイズ点のそれぞれが、悪意のある活動に関係する可能性があります。この時点でログに戻って、すべてのノイズ点を調べることは可能ですが、これは時間がかかり効率的ではありません。代わりに、Epsの設定を5から6に増やします。これにより、生成されるクラスタの数が少なくなり、疑わしいサンプルの数も少なくなるはずです。

次のコマンドで新しいハイパーパラメータの設定を適用します。

`python cluster_vectors.py -c dbscan -e 6 -m 5 -i secrepo.h5 -o secrepo.h5`

```
mldemo@mldemo-virtual-machine:~/mlbook$ python cluster_vectors.py -c dbscan -e 6 -m 5 -i secrepo.h5 -o secrepo.h5
Clustered samples into 12 clusters
Label -1 has 25 samples
Label 0 has 3570 samples
Label 1 has 66 samples
Label 2 has 42 samples
Label 3 has 13 samples
Label 4 has 15 samples
Label 5 has 7 samples
Label 6 has 40 samples
Label 7 has 53 samples
Label 8 has 7 samples
Label 9 has 11 samples
Label 10 has 5 samples
```

今回は、DBSCANによって11個のクラスタが作成され、ノイズ点は25個のみになりました。これは、かなり扱いやすい数です。k平均法の際に説明した、クラスタの検査および検証の手順は省略して、25個の疑わしいサンプルの振る舞いについて直接調査を始めます。まず、これらのサンプルのリストを次のコマンドで取得します。

`python label_notes.py -i secrepo.h5 -l -1`

```
mldemo@mldemo-virtual-machine:~/mlbook$ python label_notes.py -i secrepo.h5 -l -1
-1 66.117.9.219
-1 52.5.121.103
-1 52.8.91.105
-1 122.237.72.179
-1 107.20.245.168
-1 185.92.73.125
-1 74.6.254.140
-1 136.243.36.82
-1 180.149.143.26
-1 74.6.254.95
-1 37.110.10.31
-1 188.143.232.11
-1 212.124.109.166
-1 52.6.2.64
-1 23.253.252.202
-1 136.243.48.85
-1 108.45.93.78
-1 52.0.175.28
-1 68.180.228.155
-1 217.91.57.119
-1 54.151.42.39
-1 164.85.131.130
-1 192.187.126.162
-1 111.13.102.4
-1 188.143.232.10
```

次のスクリプトを使用して、ログファイル全体をグレップ検索して、これらのIPがどのような動きをしていたかを見つけます。

`grep -ar 192.187.126.162 datasets/http/secrepo/www.secrepo.com/self.logs/`

```
mldemo@mldemo-virtual-machine:~/mlbook/datasets/http/secrepo/www.secrepo.com/self.logs$ grep -ar 192.187.126.162
access.log.2015-05-23:192.187.126.162 - - [23/May/2015:22:35:33 -0700] "GET / HTTP/1.1" 301 483 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Windo
ws NT 5.1)"
access.log.2015-05-23:192.187.126.162 - - [23/May/2015:22:35:33 -0700] "GET / HTTP/1.1" 200 7041 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Wind
ows NT 5.1)"
access.log.2015-05-20:192.187.126.162 - - [20/May/2015:04:19:41 -0700] "GET / HTTP/1.1" 200 7041 "-" "Mozilla/4.0 (compatible; MSIE 6.0; Wind
ows NT 5.1; SV1; Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1) ; .NET CLR 1.0.3705)"
access.log.2015-05-20:192.187.126.162 - - [20/May/2015:10:29:46 -0700] "HEAD /wp-login.php HTTP/1.1" 301 202 "-" "-"
access.log.2015-05-17:192.187.126.162 - - [18/May/2015:01:37:14 -0700] "HEAD /wp-login.php HTTP/1.1" 404 220 "-" "-"
access.log.2015-05-21:192.187.126.162 - - [21/May/2015:10:26:49 -0700] "HEAD /fckeditor/editor/ HTTP/1.1" 301 207 "-" "-"
access.log.2015-05-21:192.187.126.162 - - [21/May/2015:10:26:49 -0700] "HEAD /js/fckeditor/editor/ HTTP/1.1" 301 210 "-" "-"
access.log.2015-05-21:192.187.126.162 - - [21/May/2015:10:26:49 -0700] "HEAD /include/fckeditor/editor/ HTTP/1.1" 301 215 "-" "-"
access.log.2015-05-21:192.187.126.162 - - [21/May/2015:10:26:49 -0700] "HEAD /admin/fckeditor/editor/ HTTP/1.1" 301 213 "-" "-"
access.log.2015-05-21:192.187.126.162 - - [21/May/2015:10:26:49 -0700] "HEAD /manage/fckeditor/editor/ HTTP/1.1" 301 214 "-" "-"
access.log.2015-05-21:192.187.126.162 - - [21/May/2015:10:26:49 -0700] "HEAD /common/fckeditor/editor/ HTTP/1.1" 301 214 "-" "-"
access.log.2015-05-21:192.187.126.162 - - [21/May/2015:10:26:49 -0700] "HEAD /editor/editor/ HTTP/1.1" 301 204 "-" "-"
access.log.2015-05-18:192.187.126.162 - - [18/May/2015:19:49:37 -0700] "HEAD /js/fckeditor/editor/ HTTP/1.1" 404 220 "-" "-"
access.log.2015-05-18:192.187.126.162 - - [18/May/2015:19:49:37 -0700] "HEAD /common/fckeditor/editor/ HTTP/1.1" 404 220 "-" "-"
access.log.2015-05-18:192.187.126.162 - - [18/May/2015:19:49:37 -0700] "HEAD /fckeditor/editor/ HTTP/1.1" 404 220 "-" "-"
access.log.2015-05-18:192.187.126.162 - - [18/May/2015:19:49:37 -0700] "HEAD /editor/editor/ HTTP/1.1" 404 220 "-" "-"
access.log.2015-05-18:192.187.126.162 - - [18/May/2015:19:49:37 -0700] "HEAD /admin/fckeditor/editor/ HTTP/1.1" 404 220 "-" "-"
access.log.2015-05-18:192.187.126.162 - - [18/May/2015:19:49:37 -0700] "HEAD /manage/fckeditor/editor/ HTTP/1.1" 404 220 "-" "-"
access.log.2015-05-18:192.187.126.162 - - [18/May/2015:19:49:37 -0700] "HEAD /include/fckeditor/editor/ HTTP/1.1" 404 220 "-" "-"
```

このように、これらのクラスタメンバーの大半は、侵入行為を試みたのではなく、通常とは違う行動や一時的な障害が原因でグループ化されていました。ただし、若干のボットが脆弱なサーバーを

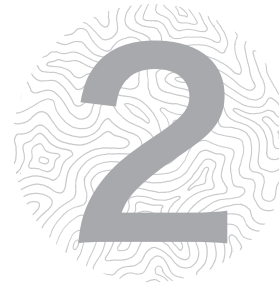
探っていることを突き止めました。たとえば、192.187.126.162は、サーバーを探索して、WordPressまたはfckeditorがインストールされているものを特定していました。ジョンドゥは、脆弱性の標的としてMossack Fonsecaを見つけるために、このようにボットを多用していた可能性があります。MFがクラスタリングを使用してこれらのプローブを検出し、脆弱性にパッチを適用していたら、パナマ文書の問題が起こることはなかったでしょう。

クラスタリングの重要点

ここまで見てきたように、クラスタリングは数学による厳密な解決方法を提供し、他の方法では、検出が困難または不可能と思われるネットワーク、アプリケーション、ファイル、およびユーザーデータ間のパターンや関係を検出します。でも、本書の分析の解説では、クラスタリングは出発点にすぎません。以降の章では、この他の統計技法、人工知能、および機械学習の技術について説明します。それらは、ネットワークセキュリティソリューションの開発と導入に幅広く採用されています。ここでは、ひとまずクラスタリングの重要点についてまとめます。

- クラスタ分析は、関連する特徴量を抽出および正規化すれば、事実上あらゆる種類のデータに適用できます。
- クラスタ分析では、サンプル間の類似性が求められ、その類似性を使用してクラスタの割り当てが決定されます。類似性は、特徴量空間内の位置に基づいてベクトル間の距離を測定することにより求められます。距離の測定基準には、ユークリッド、マンハッタン、コサインなどさまざまな手法を適用できます。
- k平均法とDBSCANは、使い方が容易で、計算効率が高く、さまざまなクラスタリングの状況に適用できます。ただし、それぞれの手法には「次元の呪い」という弱点があり、非常に次元数の多い特徴量空間で分析を行うには適さない場合があります。

- クラスタリングの結果は、現実世界のセキュリティ脅威に対して統計的に検証し、慎重に評価する必要があります。これには、この分野の豊富な専門知識と、クラスタリング手法の機能、長所および短所についての詳細な知識が必要です。
- クラスタリングは、データ探索やフォレンジック分析に特に有効です。クラスタリングによって、異常値や例外値を特定するために、大量のデータを調べることができます。さらに、サンプルやクラスタリングの結果に、以降の章で説明する分析方法などの他の分析方法を適用することもできます。



分類
ロジスティック回帰とディシジョンツリーアルゴリズムの使用

人間は、自分を取り囲む世界を理解するために、さまざまな認識方法を用いています。最も有効な方法の1つに、物や考えを個別のカテゴリへ割り当てる能力があります。この割り当ては、特徴や性質の抽象的な関係が基準になります。多くの状況でよく使用されるのは2値のカテゴリです。いくつかの食べ物は美味しく、他の食べ物は美味しくないとか、ある行為は道義的に正しく、他の行為は道義的に間違っているとかです。このようなカテゴリによって、すでに知っている物や行為を一般化して、まったく新しい物や行為の特徴を予測することができます。

楕円形の物をもらったとき、それが黄色い表面で、内側が柔らかく、甘くて刺激的な匂いだった場合、過去の知識から「果物」のカテゴリに属する物だと予測できるでしょう。予想が正確かどうかを確認したければ、その物を果物屋に持ち込めばよいでしょう。よく似た物が詰められたビンがあり、ラベルに「マンゴー」と書かれていれば、予想が正しかったと結論できます。そして、果物に関する知識の一般化により、マンゴーはとても美味しく、栄養も豊富であ

ると予測できます。次に、このカテゴリに関する知識を用いて、マンゴーを食べるかどうかを決定できます。情報に基づいた判断をするために、このように既知のカテゴリに不明な物を割り当てるプロセスは、分類という用語で呼ばれます。

機械学習では、分類は、定義済みのクラスにサンプルが属する確率を予測するための一連の計算方法を指します。すなわち、1通の電子メールが「スパム」のクラスに属するかどうかや、ネットワーク接続が無害であるか、またはボットネットに関係するかどうかの確率です。これらは、2値の分類問題の例です。たとえば、「スパム」と「非スパム」や、「ボットネット」と「無害」のように出力クラスが2つしかないものです。慣例により、検査している特徴を備えたサンプル(例、電子メールはスパムである)には、クラス「1」に属することを示すラベルが付けられます。一方、この特徴を持っていないサンプル(例、電子メールはスパムではない)には、クラス「0」に属することを示すラベルが付けられます。これらの1と0のクラスラベルは、それぞれ真と偽の状況と呼ばれます。

また、分類は、次のような問題でも使用できます。

- サンプルが、複数のクラスに同時に属することが可能な場合。たとえば、前に識別したマンゴーには、果物、黄色、トロピカルなどのクラスに対応するラベルを割り当てることができます。
- 2値の分類ではなく、多項の分類を実行する場合。この場合、サンプルは、3つ以上のクラスの1つに割り当てられます。たとえば、クラス1(無害)、クラス2(スパム)、またはクラス3(フィッシングエクスプロイト)のいずれかに属する電子メールサンプルを分類する場合などです。

ただし、この章では、2値の分類問題についてのみ考えます。

分類を実行するために使用するアルゴリズムは、クラシファイアと呼ばれます。2値の分類問題を解決するためのクラシファイアにはさまざまなものがあり、それぞれに長所と短所があります。この章では、最もよく使用される2つのクラシファイアの方法と原理に

ついて確認します。それらは、ロジスティック回帰とディシジョンツリーであり、scikit-learnで提供されています。

教師ありと教師なし学習

分類は、教師あり学習の一種です。アナリストは、調査している特徴について識別が終わっている(ラベル付けされた)サンプルを使用してモデルを作成します。たとえばスパムの場合、アナリストは分類モデルの作成で、スパム(真の状況)と非スパム(偽の状況)のいずれかのラベルがすでに付けられたサンプルのデータセットを使用します。次に、クラシファイアの作業で、各クラスの特徴量の性質が、ラベルのない新しいサンプルのクラスの予測にどのように利用できるかを確認します。一方、クラスタリングは教師なし学習の一種であり、あるグループのサンプルと他のグループを区別する特徴を見つける必要があります。

教師なしと教師あり手法を組み合わせて使用することは一般的ではありません。たとえば、クラスタリングは、ネットワークトラフィックベクトルを個々のグループに分けるために使用することができます。フォレンジックチームのメンバーは、疑わしいクラスタの要素を調べて、何らかの不適切なネットワーク活動を行っていないかを確認できます。不適切な活動を行っていた場合には、この活動に関係するベクトルにクラス1に属していることを示すラベルを付け(例、ボットネットトラフィック)、その他のすべてのベクトルにはクラス0のラベルを付けることができます(例、無害の活動)。ラベル付けの後、そのベクトルはモデルを作成するためにクラシファイアで使用できます。このモデルによって、ラベルなしの新しいベクトルが無害であるか、ボットネットワークに含まれているかを予測できます。

正確なモデルを作成するには、正しくサンプリングされ、ベクトル化され、ラベルが付けられた十分な量のデータを確保する必要があります。このデータは、通常、トレーニング、検証、およびテストのために2つまたは3つの独立した集合に分割されます。十分なデータがある場合には、3つの集合に分けることが望まれます。い

ずれの場合も、通常はトレーニングセットに最も多くの部分が割り当てられ、サンプル全体の70～90%を占めます。一般に、トレーニングセットを多くすると、クラシファイアによって正確なモデルが生成される確率が高くなります。ただし、モデルの精度に対する評価の信頼性を高めるために、十分な量のテストデータを常に確保する必要があります。

分類の作業は、通常、次の4つのフェーズで進めます。

1．トレーニングまたは「学習」フェーズでは、トレーニングデータにクラシファイアを適用することによりモデルを作成します。トレーニングには、サンプルデータの行列と、ラベルのベクトルの2つのファイルを用意します(1つのラベルは行列の各行に対応します)。ロジスティック回帰とディシジョンツリーアルゴリズムの両方で、いくつかのハイパーパラメータが提供されており出力されるモデルの作成方法を調整できます。
2．検証フェーズでは、検証データをモデルに適用してモデルの精度を評価したり、アルゴリズムのハイパーパラメータの設定を最適化するさまざまな手順を実施したりできます。この最適化の手法については、本章の最後にある関連資料の項のリンクを参照してください。
3．テストフェーズでは、トレーニングおよび検証のプロセスで使用されていないテストデータによってモデルの精度を評価します。アナリストは、モデルに対してテストベクトルを実行し、テストサンプルごとに予測されたクラスの割り当てと、実際のクラスの割り当てを比較します。結果が、要求される精度を満足し、成果の基準を上回れば、導入フェーズに進むことができます。これらを満たさない場合は、トレーニングフェーズに戻って、モデルを改善したり、再作成することができます。
4．導入フェーズでは、ラベル付けされていない新しいデータにモデルを適用して、クラスへの割り当てを予測します。

実際に行う場合は、さまざまなアルゴリズムやハイパーパラメータの設定を用いて、複数のモデルをトレーニングおよびテストできます。次に、モデルを比較して、最も精度が高く、コンピュータリソースを効率よく使用する最適な組み合わせ選択できます。

ご注意：テストフェーズでモデルの精度を評価するために使用する方法は、検証フェーズで一般的に使用される手法とほぼ同じものです。前述のとおり、独立した検証フェーズを実施することが望ましいことですが、実現できるのは、適切な規模のトレーニング、検証、および部分的なデータへのテストを実施できる十分な量のデータがある場合のみです。したがって、単純化のために検証フェーズについての詳しい説明は省略し、モデルの評価についてはテストの項で説明します。さらに、実習の例にあるデータセットは比較的小さいため、トレーニングとテストの手順のみを紹介します。

分類の課題

クラシファイアは、適切な条件の下であれば、多くの場合優れた結果を生成できます。ただし、いつも適切な条件とは限りません。次に例を挙げます。

- アナリストにとって、正確に分類され、ラベル付けされたデータを十分な量だけ収集することは、かなり難しい場合があります。
- 選択したサンプルが、真と偽の状況の実際の割合を厳密に反映していなければ、正確性が損なわれる可能性があります。さらに、分類が適切に機能するためには、真と偽の状況の比率が、許容できる範囲内に収まらなければなりません。ただし、アナリストが十分な量のデータを各クラスから収集できれば、たいていの場合、正確なモデルを作成できます。

ロジスティック回帰（LR）による分類

前の章では、特徴量空間の概念を紹介し、ベクトルのグループの類似性をベクトルと周囲の近傍との距離に基づいて評価する方法を

説明しました。特徴量空間は、ロジスティック回帰でも利用しますが、類似性を評価し、クラスにベクトルを割り当てる方法は多少違います。

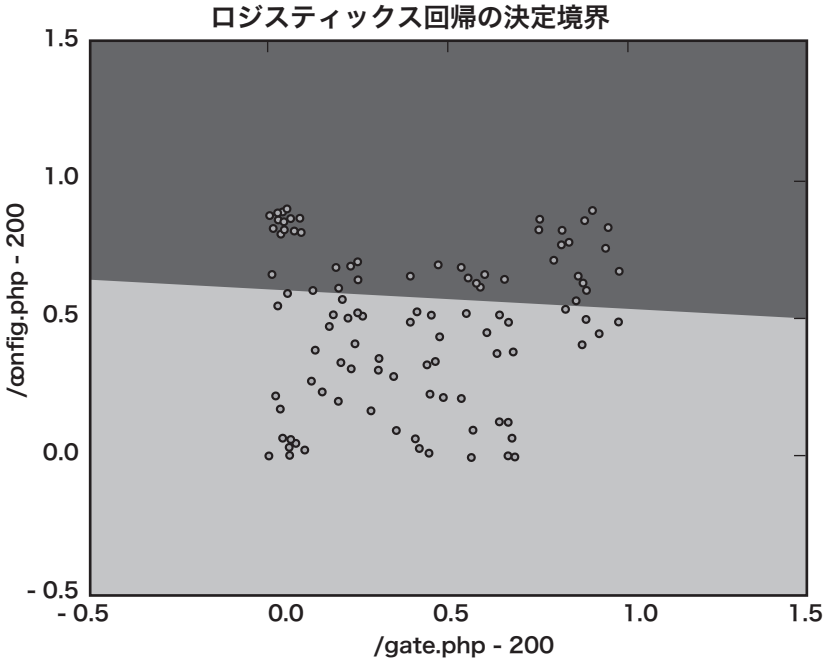


図2.1：ロジスティック回帰における決定境界

LRは、数学的には、線形クラシファイアに分類されます。線形クラシファイアとは、直線と平面を使用して、ベクトルがどちらのクラスに属しているかを判別する機能です。2値の分類では、最終的に特徴量空間を2つの領域に分けるモデルを作成します。各領域には、1つのクラスのみに属するベクトルが含まれます。このプロセスは、データのフィッティングと呼ばれます。LRでは、ある領域を他の領域から分離する直線または平面を決定境界と呼びます。2つの異なるクラスに属するベクトルを分離する決定境界の例を以下に示します。

LRには、いくつかの異なるソルバー関数があります。それらは、決定境界の位置を決定し、ベクトルをクラスに割り当てます。以下では、liblinearソルバーについて説明し、これらを実行するた

めにどのように座標降下法が使用されているかを示します。

回帰の重みの役割

回帰の重みは、特徴量とその値がクラスへのベクトルの割り当てにどの程度影響するかを決定する場合に、中心的な役割を果たします。この計算では、次のように各特徴量の値に対応する重みが掛け合わされます。

	特徴量1	特徴量2	特徴量3
特徴量の値	250	14	42
回帰の重み	0. 05	75	-6
乗算した値	12.5	1050	-252

正および負の重み値は、それぞれクラス1とクラス0の分類に寄与します。表のとおり、特徴量1の大きな正の値は、このサンプルが真のクラスに属するという予測に強く影響する可能性があることを示しています。ただし、特徴量1の影響は、小さな回帰の重みの値によってかなり減少しています。結果的に、特徴量1は、このサンプルにクラス1のラベルが付けられる可能性について少ししか影響しないことがわかります。一方、特徴量2の影響度は、大きな回帰の重みによってかなり増加しています。特徴量3の場合は、影響度が6倍に増えていますが、その予測の方向は偽のクラスへの割り当てに向いています。

実際には、1つの特徴量の値と回帰の重みの積が、サンプルのクラス割り当ての予測に与える影響は無視できる程度しかありません。これは、各ベクトルに、数百から数千にも上る特徴量の値が含まれているためです。むしろ、これらの計算を集計したものが重要になります。

LRでは、クラスを予測するために、すべての積にバイアス値を加えて合計が求められます。この合計が0以上の場合、LRはサンプルがクラス1に属すると予測します。この合計が0未満の場合、LRはサンプルがクラス0に含まれると予測します。ここで仮定している例では、ベクトルの積の総和である(12.5)+(1050)+(-252)に、

バイアス値(+5)を加算して合計の815.5が求められます。合計が0以上であるため、このサンプルはクラス1に属すると予測されます。これで、サンプルの予測済みのクラスの割り当てと、実際のクラスの割り当てを比較して、回帰の重みが正しいかどうかを確認できます。LRセッションのトレーニングフェーズのほとんどは、これらの重みを最適化する作業に費やされます。ただし、最初は、重みに初期値を指定する必要があります。このやり方は、多くの手法で使用されています。たとえば、開始時の重みの値は、乱数発生器を使用して任意に設定することもできます。結局、十分な計算時間さえあれば、クラシファイアはほとんどの場合、最適な値を算出します。

正則化とペナルティパラメータの役割

クラスタリングの章で説明したとおり、大きな値を持つ特徴量によって、モデルの結果に偏りが出る場合があります。通常、この問題は正規化によって対処されます。似たような偏りが、大きな値を持つ回帰の重みによっても発生することもあります。このため、LRアルゴリズムでは、これらの影響を緩和するために使用できるいくつかのペナルティパラメータが提供されています。

たとえば、ペナルティパラメータCを使用して、回帰の重みの範囲を圧縮することができます。これは、特徴量の値の圧縮に正規化を使用する方法とほとんど同じです。このプロセスは、正則化と呼びます。Cは、重みの値をどの程度大きくできるかを制御します。モデルに非常に高い重みを持つ範囲がある場合、トレーニングベクトルによるクラスの予測で優れた結果を残しても、テストデータに適用すると基準を下回る結果になる可能性があります。このようなモデルは、データの過剰適合と呼ばれます。このモデルの精度の差は、より積極的な正則化が必要であることを示す重要な手がかりになります。

ソルバーが、ごく一部の高い影響度を持つ特徴量に過度に集中していると思われる場合にも、正則化が役立ちます。正則化によって、制御された測定可能な方法で、ソルバーが別の特徴量を組み入

れるように強制できます。さらに、クラシファイアで回帰の重みを計算する際に、その計算へ影響し得る特徴量を制御するために正則化を使用することもできます。これはペナルティパラメータL1およびL2を使用して行います。この2つのパラメータは多少異なる動作をします。L1はしきい値を設定します。このしきい値は、LRで、予測への影響が比較的小さい特徴量を削除する際に、どの程度積極的にこれを行うかを決定します。L1に割り当てる重みが大きいほど、多くの特徴量が除外されます。一方L2は、高い相関がある特徴量のグループの影響度を最小限に抑えるため、それらを集約した影響度によって結果に誤差が生じないようにできます。

次に、scikit-learnを使用した標準的なロジスティック回帰セッションのトレーニングフェーズにおける一連の手順について説明します。

ロジスティックス回帰のトレーニングフェーズ

このフェーズの主な目的は、最適な回帰の重みを生成してデータに適合させることです。テストでは、各テストベクトルのクラスへの割り当てを予測するために、重みを適用します。次に、その結果に検証機能を適用して、予測されたクラスの割り当てが既知のクラスラベルに正確に一致しているかどうかを判定します。

手順1：データのインポートとセッションの設定

最初に、次の2つのファイルをインポートします。

1．正規化されたトレーニングサンプルの行列
2．各サンプルのクラスの割り当てを定義するラベルのベクトル

下の図のとおり、それぞれのベクトルは、特徴量空間内の点として表すことができるため、クラスの割り当てを視覚的に示すことができます。

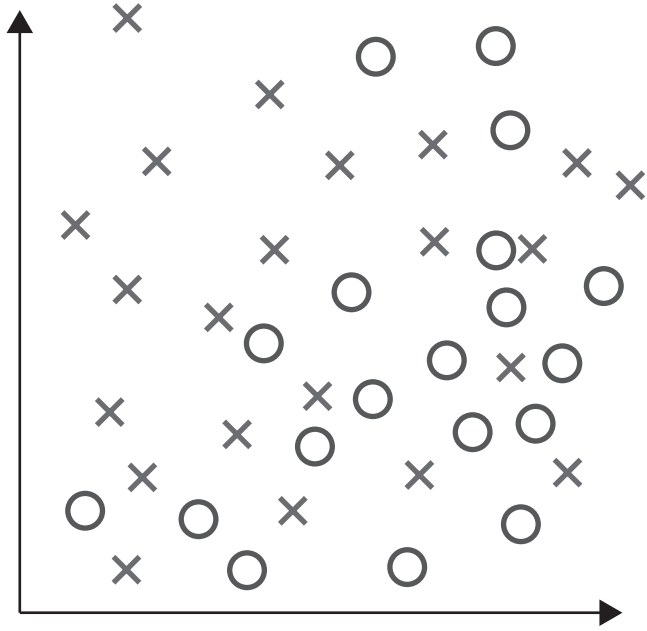


図2.2：特徴量空間におけるラベル付けされたベクトル

手順2：正則化と回帰の重みの最適化

回帰の重みの初期値を割り当てたら、尤度関数を起動します。これは、実際の真と偽の状況の数と、初期の重みを使用して予測された集計値を比較します。算出されたスコアは、重みの値ごとに正または負の調整値を計算するために使用されます。アナリストは、この調整値を特徴量ごとに制御できます。これには学習率パラメータ

を使用します。

この計算を繰り返し実行することで、回帰の重みが徐々に最適な値に近づいていきます。最適化するたびに、アナリストは、さまざまなペナルティパラメータの設定を試して、出力されたモデルを評価できます。

ただし、このようなしらみつぶしの最適化方法には、大量の計算処理と時間が必要になります。ある時点になると、モデルの精度の漸進的な改善もわずかになり、さらなる調整を行う労力に見合わなくなります。この状態になったら、アナリストはトレーニングプロセスを終了して、テストフェーズに移行できます。

手順3：クラスの予測への確率スコアの割り当て

ベクトルの分類の計算を思い出してください。すべての特徴量の値と重みの積をバイアス値とともに合計し、その合計が0以上の場合はベクトルがクラス1に割り当てられ、そうでない場合はクラス0に割り当てられます。ただし、LRは本来、あるベクトルが特定のクラスに属する確率を予測する手法です。このため、LRにはロジット関数が含まれており、以下に示すとおり、分類の結果が0から1の範囲の確率曲線に沿った点に変換されます。

点が、y=1の確率スコアに近いほど、サンプルがクラス1（真の状況）に属すると予測される確率が高くなります。同様に、点がp=0に近いほど、クラス0（偽の状況）に属すると予測される可能性が高くなります。結果が上下からp=0.5に近づいていく場合、クラスの予測についての不正確さが増加します。前に示されているとおり、決定境界はp=0.5の位置で確率曲線に沿って表されています。サンプルがこの座標上に位置する場合、両方のクラスに属する確率が等しくなり、信頼できる予測が困難になります。

手順4：ロジスティックス回帰モデルのエクスポート

これで、結果の分類モデルをエクスポートし、テストすることができます。数学的に言うと、モデルはバイアス値と、回帰の重みのベクトルで構成されています。これらを算出したら、決定境界の座

標を計算できます。特徴量が2つしかない最も単純な分類の問題では、数式が「x2 = -(m1/m2) x1 + b」の形式になります。ここで、x1とx2は特徴量の値であり、m1とm2は対応する回帰の重みで、さらにbはバイアス値です。

ただし実際の計算では、すべての回帰の重みと特徴量の値の積を合計するために、この式が展開されます。決定境界の形状は、分類に使用されている特徴量の数によって決まります。特徴量が2つある場合、決定境界は線になります。3次元の場合は、決定境界が平面になります。これよりも次元数の多い空間では、決定境界が超平面になり、特徴量が加わるたびに1つ次元が増えます。

算出されたバイアス値が0以外の場合、決定境界の原点が指定された単位数だけ移動されます。次の例では、決定境界の原点がx2軸に添って5単位上に移動されています。

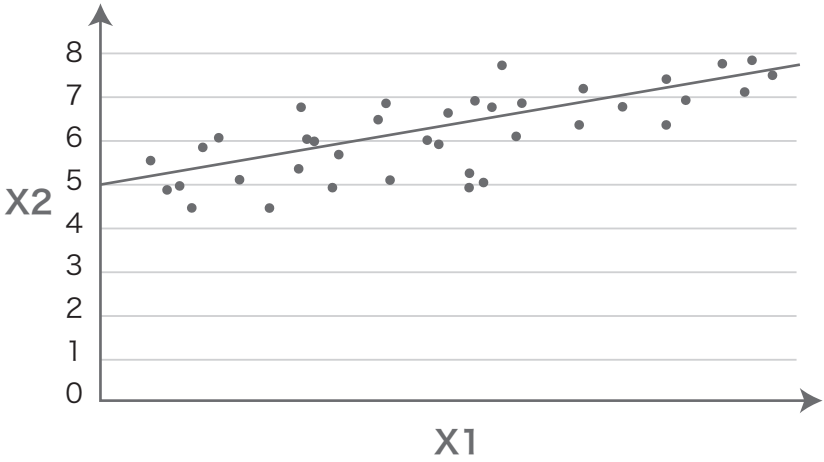


図2.3：バイアスが5のロジスティック回帰

ロジスティックス回帰のテストフェーズ

このフェーズでは、以前に使用していないデータ（テストベクトル）をモデルに適用して、その予測の精度を計測することでモデルを評価します。最も一般的な検証方法の1つに、混同行列関数を使用する方法があります。この方法では、各サンプルを順番に調べ

て、予測されたクラスの要素と実際のクラスのラベルを比較します。次に、そのサンプルの予測を次の4つのカテゴリの1つに割り当てます。

- 真陽性(TP)：クラス1に属することが正しく予測されたサンプルです。下の例では、150個のサンプルが真陽性です。
- 真陰性(TN)：クラス0に属することが正しく予測されたサンプルです。下の例では、900個のサンプルが真陰性です。
- 偽陽性(FP)：サンプルがクラス1に属するように予測されましたが、実際には0に属しています。下の例では、50個のサンプルが偽陽性です。
- 偽陰性(FN)：サンプルがクラス0に属するように予測されましたが、実際には1に属しています。下の例では、35個のサンプルが偽陰性です。

混同行列の例	モデルによる予測		
クラスの割り当て	**実際値**	**クラス0**	**クラス1**
クラス0に属するサンプル	950	900	50
クラス1に属するサンプル	185	35	150
合計サンプル数	1135	935	200
合計の真陽性	150		
合計の真陰性	900		
合計の偽陽性	50		
合計の偽陰性	35		

行列が完成したら、これを使用してさまざまな検証の測定値を計算できます。最もよく使われる測定値は、精度と再現率の2つです。

- **精度**：モデルの正しい真の予測の割合を測定します。たとえば、スパムと予測され、実際もスパムだったメッセージの割合などです。精度は、真陽性の数を真陽性と偽陽性の合計で割ることで求められます。この例の場合、精度は0.75です。
- **再現率**：モデルが真の状況を正しく分類した割合を測定します。たとえば、正しくスパムとして認識された実際のスパム

メッセージの割合などです。再現率は、「真陽性率」と呼ばれる場合もあり、真陽性の数を真陽性と偽陰性の合計で割ることで求められます。この例の場合、再現率は0.81です。

さらに、モデルの全体的な正確さや、偽の状況へ分類した際に出た誤りも測定できます。次の測定基準があります。

- **平均正確度**：真陽性と真陰性の数を合計し、その結果をサンプルの合計数で割ることで求められます。この例の場合、平均正確度の割合は約0.93です。
- **誤分類率（「誤り率」とも呼ばれます）**：1から平均正確度を引くことで求められます。この例の場合、誤り率は約0.07です。
- **偽陽性率**：モデルが偽のクラスのサンプルを真のサンプルと予測した回数を測定します。たとえば、無害のメッセージがスパムと分類された場合などです。これは、偽陽性の数を実際の偽の数で割ることで求められます。この例の場合、偽陽性率は約0.05です。
- **特異度**：モデルの偽の予測が正しい場合の回数を測定します。たとえば、無害と予測され、実際も無害だったメッセージの割合などです。これは、真陰性の数を実際の偽の数で割ることで求められます。この例の場合、特異度は約0.95です。
- **普及率**：サンプルセットにおける真のサンプルの割合を計測します。これは、実際の真の数をテストサンプルの数で割ることで求められます。この例の場合、普及率の値は0.16です。普及率によって、正確なモデルを作成する場合に、サンプリングが中心的な役割を果たす理由がわかります。普及率と世間の実情との間に差がある場合、他の計算もすべて偏ってしまい、モデルの正確さが基準を下回ってしまいます。

ここで説明しているモデルは、正確度が約0.93で、偽陽性率が約0.05と、おおむね良好な結果を示しています。ただし、上記の測定値は、モデルの偽の状況の予測精度（特異度は約0.95）が、真

の状況（再現率の0.81と、精度の0.75）よりも良いことも示しています。これは、スパム検出などの状況では許容される可能性があります。いくつかの偽陰性が特に不利にならないためです。ただし、重大なネットワークの侵入を検出するとき、このようなモデルは、必要な精度を実現するためにかなりの量の再トレーニングおよび再調整が必要になるでしょう。いつものとおり、精度のしきい値は、分類の問題の性質と、それが組織に与える影響によって決定する必要があります。

受信者操作特性曲線を使用したモデルの評価

受信者操作特性（ROC）曲線は、モデルの予測品質を評価する場合や、2値の分類問題を解決する際に、あるモデルと他のモデルの正確さを比較する場合に、便利で視覚による直感的な確認が可能な方法です。ROC曲線を描画するには、ROC空間のグラフを作成し、Y軸には真陽性率（すなわち再現率）を割り当てて、X軸には偽陽性率を割り当てます。これを可能性ある分類しきい値のそれぞれについて行います。

以下に示すとおり、モデルが正確になるほど、ROC曲線はROC空間の左上の角の近くに表示されます。点線は”分類不能”を意味し、この線の付近にある場合はモデルの予測の精度がランダムな予測とさほど変わらないことを意味します。ROC空間におけるROC曲線の説明図を次に示します。

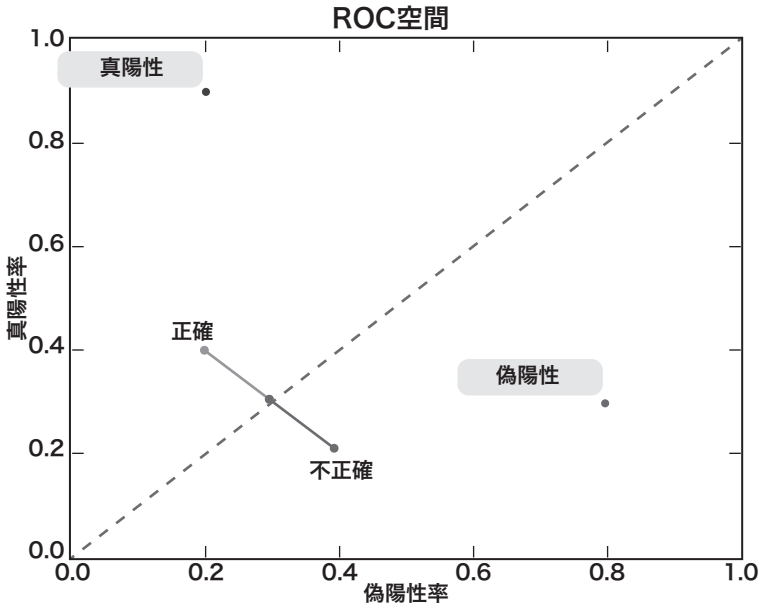


図2.4：ROC空間の説明

曲線を描画するには、最初にモデルによって正しいスコアが付けられたサンプルと正しくないスコアが付けられたサンプルの数をカウントします。これを確率しきい値ごとに行います。たとえば、下の図では、真陰性と真陽性がそれぞれ、右側と左側に棒グラフで示されています。それらの予測のそれぞれの確率スコアが、X軸に示されています。さらい、各確率スコアに対応するサンプルの数が、Y軸に示されています。右側と左側の棒グラフが重なる部分はあまりないため、モデルが、真と偽の状況を分類する正確性について良好な結果を示していることがわかります。予想どおり、ほとんどの誤りは、0.5の確率スコア付近で発生しています。モデルにとって最も難しいのが、この領域で一方と他方のクラスを区別することです。

ROC曲線を定義するために、いくつかの代表的な確率しきい値について真陽性率（再現率）と偽陽性率を計算します。真陽性率（TPR）は、Y軸の座標に割り当てます。偽陽性率（FPR）は、X軸の座標に割り当てます。ここで、0.4、0.5、および0.6の確率しきい

値に基づいて配置されたサンプルに絞ってROC曲線座標を計算します。最初に0.5のしきい値を計算します。

TPR = 真陽性の数 ÷ 真陽性と偽陰性の合計
　　= 0.92380952381（Y軸の座標）
FPR = 偽陽性の数 ÷ 実際の偽の数
　　= 0.121428571429（X軸の座標）

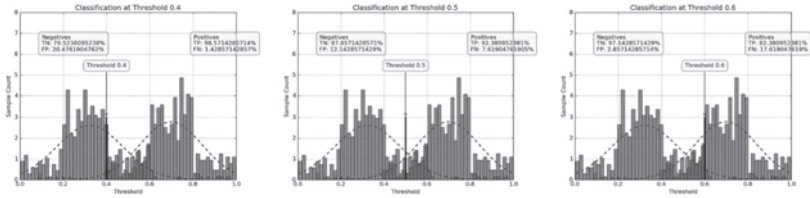


図2.5：さまざまなしきい値での分類の結果

この方法で0.4と0.6の確率しきい値も計算し、以下のROC曲線を作成します。左上の象限に曲線がある場合、モデルに、広い範囲の確率しきい値で高い精度があることを示しています。

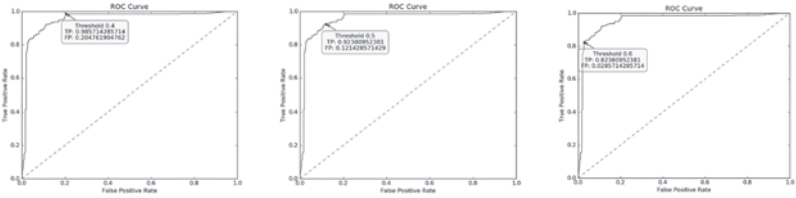


図2.6：ROC曲線

次に、精度の低いモデルに対して同じ手順を行います。まず、同じ3つの確率しきい値について、TPRとFPRを計算します。

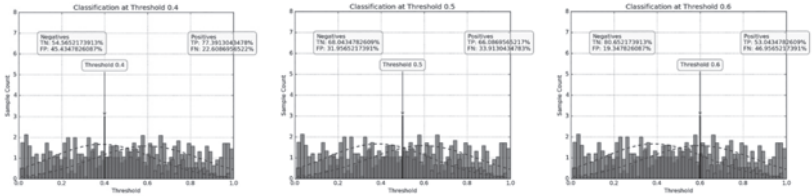


図2.7：さまざまなしきい値での分類の結果

結果のROC曲線を参照してください。

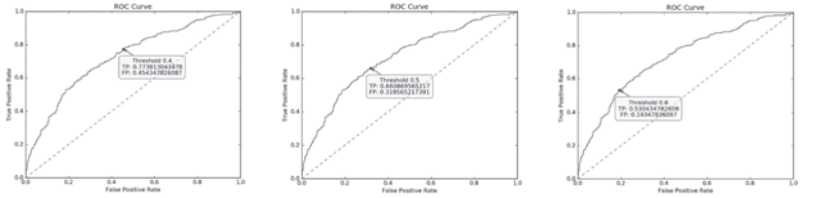


図2.8：ROC曲線

ROC曲線が、「分類不能」の線のかなり近くに位置していることがわかります。これは、このモデルがその範囲の確率しきい値において精度が低いことを示しています。

ロジスティックス回帰の欠点と制限事項

scikit-learnのロジスティックス回帰アルゴリズムは、次の制限の下では効率性に優れ、良好な結果を示します。

- 基になるデータは、本質的に線形の分類で対応できる必要があります。これは、具体的には、直線や平面で特徴量空間を分離する決定境界を使用して、ベクトルを正確に分類できることが前提になります。データセットが、この方法で線形的に分離できない場合は、特徴量の表現に、より複雑な方法を使用するか、別の分類アルゴリズムを使用する必要があります。

- LRは、データセットに多くの異常値がある場合、特徴量に極端に大きな値が含まれている場合、およびいくつかの特徴量に高い相関性のある場合には、フィッティングの精度が低下する場合があります。正規化と正則化を使えば、この影響をいくぶん緩和できる可能性があります。

ディシジョンツリーによる分類

ディシジョンツリー(DT)アルゴリズムは、ベクトルが、あるクラスと別のクラスのどちらに属するかを判定します。このアルゴリズムには、クラスの予測に至るまでの連続する「if-then-else」の判定規則が定義されています。選択するDTアルゴリズムの種類は、クラスラベルがカテゴリ値か連続値かによって異なります。

- ネットワーク接続がボットネットと関係しているどうかを予測する場合など、クラスラベルがカテゴリ値の場合、DT分類を利用します。
- 新しい製品の最適な販売価格を予測する場合など、クラスラベルが連続値の場合、DT回帰を利用します。

scikit-learnが提供するアルゴリズムはCARTディシジョンツリーであり、分類と回帰ツリーの両方を生成できます。この章では、DT分類を中心に説明します。この手法が、最も発生する可能性が高いネットワークセキュリティの問題を解決するのに適しているためです。

数学的に、DTは非線形クラシファイアに分類されます。つまり、LRと違い、DTは直線や平面の決定境界を作成しません。その代わりに、特徴量空間を矩形で区分けします。この矩形には、それぞれ1つのベクトルを含めることができます。以下では、この違いがフィッティングプロセスに重要な意味を持ち、結果のモデル精度にどのように影響するかを解説します。

ディシジョンツリーの用語

ディシジョンツリー(決定木)という名前は、ルート(根)、ブランチ(枝)、そしてリーフ(葉)を使用してクラスを予測することに由来しています。有害なURLを特定するための仮説のディシジョンツリーの例を考えてみましょう。陽性のケースは、エクスプロイトに関連付けられているURLとそうではないURLを示しています。

慣例として、ツリーはルートを一番上にしたトップダウンで構築されており、ルートにはすべてのトレーニングサンプルが含まれています。このケースでは、有害なURLと良性のURLが混在しています。私達の目標は、これらのサンプルをそれぞれの特徴や特徴値によって徐々に「純粋な」サブセットまで分岐させることです。この作業を、1つのクラスにのみ属するURLのサブセット(1つまたは複数)にたどり着くか、あるいはハイパーパラメータ設定によってモデル構築プロセスが終了するまで続けます。

まず、「ドメイン年齢」という特徴によってトレーニングセットを2本のブランチに分岐させます。分岐には、「7日未満」と「7日超」の2つの特徴値を使用します。最初の分岐条件に一致するサンプルのサブセットは「毎日の訪問者数25万人超」というノードにコピーします。同様に、他方のサブセットは「Alexaのトップサイトリストに掲載」というノードにコピーします。ノードはまだ純粋ではないため、各ノードにはクラス0とクラス1に属するサンプルが混在しています。そのため、ブランチをさらに増やす必要があります。

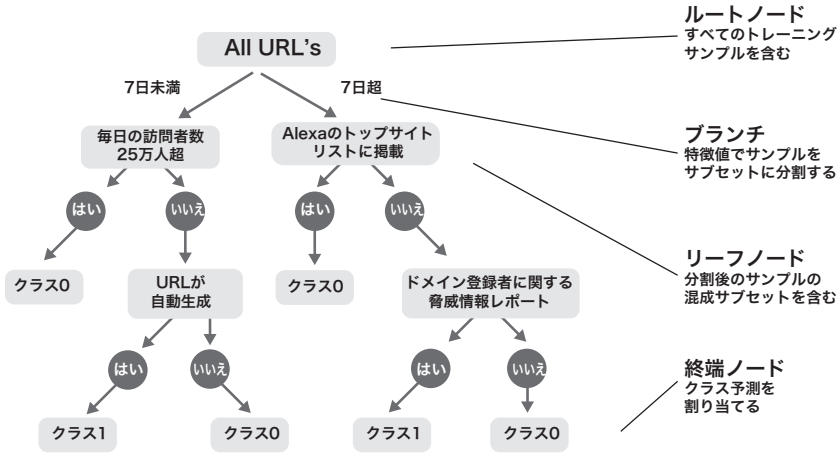


図2.9：ディシジョンツリーの例と用語

続けて、「URLが自動生成」および「ドメイン登録者に関する脅威情報レポート」の特徴値で分岐を続けます。左側のブランチ（7日未満）では、URLが次のように分類されると予測できます。

- 有害（クラス1）：毎日の訪問者数が25万人未満でURLが自動生成された場合
- 良性（クラス0）：毎日の訪問者数が25万人未満であるがURLが自動生成されなかった場合

右側のブランチ（7日超）では、URLが次のように分類されると予測できます。

- 有害（クラス1）：Alexaのトップサイトリストに掲載されておらず、ドメイン登録者に関する脅威情報レポートが発行されている場合
- 良性（クラス0）：
- Alexaのトップサイトリストに掲載されている、または
- Alexaのトップサイトリストに掲載されていないが、ドメイン登録者に関する脅威情報レポートが発行されていない場合

ディシジョンツリーのトレーニングプロセス

ディシジョンツリー（DT）クラシファイアは、各ブランチを作成

するために使用する特徴と特徴値ををどのように決定しているのでしょうか。また、ノードの純度はどのような意味を持つのでしょうか。では、DTクラシファイアが分岐点を評価してツリーを構築する際のプロセスをステップバイステップで見てみましょう。トレーニングセットに500件のサンプルがあるという仮説ケースを考えます。

1．ディシジョンツリーは、500件のベクトルをすべて調べ、評価対象となりうる最初の特徴を潜在的な分岐ポイントとして選択します。このポイントは分岐変数とも呼ばれます。
2．次に、ディシジョンツリーはこの特徴に関連している値の範囲を調べ、最初の分岐値を選びます。この例では、特徴値は1～5の範囲になります。したがって、ディシジョンツリーはまず、1～2の範囲内の分岐値を選択します。この例では、ディシジョンツリーは分岐値を1.5に設定しています。

特徴値の範囲	1	â	2	â	3	â	4	â	5
分岐変数の例		1.5		2.5		3.5		4.5	

3．ディシジョンツリーは、特徴値が1.5未満のサンプルを子ノード#1にコピーし、1.5以上のサンプルを子ノード#2にコピーします。この時点で、子ノード#1には300件、子ノード#2には200件の混成クラスサンプルが存在します。
4．ディシジョンツリーは、各子ノード(リーフ)のジニ不純度スコアを計算することで、どのくらい不純度が下がったかを評価します。次に、2つのスコアを合算して、分岐候補のベネフィットスコアを生成します。(ベネフィットスコアは、不純度ではなく「エントロピー」を使用して計算することもできます。)
5．ディシジョンツリーは次の分岐値を選択してプロセスを繰り返します。この例では、次の分岐値は2.5です。そしてもう一度、それぞれの候補である子ノードのジニ不純度スコアとベネフィットスコアを計算します。特徴のジニ不純度とベネフィットスコアがすべて計算されるまで、他の分岐値について同じプロセスが繰り返されます。

特徴値の範囲	1	â	2	â	3	â	4	â	5
分岐変数の例		1.5		**2.5**		3.5		4.5	

6．ディシジョンツリーは、特徴セットから次の特徴を選択して、ステップ2～5を繰り返し、すべての分岐値を再帰的に評価することで、新しい候補ノードセットのジニ不純度スコアとベネフィットスコアを生成します。このプロセスは、すべての特徴と分岐値を評価するまで続きます。

7．ディシジョンツリーは、全体の中で最も高いベネフィット値を生成した特徴と分岐値の組合せを選び、それを使用してツリーに新しいブランチとリーフを追加します。このレベルでは、生成されるそれぞれのリーフでは、クラス0とクラス1のサンプルがまだ混在しています。そこで、各クラスのサンプルの比率を使用して、確率スコアを計算することができます。たとえば、リーフに100件のサンプルがあり、そのうちの70件がクラス0、30件がクラス1に属しているとします。この場合、この分岐によって生成される新しいサンプルがクラス1に属する可能性は30%（30/100）クラス0に属する可能性は70%（70/100）となります。

8．ディシジョンツリーは、新しい2つのリーフのそれぞれに対して同じプロセスを開始し、これをツリーが飽和する（つまり、すべてのノードが完全に純粋になる）か、もしくは、ディシジョンツリーがいずれかのハイパーパラメータ設定に関連付けられている停止条件に遭遇するまで続行します。

9．アナリストは、モデルの精度を上げることなく単に計算を複雑にするだけの余分なブランチを必要に応じて刈り込むこともできます。

以下の例は、ボットネットのC&C（コマンド＆コントロール）システムであることを検出するディシジョンツリーの最初の4つの分岐を示しています。各ノードの特徴名、サンプル数、ジニ不純度スコア、分岐値、およびクラス予測に注目してください。本章では、完全なツリーと、それを生成するためのプロセスについて後ほど詳

しく解説します。

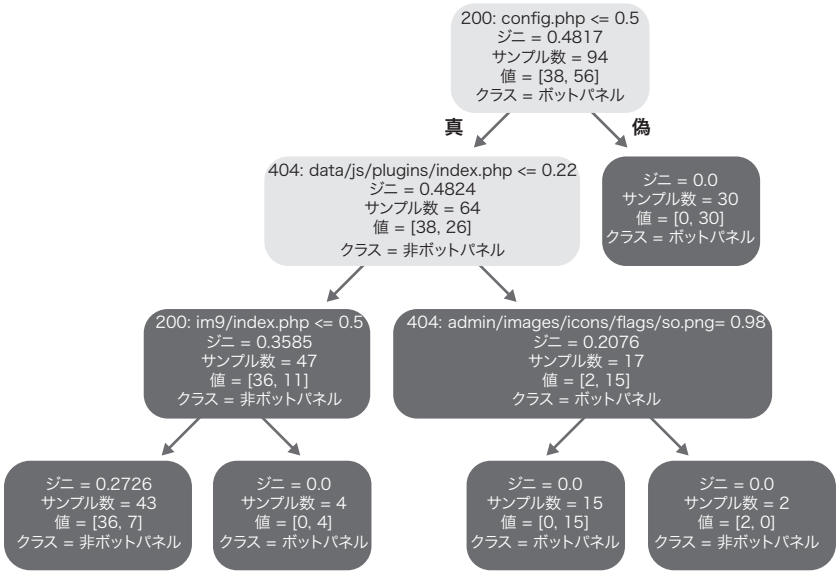


図2.10： ディシジョンツリーの詳細

ディシジョンツリーによる分類

では、scikit-learnを例として、典型的なディシジョンツリーによる分類セッションのトレーニング、テスト、および展開フェーズを構成するステップシーケンスと手法を見てみましょう。

手順1：データのインポートとセッションの設定

ここでも、アナリストは2つのファイルのインポートから開始します。

1．トレーニングサンプルのマトリクス
2．各サンプルのクラスの割り当てを定義するラベルのベクトル

ステップ2：ディシジョンツリーの生成とカスタマイズ

LRモデルの開発では、さまざまな正規化パラメータを適用して回帰の重みを最適化することに最も多くの時間を費やすことはすで

に説明しました。ディシジョンツリートレーニングでは、データを過剰に適合することなく、求められる精度のしきい値を満足できる効率的なモデルを生成することを目的として、アルゴリズムの処理時間の大半は分岐ポイントの最適化に費やされます。

多くの場合、scikit-learnのDTクラシファイアは、デフォルトのハイパーパラメータ設定を修正しなくても、この目的を達成することができます。このようなケースでは、アナリストは単純にソルバーを呼び出すだけで、アルゴリズムがディシジョンツリーと関連するジニ不純度スコアおよびROC曲線のセットを生成してくれます。一方、多くの特徴と膨大な数の異常値が存在するような場合には、ブランチを過剰に生成してデータを「過剰適合」してしまうというDTクラシファイアの傾向に対して、ハイパーパラメータ設定の修正で対応しなければならないこともあります。トレーニングでは、生成されるモデルは高い精度を提供するように見えます。しかし、データのテストに応用すると、精度スコアはかなり低くなります。アナリストはこれを一般化の失敗と呼びます。また、データを過剰適合するモデルは計算効率も悪く、展開にも手が掛かります。前述のように、アナリストは次のようなハイパーパラメータを使用して停止条件を適用することによって、ブランチの生成を制限することができます。

- max_depth. ディシジョンツリーの構築で生成できるブランチとリーフの最大数を指定します。
- min_samples_split. 分岐の対象となり得るためにノードに含まれるべきサンプルの最少数を指定します。
- min_samples_leaf. 子ノード（リーフ）を作成するために必要なサンプルの最少数を指定します。
- max_leaf_nodes. 作成できるリーフの合計数を指定します。

さらにアナリストは、分岐プロセスの各特徴に影響するハイパーパラメータ設定を使用して正規化を適用することもできます。例を示します。

- feature_importances. このパラメータは特定の特徴または特徴グループに適用できます。この値が高いほど、特徴が分割変数として使用されやすくなります。
- n_features. ディシジョンツリーを構築するために使用できる特徴の数を指定します。

クラス予測への、確率スコアの割り当て

ディシジョンツリーアルゴリズムは、すべてのリーフにおけるすべてのクラス予測で、そのリーフに含まれる陽性と陰性のサンプルの比率に基づいて、確率スコアを生成します。このスコアは、どちらかのクラスのサンプル数をそのリーフの合計サンプル数で割ることによって求められます。たとえば、クラス0のサンプルが15件、クラス1のサンプルが5件含むリーフがあるとします。このリーフのサンプルがクラス1に属する可能性は、陽性サンプル数(5)を合計サンプル数(20)で割った結果である20%となります。同様に、ツリーのリーフにコピーされるサンプルがクラス0に属する可能性は80%になります。

ステップ3:テストと展開

ディシジョンツリーモデルが構築されると、ロジスティック回帰の説明で示したのと同じテストと検証が行われます。そしてモデルが検証されると、新しい、ラベル付けされていないデータの分類用に展開されます。

ディシジョンツリーの落とし穴と制約

scikit-learnのディシジョンツリーアルゴリズムは、デフォルトのハイパーパラメータ設定でも効果的に作用し、サンプルデータの事前準備に必要な労力を少なくしてくれます。また、すべての特徴ではなく、分類に必要なサブセットのみを使用する、非常に効率的なモデルを生成します。これに対してLRモデルには、正規化によって意図的に除外しない限り、すべての特徴が含まれてしまいます。

ただし、ディシジョンツリーには特有のエラーや制約があります。

- 他のあらゆるDTクラシファイアと同様に、scikit-learnのDTクラシファイアも、データを過剰適合して実際のデータテストでは精度が低くなる、過剰に複雑なツリーを生成してしまうことがあります。一般に、特徴が多いほど、過剰適合も起こりやすくなります。
- 分岐ポイントの決定は、ディシジョンツリー全体として何が最適であるかではなく、親ノードと子ノード(リーフ)との間の「ローカルな」最適化に基づいて行われます。その結果、ディシジョンツリー全体が最適であるかどうかを保証する方法はありません。
- ディシジョンツリーは不安定になることもあります。サンプルデータがわずかに変動するだけで、まったく異なるディシジョンツリーが生成されることもあります。

この問題に対処するには、トレーニングデータのサブセットをランダムに選んでいくつかのツリーを作成します。そして、各ディシジョンツリーにおいて特定のサンプルがどのクラスに属するかを「投票」します。最も多くの票を得た予測が勝ち残りとなります。scikit-learnでは、ランダムフォレストアルゴリズムを用いてこれを実現しています。

現実世界のセキュリティ脅威への分類の適用

2013年6月5日、Microsoftは、技術パートナー、警察当局、そして大手金融機関との「共同作戦」の成果として、1462台のCitadelボットネットC&C(コマンド＆コントロール)システムと、これらボットネットの管理下にあった数百万台のコンピュータとの通信を切断したことを発表しました。当時、Citadelマルウェアは約500万人もの人に被害を及ぼし、個人や法人の被害は5億ドル以上に上っていました。

Microsoftと技術パートナーがC&Cシステムを特定するために利

用した犯罪科学手法は公開されていません。その代わりに、インターネット上に今でも存在するボットネットC&Cシステムを検出するために応用できる分類の手法を紹介します。以下では、まずディシジョンツリーを用いて、次にロジスティック回帰を用いて、これらの手法を2回説明します。どちらのケースでも、13件の異なるボットネットパネルに関連していることが判っている4,789件のWebサイトオフセットに対してHTTP要求を発行し、その結果として生成されるデータでモデルをトレーニングします。

次に、これらの要求によって返されたファイルを応答コードに基づいてグループ分けし、ssdeepプログラムを使用して各ファイルのファジーハッシュを生成します。ファジーハッシュにより、ファイルの内容を直接調べなくても、1つのファイルと他のファイルとの類似性を評価することができます。類似性の比較値は、0～100%の範囲になります。便宜上、これらの値を0～1の範囲で正規化します。

トレーニングでは、ディシジョンツリーとLRのアルゴリズムは、すべての特徴とそれらのハッシュ値を調べて、C&Cサーバーと良性サーバーそれぞれからのオフセットとの類似性を評価します。展開時には、生成されたモデルがこれらの比較結果に基づいて、特定のオフセットがボットネットC&Cシステムに関連しているかどうかを予測します。

解析を始める前に、特徴の抽出とベクトル化を行う必要があります。生成されるベクトルには、それぞれ357,947件の特徴が含まれます。各特徴は、応答コードと関連するハッシュ値で構成されます。サンプルベクトルとラベルベクトルが作成されたら、トレーニングプロセスを開始します。

モデルのトレーニングとテストが完了したら、ラベル付けされていないWebサーバーオフセットサンプルに対して展開して分類を実施します。特徴を排除するための正規化が実施されていない限り、LRモデルには、サンプルデータセットに含まれている357,947件の特徴すべてに対して、計算されたバイアス値と回帰の重みが含まれます。これに対して、ディシジョンツリーは8件の特徴だけで

ボットネットパネルを性格に検出できるため、LRモデルよりもはるかに小さく、計算効率も高くなります。

説明を簡素化するため、別々のトレーニングおよびテストフェーズを1つのトレーニング/テスト手順にまとめ、サンプルの70%はトレーニング、残りの30%はテストに使用します。

注意：これらのサンプルはすべて、2016年8月にラスベガスで開催された国際会議「ブラックハット」でサイランスが発表したツール「ID Panel」を用いて生成されています。ID Panelパネルを試してみたい、あるいは下記の分類手順を実施してみたい場合は、https://www.cylance.com/intro-to-aiから必要な指示、アプリケーション、およびデータファイルをダウンロードしてください。

サンプルデータの収集と準備

ステップ1：データの収集

モデルのトレーニングを開始する前に、適切なサンプルデータセットを収集しなければなりません。この作業はすでに済んでおり、すべてのC&Cサーバーと良性Webサーバーに対してHTTP要求を発行し、応答コードとファイルを統合してpre-vectors.jsonファイルに保存してあります。したがって、データファイルに新しいオフセットを追加したい場合を除いて、サーバーに接続する必要はありません。サンプルデータセットはすでに収集されているため、最初からpython create_prevectors.pyを（スクリプトでは実際にはHTTP要求を開始しないという前提）実行します。

```
bwall@highwind:~/code/IDPanel$ python create_prevectors.py
Making 0 requests
bwall@highwind:~/code/IDPanel$
```

もし、要求が発行される場合には、プリベクトルデータが別の場所に保存されていることが考えられます。

追加のサーバーからのオフセットを組み込みたい場合には、次のコマンドを実行して、c2_labelsディレクトリに追加してください。

```
bwall@highwind:~/code/IDPanel$ echo "https://bwall.github.io/" >> c2_labels/not_panel.txt
bwall@highwind:~/code/IDPanel$
```

その後、python create_prevectors.pyを再び実行して、新たに追加したサイトをスキャンします。

```
bwall@highwind:~/code/IDPanel$ python create_prevectors.py
```

これによって、既存のサンプルを組み立てるために使用したのと同じ4,789件の要求が生成されます。これらの要求の最初の109件は、以下のサンプルコードに示されています。

```
Requesting https://bwall.github.io/img/flags/zm.png
94 completed out of 4789
Requesting https://bwall.github.io/theme/js/tinymce/jscripts/tiny_mce/themes/advanced/skins/highcontrast/content.css
95 completed out of 4789
Requesting https://bwall.github.io/mods/mod_stats.php
96 completed out of 4789
Requesting https://bwall.github.io/rdm/img/malawi.png
97 completed out of 4789
Requesting https://bwall.github.io/aata/images/form_button.png
98 completed out of 4789
Requesting https://bwall.github.io/resources/scripts/global.php
100 completed out of 4789
Requesting https://bwall.github.io/admin/images/icons/flags/mh.png
101 completed out of 4789
Requesting https://bwall.github.io/get-functions.php
102 completed out of 4789
Requesting https://bwall.github.io/includes/Smarty-3.1.8/libs/sysplugins/smarty_internal_compile_section.php
103 completed out of 4789
Requesting https://bwall.github.io/imgs/flags/tv.png
104 completed out of 4789
105 completed out of 4789
Requesting https://bwall.github.io/admin/images/icons/flags/lu.png
Requesting https://bwall.github.io/theme/js/contextMenu/demo/disabled-callback.html
106 completed out of 4789
Requesting https://bwall.github.io/data/images/lang/32/tg.png
107 completed out of 4789
Requesting https://bwall.github.io/data/images/lang/16/bo.png
108 completed out of 4789
Requesting https://bwall.github.io/img/flags/vc.gif
109 completed out of 4789
```

以下のサンプルコードは、最後の要求（4,776～4,789）です。

```
Requesting https://bwall.github.io/includes/Smarty-3.1.8/libs/sysplugins/smarty_internal_compile_for.php
4776 completed out of 4789
4777 completed out of 4789
Requesting https://bwall.github.io/includes/design/images/modules/module_bitkinex.png
Requesting https://bwall.github.io/data/images/lang/32/sc.png
4778 completed out of 4789
Requesting https://bwall.github.io/imgs/flags/cf.png
4779 completed out of 4789
Requesting https://bwall.github.io/img/flags/aw.png
4780 completed out of 4789
4781 completed out of 4789
4782 completed out of 4789
4783 completed out of 4789
4784 completed out of 4789
4785 completed out of 4789
4786 completed out of 4789
4787 completed out of 4789
4788 completed out of 4789
4789 completed out of 4789
bwall@highwind:~/code/IDPanel$
```

手順2：特徴量の抽出

サンプルを収集したところで、ベクトルを構築するための特徴を次のコマンドで抽出します。

```
bwall@highwind:~/code/IDPanel$ python extract_features_from_prevectors.py
```

下記のように、14件の異なるラベルに関連付けられた357,947件の特徴が生成されます。これらのうちの13件のラベルは、既知のボットネットパネルサイトに関連付けられています。14番目のラベルは、こちらで用意したクリーンサーバーセットを表しています。

```
bwall@highwind:~/code/IDPanel$ python extract_features_from_prevectors.py
Loaded 421432 prevectors
Extracted 357947 features
Features cover 4789 requests
Vectors cover 14 labels
bwall@highwind:~/code/IDPanel$
```

ステップ3：ベクトル化

では次に、以下のコードを使用して、いま作成した特徴ファイルからベクトルを組み立てます。

```
bwall@highwind:~/code/IDPanel$ python vectorize_with_raw_features.py
Loading prevectors
```

この結果として、以下を含むベクトルのマトリクスが生成されます：

```
Vector for http://o-z-o.ru/Panelinator/Pony/includes/design/adsfsdf/ completed
Vector for http://www.domedicosdelnorte.com/live/Panel/ completed
Vector for http://0rgil.com/blessedoldpony/ completed
Vector for http://nellisrealestate.com/wp-includes/images/okk/panelnew/ completed
Vector for http://93.190.68.229/ completed
Vector for http://rustywood.ca/php/ completed
Vector for http://cgpdskiteam.org/wp-admin/includes/sj/ completed
Vector for http://213.155.31.195/ completed
Vector for http://drunkensheriff.glupoty.com/wp-admin/ completed
Vector for http://fausin.honor.es/ completed
Vector for http://bilimteknoloji.info/wplog/ completed
Vector for http://datarecoveryoxfordshire.co.uk/ven/ completed
Vector for http://www.osoperfume.com/errors/sols/ completed
Vector for http://45df36.de/df/ completed
Vector for http://y99978em.bget.ru/ completed
Vector for http://j906793s.bget.ru/NeT/ completed
Vector for http://189.1.162.151/tmp/ completed
Vector for http://www.sec.parmankulov.com/ completed
Vector for http://petroyeda.com/lege/server/ completed
Vector for http://datarecoveryoxfordshire.co.uk/chap/ completed
Vector for http://195.117.230.152/Panel/ completed
Vector for http://xram.onlinewebshop.net/XRAM/ completed
Vector for http://www.pardox.site11.com/Web%20Panel/ completed
Vector for http://www.thefashionspot.com/ completed
Vector for http://xbledge.com/dendroid/ completed
bwall@highwind:~/code/IDPanel$
```

ラベルのベクトルとラベルマトリクスが作成されたところで、モデルのトレーニングプロセスに移ります。

セッションワークフロー：ディシジョンツリーによる分類

前述のように、scikit-learnのDTクラシファイアは、通常はデフォルトのハイパーパラメータ設定でも優秀なモデルを構築します。以下では、–h引数を使用して、ディシジョンツリースクリプトのtrain_model.pyのハイパーパラメータ設定の一覧を表示しています。

```
bwall@highwind:~/code/IDPanel$ python train_model.py -h
usage: train_model.py [-h] [-c {gini,entropy}] [-s {random,best}]
                      [-m {auto,sqrt,log2}] [-d MAX_DEPTH]
                      [-S MIN_SAMPLES_SPLIT] [-l MIN_SAMPLES_LEAF]
                      [-w MIN_WEIGHT_FRACTION_LEAF] [-n MAX_LEAF_NODES]

Train Decision Tree

optional arguments:
  -h, --help            show this help message and exit
  -c {gini,entropy}, --criterion {gini,entropy}
  -s {random,best}, --splitter {random,best}
  -m {auto,sqrt,log2}, --max-features {auto,sqrt,log2}
  -d MAX_DEPTH, --max-depth MAX_DEPTH
  -S MIN_SAMPLES_SPLIT, --min-samples-split MIN_SAMPLES_SPLIT
  -l MIN_SAMPLES_LEAF, --min-samples-leaf MIN_SAMPLES_LEAF
  -w MIN_WEIGHT_FRACTION_LEAF, --min-weight-fraction-leaf MIN_WEIGHT_FRACTION_LEAF
  -n MAX_LEAF_NODES, --max-leaf-nodes MAX_LEAF_NODES
bwall@highwind:~/code/IDPanel$
```

デフォルト設定で問題なければ、python train_model.pyと入力して実行し、モデルのトレーニングを直ちに開始できます。

```
bwall@highwind:~/code/IDPanel$ python train_model.py
```

プロセスが完了すると、アルゴリズムはbot_model.mdlという名前のモデルファイルを出力し、その精度を評価するためのしきい値のグラフとROC曲線を表示します。この結果から判るように、このモデルは可能性しきい値の範囲全体に渡ってC&Cサイトを正確に分類しています。モデルは少なくとも部分的にはランダムな値から生成されているため、実際のグラフは少し異なる場合があります。

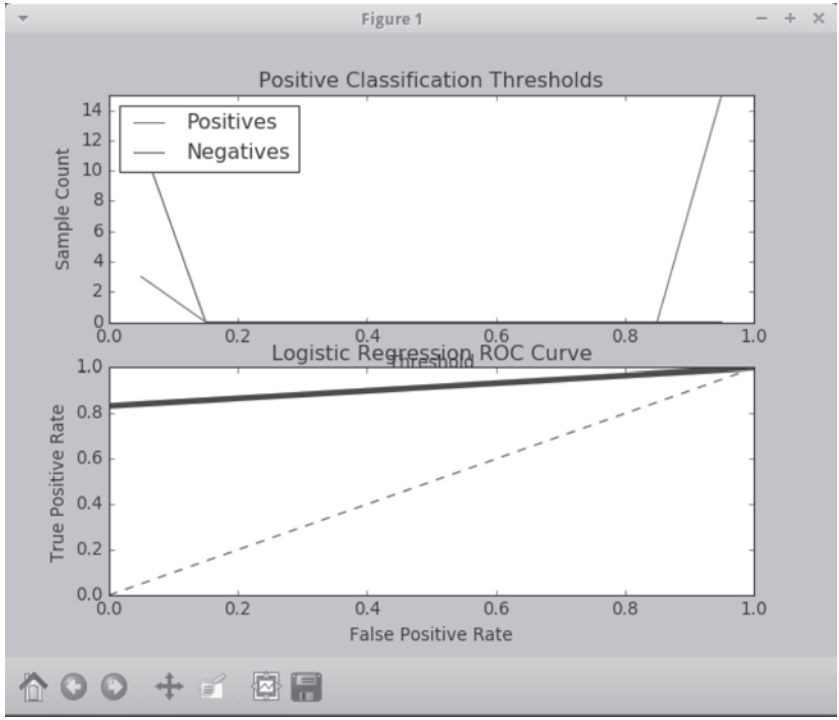


図2.11：分類のしきい値とROC曲線

グラフを閉じたら、ディシジョンツリーをグラフィックファイルとしてエクスポートして、その構造を表示することができます。こ

こでのグラフィックファイル名はtree.pngです。では、決定プロセスをルートから視覚的にトレースしてみましょう。

ルートには、クラスが混在した94件のサンプルがあります。最初の分岐変数はconfig.php特徴であり、分岐値は「0.5以下」です。この条件に合致する（条件が真の）サンプルは64件あります。これらのサンプルは左側の子ノード404にコピーされます。ジニ不純度スコアの0.4824は、このノードには両方のクラスのサンプルが含まれているため、さらなる分岐が必要であることを示しています。

一方、ルートのサンプルのうちの30件は分岐条件に合致しません（条件は偽です）。

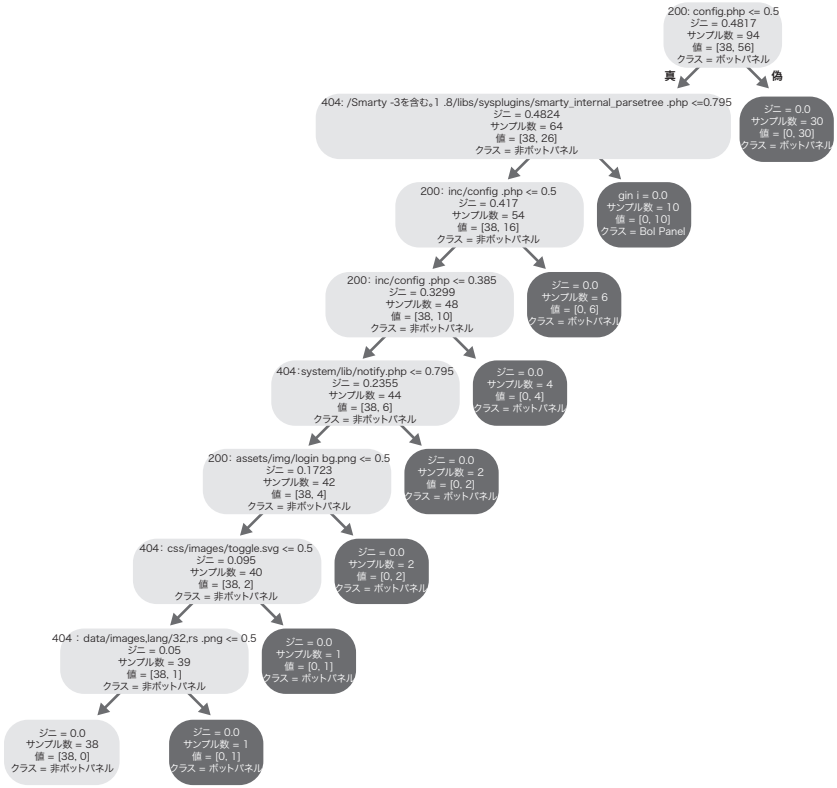


図2.12：ディシジョンツリーの詳細

したがって、これらのサンプルは右側の子ノードにコピーされま

す。このノードのジニスコアは0.0であり、ノードの純度が完全であることを示しています。サンプル数を見ると、クラス0（ボットパネルではない）に属するサンプルは0件、クラス1（ボットパネルが存在する）に属するサンプルは30件となっています。つまり、これ以上の処理が不要な最初の端末ノードが見つかったことになります。

このツリーの処理をブランチごとに下方に続行し、最終的な端末ノードセットを生成する8番目の特徴まで進みます。左側の子ノードには、良性サーバーからの38件のサンプルが含まれます。右側の子ノードには、C&Cシステムからの1件のサンプルが含まれます。すべてのサンプルの分類が完了し、ディシジョンツリーが完成したところで、モデルファイルを展開用にエクスポートします。モデルファイル名はbot_model.mdlとします。

当初の特徴の数は357,947件もありましたが、このモデルではわずか8件の特徴で高い精度でサンプルを分類できるため、極めて効率的です。さらに、

- 分類するために新しいラベル付けされていないサンプルを収集する場合は、これら8件の特徴に関連したHTTP要求のみに絞り込んで発行することができます。
- また、元のトレーニングセットから抽出した残りの357,939件の特徴は無視できるため、特徴の抽出プロセスも非常に高速かつ効率的に行えます。

では、このモデルを実際のWebサイトに適用して、C&Cシステムであるかどうかの分類を行ってみましょう。classify_panel.pyと入力し、サーバーのURLとモデルファイル名を引数として指定します。

警告：マルウェアの感染被害に対して予防策を講じていない限り、実際のインターネット上のC&Cサーバーに対してこのモデルを試すことはしないでください。

ここでも、HTTP要求を発行して応答コードとオフセットを収集します。ただし、今回は8件の特徴しか分類しませんので、必要な

要求も8件のみです。

```
bwall@highwind:~/code/IDPanel$ python classify_panel.py bot_model.mdl https://bwall.github.io
Identifying panels we can actually reach
We can reach https://bwall.github.io/
Making 8 total requests to 1 servers
https://bwall.github.io/, Not Panel, [ 1.  0.]
bwall@highwind:~/code/IDPanel$
```

次に、前回に使用した特徴抽出プロセスを繰り返し、結果をベクトル化して、モデルを通してベクトルを実行します。

前回、このモデルは、8件の要求だけでWebサーバーが良性であることを判断できました。今回はC&Cシステムであることが判っているサーバーでもう一度試してみましょう。

```
bwall@highwind:~/code/IDPanel$ python classify_panel.py bot_model.mdl http://112.213.88.68/Panel/
Identifying panels we can actually reach
We can reach http://112.213.88.68/Panel/
Making 8 total requests to 1 servers
http://112.213.88.68/Panel/, Botnet Panel, [ 0.  1.]
bwall@highwind:~/code/IDPanel$
```

同じく8件の要求を発行し、結果を抽出してベクトル化してから、モデルを通してベクトルを実行します。

モデルはサーバーをC&Cシステムとして正しく分類しました。

セッションワークフロー：ロジスティック回帰による分類

いま実行した分類手順は、元々はディシジョンツリー手法を使用してデザインされたものです。同じ分類の問題を、ロジスティック回帰を使用して解決することもできます。LRモデルをトレーニングするために、train_lr_model.pyスクリプトを使用し、用意されているコマンドライン引数を使用してハイパーパラメータ設定を調整します。

以下の例では、–h引数を使用して、train_lr_model.pyで使用できるデフォルトとオプションの引数を表示しています。

```
bwall@highwind:~/code/IDPanel$ python train_lr_model.py -h
usage: train_lr_model.py [-h] [-p {l1,l2}] [-d] [-C C] [-f]
                         [-i INTERCEPT_SCALING] [-m MAX_ITER]
                         [-s {newton-cg,lbfgs,liblinear,sag}] [-t TOL]

Train Logistic Regression Model

optional arguments:
  -h, --help            show this help message and exit
  -p {l1,l2}, --penalty {l1,l2}
  -d, --dual
  -C C
  -f, --fit-intercept
  -i INTERCEPT_SCALING, --intercept-scaling INTERCEPT_SCALING
  -m MAX_ITER, --max-iter MAX_ITER
  -s {newton-cg,lbfgs,liblinear,sag}, --solver {newton-cg,lbfgs,liblinear,sag}
  -t TOL, --tol TOL
bwall@highwind:~/code/IDPanel$
```

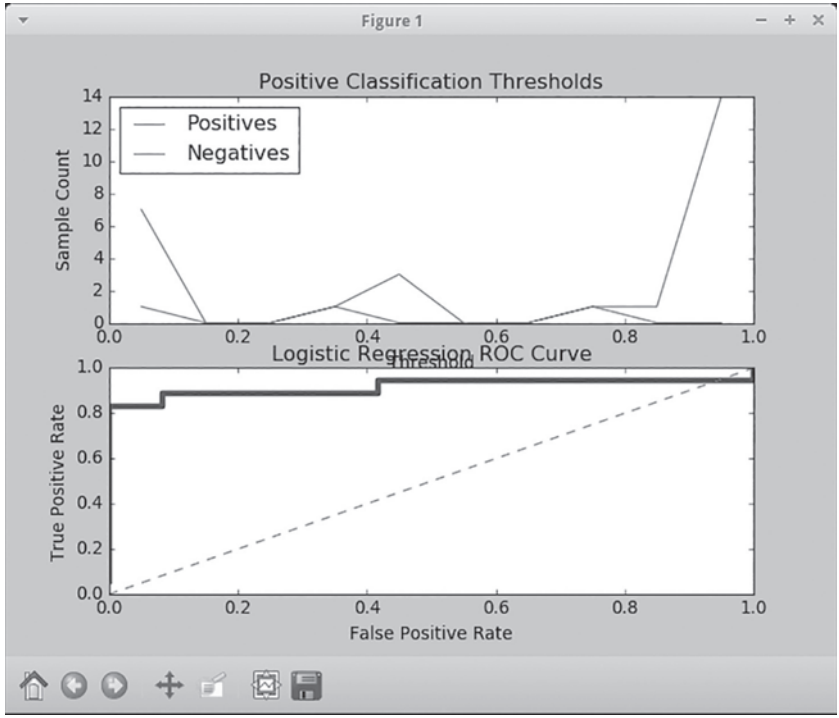


図2.13：分類のしきい値とROC曲線

前回と同じように、クラシファイアはハイパーパラメータ設定を変更せずに実行します。処理が完了すると、スクリプトはLRモデルのbot_model.lrmdlと関連するしきい値グラフおよびROC曲線と共に出力します。

今回もデフォルトのパイパーパラメータ設定で正確なモデルが生成されています。では、パイパーパラメータ設定を調整することで結果を改善できるかどうかを見てみましょう。ペナルティをデフォルトのL2からL1に変更します。

```
bwall@highwind:~/code/IDPanel$ python train_lr_model.py -p l1
Creating training and testing sets
94 samples in training set, 2 labels in training set
30 samples in training set, 2 labels in testing set
Confusion Matrix:
[[11  1]
 [ 5 13]]
```

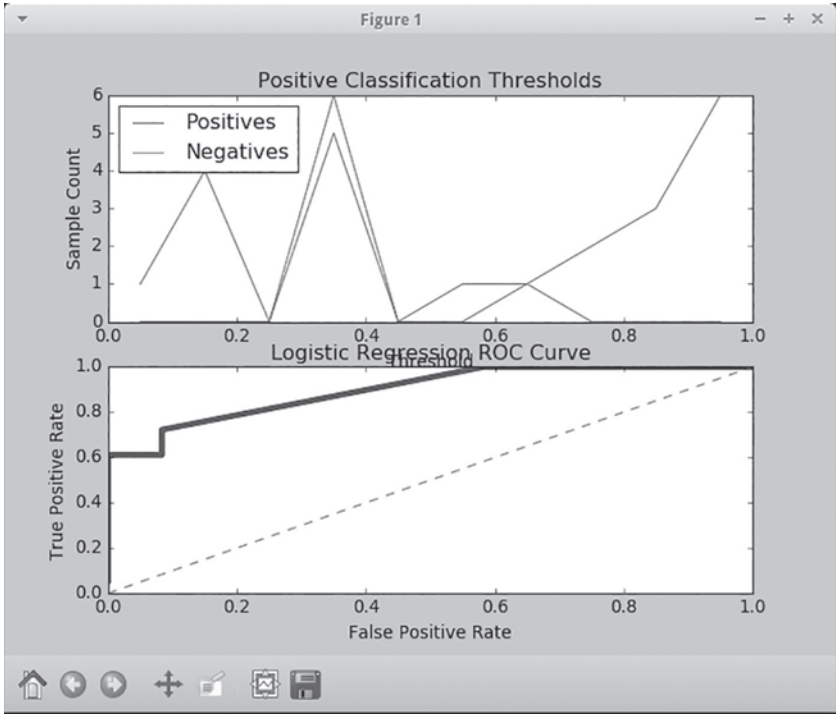


図2.14：分類のしきい値とROC曲線

この一見わずかな調整により、前回より精度の低いモデルが生成されました。分類を続行する前に、デフォルト設定でモデルを再トレーニングします。

生成されたモデルをラベル付けされていないベクトルに適用して

分類します。前回と同じコマンドを使用しますが、引数にはbot_model.lrmdlと指定します。

```
bwall@highwind:~/code/IDPanel$ python classify_panel.py bot_model.lrmdl https://bwall.github.io
Identifying panels we can actually reach
We can reach https://bwall.github.io/
Making 4789 total requests to 1 servers
https://bwall.github.io/, Not Panel, [ 0.99699357  0.00300643]
bwall@highwind:~/code/IDPanel$
```

ロジスティック回帰はディシジョンツリーほど積極的には特徴を減らさないため、ターゲットを良性と分類するには5,000件近くの要求が必要です。ただし、分類結果は正しく、信頼できます。ここでも既知のC&Cシステムに対してモデルをテストしてみましょう。

HTTP要求を発行し、結果を抽出してベクトル化してから、モデルを通してベクトルを実行します。モデルはターゲットをC&Cシステムとして正しく分類しました。

```
bwall@highwind:~/code/IDPanel$ python classify_panel.py bot_model.lrmdl http://112.213.88.68/Panel/
Identifying panels we can actually reach
We can reach http://112.213.88.68/Panel/
Making 4789 total requests to 1 servers
http://112.213.88.68/Panel/, Botnet Panel, [  5.99084234e-04   9.99400916e-01]
bwall@highwind:~/code/IDPanel$
```

分類の重要なポイント

ここまで見てきたように、分類はパワフルで効果的な教師あり学習の手法であり、さまざまなセキュリティ問題に柔軟に対応します。本章の重要なポイントは以下のとおりです：

- 分類は、十分な量のラベル付きデータが存在し、サンプルが実際の陽性および陰性ケースの比率を正確に反映していて、構成データが本質的に分類解析に適している場合に有効な教師あり学習手法です。
- 分類は、トレーニング、検証、テスト、および展開の4段階で進行します。本章では、そのうちのトレーニング、テスト、および展開について説明しました。すべてのトレーニングとテストでサンプルがどのクラスに属するか（クラスメンバーシップ）は事前に判っているため、各クラシファイア

は、各特徴と特徴値によって各サンプルの既知のクラスメンバーシップをどの程度まで予測できるかに基づいてモデルを構築します。

- ロジスティック回帰とディシジョンツリーはどちらも非常に効果的に分類を行うことができますが、アプローチが大きく異なります。LRは、各特徴がクラス予測に最終的にどの程度影響するかを決定する回帰の重みとバイアス値を計算します。ディシジョンツリーは、分岐変数と分岐値を計算して、ノードに含まれる同質のクラスメンバーの割合を徐々に増やします。
- ロジスティック回帰クラシファイアは、直線や平面を使用して特徴空間を形作るため、データが適合不足になりがちです。ディシジョンツリーは矩形を使用して特徴空間を形成し、停止条件が適用されるまで、構成するデータ環境での実際の存在比率とは無関係にすべてのベクトルに対してディシジョンルールを作り続けるため、データを過剰適合しがちです。トレーニングデータを効果的に分類できたDTモデルの精度がデータのテストにおいて顕著に低下した場合は、過剰適合が疑われます。
- アナリストは、さまざまな厳格な検証手法を使用して、モデルの精度を評価します。混同行列、確率しきい値図、ROC曲線などです。ただし、クラシファイアが予測できるのは、新しいラベル付けされていないサンプルが特定のクラスに属する確率のみです。ほとんどの場合、偽の陽性および陰性クラス割り当てが少なからず存在します。したがって、対処する問題の本質を反映した許容範囲内に誤差率が収まっていれば、分類は最適な手法であると言えます。

確率

分類の章では、ロジスティック回帰とディシジョンツリーのアルゴリズムが確率スコアを利用してサンプルを2つのクラスのいずれかに割り当てる様子を確認しました。サンプルの確率スコアがしきい値である0.5を超えていればクラス1に分類され、0.5未満であればクラス0に分類されます。この章では、予測を目的とした機械学習手法の1カテゴリーとして、確率をより広い視野から見てみましょう。具体的には、ナイーブベイズ(NB)クラシファイアと、ガウス混合モデル(GMM)クラスタリングアルゴリズムについて説明し、これらを応用してセキュリティに関連した検出と対処の問題をどのように解決できるかを考えます。

確率とは何か

人間は誰でも不確実な世界に生きています。目標を達成できるかどうかは、起こりうるイベントを正確に評価して予測できるかどうかに掛かっています。そしてあらゆる見込みを計算し、確率を考えます。しかし、ある意思決定が正しいかどうかを確実に保証する術はありません。常に成功を妨げるランダムな要因が発生するからです。たとえば、降水確率が25%のときに洗車をすべきかどうか。休暇のための航空機チケットを今買うべきか、それとも何週間か待ってもっと安いチケットが出るのを待つべきか。

このように情報が不十分あるいは不完全な場合の問題を解決するには、確率的モデルが適しています。特に不確実性のモデリングには非常に効果的です。確率的モデルを適切に応用することで、高い信頼性で意思決定を行えるレベルまで不確実性を低くすることができます。

よくある確率の問題の例

下図のように、円を四等分して赤、黄、緑、青に色分けしたルーレットを考えてみましょう。

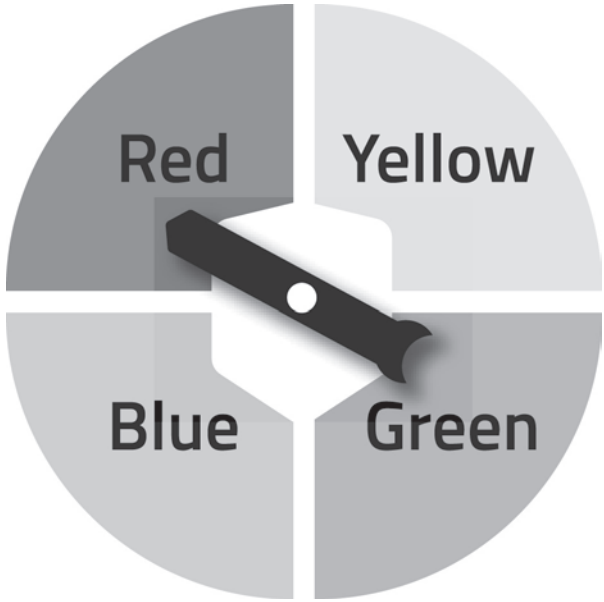


図3.1：カラースピナー

自由に回転するとして、黄、赤、青、緑のそれぞれに止まる確率はどうなるでしょうか● この問題に答えるためには、スピナーを回す必要があります。確率論の用語では、この回す行為を実験または試行と呼びます。各試行で得られるエビデンスは結果と呼ばれます。このケースで発生しうる結果は、黄、赤、青、緑です。

各結果の確率を計算するには、各結果が発生する手段の数を、発生しうる結果の合計数で割ります。

対象となる結果	この結果が発生する手段の数	発生しうる結果の数	確率の計算
P(黄)	1	4	1/4 = .25
P(赤)	1	4	1/4 = .25
P(青)	1	4	1/4 = .25
P(緑)	1	4	1/4 = .25

それぞれの結果の確率はすべて同じで、P=0.25となります。
次は少し複雑な例として、サイコロを考えてみます。

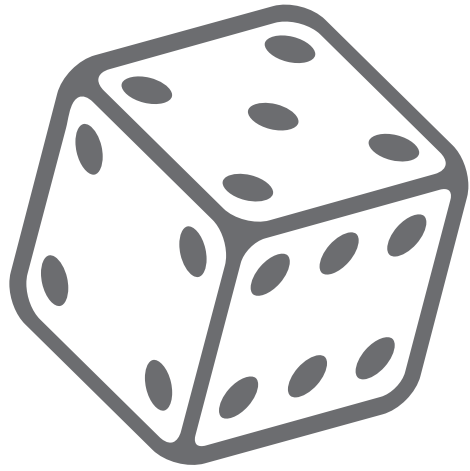


図3.2：サイコロ

同じアプローチで6面のそれぞれの目が出る確率を予想します。

対象となる結果	この結果が発生する手段の数	発生しうる結果の数	確率の計算
P(1)	1	6	1/6 = .16
P(2)	1	6	1/6 = .16
P(3)	1	6	1/6 = .16
P(4)	1	6	1/6 = .16
P(5)	1	6	1/6 = .16
P(6)	1	6	1/6 = .16

サイコロに細工がされていなければ、各目の出る確率はすべて同じです(P=0.16)。

では次に、サイコロを振って奇数が出る確率と偶数が出る確率を調べます。どの試行においても、奇数は1、3、5の3つ、偶数は2、4、6の3つです。いくつかの異なる手段で発生しうる結果をイベントと呼びます。たとえば、奇数というイベントは1、3、5の3つの結果によって発生し、偶数というイベントは2、4、6の3つの結果によって発生します。6つの発生しうる結果に基づいて、奇数イベントと偶数イベントの確率を計算してみましょう。

対象イベント	奇数/偶数イベントを発生させる結果の数	発生しうる結果の数	確率の計算
P(偶数イベント)	3	6	3/6 = .5
P(奇数イベント)	3	6	3/6 = .5

どちらのイベントの確率も同じです。ここまでの確率解析では、特定の試行の結果の予測には何も貢献しません。結果は常にランダムになります。

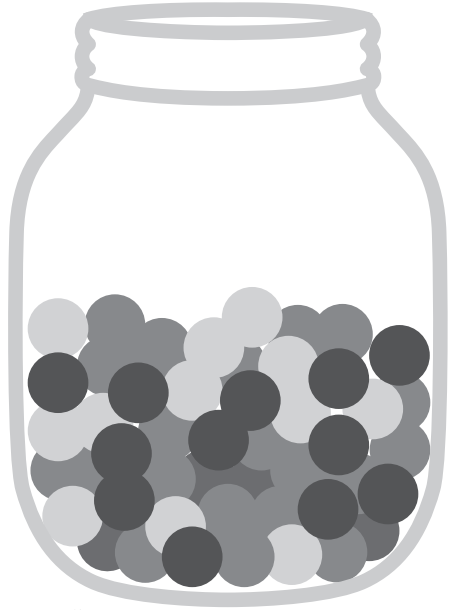


図3.3：ビー玉入りの瓶

しかし、結果の確率が等しくない問題も多く存在します。ビー玉入りの瓶を使って、このような例を考えてみましょう。

この瓶には、黄色のビー玉が3個、赤が6個、青が8個、そして緑色が5個入っています。この瓶からビー玉を1個ランダムに取りだした場合、それが黄色である確率はどうなるでしょうか。発生しうる4つの確率の可能性を下表にまとめました。

対象イベント	このイベントを発生させる結果の数	発生しうる結果の数	確率の計算
P(黄)	3	22	3/22 = .13
P(赤)	6	22	6/22 = .27
P(青)	8	22	8/22 = .36
P(緑)	5	22	5/22 = .22

この場合は、4つの結果の確率が異なります。任意の試行において、青のビー玉を取り出す確率が最も高く(P=.36)、黄色が最も低くなります(P=.13)。

この項の説明の締めくくりとして、すべての結果の発生確率が同じでも、それらの結果を発生させるイベントの確率が異なるケースを見てみましょう。この実験では、1から5までの番号が書かれた5枚のカードが入った箱から1枚を選びます。目的は、2つの異なる確率の問題を解くことです。まず、5つの発生しうる結果のうちの1つを選ぶ確率を計算します。次に、奇数または偶数のイベントを選ぶ確率を計算します。

これまでと同じように、特定の番号を選ぶための手段の数(1)を発生しうる結果の数(5)で割ります。すべての結果の発生確率はここでも同じです。

対象イベント	このイベントを発生させる結果の数	発生しうる結果の数	確率の計算
P(1)	1	5	1/5 = .2
P(2)	1	5	1/5 = .2
P(3)	1	5	1/5 = .2
P(4)	1	5	1/5 = .2
P(5)	1	5	1/5 = .2

しかしながら、イベントの発生確率は異なります。その理由は、奇数イベントが発生する結果が3つ(1、3、5)なのに対して、偶数イベントの結果は2つ(2と4)しかないためです。

	このイベントが発生する手段の数	結果の合計数	確率の計算
P(偶数)	2	5	2/5 = .4
P(奇数)	3	5	3/5 = .6

その結果、特定の試行において奇数のカードを引く確率は60%(P=0.6)で、偶数のカードを引く確率は40%(P=0.4)となります。

条件付き確率と結合確率

これまでの例は、各試行は独立したものとして考えていました。たとえば、コインを投げて表が出たという結果は、次に投げたときに裏が出るという確率には何の影響も与えません。一方で、2つの結果がどこまで関連しているのかを考える必要がある問題も多く存在します。

- 条件付き確率の問題では、イベントAに続いてイベントBが起こる確率を求めます。たとえば、雪嵐の後で自動車事故が急増する確率です。
- 結合確率の問題では、イベントAとイベントBが同時に発生する確率を求めます。たとえば、サイコロを2個投げて2個とも目が5になる確率です。

最初に条件付き確率の例を考えてみましょう。25セント硬貨が3

枚、10セント硬貨が2枚入った袋があります。最初に取り出した一枚が25セント硬貨である確率はどのくらいですか。3（25セント硬貨を選択するイベントが発生しうる回数）を5（発生しうる結果の数）で割ることで、確率スコアのP=0.6が得られます。

最初の1枚が25セント硬貨だったとして、袋には25セント硬貨と10セント硬貨が2枚ずつ残っています。では、2回目に25セント硬貨を取り出す確率はどうなるでしょうか。前回と同じく、25セント硬貨を選択するイベントが発生しうる回数（今回は3から2に減少）を発生しうる結果の数（同じく5から4に減少）で割ることで、確率スコアのP=0.5が得られます。このように、2回目の試行の結果は、最初の結果に依存します。

次は結合確率の例を見てみましょう。2個のサイコロを投げて、どちらも5の目が出ることに賭けたとします。あなたが勝つ確率はどうなるでしょうか。前述のサイコロ1個の例から、5の目が出る確率はP=0.16だとわかります。これは2個のサイコロを同時に投げようが1,000個を投げようが変わりません。それぞれのサイコロで5の目が出る確率は常にP=0.16です。2つのイベントは互いに独立しています。

しかしながら、2個のサイコロがどちらも5の目を出す確率は、1個のサイコロが5の目を出す確率より低いということは直感でわかります。結合確率の問題では、2つのイベントの確率を掛け合わせることによって、両方が発生する結合確率を以下のように計算します。

このように、勝つ確率は非常に低いことがわかります。

ナイーブベイズアルゴリズムによる分類

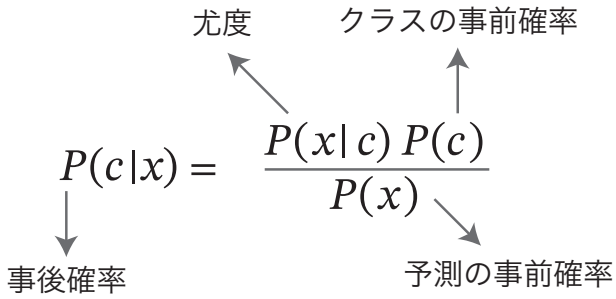
ナイーブベイズアルゴリズムは、イギリスの統計学者トーマスベイズによって18世紀に提唱され、後にフランスの数学者ピエールシモンラプラスによって現在の定理に体系化されました。ベイズの定理は、条件Bが成立したときにイベントAが発生する確率を計算するための手段を提供します。分類の問題では、ベイズの定理により、与えられた特徴属性に基づいてサンプルが特定のクラスに属す

る条件付き確率を計算することができます。第2章で説明したように、分類は教師あり学習手法の一例です。そのため、分類セッションは、トレーニング、検証、およびテストというシーケンスで実施されます。

ベイズの定理は、条件付き確率を考慮するため、解決が難しくなることがあります。分類における分岐を決定する際に、データセットに含まれるすべての特徴の間の確率関係を計算するためです。特徴の数が増えるほど、これらの関係の数と複雑さは指数的に増大します。時間とリソースのコストも急速に肥大します。

ナイーブベイズ定理は、クラスの条件付き独立性を前提とすることで、このプロセスを大幅に簡素化します。つまり、サンプルをクラスに割り当てる際に、条件付き確率の潜在的な効果を無視します。ほぼすべてのケースにおいて、この前提は真ではありません。この形式のベイズの定理が「ナイーブ」と称されるのはそのためです。しかし驚くべきことに、ナイーブベイズはわずか4つのコンポーネントのみでサンプルを分類できるため、多くのケースで優れた結果と高い効率を実現します。

ナイーブベイズ



$$P(c|x) = P(x_1|c) \times P(x_2|c) \times \cdots \times P(x_n|c) \times P(c)$$

図3.4：ナイーブベイズ確率の公式

1. **事後確率**：[P (c | x)] 事後確率とは、特徴属性 (x) が与えられたときにサンプルが特定のクラス (c) に属する確率です。実際には、可能なクラス割り当てについて1回ずつ、合計で複数回計算が実施されます。(ある時点での確率スコアは、それまでのスコアの合計を1から差し引くことで得られるため、実際にはクラス数より1回分少なく計算します。) 確率スコアが最も高い結果によってクラスの割り当てが決定します。
2. **クラスの事前確率**：[P (c)] クラスの事前確率 (Class Prior Probability: CPP) は、データセット内におけるクラスの存在比率を指します。 たとえば、データセットに10件のサンプルが存在し、そのうちの7件がクラス1に属する場合、クラス1のCPPは7/10 = 0.7となります。これは「ランダムに抽出したサンプルは70%の確率でクラス1に属する」とも表現できます。
3. **予測の事前確率**：[P (x)] 予測の事前確率 (Predictor Prior Probability: PPP) は、データセット内における特徴属性の存在比率を指します。 たとえば、データセットに20件のサンプルが存在し、そのうちの5件の特徴属性がLoanである場合、LoanのPPPは5/20 = 0.25となります。
4. **尤度**：[P (x | c)] 尤度とは、クラスcが与えられたときに特徴属性xが見つかる確率です。 特定のクラスラベルと属性を持つサンプル数を、そのクラスに属するすべてのサンプル数で割って求めます。一例として、アメリカの2つの都市における降水の仮説的研究について考えてみましょう。片方の都市は寒い冬、他方は暖かい冬となっています。各都市で10日間、降水パターンを観察し、晴天、曇天、および雨天または降雪の日数を記録しました。下記の尤度の表では、暖冬である都市1の雨天/降雪属性の尤度スコアは1/10 = 0.1と計算できます。厳冬である都市2の同じ属性の尤度スコアは7/10 = 0.7です。また、尤度表では以下の値も計算します。
   - PPP: 各行の合計をサンプル数で割って求めます。
   - CPP: 各列の合計をサンプル数で割って求めます。

尤度表

天候	都市1 (暖かい)	都市2 (寒い)	合計	尤度
晴天	6	1	7	(=7/20)
曇天	3	2	5	(=5/20)
雨天/降雪	1	7	8	(=8/20)
合計	10	10	20	
尤度	(=10/20)	(=10/20)		

ナイーブベイズにはいくつかのバリエーションがあり、それぞれが特定のデータセットや問題のシナリオに適しています。その中で以下の3種類が最もユーティリティ名です。

- **ベルヌーイナイーブベイズ**(BNB)。この手法は、特定のテキストの存在を示すようにベクトルがエンコードされているテキストベースの問題(スパムの検出など)に適しています。その文字列が1つでも含まれていれば、そのメッセージはスパムに分類されます。
- **多項ナイーブベイズ**(MNB)。この手法は、特定のテキストが出現する頻度を示すようにベクトルがエンコードされているテキストベースの問題に適しています。MNBは、文書の分類などに適しています。たとえば、「仲介」や「違反」といった文字列の出現頻度によって、文書が契約書であるかどうかを判断するようなケースです。
- **ガウスナイーブベイズ**(GNB)。この手法は、正規(ガウス)分布内の連続したデータに適しています。たとえば、アメリカの男性と女性の平均身長と平均体重を示すデータなどです。ガウス分布とその特性については、後ほどガウス混合モデルのクラスタリングアルゴリズムの説明で取り上げます。

ナイーブベイズの応用

では、実際のナイーブベイズの応用について見てみましょう。ここでの目的は、リモートコンピュータが提供しているネットワーク

サービスに基づいて、そのコンピュータでWindowsとLinuxのどちらが動作しているかを特定することです。既知のオペレーティングシステムが動作しているコンピュータのネットワークをスキャンすることによってトレーニングデータを取得します。

トレーニングデータを収集してベクトル化し、scikit-learnにインポートしたら、ナイーブベイズを呼び出して分類モデルを構築します。クラシファイアは、まず下記の表のベクトルを解析して以下を決定します。

- Windowsとラベルされたシステム(4つ)とLinuxとラベルされたシステム(3つ)。
- これらのサーバーが提供しているHTTP、SSH、SMB、およびFTPサービス。たとえば、最初のWindowsサーバーはSMBおよびFTPサービスのみを提供しており、3番目のWindowsサーバーはHTTPおよびFTPのみを提供しています。

ラベル	HTTP	SSH	SMB	FTP
Windows	0	0	1	1
Windows	1	0	1	0
Linux	1	1	0	0
Windows	1	0	0	1
Linux	0	1	0	1
Windows	1	0	1	0
Linux	0	1	0	0

次にナイーブベイズは、下記の尤度表に示すようにクラスの事前確率（CPP）と予測の事前確率（PPP）を計算します。

ラベル	HTTP	SSH	SMB	FTP	クラスごとの合計サンプル数	クラスの事前確率
Windows	3	0	3	2	4	4 / 7 = 0. 571428571
Linux	1	3	0	1	3	3 / 7 = 0. 428571429
合計	4	3	3	3	7	
予測の事前確率	4 / 7 = 0. 571428571	2 / 7 = 0. 428571429	3 / 7 = 0. 428571429	3 / 7 = 0. 428571429		

CPPを求めるため、ナイーブベイズは各クラス（WindowsまたはLinux）の出現回数をトレーニングセットに含まれるサンプル数で割ります。この例では、Windowsのクラスの事前確率は約0.57、Linuxは0.43です。これは、ランダムに選んだサンプルがWindowsシステムである確率（57%）の方がLinuxシステムである確率（43%）よりも高いという意味です。

尤度表には予測の事前確率も示されており、各特徴の（全サンプルにおける）出現回数の合計をサンプルの総数で割って求められています。たとえば、HTTP特徴のPPPは約0.57です。

どれほどサンプリングを注意深く行った場合でも、トレーニングセットに取り込み損なった特徴とクラスの組合せがデータセットに含まれる可能性は依然として残っています。これを補正しないと、ナイーブベイズは、これらの特徴属性を持つサンプルがそのクラスに属する確率スコアを0として計算してしまいます。これはナイーブベイズクラシファイアについて知られている問題であり、アナリストは「平滑化」と呼ばれる手法で対処しています。

平滑化は、実質的には確率スコアに影響することなく、どのクラスについても出現確率が0%として計算されてしまう特徴値を補正します。多くの平滑化手法を利用できます。付加的（ラプラス）平滑化では、特徴数にα値を加算します。（この例ではα値の1を使用します。）また、d* αに等しい値をクラス数に加算します。dはクラ

ス数です。補正後の値を下表に示します。Windowsのクラス数が4から6に、Linuxのクラス数が3から5に増えています。

ラベル	HTTP	SSH	SMB	FTP	クラス数
Windows	4	1	4	3	6
Linux	2	4	1	2	5

では、前述の方法で尤度値を再計算しましょう。たとえば、HTTPとWindowsの尤度は次のようになります。

$$4/6 = 0.6667$$

平滑化後の尤度表は以下のとおりです。

ラベル	HTTP	SSH	SMB	FTP
Windows	0.6667	0.1667	0.6667	0.5
Linux	0.4	0.8	0.2	0.4

では、これらの確率スコアをテストデータセットに適用して、新しいサンプルが属するクラスを正確に予測できるということを確認してみましょう。つまり、サンプルの事後確率を計算します。まず、下記の特徴値のサンプルから始めましょう。(ここでは、サンプルにLinuxというラベルが付いていることを知らないものと想定します。)

ラベル	HTTP	SSH	SMB	FTP
Linux	1	1	0	1

サンプルのクラスを予測するためには、事後確率を2回(WindowsとLinuxに対して1回ずつ)計算しなければなりません。式は次のようになります。

P (特徴 | クラス) ^ 特徴値 * (1 – P (特徴 | クラス)) ^ (1 – 特徴値)

実際に計算してみましょう。

HTTP: 0.6667^1*0.3333^0=0.6667*1.0=0.6667

SSH: 0.1667^1*0.8333^0=0.1667*1.0=0.1667
SMB: 0.6667^0*0.3333^1=1.0*0.3333=0.3333
FTP: 0.5^1*0.5^0=0.5*1=0.5

4つの特徴値が揃ったところで、これらのスコアの積と平滑化する前のクラスの事前確率を見つけて、Windowsの確率スコアを生成できます。

(0.6667*0.1667*0.3333*0.5)*0.571428571=0.010583598

Linuxについても同様に計算します。

HTTP: 0.4^1*0.6^0=0.4*1.0=0.4
SSH: 0.8^1*0.2^0=0.8*1.0=0.8
SMB: 0.2^0*0.8^1=1.0*0.8=0.8
FTP: 0.4^1*0.6^0=0.4*1.0=0.4
(0.4*0.8*0.8*0.4)*0.428571429=0.043885714

では事後確率の結果を比較します。

Windowsの確率: 0.010583598
Linuxの確率: **0.043885714**

ナイーブベイズは、Linuxクラスに属するサンプルを正しく予測できています。

ナイーブベイズセッションのプロセス

教師あり学習手法であるナイーブベイズ解析は、第2章でロジスティック回帰とディシジョンツリーを取り上げた際に説明したトレーニング、検証およびテストフェーズと同じシーケンスを実行します。トレーニングが完了すると、尤度表と関連するクラスおよび予測の事前確率値がモデルとして用意されます。このモデルは、後続の検証およびテストフェーズでデータのテストに使用され、混同行列とROC曲線で精度が評価されます。このプロセスが正常に完了したら、トレーニング済みのモデルを新しい、ラベル付けされて

いないデータに適用して、クラスを予測します。

ナイーブベイズの落とし穴と制約

ナイーブベイズは、計算された事前確率に基づいて正確な分類を行うのに極めて効果的ですが、いくつかの制約があります。

- 前述のように、ナイーブベイズは、特徴が条件付き独立性を持つということを前提としていますが、現実世界のほとんどの問題では、この前提は成立しません。それにも関わらず、ナイーブベイズは多くの場合に優れた結果を出します。
- 与えられたデータセットの密度が希薄である場合は、構成するデータ環境に存在する実際の特徴とクラスの組合せをすべて取り込めないことがあります。幸い、この状況はラプラスや他の平滑化手法によって改善できます。

ガウス混合モデルアルゴリズムでのクラスタリング

第2章では、教師なし学習の概念と、k平均法およびDBSCANアルゴリズムがクラスタにどのようにベクトルを割り当てるのかについて説明しました。どちらのアルゴリズムもベクトルの割り当てには確率を利用していませんでした。その代わりに、特徴空間における相対位置に基づいてベクトルをクラスタに割り当てていました。この項では、確率を利用する別のクラスタ手法を説明します。具体的には、ガウス混合モデル（GMM）アルゴリズムが確率スコアを使用してベクトルをクラスタに割り当てる方法と、特定の問題におけるこのアプローチの決定的な優位点について説明します。

図3.5に示すクラスタリングの例を考えてみましょう。左側のプロットは、ライトグレーとダークグレーに色分けされた2セットのベクトルを示しています。中央は、k平均法アルゴリズムで平滑化したクラスタリング結果です。そして右側はGMMで生成されたクラスタリング結果です。k平均法はユークリッド距離法を利用しており、重複するクラスタや形状が円形ではないクラスタを適切に検出することはできません。これに対してGMMはこれらのクラスタを正しく検出できます。

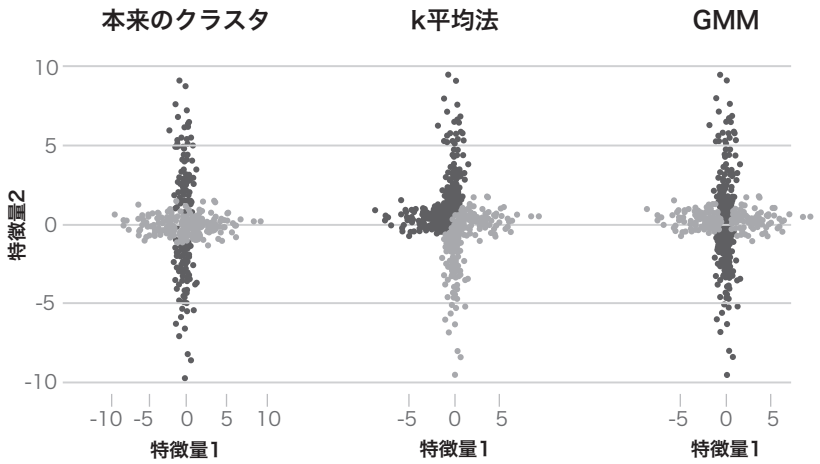


図3.5： k平均法とGMMの比較

GMMの仕組みを見る前に、いくつかの基本原理を理解しておかなければなりません。まず、ガウス(正規)分布について説明します。

ガウス分布

ニュージャージー州ジェファーソンタウンシップの某学区にある中学校の生徒2万人を対象にして、身長の分布を解析してみましょう。このデータは多くの方法で表現できます。たとえば、図3.6に示すヒストグラムでは、x軸が身長(cm)、y軸が人数を表します。それぞれの縦棒は、同じ身長範囲を共有する生徒の集まりで、これを「ビン」と呼びます。

最大の生徒数は、分布中央の160cmを表す点の付近にあります。言い換えると、この生徒の母集団において最も多く出現する身長は160cmであると言うことです。中央のすぐ右側にあるビンは、身長範囲160〜162cmに属する生徒の数を表しています。中央の左側にあるビンは、身長範囲158〜160cmに属する生徒の数を表しています。

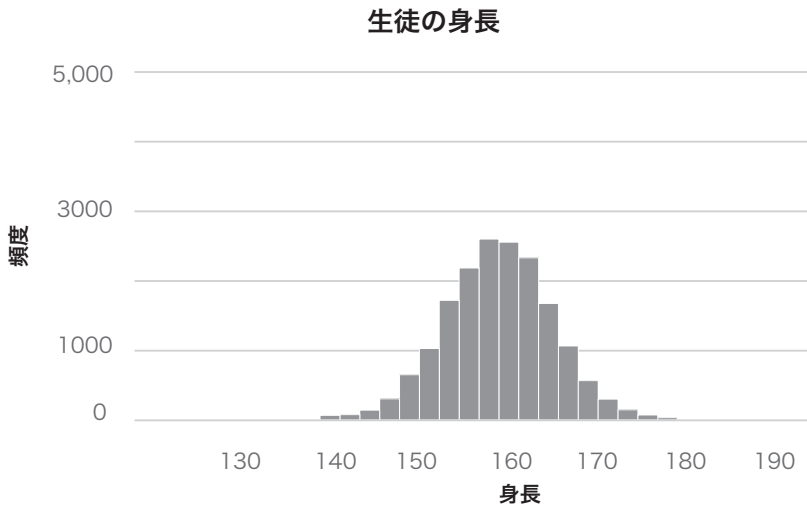


図3.6：ジェファーソンタウンシップの中学校の生徒の身長分布

中央ポイントは、平均値、中央値、または最頻値と呼ばれます。これらはそれぞれ若干異なる意味を持ちますが、ここでは平均値を用います。平均値とは、分布において測定している特徴の値の平均を出したものです。この例では、すべてのビンのすべての身長値を加算して、その合計を生徒の総数で割ることで平均値が求められます。実際に平均値を計算したら160cmになったと仮定しましょう。

図3.7は、ガウス分布の別のプロット方法を示しています。ここでは、y軸が各身長値の生徒の比率を示しています。比率は、各ビン(頻度)の生徒数を生徒の総数で割ることによって求められます。

左側のプロットは、生徒の身長の比率を表したヒストグラムで、右側のプロットはヒストグラムに適合した平滑な釣鐘曲線です。ここでも、最大の生徒数の比率(約75%)は160cmの平均値付近にあり、それより背の低い生徒や背の高い生徒は両側に対称的に分布しています。

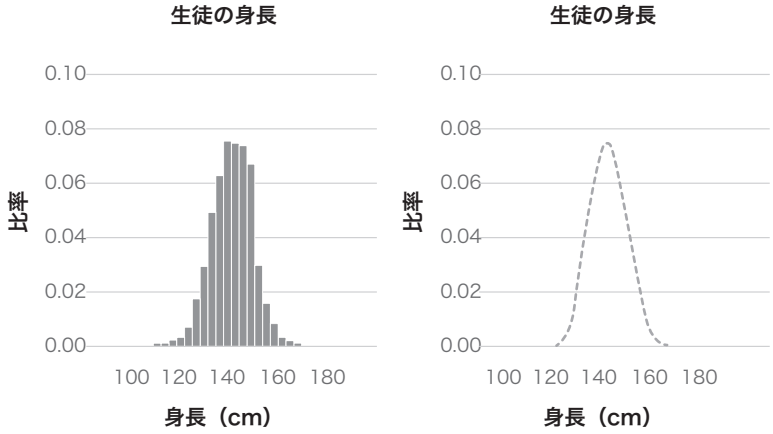


図3.7：比率として表現したガウス分布のヒストグラムと平滑曲線

最も生徒数が多いのは160cm付近の身長ですが、非常に背が高い、あるいは非常に背が低い生徒も存在します。平均値から左右に移動すると、背の低い生徒や背の高い生徒の数が対称的に減っていき、最後はゼロになります。この平均値を中心とした対称性は、ガウス分布の最も顕著な性質の1つであり、特徴的な釣鐘曲線の元となります。

標準偏差と分散の計算

すべてのガウス分布は、その平均値と分散(標準偏差)によって一意に識別できます。分散と標準偏差は、分布の幅と平均値からの多様性を示します。標準偏差または分散が小さければ、分布の幅は狭く、データポイントは平均値付近に集中していると予想できます。標準偏差と分散が大きければ、データは幅広く分布していることになります。これら2つの唯一の違いは、測定の単位です。分散は標準偏差の二乗に等しくなります。したがって、身長の例であれば、標準偏差の単位はcm、分散の単位はcm^2となります。

平均身長が160cmである母集団において、身長が155cmの生徒と165cmの生徒の分散を計算してみましょう。プロセスはシンプルです。

1．まず、対象者の身長から母集団の平均値を引いて、結果を二乗します。

$(155\text{-}160)^2=25$
$(165\text{-}160)^2=25$

2．そして、二乗した値の平均を計算します。

$(5^2+5^2)/2 = 25\text{cm}^2$

分散が求められたら、その平方根を求めることで標準偏差に簡単に変換できます。上の例では、標準偏差は$\sqrt{25} = 5$cmとなります。つまり、標準偏差は、平均値である160cmの両側5cmのエリア内に入るサンプルを包含するということです。2つの標準偏差は、平均値のどちらかの側の5cm以内のサンプルを包含します。図3.8に、平均値の片側の標準偏差内に入るデータポイントを示します。

標準偏差は、特定の属性値を持つサンプルが見つかる可能性を視覚化するのに便利です。図を見るとわかるように、身長が155cmから165cmの間に入る生徒は比較的容易に見つかりますが、左右の端に行くほど、その身長の生徒は見つかりにくくなります。

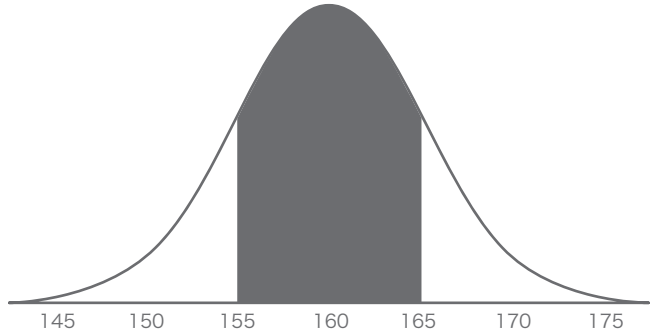


図3.8：曲線の暗い部分は、平均値の両側で1つの標準偏差内に入る分布部分を示しています。

クラスタリングへのガウス分布概念の応用

*f*と小学校の生徒の身長データを誤ってまとめてしまい、どの生徒がどちらの学校に属するのかを記録し忘れてしまったとします。

サンプルが混ざってしまった後で、元のように「分離する」ことは可能でしょうか？

小学校の生徒の平均身長は、中学校と比べて低いと考えるのが妥当です。また、小学校には幼稚園児から5年生までが在籍しますが、中学校には6年生から8年生までしかいませんので、2つのグループにおける身長の分布も異なると予想できます。

生徒の身長のように連続した変数は、通常はガウス分布に適合できます。また、すべてのガウス分布は平均値と分散パラメータによって一意に識別できることもわかっています。したがって、2つの生徒グループを分けることは理論的には可能なはずです。機械学習の用語では、生徒の身長という特徴値に基づいて各生徒を小学校または中学校のクラスタに割り当てると言います。

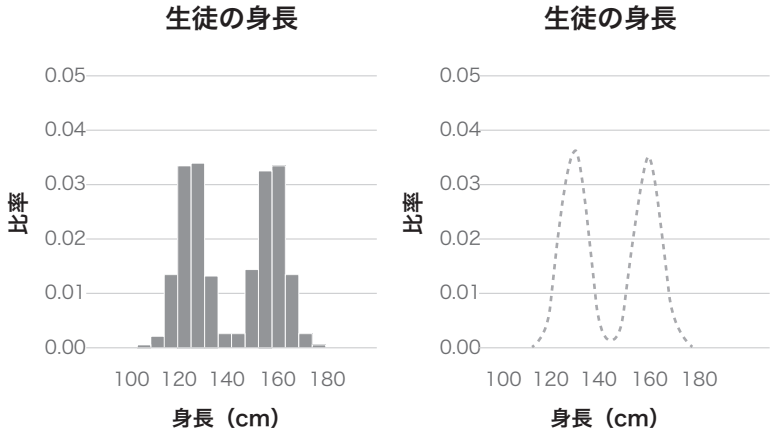


図3.9：GMMは、固有のガウス分布に基づいて小学校と中学校の生徒を別々のクラスタに分割します。

GMMは、生徒の身長のデータセットを解析して、各クラスタを定義する固有の平均値と分散パラメータを特定することで、このクラスタ割り当てを行います。次に、これらと他のいくつかのパラメータを使用して、各生徒がどちらのクラスタに属する確率が高いかを調べます。各生徒は、確率スコアが高い方のクラスタに割り当てられます。

図3.9では、GMMによって小学校と中学校の生徒それぞれのガウ

ス分布が再構築されています。前述のように、左右のプロットはガウス分布をヒストグラムと平滑曲線として表しています。

ここでシンプルなイラストと例を使ってこれらの概念を説明したのは、分かりやすく、理解を深めるためであることをお忘れなく。次に例を挙げます。

- 図3.7と図3.8では、y軸で比率を表していました。技術的な理由から、より正確な用語は密度と言います。確率の問題を解く場合の密度の役割については、https://en.wikipedia.org/wiki/ Probability_density_functionを参照してください。
- ここでの分散の説明では、特徴を1つしか使用していません。実際の確率の問題では、複数の特徴が関与し、それらに関連した共分散の関係を解析する必要があります。確率の問題を解く場合の共分散の役割については、https:// en.wikipedia.org/wiki/Covarianceを参照してください。

ここでは、GMMがデータポイントをクラスタに割り当てる際に使用するプロセスについて詳しく見てみましょう。

期待値最大化

GMMは、クラスタリングを実行する際に、期待値最大化（Expectation Maximization: EM）と呼ばれる反復した2つのステップによる最適化プロセスを使用します。EMは、他の多くの機械学習アルゴリズムでも確率問題の解決に使用されています。まず、4つの初期値を設定します。

- **固定ハイパーパラメータ：混合数**。このパラメータは、作成するクラスタ数を決定します。GMMでは、グループへのデータポイントの割り当ては、離散ガウス分布の集まり（混合要素）を「分離」することによって行うため、クラスタではなく「混合」という用語を使用します。通常、混合数は、問題の性質、求められる精度、および該当分野におけるアナリストの専門知識に基づいて、直感的に決まります。たとえ

ば、スパム検出の問題であれば、スパムメール用と通常メール用に1つずつ、合計2つの混合のみを定義すれば十分です。一方で、混合要素の数を最適化する高度な手法も使用できます。

- **パラメータ1：混合比率（MP）**。各混合要素に属するサンプルの比率の予測です。生徒の身長の例なら、小学校混合のMPを全生徒数の2/3に設定し、中学校混合のMPは残りの1/3 にすると良いでしょう。MPは確率値で表し、範囲は0～1です。その結果、小学校と中学校の混合には、それぞれ0.66と0.34のMPが割り当てられます。
- **パラメータ2：**平均値。各混合要素のデータ分布の平均を取った値です。この例であれば、小学校と中学校の各混合要素における身長の平均値です。
- **パラメータ3：**分散（標準偏差）。平均値に対してデータがどのくらい集中しているのかを定義します。多くのアナリストは、各混合要素の分散を、経験とデータセットの性質に基づいて決定します。たとえば、クラスタリングに使用する特徴の値の範囲が広い場合は、同じように大きな標準偏差パラメータを選択します。

これは、MP、平均値、および標準偏差/分散値が現実的な値となるために望ましい選択ですが、必須ではありません。これらのパラメータは、GMMを繰り返すことによって徐々に精度が高まります。GMMは、アナリストがクラスタリングモデルの性能に満足するか、最大反復回数に達するまで繰り返されます。

ステップ1：期待値

このステップでは、GMMは各混合の平均値、分散、および混合比率パラメータを使用して、0～1の確率スコアを各データポイントに割り当てます。この値は、そのデータポイントの責任を持つガウス分布はどちらであるか、つまり、そのデータポイントがそれぞれの混合要素に属する確率を示します。生徒の身長の例では、各生徒

には小学校と中学校の混合要素に対して1つずつ、合計2つの責任スコアが割り当てられます。その後、データポイントは責任スコアが高い方の混合要素に割り当てられます。

すべてのデータのポイントに責任スコアの割り当てを完了すると、GMMは最大化ステージに進みます。

ステップ2：最大化

このステップでは、重み付き平均を計算し、その結果に基づいて平均値と分散パラメータを修正します。このプロセスは次のように進みます。

1．GMMは、各混合の責任スコアを合算して、それぞれの結果をデータポイントの総数で割ることにより、混合比率を再計算します。身長の例では、中学校混合要素の責任スコアを合算してから生徒の総数で割ります。小学校混合要素に対しても同じ計算を繰り返します。
2．そして、重み付き平均を計算することで平均値を更新します。
   - 重み付き平均は、各混合メンバーの特徴値に対応する責任スコアを掛けることによって求められます。たとえば、身長120cmの生徒で、小学校混合要素に属する責任スコアが0.9である場合、新しい重み付きの身長は108cm（120 cm*.9=108cm）となります。
   - GMMは、各混合要素の重み付き平均値を新たに計算します。各混合要素の重み付き値を合算して、その合計を混合の責任スコアで割ります。この計算を残りの混合要素に対しても繰り返します。

3．また、GMMは各混合要素の分散を新たに計算します。前の説明では、分散とは身長と平均値の差を二乗した値の平均であると述べましたが、ここでは重み付き平均を使用します。

ステップ3：EMサイクルの反復

GMMは、修正したパラメータを使用してEMサイクルを繰り返

し、新しい責任スコアを生成して、混合パラメータをさらに修正します。このプロセスは、2つの条件のいずれかが発生するまで繰り返されます。1つ目の条件は、値の変化があらかじめ定義されている許容範囲内に収まった(重み付き平均値の変化が1%以内で落ち着いた)場合です。もう1つは、あらかじめ定義されているEM反復回数に達した場合です。この時点で、各データポイントには、属する確率が最も高い混合を示す責任スコアが割り当てられています。

精度の許容誤差は、アナリストが持つ分野の専門知識、クラスタリング問題の性質、および関連するリスクに基づいて決定します。たとえば、スパムメールを検出する問題であれば許容誤差は広くても受け入れられますが、銀行取引が詐欺行為であるかどうかを見極める問題では受け入れられません。

生成されるGMMモデルは、ガウス分布を定義する混合比率、平均値、および分散値を含むベクトルで構成されます。このモデルを新しいデータに適用することで、各データポイントが属する確率が最も高い混合を示す責任スコアが生成されます。

GMMの注意点と制限事項

これまで見てきたように、GMMでは確率論的手法を利用してクラスタを定義しており、その利用方法にはk平均法や第1章で説明したDBSCANアルゴリズムを上回る優位点がありますが、以下のような制限もあります。

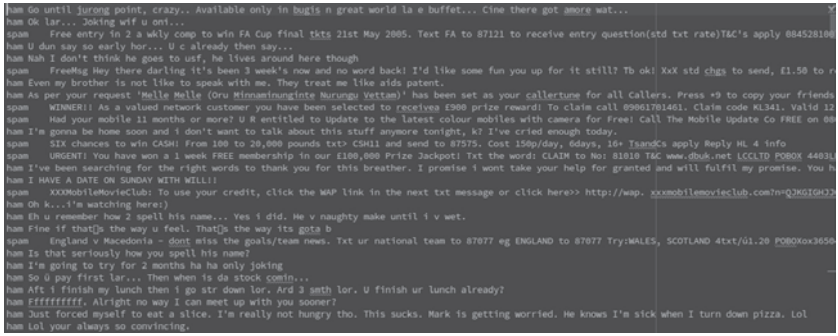
1. アナリストは、作成する混合要素の数を正確に選択しなければなりません。
2. GMMでは、例で使用した生徒の身長のように、データが連続していなければなりません。また、データがガウス分布に準拠している場合に最も優れた結果を生成します。
3. この章では、説明を分かりやすくするために特徴を1つ(身長)しか使用しませんでした。実際、GMMは、6次元以下の特徴空間で最大の威力を発揮します。それより次元数が増えると、GMMは計算コストが高まり、最終的には収束を達成

できなくなります。

ガウス混合モデルおよびナイーブベイズによるSMSスパムの検出

SMSスパムは目障りで不愉快であるだけではなく、ユーザーを騙して商品を買わせたり、個人情報をフィッシング詐欺業者に渡したりすることもあります。最近では、アプリケーション開発者も確率論的手法を利用して巧みにスパムを検出できるようになってきました。ここでは、scikit-learnでGMMおよびナイーブベイズ アルゴリズムを使用して、確率論的手法でSMSスパムを検出する例を2つ考えます。

まず、対象となるデータセットが必要です。Center for Machine Learning and Intelligent SystemsのWebサイト（https://archive.ics.uci.edu/ml/datasets/SMS+Spam+Collection）からダウンロードできるSMSスパムコレクションデータセット（SMS Spam Collection Data Set）を使用します。



次に、SMSSpamCollectionというテキストファイルを抽出して読み込みます。ファイルの内容を見てみると、各行の先頭の言葉はメッセージがスパムまたはハム（良性のメッセージ）のどちらであるかを示すラベルになっています。下記のように、テキストファイルにはスパムメッセージよりもハムメッセージの方がかなり多く入っています。クラスタリングは教師なしで実施するため、これらのラベルを使用してサンプルをクラスタに割り当てることはしません。これらのラベルは、後でクラスタリング結果の精度を評価するため

に使用します。

これらのメッセージを解析用に準備するため、すべての文字列において、連続する2つの文字間の遷移を解析します。このプロセスはバイグラム解析と呼ばれます。たとえば、最初のスパムメッセージの先頭は「Free entry」であり、Fからr、rからe、eからeのように、文字の遷移を観察します。それぞれの遷移が発生する頻度を計算することで、トレーニング対象のモデルは、メッセージが良性であるか、または宣伝や犯罪を目的とした者であるかを判断できるようになります。

ベクトル化

文字列の解析方法の1つは、各文字が特定の文字クラスに属するかどうかを調べることです。この例では4つの文字クラスを使用します。

- 文字
- 数字
- 記号
- 空白スペース

これらのクラス定義を使用して、「abc123」という文字列を文字クラスシーケンスに変換します。文字クラス0は文字に割り当て、文字クラス1は数字に割り当てます。これにより、次のシーケンスが生成されます。

文字列	a	b	c	1	2	3
文字クラスシーケンス	0	0	0	1	1	1

では、このシーケンスをバイグラムのセットに変換しましょう。

0, 0
0, 0
0, 1
1, 1
1, 1

次に、他と重複しないバイグラムの数を数えると、(0, 0)が2つ、(0, 1)が1つ、(1, 1)が2つ、そして(1,0)がゼロです。これらの合計を使用して次の行列を作ります。

2	1
2	0

最後のデータ準備ステップとして、GMMとナイーブベイズのアルゴリズムに適したベクトルに行列を平坦化します。生成される各ベクトルは16次元（先頭文字で可能な4つの文字クラス*2番目の文字で可能な4つの文字クラス）で構成されます。このように次元を制限することで、分類とクラスタリングの精度を必要以上に落とすことなく、GMMとナイーブベイズの計算をPCでも楽に行うことができます。

例1：GMMによるSMSのスパム識別

ベクトルが用意できたところで、GMM解析セッションを始めましょう。これまでと同じように、scikit-learnライブラリで提供されている関数に基づく機械学習の概念を示すために作成されているスクリプトを使用します。今回使用するGMMスクリプトの名前はcluster_with_gmm.pyです。

このGMMアルゴリズムはさまざまなハイパーパラメータを受け入れますが、ここで必要なのはデータセットへのパスのみです。

```
bwall@highwind:~/code/private/probability_example$ python cluster_with_gmm.py -h
usage: cluster_with_gmm.py [-h] [-n N_COMPONENTS] [-r]
                           [-c {full,tied,diag,spherical}]
                           dataset

Cluster SMS messages with Gaussian Mixture Models

positional arguments:
  dataset               Path to dataset to read

optional arguments:
  -h, --help            show this help message and exit
  -n N_COMPONENTS, --n_components N_COMPONENTS
                        Number of clusters to produce
  -r, --print-results   Print out results per sample
  -c {full,tied,diag,spherical}, --covariance-type {full,tied,diag,spherical}
                        Covariance type
```

cluster_with_gmm.py をデフォルト設定で実行すると、1つのクラスタのみが生成されます。

```
bwall@highwind:~/code/private/probability_example$ python cluster_with_gmm.py data/SMSSpamCollection
Cluster 0 - Total Samples: 5574 - Percent Ham: 86.5984930032 - Percent Spam: 13.4015069968
```

このクラスタのハムとスパムの比率は興味深いですが、本当に知りたいのはどれがスパムでどれがハムかということです。したがって、n（要素数）を2に設定してコマンドを再び実行します。

```
bwall@highwind:~/code/private/probability_example$ python cluster_with_gmm.py -n 2 -c full data/SMSSpamCollection
Cluster 0 - Total Samples: 4401 - Percent Ham: 98.9775051125 - Percent Spam: 1.02249488753
Cluster 1 - Total Samples: 1173 - Percent Ham: 40.1534526854 - Percent Spam: 59.8465473146
```

クラスタ0は、ほとんどがハムメッセージですから、良好な結果と言えますが、クラスタ1は6：4の割合でスパムが多いため、ハイパーパラメータの設定を見直す必要があります。共分散の設定をtied、diag、and sphericalに変更してみましょう。

```
bwall@highwind:~/code/private/probability_example$ python cluster_with_gmm.py -n 2 -c diag data/SMSSpamCollection
Cluster 0 - Total Samples: 1209 - Percent Ham: 41.7700578991 - Percent Spam: 58.2299421009
Cluster 1 - Total Samples: 4365 - Percent Ham: 99.0148911798 - Percent Spam: 0.98510882016
```

```
bwall@highwind:~/code/private/probability_example$ python cluster_with_gmm.py -n 2 -c spherical data/SMSSpamCollection
Cluster 0 - Total Samples: 3006 - Percent Ham: 98.9354624085 - Percent Spam: 1.06453759148
Cluster 1 - Total Samples: 2568 - Percent Ham: 72.1573208723 - Percent Spam: 27.8426791277
```

```
bwall@highwind:~/code/private/probability_example$ python cluster_with_gmm.py -n 2 -c tied data/SMSSpamCollection
Cluster 0 - Total Samples: 5028 - Percent Ham: 95.7239459029 - Percent Spam: 4.27605409706
Cluster 1 - Total Samples: 546 - Percent Ham: 2.5641025641 - Percent Spam: 97.4358974359
```

tied設定が最も良い結果となり、約95%のハムサンプルがクラスタ0に、97%のスパムサンプルがクラスタ1に割り当てられました。

教師なしでのクラスタリングを実施しましたが、データセットに含まれているラベルから、これらのクラスタ割り当てが非常に正確であることが分かります。最後に各クラスタのサンプルをエクスポートして、実際のメッセージを確認します。ここでは–rフラグ(サンプル毎の結果を出力)を使用します。

```
Cluster 0 - Total Samples: 5028 - Percent Ham: 95.7239459029 - Percent Spam: 4.27605409706
Cluster 1 - Total Samples: 546 - Percent Ham: 2.5641025641 - Percent Spam: 97.4358974359
Cluster 0  -  Go until jurong point, crazy.. Available only in bugis n great world la e buffet... Cine there got amore wat...
Cluster 0  -  Ok lar... Joking wif u oni...
Cluster 0  -  U dun say so early hor... U c already then say...
Cluster 0  -  Nah I don't think he goes to usf, he lives around here though
Cluster 0  -  FreeMsg Hey there darling it's been 3 week's now and no word back! I'd like some fun you up for it still? Tb ok! XxX std chgs to send, £1.50 to rcv
Cluster 0  -  Even my brother is not like to speak with me. They treat me like aids patent.
Cluster 0  -  As per your request 'Melle Melle (Oru Minnaminunginte Nurungu Vettam)' has been set as your callertune for all Callers. Press *9 to copy your friends Callertune
Cluster 0  -  I'm gonna be home soon and i don't want to talk about this stuff anymore tonight, k? I've cried enough today.
Cluster 0  -  I've been searching for the right words to thank you for this breather. I promise i wont take your help for granted and will fulfil my promise. You have been wonde
Cluster 0  -  I HAVE A DATE ON SUNDAY WITH WILL!!
```

```
Cluster 1  -  I don't know u and u don't know me. Send CHAT to 86688 now and let's find each other! Only 150p/Msg rcvd. HG/Suite342/2Lands/Row/W1J6HL LDN. 18 years or over.
Cluster 1  -  SMS SERVICES For your inclusive text credits pls gotto www.comuk.net login 3qxj9 unsubscribe with STOP no extra charge help 08702840625 comuk.220cm2 8AE
Cluster 1  -  Valentines Day Special! Win over £1000 in our quiz and take your partner on the trip of a lifetime! Send GO to 83600 now. 150p/msg rcvd. CustCare:08718720201
Cluster 1  -  For ur chance to win £250 cash every wk TXT: PLAY to 83370. T's&C's www.music-trivia.net custcare 08715705022, 1x150p/wk.
Cluster 1  -  Final Chance! Claim ur £150 worth of discount vouchers today! Text YES to 85023 now! SavaMob, member offers mobile! T Cs SavaMob POBOX84, M26JGZ. £3.00 Subs 16
Cluster 1  -  Urgent! call 09066612661 from landline. Your complementary 4* Tenerife Holiday or £10,000 cash await collection SAE T&Cs PO Box 3 WA14 2PX 150ppm 18+ Sender: Hol Offer
Cluster 1  -  WINNER! As a valued network customer you hvae been selected to receive a £900 reward! To collect call 09061701444. Valid 24 hours only. ACL03530150PM
Cluster 1  -  U have won a nokia 6230 plus a free digital camera. This is what u get when u win our FREE auction. To take part send NOKIA to 83383 now. POBOX114/14TCR/W1 16
Cluster 1  -  Text82228>> Get more ringtones, logos and games from www.txt82228.com. Questions: info@txt82228.co.uk
Cluster 1  -  FreeMSG You have been awarded a FREE mini DIGITAL CAMERA, just reply SNAP to collect your prize! (quizclub Opt out? Stop 80122300p/wk SP:RWM Ph:08704050406)
Cluster 1  -  Congrats 2 mobile 3G Videophones R yours. call 09063458130 now! videochat wid ur mates, play java games, Dload polyph music, noline rentl. bx420. ip4. 5we. 150p
Cluster 1  -  U are subscribed to the best Mobile Content Service in the UK for £3 per ten days until you send STOP to 83435. Helpline 08706091795.
Cluster 1  -  <Forwarded from 21870000>Hi - this is your Mailbox Messaging SMS alert. You have 40 matches. Please call back on 09056242159 to retrieve your messages and matches cc100p/min
Cluster 1  -  FREE RING TONE just text "POLYS" to 87131. Then every week get a new tone. 0870737910216yrs only £1.50/wk.
Cluster 1  -  URGENT! Your mobile No 077xxx WON a £2,000 Bonus Caller Prize on 02/06/03! This is the 2nd attempt to reach YOU! Call 09066362206 ASAP! BOX97N7QP, 150ppm
Cluster 1  -  You are guaranteed the latest Nokia Phone, a 40GB iPod MP3 player or a £500 prize! Txt word: COLLECT to No: 83355! IBHltd LdnW15H 150p/MtmsgrcvdG18
Cluster 1  -  8007 FREE for 1st week! No1 Nokia tone 4 ur mob every week just txt NOKIA to 8007 Get txting and tell ur mates www.getzed.co.uk POBox 36504 W4 5WQ norm 150p/tone 16+
Cluster 1  -  we tried to contact you re your response to our offer of a new nokia fone and camcorder hit reply or call 08000930705 for delivery
```

ほぼすべてのスパムメッセージとハムメッセージが正しく分類されています。

例2:ナイーブベイズによるSMSのスパム識別

今度は教師あり学習アプローチで、データセットに含まれるラベルを使用してスパムを検出します。前回と同じベクトルと文字クラスシーケンスを再利用できます。通常は、サンプルを個別のトレーニングセット、テストセット、および検証セットに分割するところから始めます。ここでは説明を簡単にするために、トレーニングプロセスのみを説明します。

scikit-learnで提供されているMultinomialNBという名前の多項ナイーブベイズ関数を使用してモデルを構築します。今回も必要なパラメータはデータセットへのパスのみです。

```
bwall@highwind:~/code/private/probability_example$ python train_nb.py -h
usage: train_nb.py [-h] [-f] [-a ALPHA] dataset

Train model to classify SMS as spam or ham

positional arguments:
  dataset               Path to dataset to read

optional arguments:
  -h, --help            show this help message and exit
  -f, --fit-prior       Learn class prior probabilities
  -a ALPHA, --alpha ALPHA
                        Smoothing Parameter
```

デフォルト設定（alpha = 1.0、fit_prior = False）で実行すると、クラシファイアは約95%の平均精度を達成します。

```
bwall@highwind:~/code/private/probability_example$ python train_nb.py data/SMSSpamCollection
Predict(Sunshine Quiz Wkly Q! Win a top Sony DVD player if u know which country the Algarve is in? Txt ansr to 82277. £1.50 SP:Tyrone)
[[  1.32093912e-09   9.99999999e-01]]
Predict(What time you coming down later?)
[[ 0.9859347  0.0140653]]
Classification report for testing set
Accuracy: 0.954088952654
             precision    recall  f1-score   support

          0       0.98      0.97      0.97      1202
          1       0.81      0.88      0.84       192

avg / total       0.96      0.95      0.95      1394
```

では、ナイーブベイズにクラスの事前確率を学習させることで精度を改善できるかどうかを見てみましょう。これを行うため、fit_priorハイパーパラメータの設定をTrueに変更します。

```
bwall@highwind:~/code/private/probability_example$ python train_nb.py data/SMSSpamCollection -f
Predict(Sunshine Quiz Wkly Q! Win a top Sony DVD player if u know which country the Algarve is in? Txt ansr to 82277. £1.50 SP:Tyrone)
[[  8.18576852e-09   9.99999992e-01]]
Predict(What time you coming down later?)
[[ 0.99738247  0.00261753]]
Classification report for testing set
Accuracy: 0.963414634146
             precision    recall  f1-score   support

          0       0.98      0.98      0.98      1206
          1       0.85      0.88      0.87       188

avg / total       0.96      0.96      0.96      1394
```

この結果、モデルの平均精度は約96%に向上しました。このモデルを新しいサンプルに適用します。

モデルはメッセージがハムとして分類されるべきであると予測しています。

```
bwall@highwind:~/code/private/probability_example$ python classify_nb.py "I bought a dog"
[ham]  0.978657044375
[spam]  0.021342955625
Prediction: ham
```

このメッセージはスパムとして分類されていました。

```
bwall@highwind:~/code/private/probability_example$ python classify_nb.py "Want to buy Male Verility Treatment! Txt ansr to 12345"
[ham]  1.23879674572e-06
[spam]  0.999998761203
Prediction: spam
```

確率モデリングの重要なポイント

この章では、さまざまな確率の概念を説明し、ナイーブベイズとガウス混合モデルのアルゴリズムが確率手法をどのように利用して分類とクラスタリングを行っているかを実際に示しました。重要なポイントは以下のとおりです：

- 条件付き確率と結合確率の違い。条件付き確率は、イベントAに続いてイベントBが発生する尤度を決定するもので、結合確率は両方のイベントが同時に発生する尤度を決定します。
- ベイズの定理とは対照的に、ナイーブベイズのクラシファイアは、クラスの条件付き独立性を前提とします。つまり、分類の決定において、各特徴は他の特徴に影響しないということです。実際には、この前提はほとんど成立しませんが、ナイーブベイズはわずか4つのコンポーネントのみでサンプルを分類できるため、多くのケースで優れた結果と高い効率を実現します。これらのコンポーネントとは、事後確率、クラスの事前確率、予測の事前確率、および尤度です。
- すべてのガウス分布は、その平均値と分散(標準偏差)によって一意に識別できます。GMMは、反復的な期待値最大化プロセスによってクラスタリングを実行し、各データポイントに対して責任を持つガウス分布を示すスコアを生成します。その結果、GMMは、重複するクラスタや形状が円形ではないクラスタを適切に検出することができます。GMMでは、1

つのハイパーパラメータ(混合数)のみを設定する必要があります。GMMは、それぞれの混合要素について混合比率、平均値、および分散を計算することで、データポイントをクラスタに割り当てます。ただし、GMMは、構成データが連続値で、サンプルがガウス分布に準拠している問題にのみ適しています。

深層学習

これまでの章では、さまざまなの機械学習アプローチを適用してクラスタリングや分類の問題を解決する方法を探りました。詳細は異なりますが、どのプロセスも比較的シンプルな処理シーケンスを実施していました。アルゴリズムはベクトルセットを入力として受け取り、計算を実行して、ラベルまたはクラスタ割り当てを出力として生成します。たとえば、ロジスティック回帰では、分類エンジンは回帰の重みを計算します。クラスタリングを行う場合、k平均法ではベクトルとセントロイドの間のユークリッド距離や他の距離を計算します。

アナリストは、これらの計算の実行方法や生成される結果を、さまざまなパラメータやハイパーパラメータを使用して制御できます。ロジスティック回帰であれば、正規化パラメータやペナルティパラメータを使用して、重みの計算方法を制御することができます。また、各種の最適化関数を使用することで、各重み補正の最小サイズや、十分な収束レベルなどを決定できます。しかしながら、これらは基本的に各アルゴリズムの動作を微調整するための手段であり、燃料の流れや燃焼率を調整して自動車を制御するのと同じです。これらの補助的な関数は不可欠ではあるものの、これらだけでは分類やクラスタリングの決定は行えず、1つのアルゴリズムからの出力を別のアルゴリズムの入力として送ることもありません。

深層学習は、処理の階層を組み合わせ、各階層で異なる計算をするという根本的に異なるアプローチを基本としています。サンプルは階層ごとにステップ単位で処理され、各階層からの出力は次の階層の入力となります。これらの処理階層のうち、最低1つは「隠れ」ます。深層学習が他のすべての機械学習手法と大きく異なる点は、この隠れ層を持つ多層アプローチです。

深層学習という用語は、ニューラルネットワークを中核として使用する幅広い教師なし、半教師あり、教師あり、および強化学習方法を指します。ニューラルネットワークはアルゴリズムのクラスであり、その名前は、人間の脳内で密接に相互接続されたニューロンの働きを模していることに由来します。

1950年代に初めて考案されたニューラルネットワークは、人間のように「考えて」問題を解決できる「インテリジェントマシン」を生み出す可能性から、1980年代に大きく注目されました。しかし、1980年代の終わりまでに、この新しい技術への関心は薄れて行きました。当時のコンピュータは貧弱で、複雑で興味深い問題を解決するために必要な負荷を処理しきれなかったのです。また、研究者達は、ニューラルネットワークモデルを適切にトレーニングするのに必要な大量のデータセットを取得することが難しいことにも気付きました。その結果、多くの研究者は、データセットや処理負荷の観点からより手軽な機械学習アプローチに目を向けたのです。

ニューラルネットワークへの関心が再燃したのは2005年から2006年に掛けてのことで、Geoff Hinton、Yoshua Bengio、Yann Lecun、そしてJurgen Schmidhuberらの人工知能の研究者達が、ニューラルネットワークは実現可能であるばかりでなく、音声認識や画像分類といった複雑な問題を既存の手法よりはるかに正確に解決できることを実証したことが発端となりました。ゲーム機などに組み込まれているグラフィック処理ユニット(GPU)は、すでにニューラルネットワークのアルゴリズムを効率的に、そして手頃なコストで実行できるだけの能力を備えていました。また、インターネットの爆発的な成長により、大量のオンラインデータベースも利用できるようになっていました。そして、GoogleやMicrosoftと

いったソフトウェアの巨大企業は、深層学習の研究に巨額の投資を始めました。その後すぐに、深層学習技術を組み込んだ新製品が次々と登場したのです。現在、ニューラルネットワークはGoogleのAndroid OSやAppleの音声認識アシスタントSiriに応用されています。深層学習は、幅広いネットワークセキュリティ問題への対応においても非常に効果的であることが示されています。

この章では、次の2種類のニューラルネットワークアルゴリズムについて説明します。

1．長・短期記憶(Long Short-Term Memory: LSTM) － 再帰型ニューラルネットワーク (Recurrent Neural Network: RNN) の一種
2．畳み込みニューラルネットワーク (Convolutional Neural Network: CNN)

注意：ニューラルネットワークは極めて柔軟性の高い汎用アルゴリズムであり、無数の問題を無数の方法で解決できます。たとえば、他のアルゴリズムとは異なり、ニューラルネットワークは数百万あるいは数十億ものパラメータを適用してモデルを定義できます。説明を簡単にするため、ここでは代表的なハイパーパラメータを使用して、分類の問題を解決するためのニューラルネットワーク機能のみを取り上げます。最後の実用的な実習例では、LSTMモデルとCNNモデルを使用して、テキストのサンプルを難読化するためのXORキーの長さを決定します。

汎用ニューラルネットワークアーキテクチャ

図4.1に示すように、ニューラルネットワークは、入力層、隠れ層、および出力層に含まれるノードから構成されています。

各階層は、分類の計算において特定の役割を果たします。この図のような全結合ネットワークでは、特定の階層のすべてのノードからの出力は、その後の階層のすべてのノードの入力に接続されています。これは順伝播 ニューラルネットワークの例でもあり、情報が1つの階層から次の階層に直接渡され、後戻りすることなく、分

類の決定が割り当てられる出力層まで送られます。

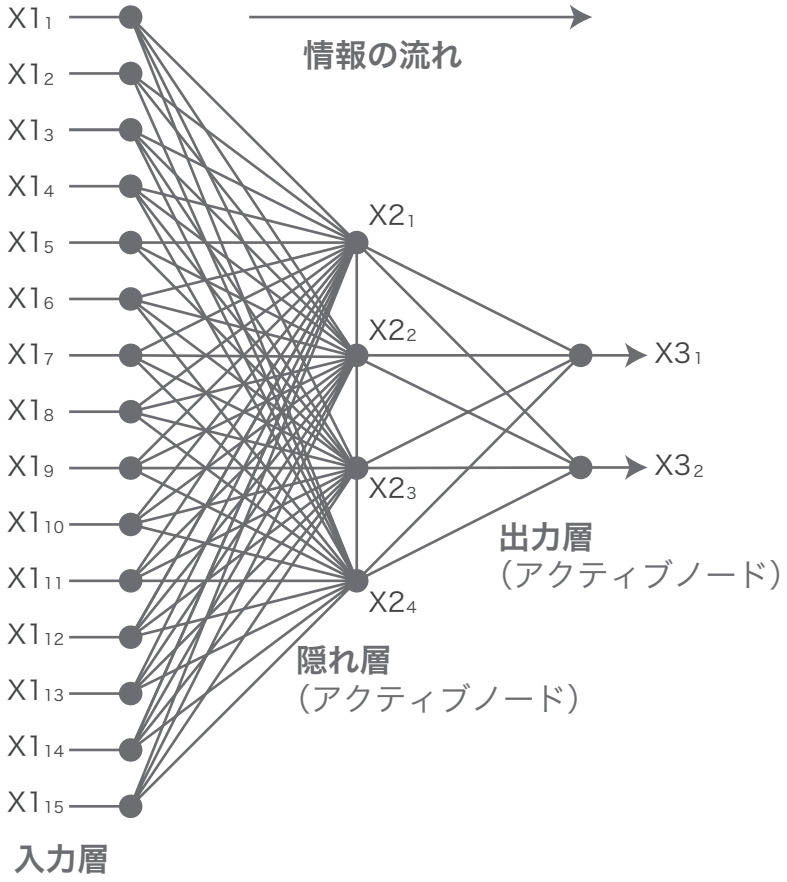


図4.1: 汎用ニューラルネットワークアーキテクチャ

これは多くの実装可能な構成の一例に過ぎません。また、ニューラルネットワークでは、階層間でフィードバックループを取り入れて、情報の流れを特定のノードに限定する部分的結合アーキテクチャを利用することもできます。とは言えまず最初は、順伝播型の全結合ニューラルネットワークについて、構成する階層をみていきましょう。

入力層

入力層のノードは受動的です。特定サンプルの属性値を受け取っ

たら、処理を行う最初の隠れ層の全ノードに渡すだけです。そのため、入力層には、サンプルセットのすべての特徴について1つのノードが存在しなければなりません。たとえば、解像度が64x64画素の画像を分類する場合、入力層には各画素に1つずつ、合計4,096の入力ノードが必要です。言語処理の問題を解決する場合なら、解析対象のサンプルに含まれる固有の単語の数や、各単語の出現頻度などが関連する特徴となります。

隠れ層

隠れ層は、図4.2に示すようなノードで構成され、深層学習プロセスにおける重要な仕事を行います。これは最初の隠れ層の最初のノードであるため、処理は次のように進みます。

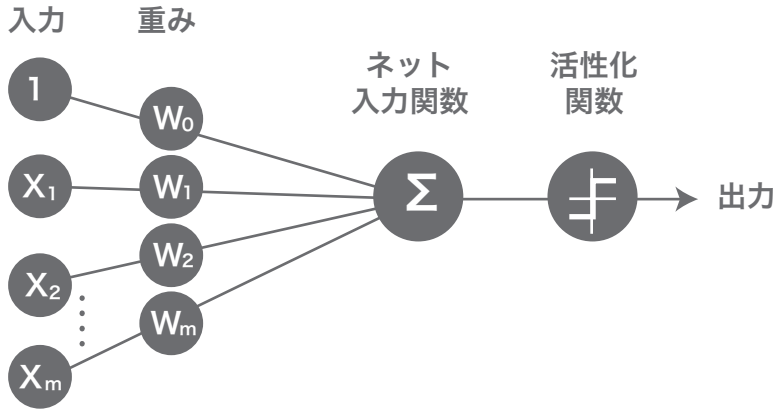


図4.2：隠れ層1のノード1

- **入力層から特徴値を受け取る：**最初のサンプルのすべての属性値はノードの入力x^1～x^mで受け取ります。
- **重みを適用する：**各属性値に対応する重み値を掛けます。たとえば、入力x^1の属性値に重みw^1を掛け、入力x^2の属性値に重みw^2を掛けます。 重み値が1より大きい場合、最終的な分類の決定に対するその特徴の貢献度は徐々に大きくなります。重み値が1より小さい場合は貢献度が小さくなります。この処理は、ロジスティック回帰で回帰の重みを利用し

たのと似ています。ただし、ニューラルネットワーク場合は、特定の隠れ層での処理だけではなく、すべての隠れ層での総計的な処理によって分類が決定します。

特徴重みw_1～w_mに加えて、w_0というラベルの重みもあります。これはノード1の偏りまたはy切片値です。これまでの章で見てきたように、偏りは、特徴空間における決定境界を、y軸原点からサンプルデータに最適な位置まで移動します。

隠れ層のすべてのノードに対するすべての入力は、異なる重み値で初期化され、この値は求められる精度レベルに達するまで徐々に最適化されます。アナリストは、初期の重み値をランダムに設定するか、関数を利用しておよその初期値を概算するか、もしくは以前に同様の問題とデータセットを扱った経験を元にして設定します。

- **積を合計する：**積は、ネット入力関数に送られ、積和が計算されて、その結果が活性化関数でさらに処理されます。
- **活性化関数を適用する：**活性化関数は、その階層に対して指定されている特定の計算を実行します。アナリストは、問題の性質と、ソリューションを導き出すのに必要な計算シーケンスに基づいて、大量の活性化関数の中から必要な関数を選択できます。
- **結果を出力する：**この活性化関数の結果は、そのノードの処理を合計した効果を反映した数値です。これらのそれぞれの値は、重みと特徴属性の組合せの異なる比率を表します。これらの組合せをすべて処理し、結果を後続の隠れ層に送ることで、ニューラルネットワークは、サンプルのクラス割り当てを最も正確に予測できる特徴と重みの組合せを、ステップバイステップで決定することができます。

隠れ層2～nでの処理

隠れ層2の各ノードは、隠れ層1のすべてのノードからの出力値を受け取ります。ここでも、各値には特定の重み値が掛けられ、積

が合算され、その結果が活性化関数に送られて、新しい出力が生成されます。新しい出力は次の階層に送られて同じように処理されます。この処理は、すべての隠れ層を通過して、計算の結果が出力層に届くまで続けられます。

出力層

出力層は、ニューラルネットワークの最後の階層です。この章では分類の実習を行っていますので、出力層には、すべての可能なクラス分類それぞれについて1つのノードがあります。隠れ層のノードと同様に、これらのノードもアクティブであり、活性化関数を組み込んでいます。たとえば、分類決定を確率スコアに変換するためには、logitまたはsoftmax活性化関数を使用できます。割り当てるクラスラベルは、スコアが最も高いノードによって決定されます。

各トレーニングサイクルの後で、loss関数によって分類決定をクラスラベルと比較して、より正確な結果を導き出すには、すべての隠れ層の重みをどのように修正すべきかを決定します。

この処理は、候補モデル群が検証およびテストフェーズに進めるようになるまで、何度も繰り返されます。

注意：最適化アルゴリズムが重みの修正値を計算して適用するために逆伝播プロセスがどのように働くのか、またLSTMモデルとCNNモデルのトレーニングにおいてこれらのプロセスの適用方法がどのように異なるかについては、本書の範疇を超えています。もっと深く学習したい方は専門の技術書籍を読まれることをお勧めします。

抽象レベルの増大

前述のように、サンプルの特徴値は、入力層と最初の隠れ層のノードにしか認識されません。後続のすべての階層では、1つ前の階層のノードからの組合せ出力値しか「認識」できず、抽象レベルが徐々に増大した全体像としてしかサンプルを「観察」できません。これは、人間の脳が知覚入力を理解して解釈する過程と似ています。たとえば視覚であれば、入力層の役割を果たすのは網膜であ

り、網膜は光子の「サンプル」から受け取った光エネルギーの強さに応じた電気信号を視覚野に送ります。その後、複数の「隠れ層」が異なる種類の視覚「活性化関数」を適用します。たとえば、1つの階層ではエッジ処理を行い、別の階層ではエッジを形状として統合し、さらに別の階層ではその形状を「顔」などのカテゴリーに分類します。ニューラルネットワークは、このような処理を非常に微細な方法で行うことができ、順序付けられた多層計算を通して低レベルの信号から複雑な決定までを処理します。

必要な隠れ層の数

一般に、隠れ層が多く、使用する全体の容量が大きいほど、ニューラルネットワークが解決する問題も複雑になり、大量のデータを扱えるようになります。しかし何にでも言えることですが、ここにもトレードオフがあります。過剰な容量のニューラルネットワークは、データを過剰適合するモデルを生成しがちですし、逆に容量が不十分なニューラルネットワークは、誤差が許容範囲を超えるほど大きくなるモデルを生成しがちです。

通常、アナリストは、データの過剰適合が観察されるまでモデルの容量を増やし、必要に応じて階層数を減らします。階層とノードの密度は、不要な、あるいは冗長な階層を除去するような補助的アルゴリズムを適用するプログラムによって修正することもできます。特定のメカニズムは若干異なるものの、その効果は、ハイパーパラメータを適用してディシジョンツリーから不要なブランチを刈り込む方法と似ています。

長・短期記憶（LSTM）ニューラルネットワーク

これまでの説明では、図4.1に示すような順伝播の全結合ニューラルネットワークを取り上げてきました。このアーキテクチャは、処理対象のサンプルが入力層に到達する順序を考慮しなくてもよい問題を解く場合に適しています。たとえば、一連の動物の画像を分類する場合には、最初の画像が猫として分類されたという事実が、2番目の画像が犬として分類されることにどのように影響するかを

考慮する必要はありません。順伝播ネットワークには、時間という考えがありません。常に今の瞬間のみを扱い、以前に処理したサンプルを「覚えておく」能力はありません。

しかしながら、現実の多くの問題では、1つのサンプルの持つ意味が、その後で発生する別のサンプルの意味にどう影響するかを考慮しなければなりません。たとえば、映画の1コマを見て、そのストーリーを推測することはできません。時系列に沿った一連のコマを見ることで、始めて意味とコンテキストが発生するのです。同様に、1つのパケットを調べるだけでは、疑わしい接続を特定することはできません。時系列に沿った一連のパケットを調べることで、エクスプロイトが発生したかどうかを見極めることができるのです。このような時系列の問題には、再帰型ニューラルネットワーク(RNN)がより適しています。

RNNは、順伝播型のニューラルネットワークとは異なり、現在の入力だけではなく、直前の入力との関係まで考慮することができます。そのため、再帰型ネットワークは「記憶」を持つと言われます。再帰型ネットワークは、フィードバックループを使用して、これを実現しています。図4.3に示すように、出力層からのサンプル値は、最初の隠れ層のノードにコピーされます。コピーされた値は、時系列で次のサンプルの入力値と組み合わされます。すべてのサンプルが処理されるまで、このプロセスが繰り返されます。

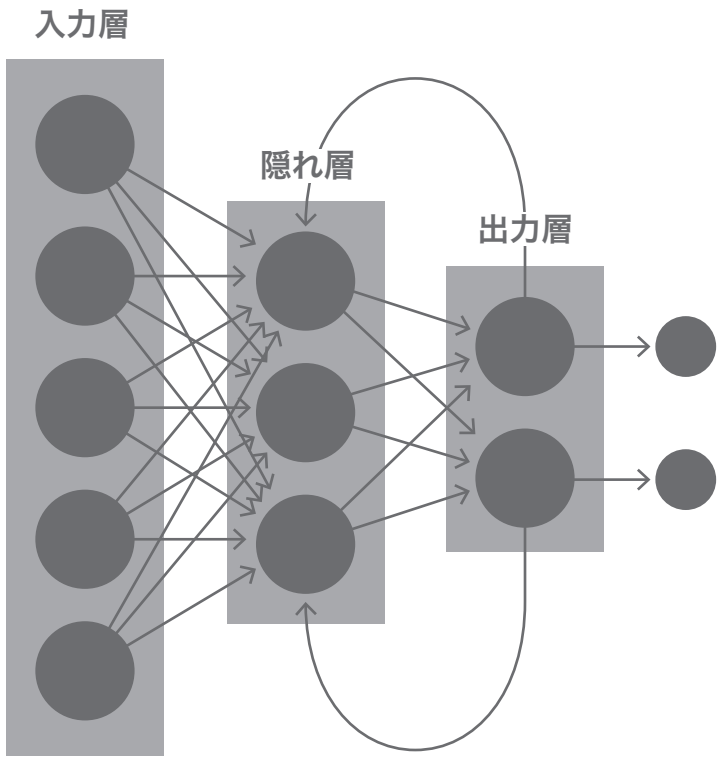


図4.3：シンプルな再帰型ニューラルネットワーク

これをもう少し技術的に説明すると、再帰型ネットワークはサンプルの隠れ状態を追跡します。1つの隠れ状態は、隠れ層のすべての値から成ります。図4.4に示すように、再帰型ネットワークは、先行する時間ステップの状態を、現在の時間ステップの隠れ状態への入力として利用します。これにより、ニューラルネットワークは、1つの時間ステップから次の時間ステップにおいて隠れ状態がどのように変化するかを追跡できます。

最もシンプルなRNNでは、1つのシーケンスで記憶できるタイムステップ数が限られています。つまり、シンプルなRNNではコンテキストの維持が難しいのです。たとえば、疑わしいを分類するために、RNNがシーケンスの最初のパケットから10番目のパケットまでを記憶できなければならないような場合、シンプルなRNNが9番目までのパケットのコンテキストしか維持できないとしたら、分

類は失敗します。この制限に対応するためには、RNNアーキテクチャを修正して長・短期記憶ユニットを取り入れた、LTSMニューラルネットワークというアーキテクチャが生まれました。

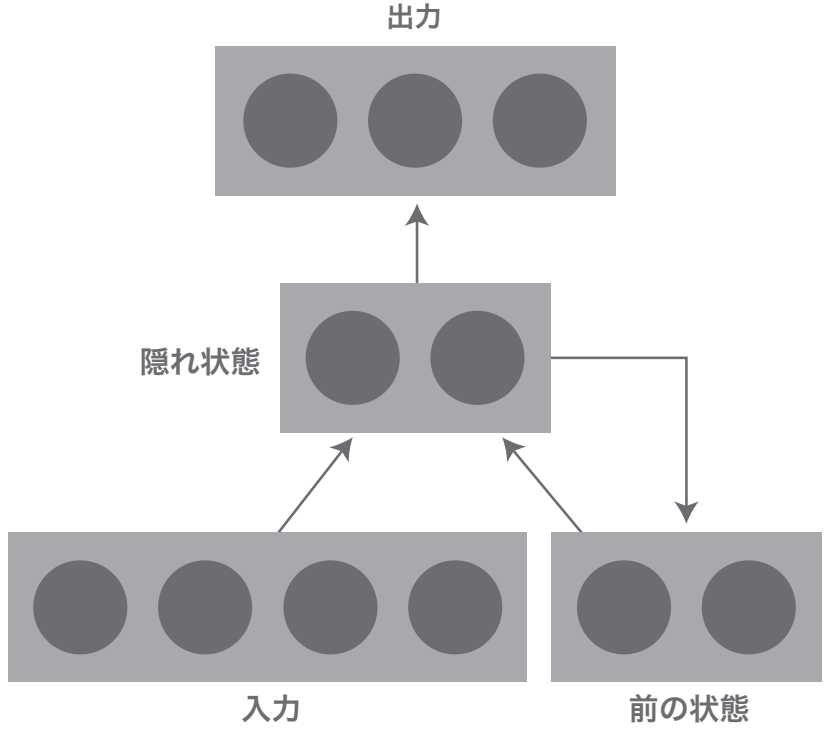


図4.4：隠れ状態の追跡

図4.5に示すように、LSTMネットワークでは、それぞれの隠れ層のノードはメモリブロックに置き換えられ、各メモリブロックには1つまたは複数のメモリセルが入っています。隠れ層ノードとは異なり、メモリブロックは活性化関数には関連付けられません。その代わりに、メモリブロックはゲートを利用して、各セルに格納されている状態をいつどうやって更新し、後続の隠れ層のメモリブロックに渡すかを決定します。

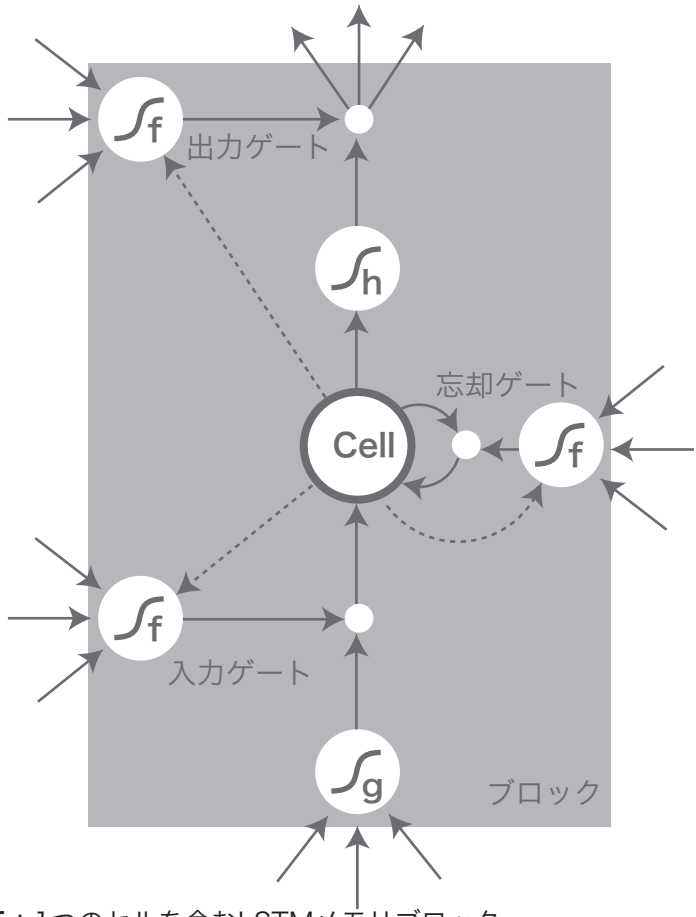


図4.5：1つのセルを含むLSTMメモリブロック

3種類のゲートがあり、それぞれ異なる重み設定が適用されています。

- 忘却ゲートは、各セルの現在の状態を重み値1で再帰的に乗算します。これにより、新しいサンプルやフィードバックループからの状態値が供給されるまで、セルの現在の状態を無限に維持できるという効果があります。忘却ゲートの重み値を高く設定すると、新しい状態情報がセルに書き込まれ、現在の状態値を部分的に、あるいは完全に上書きします。低く設定すると、現在の状態値はほとんど変化しません。
- 入力ゲートが開くと新しい状態値が受け入れられ、閉じると現在の状態値が維持されます。
- 出力ゲートが開くと現在の状態値が次の隠れ層のメモリブロックに渡され、閉じている間は渡されません。

LSTMは、ゲートを利用することで、時系列データの非常に長いシーケンスに渡って、関連情報を維持または再利用できます。このプロセスは、人間の学習とある意味では似ています。脳内では、ニューロンの「メモリブロック」から次のニューロンへ情報を渡すかどうかを「ゲート」によって決定します。意味のある情報を渡すニューロンのみが強力な結びつきを許され、それぞれの状態値を次の隠れ層のメモリブロックに渡すことができます。

畳み込みニューラルネットワーク（CNN）

前述のニューラルネットワークとは異なり、CNNは全結合アーキテクチャを使用しません。その代わりに、それぞれの畳み込み階層は、自分の階層にあるノードを、前の階層の連続したノード群にのみ接続します。図4.6の例では、階層2のノード1は、階層1のノード1～3にのみ接続されています。技術的な観点から言えば、ノード1はこれら3つのノードの関数です。同じ階層のノード2とノード3についても同様です。それぞれのノードは、階層1でそれぞれが接続されている連続ノードの関数です。

これらの接続は、CNNのハイパーパラメータであるsizeとstrideを使用して定義します。sizeは、このような接続における連続ノードの数を指定します。strideは、スキップするノード数を指定しま

す。図4.6の例では、sizeは3、strideは1に設定されています。その結果、階層1のノードは1つもスキップされていません。

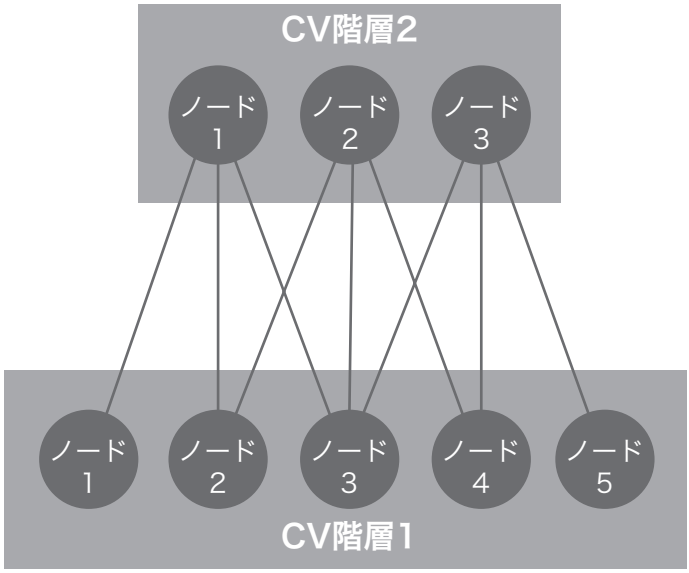
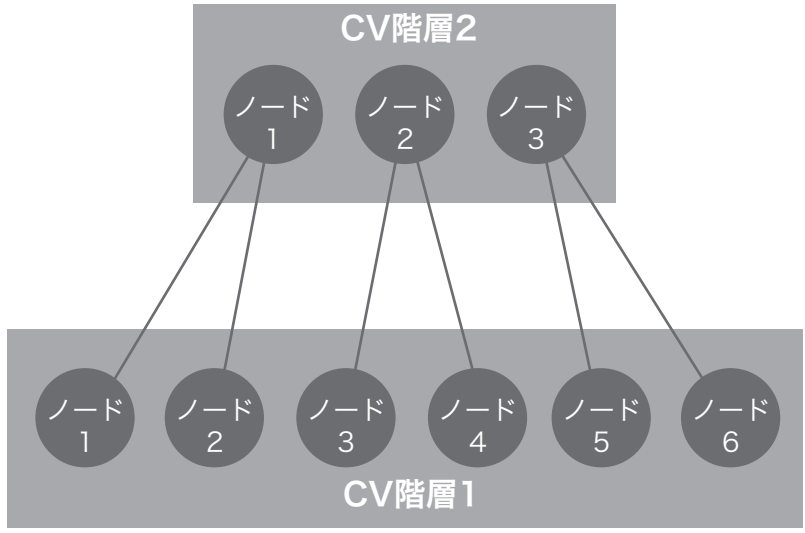


図4.6：Size=3、Stride=1

これに対して図4.7では、sizeとstrideはどちらも2に設定されています。この結果、階層2の各ノードは、階層1の2つのノードの関数となっています。また、strideの設定により、階層2のノード2は、階層1の最初の2つのノードをスキップしてノード3〜4に接続されています。そして、ノード3はノード5〜6に接続されています。

隠れ層の各ノードには固有の重みが割り当てられ、全階層の重み値の合計によって正しい分類決定が導き出されるまで、重み値が徐々に調整されるということを以前に説明しました。CNNでは、これらの重み値はそれぞれの接続ノード群にのみ“適用”されます。



階層2の各ノードは
階層1の2連続ノードにのみ接続されている

図4.7：Size=2、Stride=2

重みの拘束により、計算時にはこれらのすべてのノードに同じ重み値が適用されます。この結果、CNNモデルが計算しなければならないパラメータ数が大幅に削減されます。図4.8に示すCNNの例では、重み値は2つ（W_1とW_2）しかありません。同じノード数の全結合アーキテクチャでは、ネットワークは18もの重み（階層1の6つの重み*階層2の3つの重み）を計算しなければなりません。W_1やW_2のような拘束重みセットはフィルターと呼ばれます。すぐ後で説明しますが、フィルターは極めて便利です。では、情報が1つのCNN 階層から次の階層にどのように流れるか、そして分類決定において各階層がどのような役割を果たすのかを考えてみましょう。ここでは、画像分類の観点からプロセスを調べます。入力データは2次元の画素値の行列で、0は白、1は黒の画素をそれぞれ表します。プロセスは、自然言語処理や他の1次元データ問題と同じように適用されます。XORキー長を検出した問題と同じように、入力はバイナリ文字のシーケンスとなります。

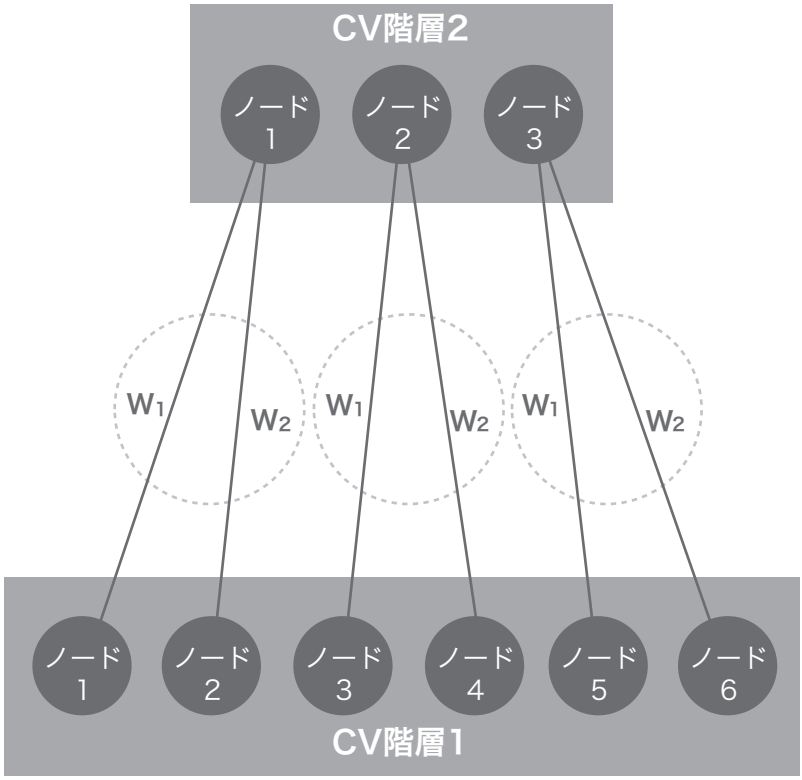


図4.8：接続ノード間で拘束されている重み

畳み込み階層

前の説明では、フィルターとそのsizeおよびstride設定によって、隣接する畳み込み階層で接続されるノードと接続されないノードがどのように決定されるかについて述べました。フィルターは、特徴を識別し、それらの値を合算して漸進的な抽象表現へと組込み、分類決定を導き出すというプロセスにおいても中心的な役割を果たします。

図4.9は、5x5画素の白黒画像に対応する特徴値の行列を示しています。その隣には、3x3の受容野と、初期重みの値が示されています。まず、最初の画素群の上でフィルターを”スライド”させてから、画素値に対応するフィルターの重み値を掛けます。

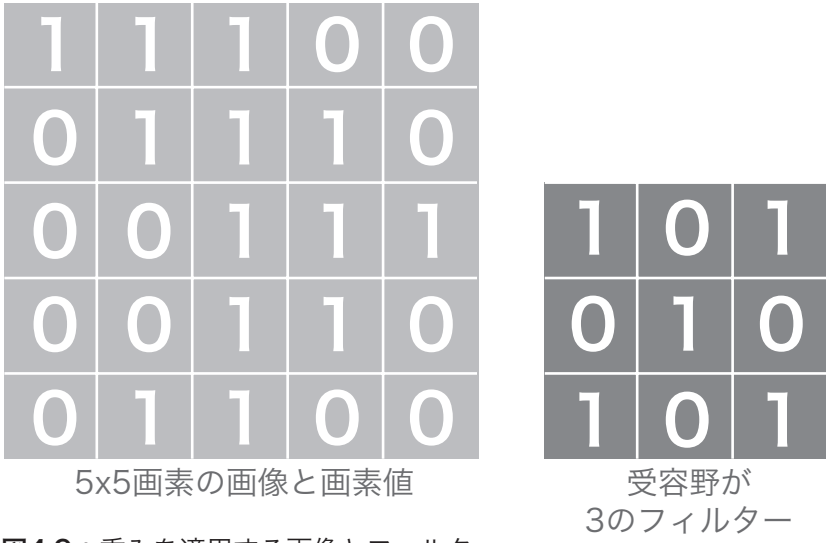


図4.9：重みを適用する画像とフィルター

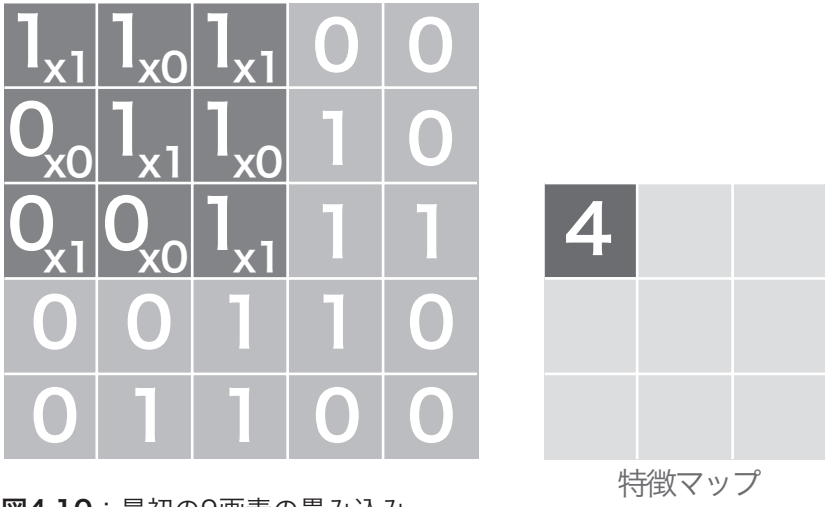


図4.10：最初の9画素の畳み込み

次に、この要素単位の乗算の結果を合算して、関連する特徴マップの最初のセルに結果を挿入します。特徴マップが完成したら、得られた活性化値を次の隠れ層の接続ノードに出力して、次の処理ステージに送ります。最初の畳み込みで得られた活性価値は図4.10のとおりです。

この説明のCNNは、strideが1に設定されています。したがって、図4.11に示すように、フィルターを1ピクセル右にスライドさせることで、次の畳み込みを実行します。今回も、特徴マップに活性化を記録します。

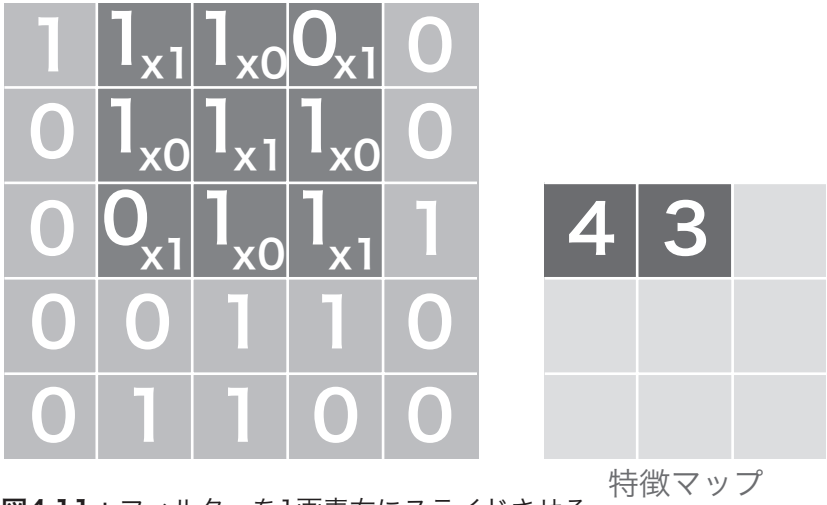


図4.11：フィルターを1画素右にスライドさせる

同じように画像の横方向と縦方向に処理を進めて、図4.12に示すように特徴マップを完成させます。特徴マップが画像をより抽象的に表現し、以降の階層が処理するパラメータを減らしている点に注目してください。

各畳み込み層では数百から数千もの異なるフィルターを使用するのが普通で、各フィルターは異なる特徴や特徴の組合せを検出します。たとえば、1つのフィルターセットはエッジを検出し、別のフィルターセットはさまざまな向きの曲線を検出します。

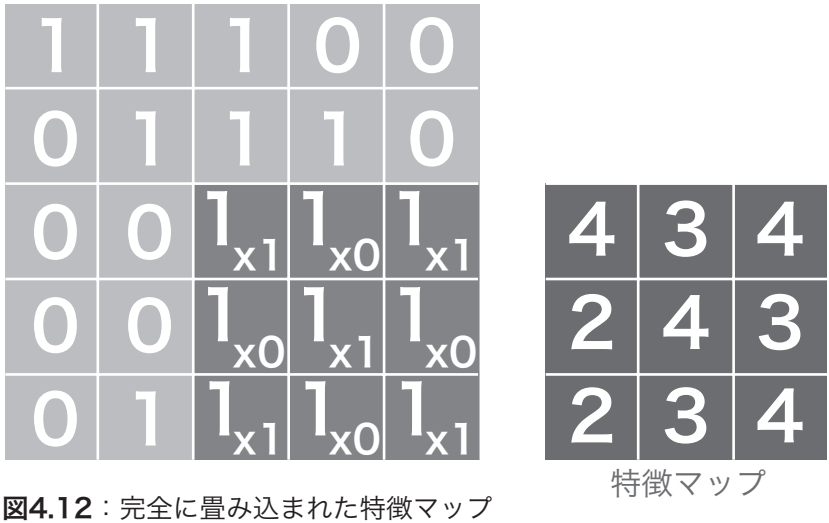


図4.12：完全に畳み込まれた特徴マップ

すべての活性化を計算したら、結果を次の階層の接続ノードに出力して次の処理に回します。

バッチ正規化層

1990年代の後半、アナリスト達は、モデルへの入力を正規化することでニューラルネットワークの最適化が容易になることを発見しました。モデルへの各入力は、平均値が0で、標準偏差が1です。前の階層の活性化も同様に正規化されているという前提に基づき、ランダムな初期重み値を設定するための多くの戦略が構築されました。しかし現実には、この前提が成り立つことはほとんどありません。

最適化をスピードアップするためには、各畳み込み層の後にバッチ正規化層を設けることができます。バッチ正規化により、平均値0、標準偏差1を実際に強制することなく、隠れ活性化が正規化に近い値を取るようになります。実際に、バッチ正規化によってトレーニング時間を1/10以下に短縮することができます。

正規化線形(RECTIFIED LINEAR UNIT: RELU)階層

説明を簡単にするため、フィルターの重みと生成される特徴マッ

プでは正の値を使用します。ただし、ロジスティック回帰の説明で述べたように、重みは負の値も取り得ます。CNNでは、これらの負のパラメータを計算するために必要な追加の処理では、モデルの精度や性能を高めるような意味のある結果は生まれません。そのため、図4.13に示すように、各畳み込み層の後にRELU層を組み込んで、これらの負の値を0に変換します。

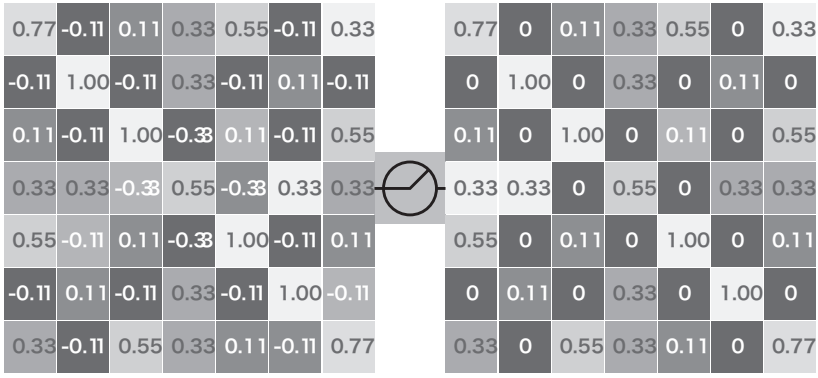


図4.13： RELU層は負の値を0に変換する

プーリング層

より多くのフィルターを使用するほど、後続の階層で処理しなければならない活性化関数も多くなります。プーリング層は、この階層を通って後続の階層に到達できる活性化関数を制限することによって、この処理を減らし、過剰適合の可能性も低減します。このプロセスは、入力のサブサンプリングとも呼ばれます。プーリングを実装するための最も一般的な手段の一つは、max pooling活性化関数であり、以下のように処理が進みます。

図4.14：入力特徴マップへの最大プーリングの適用

まず、入力された特徴マップにフィルターを適用し、その受容野内にある活性化関数を評価します。通常は、受容野が2x2、stride値が2のフィルターが使用されます。max poolingで1-maxが指定されている場合、その受容野で最大の値となる活性化関数のみが、縮小された新しいマップにコピーされます。次に、フィルターを次の場所（この例では2つ右のセル）まで滑らせて、同じプロセスを繰り返します。これを新しい特徴マップが完成するまで続行します。図4.14にこのプロセスを示します。このようにプーリングを行うことで、計算すべき重みの数を16から4まで減らしています。

多くの場合、最終的な全体最大プーリング層を組み込むことで、入力された活性化関数を出力層のノードに渡す前に、サブサンプリングに掛けます。たとえば、図4.14に示されている元の特徴マップに対して全体最大プーリングを適用すると、出力値は8のみとなります。

出力層

出力層は、1つ前のすべてのノードに全結合されています。各出力ノードは、入力として合算された活性化値を受け取り、それによって分類を決定します。softmax活性化関数を使用して、このクラス分類に確率スコアを割り当てることもできます。

典型的な深層学習の分類セッション

典型的なアナリストが深層学習の分類セッションの各ステージをどのように進めていくのかを考えてみましょう。これらのステップのいくつかは、すでに説明したものと同じです。

サンプルの取得とベクトル化

ステップ1：サンプルの選択

今回も、代表的なサンプルデータを選ぶところから始めます。これは教師あり学習セッションであるため、データセットには各サンプルのクラス所属を定義するラベルが含まれます。

ステップ2：サンプルのサブセットへの分割

次に、合計データセットの70～90%をモデルのトレーニング用に選り分けます。用途によっては、残りのサンプルも検証とテスト用に分割する場合もあります。分割後のセットには、トレーニングサイクルが完了するまでは、モデルを適用しません。

検証セットとトレーニングセットは、若干異なる目的で使用します。一般に検証セットは、いくつかのモデルの精度と性能を比較して、実際に使用するモデルとテストに掛けるモデルを選り分けます。検証ステップは、過剰適合が発生しているかどうかを見極めるのにも便利です。トレーニングセットでのモデルの精度が、検証セットに適用した場合よりはるかに高い場合には過剰適合が明らかになります。

それでも、トレーニングセットと検証セットが構成データ環境の本質を正確に反映していない場合は、モデルの選択プロセスが偏ることがあります。この場合は、選択したモデルをテストデータセットに適用し、さまざまな手法で精度を評価します。検証フェーズとテストフェーズの結果に性能や精度のばらつきがある場合は、モデルの性能が不十分であるため、再トレーニング必要であると考えることができます。

余談ですが、テストセットに含めるデータの選択は、任意である

必要はありません。たとえば、トレーニングセットや検証セットのサンプルより分類が難しいサンプルをあえて集めてモデルをテストすることもできます。

手順3：特徴の抽出とベクトル化

特徴の抽出と処理(ベクトル化)は、機械学習プロセスにおいて最も複雑で難しく、時間の掛かる作業です。特徴の定義が貧弱であったり、主要な属性が除外されていたりすると、生成されるモデルは本質を反映せず、現実世界のデータに適用しても適切に機能しません。幸い、ニューラルネットワークには、特徴を自動で抽出し、トレーニングデータを未加工の表現に近い状態でも受け入れることのできる能力があります。たとえば、バイナリデータのシーケンスを学習することが目標でれあば、これらのシーケンスに関する特徴を抽出する必要はありません。その代わりに、未加工のバイナリデータを直接ニューラルネットワークに送って、どのシーケンスが有意であるかをネットワークに判断させることができます。いずれにしても、トレーニング用のサンプルをニューラルネットワークの入力層に読み込ませるためには、サンプルを別々のベクトルとラベルの行列として再構築しなければなりません。

モデルの選択とインスタンス化

サンプルを読み込んだら、モデルの選択とインスタンス化を行います。選択するモデルの種類とそのアーキテクチャは、解析するデータの種類と問題の性質によって決定します。たとえば、画像データを分類する場合には、畳み込みアーキテクチャが適しています。また、テキスト、バイト、ネットワークキャプチャデータなどのシーケンシャルなデータを分類する場合は、再帰型ニューラルネットワークが適しています。すでに特徴の抽出が終わっていて、いくつかの異なる種類のモデルの性能を比較したいだけであれば、全結合ネットワークを使用します。ここでは、適切なアーキテクチャの選択が終わっていて、モデルのトレーニングを開始できる状態であるとします。

モデルのトレーニング

このフェーズでは、モデルによってデータのトレーニングを行い、モデルの精度と性能が受け入れられるレベルに達するまで、それぞれの隠れ層の重みを徐々に最適化します。モデルのアーキテクチャを定義し、データセットを含むディレクトリへのパスを指定したら、トレーニングを開始できます。デフォルトのハイパーパラメータで問題ないという前提であれば、単純に次のような関数を実行します：

```
model.fit(train_x,train_y, nb_epochs=10)
```

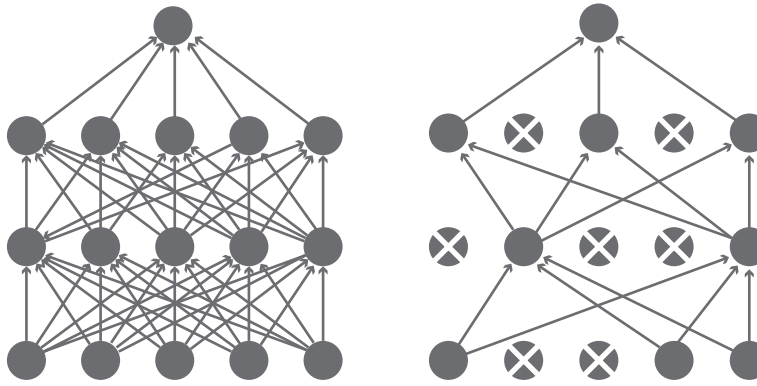
ここで、train_xはトレーニングデータの行列、train_yはトレーニングラベルのベクトル、そしてnb_epochsはトレーニングを停止してモデルの精度と性能の変化を評価するまでに完了しなければならないエポック数をそれぞれ表します。エポックとは、モデルをトレーニングセット全体に適用する計算サイクルを意味します。

モデルがトレーニングセット全体を処理するには長い時間が掛かります。したがって、多くの場合は小さいサブセット（バッチ）を使用してトレーニングを行った方が効率的です。各バッチは処理の時間が短くて済むため、各バッチの結果を検証セットの断片に適用することで、モデルがどのように改善されつつあるのかをより早く知ることができます。モデルが期待どおりの性能を示していないと判断した場合は、ハイパーパラメータ設定を変更して、トレーニングプロセスをやり直すことができます。後ほど、LSTMおよびCNN深層学習セッションでXORキー長を分類する実習を行う際に、実際にバッチアプローチを使用します。

ニューラルネットワークには、モデルの構成に影響するハイパーパラメータが無限にあるように見えます。最も一般的なハイパーパラメータは以下のとおりです：

- **隠れ層の数**：誤差率が過剰であればこの数を増やし、過剰適合が疑われる場合は減らします。

- **隠れ層あたりのノード数**：通常はデフォルト設定を使用し、各隠れ層のノード数は同じになります。隠れ層ごとにノード数を変えることが効果的だと思われる場合は、異なるノード数を割り当てることができます。
- **学習速度**：最適化において重みを増分的に調整する量を指定します。つまり、重み空間においてモデルが収束に向かう速度です。学習速度が高すぎると、重み値は重み空間内で定まらなくなり、収束に至ることができません。学習速度が低すぎると、最適化アルゴリズムは重みを微細な量しか修正しなくなるため、「収束に至る時間」が不要に長くなります。モデルを合理的な時間内にトレーニングしたければ、「ちょうどよい」値を見つけるための学習速度を試す必要があります。



(a)標準的なニューラルネットワーク　(b)ドロップアウトの適用後

図4.15：ドロップアウト正則化の適用前と適用後

- ドロップアウト正則化：ニューラルネットワークは、その複雑さと能力のため、すべての隠れ層のそれぞれの重みをすべての入力サンプルに特有の値に設定してしまい、結果としてトレーニングデータを過剰適合してしまうことがあります。これは、トレーニングセットのすべてのサンプルに対してノードを作成してしまうディシジョンツリーと同じです。これを防止するため、ドロップアウト正則化ハイパーパラメー

タを適用して、各エポックにおいて、各隠れ層内のノードの一部をモデルに無視させます。ドロップアウト正則化による効果を図4.15に示します。

このハイパーパラメータは、それぞれの隠れノードがサンプルの処理中に無効化される尤度を示す確率値で表されます。たとえば、ドロップアウト設定値が0.2であれば、隠れノードの20%がランダムに無効化されることを意味します。ドロップアウト設定値が低いと、トレーニングセットのすべてのニュアンスを取り込む非常に表現力に富んだモデルが生成されますが、過剰適合を起こしやすくなります。0.8などの高い値に設定すると、多くのドロップアウトが発生するため、モデルの精度が下がります。実際には、デフォルト値の0.5から開始して、値を上げたり下げたりしながら結果を評価します。

トレーニングでは、これらや他のハイパーパラメータに幅広いランダムな設定を適用して、後続の検証およびテストフェーズで評価するために数十種類のモデルを生成するのが一般的です。

モデルの検証

前述のように、バッチ単位でトレーニングプロセスと検証プロセスを交互に行うか、あるいはトレーニングが完了するのを待ってから複数のモデルを検証することができます。分類の章で説明した精度や再現率といった測定値など、さまざまな検証手法や関数を利用できます。いずれにしても目的は、トレーニングデータを過剰適合してしまうモデルを不用意に選ばないようにすることです。そのため、トレーニングしたモデルを検証データセットに適用し、性能と精度の最良の組合せを実現するモデルを選んでテストします。

モデルのテストと展開

「最良の」モデルを選べたら、今度はテストデータに適用して結果を測定します。現実世界でのモデルの実力を評価するため、テスト

データに特に難しいサンプルばかりを入れることも珍しくはありません。

たとえば、目標がXORで難読化されたバイトシーケンスのキー長の検出であれば、特に解読が難しいサンプルをテストセットに入れることもできます。もしくは、XORした英語のASCII文字でトレーニングしたモデルを、XORしたbase64エンコードテストデータに適用することで、異なる種類のバイトシーケンスの分類にどの程度応用できるかを評価できます。

実用的な学習の実習

これまでの学習内容を応用して、バイナリデータのシーケンスを解析することでLSTMとCNNが分類の問題をどのように解決できるかを見てみましょう。この場合の目的は、テキストのサンプルを難読化するために使用したXORキーのビット長を特定することです。

scikit-learnツールキットには、いくつかのニューラルネットワーク実装が用意されていますが、Theano pythonライブラリで提供されている、より強力なバージョンを使用します。使用するスクリプトとデータは以下からダウンロードできます。

https://www.cylance.com/intro-to-ai

前述のように、ニューラルネットワークのトレーニングと最適化に影響するハイパーパラメータ設定は無数にあります。説明を簡単にするため、これらのスクリプトでは、有益であることがわかっているハイパーパラメータ設定がすでに定義されています。しかしながら、これらのスクリプトを修正してみたり、自分で新しいスクリプトを作成したりすることを強くお勧めします。そうすることで、ニューラルネットワークの異なるパイパーパラメータ設定やアーキテクチャが分類プロセスにどのように影響するか、そして生成されるニューラルネットワークモデルの精度や性能がどのように変わるかを実感できるからです。

XORでの暗号化

XOR（排他的論理和）は、さままな可変長キーを使用したバイナリデータの暗号化と解読に応用できる論理演算です。その名前は、バイト値を修正する方法に由来しています。キーに含まれる各ビットは、サンプルセットの対応するビットと比較されます。次に、XORは、2つのビット値が異なる場合は1（真）、同じである場合は0（偽）を戻します。処理結果に同じキーを適用することで、修正されたビットを元の値に簡単に戻すことができます。XORはセキュリティの高い暗号化手法とは言えませんが、実装が簡単であり、データを簡単に修正できることから、暗号化方式の一部として広く利用されています。たとえば、データセット全体に渡って同じXORキーを繰り返し適用することができます。仮にキーが「password」で、データが「Encrypt this data」であれば、XORはこの8ビットキーをデータの各バイトに下表のように適用します。（バイナリASCIIデータを使用しているため、8ビットのうちの7ビットのみが暗号化されます。8番目のビットは無視されます。）

ASCII文字を暗号化するために適用されたXOR

キー	プレーンテキスト	暗号化後
p	E	0x35
a	n	0x0f
s	c	0x10
s	r	0x01
w	y	0x0e
o	p	0x1f
r	t	0x06
d		0x44
p	t	0x04
a	h	0x09
s	i	0x1a
s	s	0x00
w		0x57
o	d	0x0b
r	a	0x13
d	t	0x10
p	a	0x11

2バイト(16ビット)のキーと各8ビットの160文字のデータセットを使用した場合でも、プロセスは同じです。この場合、キーは順に10回適用されます。たとえば、キーの最初のビットは先頭バイトの先頭ビットに適用され、次は第3バイトの先頭ビットに適用され、その後も同じように繰り返されます。XORキーの長さがわかれば、頻度分析と呼ばれる技法を使用して、元の文字を推測することができます。通常、このアプローチは単独バイトのキーでのみ効果を発揮します。ただし、キー長がわかれば、同じ手法を暗号テキストストリームにも応用できます。頻度分析の詳細と暗号化および解読における役割については、本章の範疇を超えています。ここでは、LSTMおよびCNNアルゴリズムを使用してX ORキーの長さを特定する方法のみを説明します。

データセットの生成

これまで同様、処理対象となる代表的なデータセットを取得しなければなりません。ここで使用するデータセットは、英語のASCII文字を表すバイト恣意から構成されます。そのため、各ベクトルは8次元となり、各次元が8つの可能なビット値のそれぞれを表します（ただし、前述のように8番目のビットは無視されます）。バイナリ形式であるため、各特徴は0と1のどちらかの値のみを取ります。

データセットを生成するため、enwik8データセットからランダムなプレーンテキストのセクションをダウンロードします。このデータと関連文書は、以下のリンクからダウンロードできます。

https://cs.fit.edu/~mmahoney/compression/textdata.html

このプレーンテキストデータの約70%をトレーニングセットのために確保します。残りのデータは、トレーニング中の検証に使用します。また、モデルをテストできるようになったら、追加の検証データを少し作成します。このデータは、Pythonスクリプトのgenerate_xor.pyを使用して作成します。このスクリプトは、enwik8プレーンテキストを読み込み、指定長のランダムなXORキーを使用して暗号化します。ここでは、キー長は8ビットとします。

後ほど、この暗号化データを使用してモデルをテストし、8ビットのキー長をどれほど正確に予測できるかを評価します。

下記のように、スクリプトはランダムな8ビットキーと、残りのプレーンテキストデータの暗号化バージョンを戻します。キーとデータは、どちらも指定されたファイルパスに書き込まれます。

では、プレーンテキストトレーニングセットのバッチを2つのニューラルネットワークに供給します。最初はLSTMです。

LSTMモデルによるXORキー長の特定

LSTMのような再帰型ニューラルネットワークは、サンプルどうしのシーケンシャルな関係によってクラスの分類が決定されるような問題に適しています。モデルをインスタンス化してから、pythonスクリプトのtrain_model.pyでトレーニングプロセスを開始します。デフォルトでは、この関数はLSTMモデル（conv演算子がfalseに設定されている点に注意）を作成し、デフォルトのパイパーパラメータ設定を定義します。下記のスクリーンショットを参照してください。

```
root@                                   # KERAS_BACKEND="theano" THEANO_FLAGS=device=gpu,floatX=float32 python train_model.py
Using Theano backend.
Using gpu device 0: GRID K520 (CNMeM is disabled, cuDNN not available)
{'output_dim': 256, 'output_activation': 'softmax', 'activation': 'relu', 'conv': False}
Loading data
Starting vectorization threads
Model Summary
____________________________________________________________________________________________________
Layer (type)                     Output Shape          Param #     Connected to
====================================================================================================
input_1 (InputLayer)             (None, 64, 8)         0
____________________________________________________________________________________________________
lstm_1 (LSTM)                    (None, 64, 256)       271360      input_1[0][0]
____________________________________________________________________________________________________
lstm_2 (LSTM)                    (None, 64, 256)       525312      lstm_1[0][0]
____________________________________________________________________________________________________
globalmaxpooling1d_1 (GlobalMaxP (None, 256)           0           lstm_2[0][0]
____________________________________________________________________________________________________
dense_1 (Dense)                  (None, 32)            8224        globalmaxpooling1d_1[0][0]
====================================================================================================
Total params: 804,896
Trainable params: 804,896
Non-trainable params: 0
____________________________________________________________________________________________________
```

図4.16に示すように、モデルには1つのLSTM層（256のノード）と、同じノード数の隠れLSTM層があります。この隠れ層からの出力は、全体最大プーリング層に渡され、その後で出力層に渡されます。そしてsoftmax活性化関数が適用され、各サンプルが分類されて、XORキーの長さが予測されます。ではトレーニングプロセスを開始しましょう。

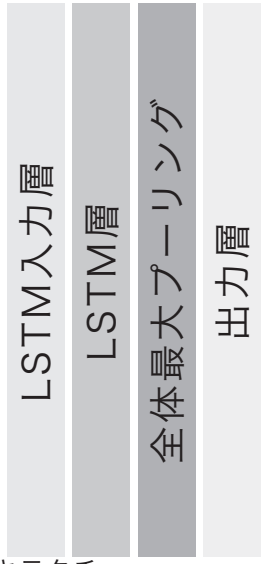


図4.16：LSTMのアーキテクチャ

LSTMモデルのトレーニングと最適化

LSTMモデルはバッチ単位でトレーニングします。各バッチのトレーニングが完了したら、暗号化されている検証セットに適用してモデルの精度を評価します。トレーニングと検証を10件のバッチで繰り返しても精度スコアが向上しない場合は、学習速度が高すぎることを意味するため、少し下げます。このようにモデルの微調整を無制限に行うこともできますが、トレーニングを中断して、異なる構成とハイパーパラメータ設定で新しいモデルを構築することもできます。このようなトレーニングと検証のプロセス進行状況を、次ページのスクリーンショットに示します。

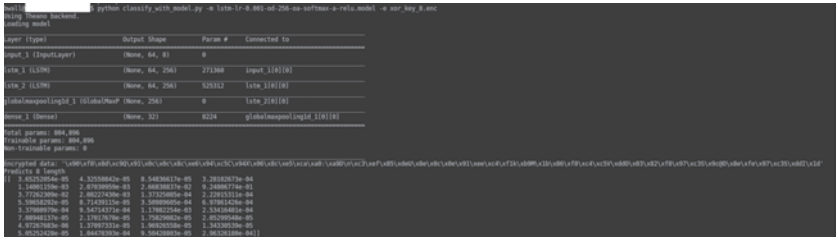
```
Total params: 804,896
Trainable params: 804,896
Non-trainable params: 0
_________________________________________________________________
Epoch 1/1
16384/16384 [==============================] - 84s - loss: 3.4259 - acc: 0.0447
New best validation score: 0.118347167969 (saving)
Epoch 1/1
16384/16384 [==============================] - 84s - loss: 2.4053 - acc: 0.2164
New best validation score: 0.344116210938 (saving)
Epoch 1/1
16384/16384 [==============================] - 84s - loss: 1.9004 - acc: 0.3696
New best validation score: 0.543029785156 (saving)
Epoch 1/1
16384/16384 [==============================] - 84s - loss: 1.2417 - acc: 0.5870
New best validation score: 0.608764648438 (saving)
Epoch 1/1
16384/16384 [==============================] - 84s - loss: 1.1356 - acc: 0.6335
New best validation score: 0.621643066406 (saving)
Epoch 1/1
16384/16384 [==============================] - 84s - loss: 0.8702 - acc: 0.7270
New best validation score: 0.749877929688 (saving)
Epoch 1/1
16384/16384 [==============================] - 84s - loss: 0.7887 - acc: 0.7664
Validation score: 0.73828125
Epoch 1/1
16384/16384 [==============================] - 84s - loss: 1.0136 - acc: 0.7015
Validation score: 0.54443359375
Epoch 1/1
16384/16384 [==============================] - 84s - loss: 0.7927 - acc: 0.7715
New best validation score: 0.762512207031 (saving)
Epoch 1/1
 2144/16384 [==>...........................] - ETA: 73s - loss: 0.7446 - acc: 0.7873
16384/16384 [==============================] - 84s - loss: 0.6921 - acc: 0.8035
New best validation score: 0.83056640625 (saving)
Epoch 1/1
```

ニューラルネットワークのトレーニングには非常に長い時間が掛かります。この時点で、モデルは約0.83という検証スコアを達成しています。初期スコアは0.12でしたから大幅な改善となりますが、それでもまだ必要なレベルには達していません。最低でも0.90の精度スコアになるまで、バッチを追加してトレーニングを繰り返します。

```
Epoch 1/1
16384/16384 [==============================] - 84s - loss: 0.1474 - acc: 0.9625
Validation score: 0.970397949219
Epoch 1/1
16384/16384 [==============================] - 84s - loss: 0.1316 - acc: 0.9655
Validation score: 0.967346191406
Epoch 1/1
16384/16384 [==============================] - 84s - loss: 0.1333 - acc: 0.9656
Validation score: 0.970703125
Epoch 1/1
16384/16384 [==============================] - 84s - loss: 0.1386 - acc: 0.9644
Validation score: 0.968688964844
Epoch 1/1
16384/16384 [==============================] - 84s - loss: 0.1377 - acc: 0.9644
Validation score: 0.970031738281
Epoch 1/1
16384/16384 [==============================] - 84s - loss: 0.1223 - acc: 0.9679
New best validation score: 0.973571777344 (saving)
Epoch 1/1
16384/16384 [==============================] - 84s - loss: 0.1316 - acc: 0.9667
Validation score: 0.970520019531
```

LSTMモデルのトレーニング

多くのバッチを追加してトレーニングを続けた結果、LSTMモデルの検証スコアは0.97を超えました。では、このモデルを使用してXORキーの長さを予測してみましょう。モデルを保存したら、XOR処理したテストデータに適用します。classify_with_model.pyスクリプトにテストデータへのパスとモデル名の引数（lstm-lr-0.001-od-256-oa-softmax-a-relu.model）を渡して実行します。



LSTMモデルは正しいキー長である8ビットを予測しました。

CNNモデルによるXORキー長の特定

CNNには、RNNのように膨大な記憶を可能にするゲートはありませんが、それでも、サンプルの時系列や空間での隣接性によって分類が決定されるような複雑な分類の問題を解決することができます。そのため、CNNは画像の分類（隣接する画素の近傍を解析）か

ら自然言語処理(単語の近傍を解析)まで、幅広い問題に応用されています。まず、概念レベルでこれらの能力がXORキーの検出にどのように応用できるかを考えてみましょう。

初期の畳み込み層では、各入力ノードが8ビットでエンコードされた一連のASCII文字バイトを受け取ります。そして、フィルターを適用し、CNNのsizeハイパーパラメータで設定されている近傍サイズで文字の近傍の重みを計算します。次に、2番目の畳み込み層の接続ノードに出力を渡し、追加のフィルターと活性化関数を適用して、サンプルをより高い抽象レベルに解釈します。さらに何ステージかの処理を行った後、全体最大プーリング層が入力シーケンス全体での最大ノード値を選択して、最終畳み込み層のフィルターに渡します。最終的に、サンプルは全結合層を通って出力層に渡され、キー長の分類が割り当てられます。

この例で使用するCNNは、もう少し複雑で、それぞれが256のノードを持つ4つの畳み込み層を使用します。それぞれの畳み込み層の後には、バッチ正規化、RELU活性化、および最大プーリングを行うための階層があります。出力は全体最大プーリング層に渡され、最終的には出力層に渡されて、softmax活性化関数によって分類の予測が生成されます。CNNのアーキテクチャを図4.17に示します。

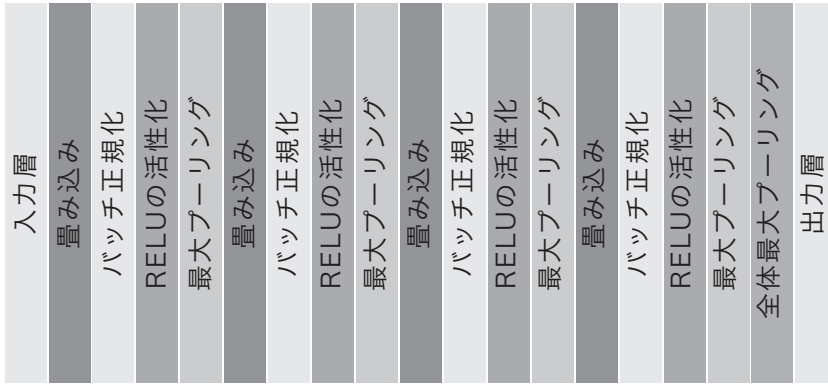


図4.17：CNNのアーキテクチャ

CNNモデルのトレーニングと最適化

これまでどおり、モデルをインスタンス化してから、pythonスクリプトのtrain_model.pyでバッチ単位のトレーニングを開始します。今回はconv演算子をtrueに変更して、デフォルトのLSTMではなくCNNを生成します。構成の詳細は下記のスクリーンショットを参照してください。

```
root@                                   # KERAS_BACKEND="theano" THEANO_FLAGS=device=gpu,floatX=float32 python train_model.py --conv
Using Theano backend.
Using gpu device 0: GRID K520 (CNMeM is disabled, cuDNN not available)
{'output_dim': 256, 'output_activation': 'softmax', 'activation': 'relu', 'conv': True}
Loading data
Starting vectorization threads
Model Summary
____________________________________________________________________________________________________
Layer (type)                     Output Shape          Param #     Connected to
====================================================================================================
input_1 (InputLayer)             (None, 64, 8)         0
____________________________________________________________________________________________________
convolution1d_1 (Convolution1D)  (None, 64, 256)       33024       input_1[0][0]
____________________________________________________________________________________________________
batchnormalization_1 (BatchNorma (None, 64, 256)       1024        convolution1d_1[0][0]
____________________________________________________________________________________________________
activation_1 (Activation)        (None, 64, 256)       0           batchnormalization_1[0][0]
____________________________________________________________________________________________________
maxpooling1d_1 (MaxPooling1D)    (None, 32, 256)       0           activation_1[0][0]
____________________________________________________________________________________________________
convolution1d_2 (Convolution1D)  (None, 32, 256)       1048832     maxpooling1d_1[0][0]
____________________________________________________________________________________________________
batchnormalization_2 (BatchNorma (None, 32, 256)       1024        convolution1d_2[0][0]
____________________________________________________________________________________________________
activation_2 (Activation)        (None, 32, 256)       0           batchnormalization_2[0][0]
____________________________________________________________________________________________________
maxpooling1d_2 (MaxPooling1D)    (None, 16, 256)       0           activation_2[0][0]
____________________________________________________________________________________________________
convolution1d_3 (Convolution1D)  (None, 16, 256)       1048832     maxpooling1d_2[0][0]
____________________________________________________________________________________________________
batchnormalization_3 (BatchNorma (None, 16, 256)       1024        convolution1d_3[0][0]
____________________________________________________________________________________________________
activation_3 (Activation)        (None, 16, 256)       0           batchnormalization_3[0][0]
____________________________________________________________________________________________________
maxpooling1d_3 (MaxPooling1D)    (None, 8, 256)        0           activation_3[0][0]
____________________________________________________________________________________________________
convolution1d_4 (Convolution1D)  (None, 8, 256)        1048832     maxpooling1d_3[0][0]
____________________________________________________________________________________________________
batchnormalization_4 (BatchNorma (None, 8, 256)        1024        convolution1d_4[0][0]
____________________________________________________________________________________________________
activation_4 (Activation)        (None, 8, 256)        0           batchnormalization_4[0][0]
____________________________________________________________________________________________________
maxpooling1d_4 (MaxPooling1D)    (None, 4, 256)        0           activation_4[0][0]
____________________________________________________________________________________________________
globalmaxpooling1d_1 (GlobalMaxP (None, 256)           0           maxpooling1d_4[0][0]
____________________________________________________________________________________________________
dense_1 (Dense)                  (None, 32)            8224        globalmaxpooling1d_1[0][0]
====================================================================================================
Total params: 3,191,840
Trainable params: 3,189,792
```

今回はトレーニングがかなりスピーディに進行します。

```
Trainable params: 3,189,792
Non-trainable params: 2,048

Epoch 1/1
16384/16384 [==============================] - 66s - loss: 1.6461 - acc: 0.5364
New best validation score: 0.630859375 (saving)
Epoch 1/1
16384/16384 [==============================] - 66s - loss: 0.3217 - acc: 0.9221
Validation score: 0.442932128906
Epoch 1/1
16384/16384 [==============================] - 66s - loss: 0.2563 - acc: 0.9385
New best validation score: 0.857849121094 (saving)
Epoch 1/1
16384/16384 [==============================] - 66s - loss: 0.2221 - acc: 0.9479
New best validation score: 0.888427734375 (saving)
Epoch 1/1
16384/16384 [==============================] - 66s - loss: 0.2140 - acc: 0.9478
Validation score: 0.886352539062
Epoch 1/1
16384/16384 [==============================] - 66s - loss: 0.1919 - acc: 0.9542
New best validation score: 0.934265136719 (saving)
Epoch 1/1
16384/16384 [==============================] - 66s - loss: 0.1789 - acc: 0.9564
Validation score: 0.893249511719
```

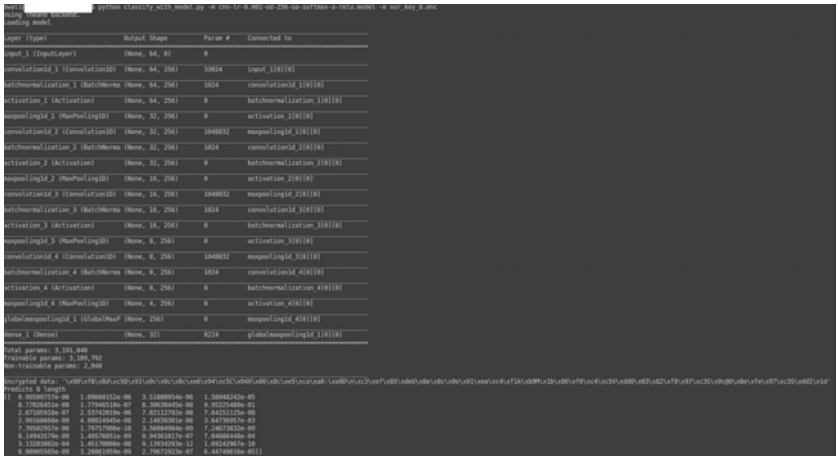
それほど待たずに、検証スコアが初期値の約0.63から0.99超へと改善しました。

```
Epoch 1/1
16384/16384 [==============================] - 66s - loss: 0.0381 - acc: 0.9885
Validation score: 0.991638183594
Epoch 1/1
16384/16384 [==============================] - 66s - loss: 0.0359 - acc: 0.9893
Validation score: 0.991088867188
Epoch 1/1
16384/16384 [==============================] - 66s - loss: 0.0409 - acc: 0.9886
Validation score: 0.991577148438
Epoch 1/1
16384/16384 [==============================] - 66s - loss: 0.0383 - acc: 0.9888
New best validation score: 0.993041992188 (saving)
Epoch 1/1
16384/16384 [==============================] - 66s - loss: 0.0395 - acc: 0.9896
```

この時点でモデルをテストすることができます。

CNNモデルのトレーニング

モデルを保存したら、classify_with_model.py スクリプトを前回と同じ引数で実行して、テストセットに適用します。異なるのはモデル名のみです。今回はcnn-lr-0.001-od-256-oa-softmax-a-relu.modelです。



CNNモデルも正しいキー長である8ビットを予測しました。

深層学習の重要なポイント

本章では、ニューラルネットワークを応用してさまざまな深層学習の問題を解決する方法を説明し、3つの異なるニューラルネットワークアーキテクチャを見てきました。本章の内容の重要なポイントは以下のとおりです：

- ニューラルネットワークは極めて柔軟性の高い汎用アルゴリズムであり、無数の問題を無数の方法で解決できます。たとえば、他のアルゴリズムとは異なり、ニューラルネットワークは数百万あるいは数十億ものパラメータを適用してモデルを定義できます。
- ニューラルネットワークは階層的な処理を行い、各階層とその中にあるノードが特定の計算を実行します。これらの階層のうち、最低1つは隠れ層です。深層学習が他のすべての機械学習手法と大きく異なる点は、この隠れ層を持つ多層アプローチです。
- 各隠れ層のすべてのノードには、サンプルセットに含まれる各特徴について1つずつの重み値がランダムに割り当てられます。処理中、各ノードはそれぞれの特徴に対応する重みを

特徴値に掛け、その積を合算して、結果を活性化関数に渡します。活性化関数は、その階層で指定されている計算を実行します。この活性化関数の結果は、そのノードの処理を合計した効果を反映した活性化値です。

- 各トレーニングサイクルの後で、出力層で割り当てられた分類決定とトレーニングセットのクラスラベルをloss関数によって比較し、より正確な結果を導き出すには、すべての隠れ層の重みをどのように修正すべきかを決定します。この処理は、候補モデル群が検証およびテストフェーズに進めるようになるまで、何度も繰り返されます。
- LSTMのような全結合ネットワークでは、すべてのノードからの出力は、その後の階層のすべてのノードの入力に接続されています。それに対してCNNでは、部分的結合アーキテクチャを使用し、1つの隠れ層のノードは1つ前の隠れ層の連続したノード群にのみ接続されます。これらの接続は、フィルター設定のsizeとstrideで制御できます。
- 順伝播ネットワークでは、情報は入力層から出力層まで後戻りすることなく流れます。それに対してLSTMや他の再帰型ニューラルネットワーク では、フィードバックループやゲートを使用することで、ノードの内容をいつどうやって更新し、後続の隠れ層のノードに渡すかを決定します。
- 順伝播ネットワークは、サンプルが入力層に到達する順序を記憶しなくてもよい問題を解く場合に適しています。連続するサンプルのコンテキストを考慮しなければならない場合は、LSTMのような再帰型ニューラルネットワーク（RNN）の方が適しています。LSTMは、活性化関数の代わりにゲートを利用することで、時系列データの非常に長いシーケンスに渡って、状態情報を維持および再利用できます。
- CNNは、画像認識のように、サンプルの特徴が空間的に関連している問題を解くのに適しています。ただし、XORキー長の分類のように、連続ノード内での一連のローカル接続に基づいてバイナリ文字のシーケンスにおける関係性が決定で

きるような問題であれば、CNNも適しています。

- サンプルの特徴値は、入力層と最初の隠れ層のノードにしか認識されません。後続のすべての階層では、1つ前の階層のノードからの組合せ出力値しか「認識」できず、抽象レベルが徐々に増大した全体像として特徴を「観察」します。たとえば、1つの階層ではエッジ処理を行い、別の階層ではエッジを形状として統合し、さらに別の階層ではその形状を「顔」などのカテゴリーに分類します。ニューラルネットワークは、このような処理を非常に微細な方法で行うことができ、順序付けられた多層計算を通して低レベルの信号から複雑な決定までを処理します。

おわりに

人工知能を活用してサイバー空間でウィルスと戦うと言ったら、数年前であれば映画の世界の話だと笑われたかもしれません。人工知能の関連技術は画像認識や言語認識の領域をはじめ、今後も数多くの分野で活用が進んでいくことが期待されていますが、その1つがサイバーセキュリティです。これまでセキュリティ技術者といえば、過去に見られたマルウェアや攻撃の痕跡情報を元にシグネチャを作成して防御したり、発生したログを解析して攻撃を検知したり分析するという、既知の脅威へのパッシブなアプローチが一般的でしたが、これからはログ、ネットワークセンサー、およびエンドポイントエージェントからの大量のデータを元に、機械学習の技術を活用し、未来の脅威を予測して防御するというアクティブなアプローチが可能になるでしょう。サイバーセキュリティの歴史が始まってから、常に攻撃者が先行していてそれを守る側が追いかけるというイタチごっこが続いていましたが、人工知能がはじめてその図式を変えることになるかもしれません。

本書はセキュリティ技術者が人工知能そして機械学習を学ぶために基礎的なアルゴリズムとその手法について解説しました。サイバーセキュリティの世界で人工知能の技術を活用するという試みはまだ始まったばかりですが、本書に記載されている提案を元に、日々のセキュリティ運用に携わる人たちがより効率的で実用的な方法をみつけることができ、またその人たちによって支えられたシステムによって、ディジタル世代の人々も安心して暮らすことができる世界が実現されることを願ってやみません。

Apply Artificial Intelligence to Information Security Problems

情報セキュリティの世界は情報で溢れています。ログの確認からマルウェアの分析まで、処理すべき情報は膨大な量に上っており、作業人員の許容量を超えています。人工知能は、知性を用いて大量のデータから有意義な結果を導き出すことに優れた研究分野です。本書では、実践的な機械学習の手法を説明しているため、データを扱う他の領域でも応用できるはずです。クラスタリングでは、アイテムのグループ化と異常値の特定方法を解説します。分類では、入力のクラスを分類するためのモデルのトレーニング方法について解説します。確率では、「どのくらいの可能性があるか」という問題に答えて、その結果を利用します。深層学習では、生物学を期限とする今日最も機械学習でも有効といわれる手法とその領域について詳しく解説します。

About the Authors

サイランスのデータサイエンスチームは、さまざまな分野の専門家で構成されています。Brian Wallace は、本書の制作に協力したチームメンバーの 1 人で、セキュリティ研究員からデータサイエンティストに転向した経歴を持ち、情報セキュリティとデータサイエンスの領域を統合するビルディングツールを作っています。データサイエンティストの Sepehr Akhavan-Masouleh は、米カリフォルニア大学アーバイン校で博士号を取得し、統計学の応用と、サイバーセキュリティにおける機械学習モデルを担当しています。Andrew Davis は、米テネシー大学でコンピュータ工学の博士号を取得した、ニューラルネットワークの魔術師です。

Mike Wojnowicz は米コーネル大学で博士号を取得したデータサイエンティストであり、大規模な確率的モデルの優れた解釈能力に惹かれて、その開発と導入に取り組んでいます。データサイエンティストの John H. Brock は、米カリフォルニア大学アーバイン校でコンピュータサイエンスの修士号を取得し、静的マルウェアの検知と分析に対する機械学習の適用について研究しています。彼は、いつもは、Lovecraftian オープンソースコードのデバッグをしており、ユニットテストの長所についてぶつぶつと独り言をつぶやいています。

Computers / Intelligence (AI) & Semantic