『データサイエンティスト教程　応用』

（数理人材育成協会 編，学術図書出版社）

節末問題の解答例

2021 年 10 月 1 日版

● 第 1 章　ビジネスにおけるデータ

1.1 図 1.4 は 1 つのモデリングの例である．

留意点 小人数が協力することで，入力情報やその処理作業が拡大し，意見交換することで理解が深化したり，1 人では得られないような視点やアイデアが得られる．グループワークでは これらのメリットを最大限に生かすように工夫することが大切である．グループ内で同一の作業が重複したり，方針が定まらないといったようなロスを防ぐために，作業分担を適切に行うことが有効である．たとえば，1 人がインタビュアー役となり，業務フローをよく知る者から情報を得る．また，作図役，モデリング規則を満たしているか点検する役，改善案の議論におけるファシリテータ役などを設ける．5 人くらいのグループだと，リーダーシップを含めた役割分担や，有機的な意見交換の機会を組みやすい．またウェブを使った共同作業に習熟することも，社会人として基本的なスキルになっている．失敗をおそれずに挑戦してほしい．

1.2 図 A.1 は図 1.4 に示すプロセスを改善したものである．ここでは，「ループをなくす」「プロセスの評価指標が改善される」ことを主眼とした修正がなされている．

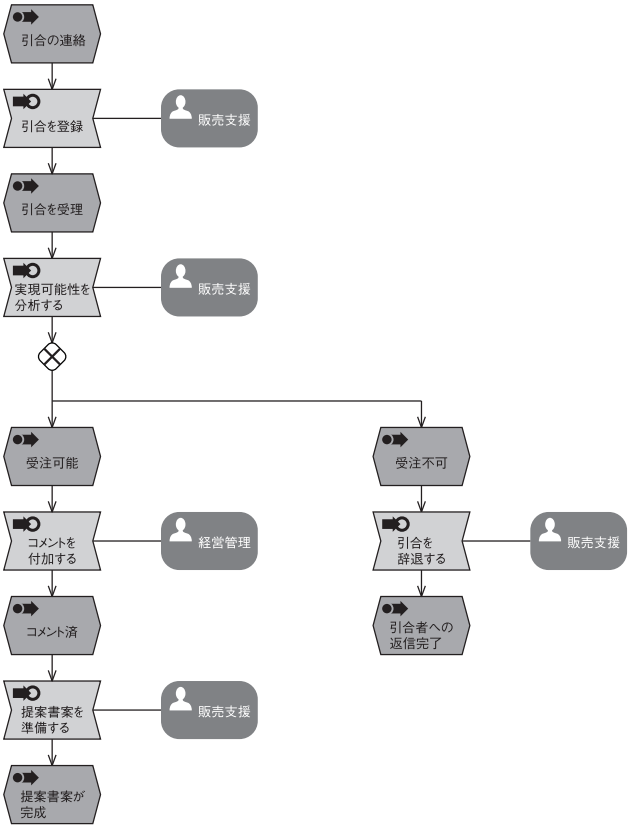


図 A.1　改善されたプロセス

● 第 2 章　データサイエンスの活用

2.1　下の表は (1), (2), (3) の解答例をまとめたものである．自身の興味に従い，個々の例についてそれぞれのツールをどのように適用するか考えて，実装を試みることをすすめる．

	回帰	識別
サポートベクターマシン	災害危険度評価 人口推定の予測	スパムメール検出 数字画像の認識 顔画像の認識 医療現場の事例発見
決定木・ランダムフォレスト	気象予報 電力需要予測 画像中の物体検出	数字画像の認識 姿勢推定 文書の分類・トピック推定 行動認識

与えられた課題やデータに対して，どのような手法をどのように適用するかを設計することが，データサイエンティストの腕の見せ所である．ここでは，カメラで人間の動きを計測したデータから行動を認識する課題について述べてみよう．求められているのは，たとえば，歩いている人を撮影した画像・動画データを「歩行」と分類するような機械である．この課題では，分類する行動パターンの種類が多い．多クラス分類にはサポートベクターマシンも多クラス問題に使用できるが，その簡便性からランダムフォレストを採用した（表の「行動認識」）．

留意点 自分の業務内容を書き入れる場合は，秘匿事項に注意し，公開できるもののみ紹介するにとどめる．

2.2 （解答例）

(解答 1) 画像圧縮に使われている．画像は各画素が持つ数値データの集まりであり，一般的に情報量が大きい．そこで画像をパッチとよばれ小領域に分割する．各パッチの画素を一列に並べることによって，パッチをベクトル数値化することができる．各パッチを辞書ベクトルの重みづけ線形和として復元できるような辞書ベクトルと，少数の重み係数をスパースモデリングによって求めることができる．画像を低次元の辞書ベクトルと少数の重み係数に変換することによって，画像を情報量の少ない数値データとして圧縮することができる．

(解答 2) 高解像度画像の復元に使われている．高解像度画像のパッチの学習データから，スパースモデリングによって辞書ベクトルを求め

ておく．高解像度辞書ベクトルに人工的にダウンサンプリングを施して，低解像度画像を表現するための辞書ベクトルを作成する．入力された低解像度画像は，スパースモデリングによって低解像度辞書ベクトルの重みづけ線形和として表現できる．求められた重み係数は維持して低解像度辞書を高解像度辞書に置き換えることによって，高解像度画像を生成することができる．

2.3 （解答例）

潜在意味解析（トピックモデル）で使われている．文書は単語の集合であり，各文書に含まれている単語から文書を分類することや文書の意味・トピックを抽出することが求められることがある．そこで，各文書には「複数のトピックが含まれている」,「トピックによって使用される単語の頻度が変わる」という仮定に基づき，各文章に含まれるトピックの分布と各トピックから単語が生成される分布を潜在変数・モデルパラメータとみなす．各文書のトピック分布に沿ってトピックが生成され，そのトピックから単語が生成されると考えると，各文書中の単語が生成される確率が計算できる．この確率に対して変分ベイズを適用することによって，トピック分布・単語分布が推定される．各文書に含まれるトピックの強さや，それらの分布から文書の意味を解析・分類することができる．

● コラム A.1　パーティクルフィルターによる物体追跡

動く物体をカメラで撮影して，その位置を推定する物体追跡にパーティクルフィルタを利用することを考える．観測値は特定サイズの領域を切り出した画像と状態量をカメラ画像中における位置（x, y 座標値）とする．

この場合，パーティクルフィルタにおいて非観測な状態量の変化を記述する状態方程式，状態量と観測値の関係を記述する観測方程式，および実際の観測値に状態量を照らし合わせて各状態量の評価法を設計する必要がある．一般に，時間経過につれて状態が変化する現象に対しては，状態の移り変わりを状態方程式として推測・予測する機能を数理モデルとして内包することにより，時間変動する現象をとらえることにパーティクルフィルター―が有用である．

パーティクルフィルターを用いた物体追跡法の一例である．

(手順 1)　物体の初期位置，初期速度（x, y 座標値の変化量）をランダムに設定

し，その位置での領域画像を取得する．

(手順 2)　時刻 t での位置から時刻 $t+1$ での位置を計算する状態方程式（現在位置に速度と雑音を加算するなど）に従って，時刻 $t+1$ の位置を推定する．雑音を変えると位置が変わるので，さまざまな雑音に対して時刻 $t+1$ の位置を計算して，位置を状態量 x_{t+1} とするパーティクルを生成する．

(手順 3)　時刻 t で撮影した画像において，現在位置での領域画像 $I_t(x_t)$ を取得する．時刻 $t+1$ の画像に対して，手順 2 で求めた位置における領域画像 $I_{t+1}(x_{t+1})$ を取得する．これが状態量から観測値を求める観測方程式の役割をはたし，各パーティクルに対して各々の領域画像を取得することができる．

(手順 4)　時刻 t での領域画像 $I_t(x_t)$ と時刻 $t+1$ の領域画像 $I_{t+1}(x_{t+1})$ を比較する．物体追跡が成功していれば同じ物体を補足しているので，2 つの画像は類似していることになる．そこでこの 2 つの画像の類似度を計算し，類似度が大きい状態量（次時刻のでの位置）を高く評価し，類似度が小さい状態量を低く評価して，評価値が高い状態付近で分布が高くなるように状態を生成する．この状態を次の時刻での位置として，手順 2 に戻る．以下これを繰り返す．

● 第 3 章　AI

3.1　（解答例）

(解答 1)　AI による自動運転は，まだ実現していない．ヒトかモノかを判別するためのセンシングの情報量が多く，予期しない障害物が出現したときの判断が難しいのが，1 つの原因である．実現すれば移動のための拘束から解放されるので，運転で使われていた時間を仕事やプライベートのために使えるようになる．また労働力不足の解消に役立ったり，車が部屋になることで，専用のキットや家具などの市場を開拓することができる．

(解答 2)　AI による製造業の需要予測は，まだ実現していない．在庫管理の

場面で，商材による特性を反映した各事業へのカスタマイズが難しく，担当者の経験に頼っているのが現況である．実現すれば，環境に配慮した資源の有効利用が達成できる．

(解答 3) 数式を理解できる AI は，まだ実現していない．定義を与えて，シンプルで合理的な数式を導き出すことはすぐできそうであるが，定義や例から AI が独自で抽象的な概念を導き出すのは，現在の技術水準では原理的な困難がある．実現すれば，特徴量エンジニアリングが自動化され，研究開発においてスモールデータの活用が可能になる．

(解答 4) クロスワードパズルの自動解法は，まだ実現していない．もしかすると，実用間近かも知れない．

(解答 5) AI による薬剤師は，まだ実現していない．顧客の質問や相談に対応するサービスとして 文字チャットがあるが，不完全である．知識の蓄積管理が大きな課題である．実現すれば，音声で特定分野の質疑応答ができるようになる．

(解答 6) AI を用いて医療における服用薬や投与量を個別に最適化することは，まだ実現していない．困難な理由は生体反応の複雑さで，個体差に加えて日内変動もある．実現すれば，効果的な治療や副作用の軽減ができるようになる．

(解答 7) 介護や掃除を行うロボットは，まだ実現していない．汎用性や多用途が求められ，先例がないような状況に関わる要素が大きいのが原因である．実用すれば，スポーツの審判や日常の判断・交渉ができるようになるかも知れない．

(解答 8) 人の感性に関わるような笑い，小説などを作り出す AI は，まだ実現していない．現時点では，AI の枠組み自体はヒトが定義して，パラメータのみを AI で決める形になっていることがその原因である．実現すれば，デザイン，創作物，AI の設計など，ヒトと AI が切磋琢磨する場面を構築することができる．

(解答 9) 間取り，構造，法令などに従って住宅を設計する AI は，まだ実現していない．過去の優れたプラン集はあるが，最適の定義が人によって異なり，環境工学が必ずしも快適さにつながらないことや，

AI には法律の解釈ができず，人がするべき仕事と考えられていることが原因である．実現すれば，周辺環境を考慮したシミュレーションや子どもの成長に応じた最適配置が可能になる．

留意点 決められた解答があるわけではない．夢をもつこと，同時にその夢の実現可能性を冷静に分析することが，次のブレークスルーをもたらす．技術革新が想定を超えて進む現在，正確な予測は困難ではあるが「西暦何年に実用可能と思われるか」という問いかけも有益であろう．

3.2 （解答例）

(解答 1) 薬物動態・薬効の個体差の原因となりうる共変量を探索するアルゴリズムで活用できる．必要なデータは，フィッティング対象となる解析データ（遺伝子を含む）と候補となる変数である．期待される効用としては，マニュアルでの探索と比べて時間を節約できること，恣意的なモデル構築を避けられることがある．予想される困難として，データ数が限られている場合，オーバーフィッティングや組み合わせによる計算量の爆発が発生することがある．対処法として，事前のモデルの絞り込みがあげられる．

(解答 2) 生涯収入に対して自分のライフスタイルに合わせて使用できるように，不動産への投資額決定を迷路探索シミュレータで行うときに活用できる．期待される効用としては，人生設計に合わせた満足な買い物が可能になることがある．予想される困難として，不動産の特徴量やその人の性格などをどのように取り込むかということや，精度を上げようとすると項目が膨大になることがある．対処法として，家に住んでみた顧客の満足度のデータを集め，それを最大化することがあげられる．

(解答 3) ツイッターのつぶやきから，ユーザーの好みの特徴量を取得し，婚活のマッチングサービスに活用できる．期待される効用としては，最適なパートナーが選べることがある．予想される困難として，データを集める労力が膨大になることや，マネタイズが難しいことがある．対処法としては，このサービス自体は無料にして，広告収入を得たり，デート先のクーポンを無料配布したりすることがあげられる．

留意点　この節では，演繹的な探索・推論を解説した．課題に対しては，機械学習による帰納的な学習の例をあげてもよいが，違いを認識していることが必要である．またアイデアを実装しようとすると，現在の技術だけでは難しいこともある．その場合，何が課題となるか，なぜ現時点で実用化されていないのかを明確にすることも有用である．たとえば，特許情報や学術文献などから必要な情報を検索するときに活用しようとする場合，現在の自然言語処理では，文章の中の因果関係を理解した上での情報検索が困難であるため，皆が常識としているような内容は検索できても，学術性が増すほど有用な情報をデータベースから得ることは難しくなっている．オントロジーや知識表現に基づいて今後可能になるかも知れない．その場合には，シンボルグラウンディング問題なども解決しなければならないであろう．

3.3　（解答例）

(1) 気候，金融関連指標，電力エネルギー需要などの予測電子商取引などの購買行動モデリング，商品在庫管理や商品推薦システム医療検診データから病気の検出と最適処方の提案などがある．

(2) 医療現場向けの AI を設計する場合，診察や検査の過程で収集されるデータを学習させる必要がある．これらのデータは個人情報であるので，患者に十分な説明をした上で同意を得たり，データから個人が特定されないように匿名化するなどの対策を十分に検討しておくことが必要である．総務省 e-Stat ページに公開されている政府系の調査データ（国勢調査，経済センサスなど）のように，オープンデータを活用すると，データ収集が効率化できる．

(3) EC サイトでの各ユーザーの購買行動ビッグデータからユーザーの属性・嗜好に応じたグループ分け，特徴が類似している他のユーザーの購買傾向を参考にしながら，購入するべき商品を推薦するサービスが提供できる．売上向上や企画部署へフィードバックして新商品開発につなげることができる．

(4) 学習データは AI の性能に直接的な影響を及ぼすため，膨大なデータを常に収集しながら，新たな学習データを用いて AI を改善し続けることが求められる．シェアリングエコノミーやサブスクリプションなどでは，購入した商品の使い切りビジネスと異なり，頻繁に消費者と企業のやりとりがある．接点を増やすことで，消費者行動のビッグデー

タが収集できる．ビッグデータを学習した AI はサービスの質を高め，顧客の離反を防ぎ，新たな顧客を呼び込むのに大きく貢献している．

留意点 AI は多様であり，ニューラルネットに限らず，他の機械学習を用いた AI も含めて調査することが望ましい．

3.4 （解答例）

(1)
- 画像中の物体認識のための特徴抽出
- 正常・異常の検知アルゴリズム
- ノイズ除去フィルタリング
- 深層学習ニューラルネットワークの事前学習

(2) オートエンコーダの代表的な機能は，エンコーダ（情報圧縮）とデコーダ（情報生成）である．上述の例では，以下のようなエンコーダ，デコーダが活用されている

- 画像の高次元データを少数のノードから構成される中間層の数値データに変換（エンコード）し，この特徴量から画像中の物体を分類する．
- そもそも異常データを学習データとして収集することは困難である．正常データのみを学習したオートエンコーダは，正常データを入力すると，入力と類似したデータを生成する（デコード）．一方，異常データを入力すると，入力と大きく異なるデータを生成する．これらの入力と出力の相違から正常・異常を判定することができる．
- ノイズのないデータを学習したオートエンコーダは，きれいなデータを生成する機能（デコーダ）を獲得する．このオートエンコーダに，ノイズのあるデータを入力しても，きれいなデータの特徴量に変換され，そこを経由してきれいなデータが復元される．
- 多くの中間層を有するニューラルネットワークでは，学習時に勾配消失によって適切に入出力関係を学習できないことがある．オートエンコーダによって，深層学習ニューラルネットワークの一部を学習する（事前学習）ことで，勾配消失などの学習時の問題を解決することができる．

留意点 ニューラルネットワークはさまざまな分野に活用されているので，広い視野を持って多くの利用例を探し出して，理解を深めることが望ましい．

3.5 何らかの特徴を持った絵，写真，図を複数提示する．これらに対して，畳み込み処理をするためのカーネルを提示する．以下はその例である．

(1) 縦エッジフィルタ：縦方向のエッジを検出する．
横エッジフィルタ：横方向のエッジを検出する．
平滑化フィルタ：領域中の画素値の平均を算出する．

縦エッジフィルタの例

0	0	0
−1	0	1
0	0	0

横エッジフィルタの例

0	−1	0
0	0	0
0	1	0

(2) 従来の画像認識のエッジなどの局所特徴量の検出では，人間があらかじめ設計したフィルタを掛けることによって特徴を抽出していた．これに対し，畳み込みニューラルネットワークは，フィルタは重みパラメータに相当し，画像認識の精度が向上するようにパラメータを自動的に最適化し，同時に画像の特徴量も学習する．識別器としての機能だけでなく，特徴抽出の機能も有していることが大きな強みとなっている．

留意点 絵，写真，図は身近にある単純なものを用意し，適切と思われるカーネルを自身で決める．カーネルによってどの部分が抽出され，プーリング層の中に情報として残っていくかについて推定し，それぞれどのように特徴が抽出され，絵図がどのようにして最終的に分離するかを説明する．深層ニューラルネットワークの中身をブラックボックスとするのではなく，作業を通してどのような処理が行われており，それがどのような役割を演じているのかを理解するように努めることが有用である．
画像とカーネルのどの部分にどのような類似性があれば目的を達成できるかについて正しく理解するために，使用する絵図はできるだけ単純なほうがよい．解答をどのように評価するかについては，担当者の経験も必要である．

3.6 （解答例）

(1) 2000 年以前ではこのような例は見出せない．最近の製品として，顔認

証を行うカメラがある．また，映像中の物体認識機能によって自動車の安全運転に役立てる自動運転装置，映像を用いた体温計測などがある．

(2) これらのカメラでは，撮影する際になるべく顔に焦点を合わせてそれ以外の背景などはぼかすなどの処理を行っている．

留意点 本課題は，第三次 AI ブームがそれ以前と比べてどのような進歩を遂げたのか，以前に解決できなかった問題点が解決したのか，しなかったのかなどについて考えることが目的である．確実な動作が保証される場面では最終的には人に任せることになるが，補助として用いるにはとても便利となる分野で，より良い解決策として用いられるケースが増えていることが要点である．

3.7 （解答例）ニュース記事分類への説明付与の場面で適用できる．すでにニュース記事の分類には，機械学習の技術が用いられている．多くのニュースサイトでは，政治，経済，スポーツ，芸能などのカテゴリに分類する形で各記事が掲載されている．これに加えて，たとえば，スポーツの記事の中でも，サッカーの記事，バスケットボールの記事などさらに詳細な分類ができれば，各個人の選好に合った記事を提示することができる．カテゴリが詳細化されると，分類精度を維持できるかどうかが問題になるので，ユーザからのフィードバックを獲得し，それを分類モデルの修正に活用する改善策が考えられている．

現在の技術では，年齢や性別などある属性を持ったユーザに対して，ある記事を提示して，クリックされたかどうかといったデータがフィードバック情報として取得されているが，クリックされて閲覧された記事であっても，その記事にユーザがどれくらい満足したかはわからない．like, dislike のボタンを設けて押してもらうなどの設計もあるが，分類の適切・不適切についての意見か，記事内容自体の好き嫌いに関する意見かが明確でないし，ユーザがシステム改善に貢献しているという印象を与えにくい点も問題である．ユーザからより多くのフィードバックを得るために，説明を提示するのはどうであろうか．

たとえば機械学習によって得られた分類モデルに対して，分類結果に対する確信度が低い場合，「こういう理由でこのカテゴリに分類しましたので，あなたに提示しています」といったようなメッセージを提示する．用いらる説明は，分類根拠となった単語をハイライトする，下線を引くなどである．

さらに理由提示の場面を限定して，ユーザに煩わしさを感じさせることを抑制する．このようにすれば，クリックの有無だけでなく，説明に納得するかしないかという情報を得ることができるだろう．

さて，説明可能な AI に関する既存技術の多くは，判断に影響を与えた特徴量（単語や画像のパッチなど）を示すことはできるが，特徴量間の相互作用を十分に考慮に含めることはできない．よって，以下で説明するような問題が生じる．「サッカー」と「バスケットボール」の二値分類を行うとして，4 つの単語「ドリブル」「キック」「反則」「手」を考え，この 4 単語について分類モデルが以下のように振る舞うとする．

- 「ドリブル」「キック」が含まれていると，その一文はサッカーと分類される
- 「反則」「手」が含まれていると，サッカーと分類される
- 「ドリブル」「手」が含まれていると，バスケットボールと分類される
- 「反則」「キック」が含まれていると，バスケットボールと分類される

このとき，「ドリブル」を含むサッカーに関するニュース記事があったとする．ドリブルは当該記事をサッカーに分類する根拠と考えられるが，ドリブルだけではサッカーとバスケットボールを分類できない．そのため，単語「ドリブル」は記事をサッカーに分類した主要な根拠として提示されないことになる．上記の単語の組を特徴量として扱うことも考えられるが，組み合わせの数が膨大になり，計算量の点で問題が生じるおそれがある．

このように，ニュース配信において，説明を介してユーザの積極的関与を求め，システム全体の性能改善を図るという考え方もあるが，その実現には，特徴量間の相互関係の扱い方の検討が必要である．

留意点　本分野に関する研究は，論文だけでなく，コードが公開されている場合も多い．適当なデータセットを準備し，生成された説明に基づいて議論を展開できるとなおよい．

● 第 4 章　マルチメディア

4.1 （解答例）

(1) 文書は，各単語の出現頻度を成分とするベクトルによって特徴を数値化することができるが，逆に，単語が各文書に出現するかどうかを成分とするベクトルで単語の特徴を表現することもできる．ある文書中に，この単語が出現する場合は 1，出現しない場合は 0 とすれば，文書数に対応する次元でその単語の出現度を表すことができる．文書の特徴ベクトルを単語の特徴ベクトルに置き換えて同じ計算をすれば，単語間の類似度や潜在意味解析をすることができる．

(2), (3) テキスト解析の実施例，適用できる課題，期待できる効用としては次のようなものがある．

- 自由記述式アンケート回答のデータ解析では，アンケート回答に使用される単語の bag of words から回答者をグループ分けすることができる．
- 新聞や雑誌記事の解析では，潜在意味解析を利用して文書間の関係性やトピック推定を行うことができる．
- 検索キーワードを用いたインフルエンザ予測では，Google 検索キーワードのトレンドと疾病予防管理センターが公表するインフルエンザ感染者数のビッグデータから，インフルエンザの流行と相関を高い検索キーワードを選定し，このキーワードにて検索される件数からインフルエンザが流行しているかどうかを予測する．
- スパムメールの検出では，メール文から内容の特徴ベクトルを計算する．この特徴ベクトルと，サポートベクターマシンやランダムフォレストなどの識別器を組み合わせることによって，受信メールがスパムかどうかを識別する．

留意点 テキストの特徴量を用いたテキストマイニングを調査する．本書で紹介した各種機械学習法と組み合わせた事例もあるので合わせて検討する．

4.2 （解答例）

(1) 画像分類，オブジェクトや顔検出，手書き文字の読み取りなどのサービスは `https://cloud.google.com/vision?hl=ja` にある．また音

声をテキストに変換するサービスが `https://cloud.google.com/speech-to-text?hl=ja` にある．

(2) (解答 1) 会話の声や音程から言葉と感情の特徴量を抽出する．出席者の基本情報を活用して，会場に合わせた音楽を流したり，光量や室温の設定を行う．

(解答 2) 画家の絵画の特徴量を検出し，似ているタッチの絵画を生成モデルによって作成する．教師データが足りない場合，他の画家で学習したモデルによる転移学習を行う．

(解答 3) 音声処理によるウソ発見器や，動物の音声を翻訳する機械を作製する．

(解答 4) 音声を文字に変換し，内容を要約する図や設計書を作成する．

(解答 5) レストランの入口や学校などで，マスクを正確につけているか，手洗いが適切かどうかを画像処理によって検出する．

(解答 6) 怒りの感情を音声で数値化し，咀嚼音を客観的に表示して自身の振り返りに役立てる．

(解答 7) 衣服の洗濯タグを画像処理で読み取って，洗濯機の洗濯モードを設定する．

(解答 8) 公園など昔の風景を 2 次元写真の記録などから 3 次元的に再現し，時代をさかのぼって世界中どこでもバーチャルに楽しめるものを作る．Google Earth などがなかった時代の復元などはどうであろうか．

留意点 漠然とした質問だが，サービスの実現可能性について，自由に発想していただくとよい．「ここ数年以内に実現可能である」という条件をつけてみると，具体的なイメージが湧いてくるかも知れない．

● 第 5 章　データエンジニアリング

5.1 （解答例）

(解答 1) オンラインショップの商品情報や顧客情報の管理，注文に応じた在庫の確認と発送．

(解答 2) 大学の教務システム．各学生の個人情報，開講されている授業科目情報，学生の履修情報，履修登録や成績管理．

(解答 3) 大学や会社の e-Learning システム．講義映像や学習コンテンツをデータベース化したもの．

(解答 4) スーパーやコンビニの POS システム．チェーン店での各店舗での商品販売時に，時間帯や顧客の種別をあわせて登録し，中央のデータベースに集約する．店舗や時間帯による流通状況を確認し，商品の在庫管理やマーケティングに利用する．

(解答 5) メンバーズカードによる，会員登録された顧客の購入履歴のデータベース．商品販売に活用する．

(解答 6) 製造業において，製品の在庫管理，部材，販売，人事などの，企業に関するあらゆる情報を一元管理し，経営の効率化や製品展開に活用する．

(解答 7) ある疾患の画像を器官レベルから細胞レベルまで，また CT や MRI，病理染色などさまざまな医療機器のものを集めたデータベース．深層学習のような機械学習手法と組み合わせて用いることで，疾患の診断を可能にする．

留意点 今回扱ったリレーショナルデータベースと，後の講で扱う別のタイプのデータベース（たとえばビッグデータの取り扱いに適した NoSQL データベース）との違いをみて，それぞれに向いた実例を考えてみる．

5.2 （解答例）

(1) インターネットを通じて中古商品購入が可能な Web ページを利用した．

(2) 自動車のブレーキパーツだけで 600 万点以上の商品があり，メーカー別，車種別，パーツの種類別に分類されており，一点につき複数の写真が掲載されていた．また価格は時間と共に刻々と変化し，そのため各パージでの価格反映にタイムラグが生じていることがわかった．

(3) セキュリティーについては，出品者情報とそれに付随する決済口座関連情報などが洩れる可能性があるが，それ以外の商品に関する情報漏洩については特に大きな問題はないと考えられた．

留意点 本課題に取り組むことで，実際の現場でのデータベース（DB）の応用の広さや，それを実現するためのシステム規模などに考慮が至ることが重要である．またDBシステムの書き換えが発生するようなシステムであるかどうかなども確認事項としたい．グループワークによって，さまざまな利用法について気づくことが望ましい．

5.3 （解答例）学歴情報管理システムをブロックチェーンを使って作成し，サーバーノードのないDBで管理することになった．改ざんなどのリスクが少なく，時間・コストの削減にもつながると考えたが，トランザクションの内容は利用者全員に公開されてしまうため，情報に秘匿性を持たせられない場合の問題点を検討する必要が生じた．

留意点 従来型のDB技術の持つ短所でもあるサーバーノードの存在の必要性や，コストの存在，データベースの改ざんなどの問題点のいくつかを新しい技術で置き換えることを検討することで，新旧DBシステム全体の理解を深めることができる．特に新技術で置き換えなくてもよいことがあることに留意する必要がある．

5.4 （解答例）表計算ソフトによって作られたファイルを共有して利用してきたが，複数の人が読み書きしているうちに複数のバージョンができてしまい，一部処理結果が失われるなどのトラブルが発生し，顧客に損害を与えてしまった．

留意点 システムの変更では，発生するトラブルによる損失の大きさとアクセス性，利便性の評価が出来ているかどうかに着目して意思決定する必要がある．

5.5 （解答例）

(1) 分散管理によりデータ管理者が不要となり，システムの故障，管理者による改ざん，中央集権的な運用などのリスクからシステムやデータを守ることができる．応用例として，決済、契約履行の記録などのスマートコントラクト，コピー可能なデジタルデータの所有権を明確にできる可能性のあるデジタル所有権，医療情報や学歴・職歴データの公共データベース化などの多数の例が考案されている．

(2) 「特許情報をブロックチェーンによって管理する．」利点として，データを改ざんや誤消去から守り，運用コストが最小化できる．実現可能性のための課題として，ブロックチェーンネットワークへ参加する動

機をどのように維持するかということが考えられる.

● 第 6 章　情報理論の基礎

6.1 （解答例）

(1) デジタル電圧計.

(2) 安価な製品であり，高いサンプリング周波数は必要としない.

(3) 高い（電圧）分解能（Bit）や直線性誤差などを実現する変換方式が使われているものと考えられる.

留意点 どのような変換方式が利用されているか，カタログなどで情報が公開されている機器もあるので，可能であれば確認するとよい.

6.2 （解答例）

(解答 1) 日本語で,「おはようございます」などの長い定型的な表現を圧縮する．メールなどのデータ量が圧縮されて，表現と概念の分離が可能となり，多言語対応も促進される．困難な点は，圧縮の対応表を作ることである.

(解答 2) 画像・動画・音声・電気信号を符号化してデータ圧縮する．医療の遠隔診断，子どもやペットの監視，テレワーク，農業 IoT，気象衛星の観測など，さまざまな応用が可能になる.

(解答 3) ゴルフなどのスポーツにおけるフォームの改善にデータ構造・アルゴリズムを活用する．映像，位置，加速度に加えて地形のデータを使い，平坦な打ちっぱなしでは打てるがゴルフ場ではうまく打てない問題が解決できるかも知れない．まずゴルフ場を想定し，AI キャディーのようなアプリケーションを作製する．大量の教師データも集める．たとえば，正しいフォームは体型に細かく依存するし，ボールはどこにでも飛ぶので，地形データも膨大に必要で，ゴルフ場での自分の位置を正確に測定することの困難もあるだろう．画像でなく，ウェアラブルとジャイロセンサーの組み合わせも使えるかも知れない.

(解答 4) データ構造・アルゴリズムを活用して歩き方を分析し，メタボの予測・予防につなげる．睡眠の質の向上に役立つであろう．

(解答 5) 製造業の製造工程のパラメータを符号化し，目的物の作業工程の構築に役立てる．

(解答 6) 人事履歴を木構造で表現する．データ構造を活用した文書管理の効率化を図ることができる．

留意点 課題によって具体例を自由に考える．本節は理論に重点を置き，具体的なアルゴリズムの記述は少なかったが，たとえばグラフアルゴリズムはさまざまな探索に使うことができる．

6.3 （省略）

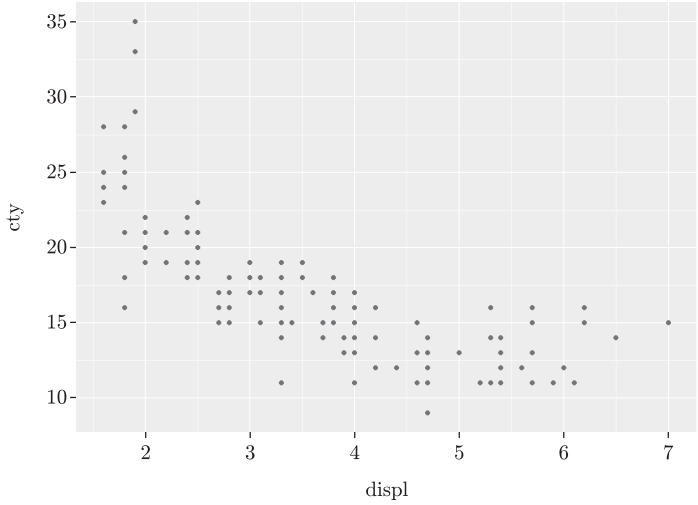
● 第 7 章　標準ソフトの基本動作

7.1 mpg は 1999 年と 2008 年に製造された車両の燃費についての 234 台分のデータセットで，1999 年から 2008 年までの間に毎年新発売されたモデルのみが含まれる．メーカー名，モデル名，エンジン排気量，市街地走行距離，製造年，燃料の種類，車種などの情報が含まれている．

```
1 library(ggplot2)
2 data(mpg)
3 str(mpg)
```

```
## tibble [234 x 11] (S3: tbl_df/tbl/data.frame)
##  $ manufacturer: chr [1:234] "audi" "audi" "audi" "audi" ...
##  $ model       : chr [1:234] "a4" "a4" "a4" "a4" ...
##  $ displ       : num [1:234] 1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 ...
##  $ year        : int [1:234] 1999 1999 2008 2008 1999 1999 ...
##  $ cyl         : int [1:234] 4 4 4 4 6 6 6 4 4 4 ...
##  $ trans       : chr [1:234] "auto(l5)" "manual(m5)" ...
##  $ drv         : chr [1:234] "f" "f" "f" "f" ...
##  $ cty         : int [1:234] 18 21 20 21 16 18 18 18 16 20 ...
##  $ hwy         : int [1:234] 29 29 31 30 26 26 27 26 25 28 ...
##  $ fl          : chr [1:234] "p" "p" "p" "p" ...
##  $ class       : chr [1:234] "compact" "compact" "compact" ...
```

横軸に displ（engine displacement, in litres），縦軸に cty（city miles per gallon）を取って，散布図を作成すると，エンジンが大きくなると燃費が低くなる傾向を読みとることができる．



留意点　余裕があれば，R documentation の内容だけでなく，データにおける変数や内容がどのようなものかを調べたり，他の可視化手法を使用して課題に取り組む．

7.2　（解答例）

(1)　化合物に関するデータ，歴代の首相の演説に関するデータ，SNS 上のデータ，特許データなどがある．

(2) (a)　歴代首相の演説に関するデータ．これまでの演説内容を名詞・形容詞などの内容に関する特徴量を用いて分析することで，演説を時代や政党などに分類できる可能性がある．

(b)　化合物に関するデータ．これまで蓄積してきたデータをクラスタリングによって統計的な観点から分類し，分類結果を参考に素材の特性を捉え，それを基礎に新たな製品の開発に活用する．

留意点　使用するデータによってはクラスタリング・グルーピングの手法よりも適した分析手法があるかも知れない．グループワークでそのような点についての議論ができれば，より有益である．

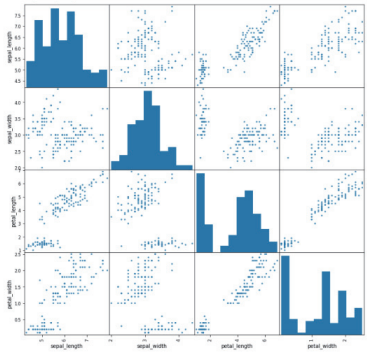
7.3 UCI machine learningrepository からアヤメのデータをダウンロードし，「外花被片の長さ」·「外花被片の幅」·「内花被片の長さ」·「内花被片の幅」·「アヤメの種類」に対して回帰分析および主成分分析を行った後，機械学習により「アヤメの種類」を分類することを学ぶ．

● データの読み込み ●　読み込んだデータに対して，sepal-length（外花被片の長さ），sepal-width（外花被片の幅），petal-length（内花被片の長さ），petal-width（内花被片の幅），speicies（アヤメの種類）のように，変数名を定義する．これが図 A.2 (a) で，図 A.2 (b) はこのデータの散布図である．

	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa
...	...	...	...	...	...
145	6.7	3.0	5.2	2.3	Iris-virginica
146	6.3	2.5	5.0	1.9	Iris-virginica
147	6.5	3.0	5.2	2.0	Iris-virginica
148	6.2	3.4	5.4	2.3	Iris-virginica
149	5.9	3.0	5.1	1.8	Iris-virginica

150 rows × 5 columns

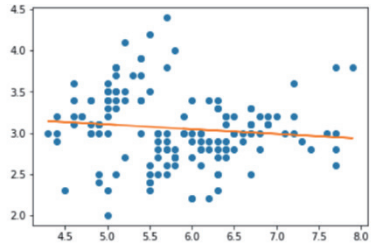
(a)



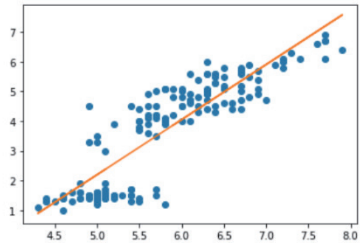
(b)

図 A.2　アヤメのデータ及び散布図

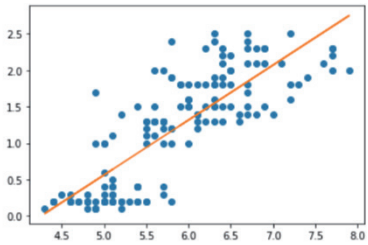
● 回帰分析 ●　散布図を見ただけでは，データ同士の関係を解析することができないので，回帰分析を行う．図 A.3 は 6 通りの組み合わせにおける回帰変数と切片である．ここから，(sepal-length，petal-width）の組み合わせが最も相関が高いことがわかる．



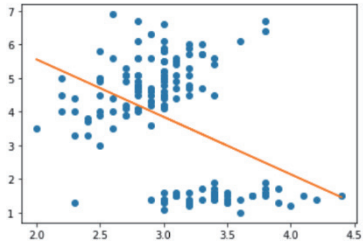
（sepal-length，sepal-width）



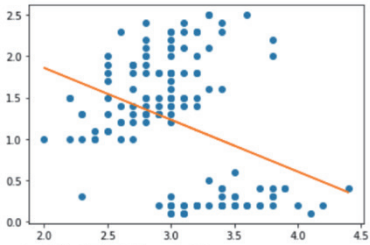
（sepal-length，petal-length）



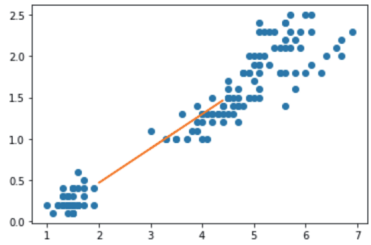
（sepal-length，petal-width）



（sepal-width，petal-length）



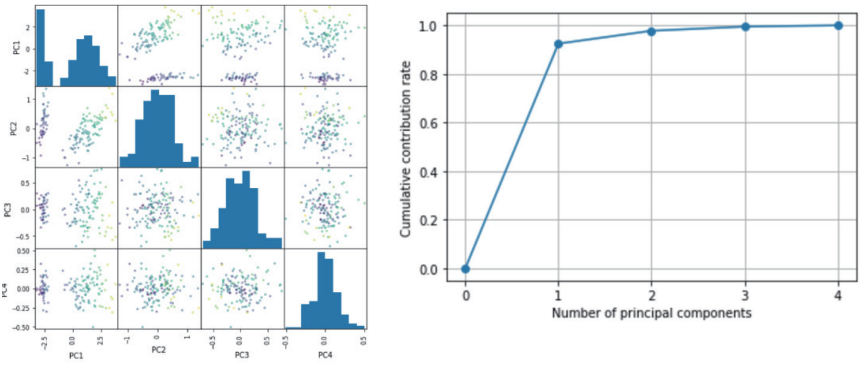
（sepal-width，petal-width）



（petal-length，petal-width）

図 A.3　回帰変数と切片

主成分分析・因子分析　前項のように，回帰分析の段階でシンプルなデータ構造（変数：5，サンプル数：150）であれば関連性を見出すことができるが，膨大な変数を持つデータを扱う場合には主成分分析を行うことで，次元圧縮を行う．アヤメのデータに対して主成分分析を行うと，第 2 主成分で累積寄与率が 95 %を超えているので，元データ変数 5 つに対して次元を 3 つ落とし，2 次元にする．最後に固有ベクトル（表 A.1）を確認すると，PC1 は sepal-length・petal-width・petal-length の特徴を持ち，PC2 は sepal-width の特徴を持っていることがわかる．



(a)　(b)

図 A.4　主成分の散布図と累積寄与率

表 A.1　固有ベクトル

	sepal_length	sepal_width	petal_length	petal_width
PC1	0.522372	−0.263355	0.581254	0.565611
PC2	0.372318	0.925556	0.021095	0.065416
PC3	−0.721017	0.242033	0.140892	0.633801
PC4	−0.261996	0.124135	0.801154	−0.523546

主成分分析がデータを次元圧縮して，各主成分が有する特徴量を取り出すのに対し，多数ある変量同士で共通する因子を抽出し，それらの共通因子を利用して解析を行う方法を因子分析という．表 A.2 は，因子数を 3 として因子分析を行って得られた因子負荷行列を示している．この表から，第

1 因子が sepal-length・petal-width・petal-length の共通因子，第 2 因子が sepal-width の共通因子を有し，主成分分析と同傾向の結果が得られていることがわかる．

表 A.2　因子負荷行列

	第 1 因子	第 2 因子	第 3 因子
sepal_length	0.883318	0.377790	−0.153408
sepal_width	−0.390143	0.749030	0.313899
petal_length	0.994271	−0.030180	−0.031255
petal_width	0.972270	−0.039958	0.166987

Neural Network: NN　ここまではデータ自体の解析を行ってきたが，ここからは機械学習（特に，ニューラルネットワーク）によってアヤメのデータ分類を行う．データセットを以下の 3 つに分けることが，これまでと決定的に異なる点である．

- トレーニングセット（学習セット）
- バリデーションセット（学習セット）
- テストセット

トレーニングセット（教師データ）は，予測器構築のためのパラメータを決めるのに使うデータセットで，これを何度も何度も読み込ませて，目的関数を逐次的に最小化し，最終的なパラメータを決めます．バリデーションセットは，過学習がおきていないことを確認するために用いる疑似テストセット，テストセットは構築した予測器の性能を測るためのデータセットで，学習セットとは独立でなければならない．簡単のため，ここではアヤメのデータに対してバリデーションセットは用意せず，トレーニングセット（訓練データ）とテストセット（検証データ）の 2 種類のデータを用いる．

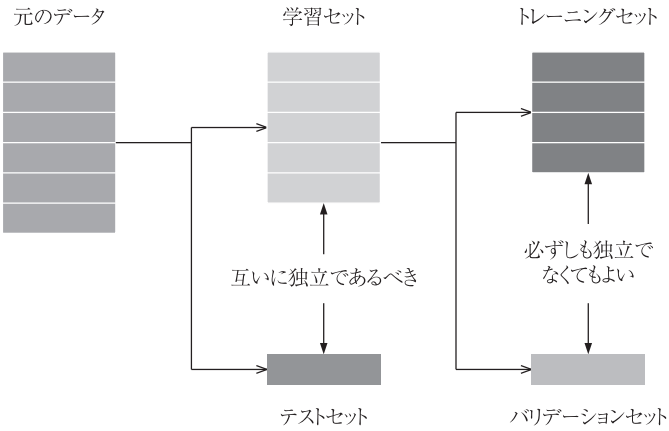


図 A.5　データセットの分割

準備として必要なパッケージをインポートする．

```
import numpy as np
import pandas as pd
from sklearn.neural_network import MLPClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import GridSearchCV
import matplotlib.pyplot as plt
```

図 A.2 (a) から，ダウンロードしたデータは 1〜4 列目に sepal-length, sepal-width, petal-length, petal-width, 5 列目に speicies が 150 サンプル格納されている．ニューラルネットワークで学習させて，sepal-length, sepal-width, petal-length, petal-width から speicies を推定するため，図 A.2 (a) のデータを図 A.6 のように 1〜4 列目（sepal-length, sepal-width, petal-length, petal-width）と 5 列目（speicies）に分解し，それぞれデータ名を feature, target と定義する．この feature, target に対して数学的な演算が可能となるように，`np.array()` を用いて Numpy 形式にデータを変換し，変換後のデータ名を X, Y とする．feature は全て長さの単位であり，オーダーも揃っていることから比較的扱いやすい状況にあるが，一般には，統計解析を行う際にはデータを標準化（平均 0, 標準偏差 1）させておくこ

	sepal_length	sepal_width	petal_length	petal_width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
...	...	...	...	...
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3
149	5.9	3.0	5.1	1.8

150 rows × 4 columns

(a) feature

	species
0	Iris-setosa
1	Iris-setosa
2	Iris-setosa
3	Iris-setosa
4	Iris-setosa
...	...
145	Iris-virginica
146	Iris-virginica
147	Iris-virginica
148	Iris-virginica
149	Iris-virginica

150 rows × 1 columns

(b) target

```
array([[5.1, 3.5, 1.4, 0.2],
       [4.9, 3. , 1.4, 0.2],
       [4.7, 3.2, 1.3, 0.2],
       [4.6, 3.1, 1.5, 0.2],
       [5. , 3.6, 1.4, 0.2],
       [5.4, 3.9, 1.7, 0.4],
       [4.6, 3.4, 1.4, 0.3],
       [5. , 3.4, 1.5, 0.2],
       [4.4, 2.9, 1.4, 0.2],
       [4.9, 3.1, 1.5, 0.1],
       [5.4, 3.7, 1.5, 0.2],
       [4.8, 3.4, 1.6, 0.2],
       [4.8, 3. , 1.4, 0.1],
       [4.3, 3. , 1.1, 0.1],
       [5.8, 4. , 1.2, 0.2],
       [5.7, 4.4, 1.5, 0.4],
       [5.4, 3.9, 1.3, 0.4],
       [5.1, 3.5, 1.4, 0.3],
       [5.7, 3.8, 1.7, 0.3],
       [5.1, 3.8, 1.5, 0.3],
       [5.4, 3.4, 1.7, 0.2],
       [5.1, 3.7, 1.5, 0.4],
       [4.6, 3.6, 1. , 0.2],
       [5.1, 3.3, 1.7, 0.5],
       [4.8, 3.4, 1.9, 0.2],
       [5. , 3. , 1.6, 0.2],
       [5. , 3.4, 1.6, 0.4],
```

(c) X(一部抜粋)

```
array(['Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-setosa', 'Iris-setosa',
       'Iris-setosa', 'Iris-setosa', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor', 'Iris-versicolor', 'Iris-versicolor',
       'Iris-versicolor'. 'Iris-versicolor'. 'Iris-versicolor'.
```

(d) Y(一部抜粋)

```
array([[-9.00681170e-01,  1.03205722e+00, -1.34127240e+00,
        -1.31297673e+00],
       [-1.14301691e+00, -1.24957601e-01, -1.34127240e+00,
        -1.31297673e+00],
       [-1.38535265e+00,  3.37848329e-01, -1.39813811e+00,
        -1.31297673e+00],
       [-1.50652052e+00,  1.06445364e-01, -1.28440670e+00,
        -1.31297673e+00],
       [-1.02184904e+00,  1.26346019e+00, -1.34127240e+00,
        -1.31297673e+00],
       [-5.37177559e-01,  1.95766909e+00, -1.17067529e+00,
        -1.05003079e+00],
       [-1.50652052e+00,  8.00654259e-01, -1.34127240e+00,
        -1.18150376e+00],
       [-1.02184904e+00,  8.00654259e-01, -1.28440670e+00,
        -1.31297673e+00],
       [-1.74885626e+00, -3.56360566e-01, -1.34127240e+00,
        -1.31297673e+00],
       [-1.14301691e+00,  1.06445364e-01, -1.28440670e+00,
        -1.44444970e+00],
       [-5.37177559e-01,  1.49486315e+00, -1.28440670e+00,
        -1.31297673e+00],
       [-1.26418478e+00,  8.00654259e-01, -1.22754100e+00,
        -1.31297673e+00],
       [-1.26418478e+00, -1.24957601e-01, -1.34127240e+00,
        -1.44444970e+00],
       [-1.87002413e+00, -1.24957601e-01, -1.51186952e+00,
        -1.44444970e+00],
       [-5.25060772e-02,  2.18907205e+00, -1.45500381e+00,
        -1.31297673e+00],
       [-1.73673948e-01,  3.11468391e+00, -1.28440670e+00,
        -1.05003079e+00],
```

(c) 標準化後のX(一部抜粋)

図 A.6　元データの分解

とが推奨されている．そこで下記のコマンドで，標準化を実装する．

```
scaler = StandardScaler()
X = scaler.fit_transform(X)
```

ここで，これまで用意してきたデータを訓練データと検証データに分解する．まず下記のコマンドで，訓練データ（X_train，y_train）から検証データ（X_test，y_test）を生成する．

```
X_train, X_test, y_train, y_test = train_test_split(X, y)
```

次に，ニューラルネットワークを準備するため，下記のようなコマンドを記述する．ここで，max_iter=10000 は学習回数を表しているが，十分に誤差が小さくなれば max_iter 回以下でも計算は自動的に終了する．

```
clf = MLPClassifier(max_iter=10000)
```

ニューラルネットワークを利用する場合には下記のように記述し，学習用データ（X_train，y_train）を入力すると学習が開始される．

```
clf.fit(X_train, y_train)
```

学習が終了したら，出力と正解カテゴリーの誤差が clf.loss_curve_に格納されているので，可視化して確認する．学習回数が増えるほど誤差が小さくなっていることがわかる．最後に，学習の成果を確認するために，clf.predict コマンドを使うと予測結果が表示される（図 A.7 (a)）．また，出力と正解カテゴリーの誤差が clf.loss_curve_に格納されているので可視化して確認すると図 A.7 (b) となり，学習回数が増えるほど誤差が小さくなっていることもわかる．

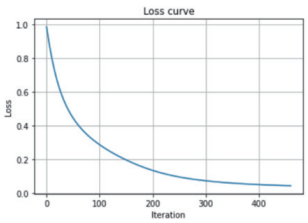
次に，clf.score コマンドを使って正解率を得る．

```
c=clf.score(X_test,y_test)
print("正解率は{}%です.".format(c*100)) # 検証データでの成績表示
```

```
正解率は 92.10526315789474%です.
```

Python では，その他の予測器を実装することも容易である．実際，ニューラルネットワークの場合に

(a)



(b)

図 A.7　学習結果

```
1 from sklearn.neural_network import MLPClassifier
2 clf = MLPClassifier(max_iter=10000)
```

としていた行を，サポートベクターマシンの場合には

```
1 from sklearn import svm
2 clf = svm.SVC(max_iter=10000)
```

ランダムフォレストの場合には

```
1 from sklearn.ensemble import RandomForestClassifier
2 clf = RandomForestClassifier(random_state=1234
```

ロジスティック回帰の場合には

```
1 from sklearn.linear_model import LogisticRegression, LinearRegressio
      n
2 clf = LogisticRegression(max_iter=10000)
```

などと，書き換えるだけでよい．ただしパラメータチューンングは対象とする問題に応じて適宜，変更する必要があるので注意する．

留意点　実データでは，機械学習を行う前に主成分分析などを行い，特徴量を抽出した後，それらを入力データとする方が正解率の向上がみられる場合がある．ただし，これらの操作ではデータの性質によって対応を変える必要があり，一概に確実な方法があるとはいえないようである．また，実データでは分類器を用いて多くのハイパーパラメータをチューニングする必要があり，これらもデータの性質によってやり方を変更する必要がある．

● コラム A.2　ビッグデータの学習

気象・金融などの膨大な時系列データ，画像データ，蛋白質構造データなどを学習させる場合には，入力するデータだけで数〜数百 Tb 必要とする場合もある．入力データを格納可能な Memory を準備したり，スパースモデリングなどで次元圧縮を行ったデータを入力データとすることも求められる．ソフトウェアについても，比較的小規模なデータであれば R で十分だが，上述のような大規模データとなると Python を用いることも必要である．Python には，Numpy や PyTorch，Scikit-Learn などの特に機械学習向けに準備されている科学計算ライブラリーが整備されているので，まずはこれらを活用する．計算機環境についても，格納する入力データに依存はするが，GPU を用いることで学習を高速に進めることができる．1.2 節の節末問題解答で注意したように，各企業や研究室で秘匿情報については絶対に開示しないことはいうまでもない．

● コラム A.3　プログラミングと有限オートマトン

コンピュータを使って，売上データの集計をしたりゲームでキャラクターを描画するといったことをするためには，プログラムを書かなければならない．AI エージェントに対しては，日常の言語を使ったコミュニケーションも可能になりつつあるが，プログラムに関しては，指示が多段になることや，厳密性が要求されることから，まだまだ専用の言語を使う必要がある．たとえば，プログラムに一か所「;」が抜けているだけでも，エラーとなって実行できなってしまう．融通が利かないともいえるが，人間が意図しない計算結果が出力されることを避けるためにはやむを得ない．

プログラムは，たとえば，以下のような形で表現されるものである．

```
1 import java.io.*;
2 class Sample throws IOException{
3   public static void main(String[] args){
4     BufferedReaderbr= ....
5   }
6 }
```

これは，Java というプログラミング言語で書かれたもので，1 行目は，人間が見ると，java.io.*というライブラリ（モジュール）を取り込む（import）と読める，コンピュータにとっては，ｉｍｐｏｒｔといった単なる文字列にしか見えない．つまり，どこからどこまでが意味のあるかたまりかわからないのである．

そこで登場するのが字句解析という技術で，一文字ずつ認識するよりも，「`import`」,「`java.io.*`」,「`;`」,「`class`」,「`Sample`」,「`throws`」,「`IOException`」とった単語（トークン）から構成されるものとするものである．ではトークンに分けるにはどうすればよいであろうか．スペース（空白）が現れればそこで区切るというのも 1 つのアイデアであるが，「`IOException{`」の部分など，それですべてが扱えるというわけではない．

ここで役立つのが有限オートマトンの技術である．この技術を適用すれば，どのような文字列であれば，そこで区切って受け入れるかを表現することができる．たとえば，図 A.8 は奇数個の文字からなり 0,1 の列を受け入れる．従って 0, 1, 001, 111 などは受け入れるが，00, 10 などは受け入れない．詳細は述べられないが，下図は，ある文字列が与えられたとき，それを意味のあるかたまりとしてのトークンに分割していく方法を示している．

人間が理解できるプログラムの表現を，コンピュータが理解できる表現に変換する技術をコンパイラとよぶ．字句解析はコンパイラの機能の 1 つで，この処理の後，構文解析→意味解析→最適化→コード生成といった段階をたどって，はじめてコンピュータが実行可能な形態になる．

身の回りにはスマートフォンをはじめ，多くの情報機器があり，その情報機器の中では数多くのプログラムが動作している．そのプログラムを作るという部分で，コンパイラが活躍し，コンパイラの実現に有限オートマトンの考え方が使われているのである．

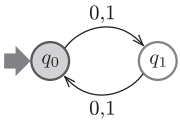


図 A.8