第四章

1.数百万個もの特徴量を持つ訓練セットがあるときに使える線形回帰訓練アルゴリズムは何か。

2.訓練セットの特徴量のスケールがまちまちだとする。これによって悪影響を受けるアルゴリズムは何で、どのような影響があるか。その問題にはどのように対処すればよいか。

3.ロジスティック回帰モデルを訓練しているときに、勾配降下法が局所的な最小値から抜け出せなくなることはあるか。

4.十分な実行時間を与えれば、すべての勾配降下法アルゴリズムは同じモデルに帰着するか。

5.バッチ勾配降下法を使っていて、エポックごとに検証誤差をプロットしているものとする。検証誤差が絶えず大きくなっていることに気づいた場合、何が起きていると考えられるか。この問題はどのように修正すればよいか。

6.検証誤差が上がりだしたときにミニバッチ勾配降下法をすぐに中止するのはよいことか。

7.本書で取り上げた勾配降下法アルゴリズムの中で、最適な解の近辺に最も早く到達するのはどれか。それは実際に収束するか。ほかの勾配降下法はどうすれば収束するか。

8.多項式回帰を使っているものとする。学習曲線をプロットしたところ、訓練誤差と検証誤差の間に大きな差があった。何が起きているのか。この問題を解決するための方法を3つ挙げなさい。

9.Ridge回帰を使っていて、訓練誤差と検証誤差がほとんど同じだが非常に高いことに気づいたとする。バイアスと分散のどちらが高いとそうなるか。正則化ハイパーパラメータのαは上げるべきか、下げるべきか。

10.以下を説明しなさい。

・線形回帰（正則化項なし）ではなくRidge回帰を使うべき理由

・Ridge回帰ではなくLasso回帰を使うべき理由

・Lasso回帰ではなくElastic Netを使うべき理由

11.写真を屋内/屋外、日中/夜間に分類したいものとする。2つのロジスティック回帰分類器を作るべきか、それとも1つのソフトマックス回帰分類器を作るべきか。

12.scikit-learnを使わず、ソフトマックス回帰のための早期打ち切り機能を持つバッチ勾配降下法を実装しなさい。

第五章

1.サポートベクトルマシンの基本的な考え方は何か。

2.サポートベクトルとは何か。

3.**SVM**を使う時に入力をスケーリングするのが重要なのはなぜか。
4.**SVM**分類器はインスタンスを分類するときに確信度のスコアを出力できるか。確率はどうか。
5.特徴量が数百個、インスタンスが数百万個の訓練セットでモデルを訓練するとき、**SVM**の主問題と双対問題のどちらを使うべきか。
6.**RBF**カーネル付きの**SVM**分類器を訓練したとする。訓練セットに過小適合しているように見えるが、γを増やすべきか、それとも減らすべきか。
7.できあいの**QP**ソルバーを使ってソフトマージン線形**SVM**分類器の問題を解決するためには、**QP**パラメータ(**H,f,A,b**)をどのように設定すべきか。
8.線形分割可能なデータセットで**LinearSVC**を訓練しなさい。次に、同じデータセットで**SVC**と**SGDClassifier**を訓練しなさい。ほぼ同じモデルが出来るかどうかを確かめなさい。
9.**MNIST**データセットを使って**SVM**分類器を訓練しなさい。**SVM**分類器は二項分類器なので、**10**種類のすべての数字を分類するためには、**OVA**法を使う必要がある。小さな検証セットを使ってハイパーパラメータを調整し、プロセスを高速化したい。どの程度の正解率が得られるか。
10.カリフォルニアの住宅価格データセットを使って**SVM**分類器を訓練しなさい。

第六章
1.百万個のインスタンスを持つ訓練セットで決定木を訓練するとき、おおよその深さはどれくらいか。
2.ノードのジニ不純度は一般に親よりも高いか、それとも低いか。それは一般にか常にか。
3.決定木が訓練セットを過学習している場合、**max_depth**を下げるとよいか。
4.決定木が訓練セットに過小適合している場合、入力特徴量を増やすとよいか。
5.インスタンスが百万個ある訓練セットを対象として決定木を訓練するために一時間かかるとき、インスタンスが一千万個の訓練セットを対象として別の決定木を訓練するためにどれくらいの時間がかかるか。
6.訓練セットのインスタンスが**100000**個あるとき、**presort=True**を設定すると訓練のスピードは上がるか。
7.次の手順で**moons**データセットで決定木を訓練し、微調整しなさい。
a.**make_moons(n_samples=10000**、**noise=0.4)**を使って**moons**データセットを生成しなさい。
b.**train_test_split()**を使ってデータセットを訓練セットとテストセットに分割しなさい。

c.グリッドサーチと交差検証を使って**DecisionTreeClassifier**のハイパーパラメータ値としてよいものを探しなさい。
d.見つけたハイパーパラメータを指定して訓練セット全体を対象に訓練を行い、テストセットでモデルの性能を測定しよう。**85%**から**87%**の正解率が得られるはずだ。
8.次の手順で森を育てなさい。
a.前問に引き続き、訓練セットからそれぞれ**100**個のインスタンスを無作為に選択した**1000**個のサブセットを作りなさい。
b.前問で見つけた最良のハイパーパラメータを指定して個々のサブセットごとに**1**つの決定木を訓練しなさい。そして、テストセットを対象として**1000**個の決定木を評価しなさい。小規模なセットで訓練したものなので、これらの決定木は**80%**くらいの正解率しか得られないだろう。
c.ここで魔法を起こす。個々のテストセットインスタンスについて、**1000**個の決定木で予測を行い、もっとも多くの決定木が予測した値だけを残す。こうするとテストセットに対する多数決予測が得られる。
d.テストセットに対する予測を評価しよう。最初のモデルと比べてわずかに高い性能が得られるはずだ。おめでとう。あなたはランダムフォレスト分類器を訓練したことになる。

第七章
1.全く同じ訓練データを使って**5**つの異なるモデルを訓練し、それらがすべての**95%**の適合率を達成したとき、それらのモデルを組み合わせたらもっと良い結果が得られる可能性はあるか。もしそうだとすれば、どうすればそのような結果が得られるか。そうでないとすればそれはなぜか。
2.ハード投票分類器とソフト投票分類器の違いは何か。
3.複数のサーバーで分散処理することによってバギングアンサンブルのスピードを上げることはできるか。ペースティングアンサンブル、ブースティングアンサンブル、ランダムフォレスト、スタッキングアンサンブルではどうか。
4.**OOB**検証の長所は何か。
5.**Extra-Trees**分類器が通常のランダムフォレストよりも無作為的なのは何によるものか。この余分に無作為的なことにはどのような意味があるか。**Extra-Trees**は、通常のランダムフォレストと比べて遅いか、それとも速いか。
6.手元のアダブーストアンサンブルが訓練データに過小適合している場合、どのハイパーパラメータをどのように調整すべきか。

7.勾配ブースティングアンサンブルが訓練セットを過学習している場合、学習率を上げるべきか、下げるべきか。
8.MINISTデータをロードし、それらを訓練セット、検証セット、テストセットに分割し、ランダムフォレスト分類器、Extra-Trees分類器、SVMなどの様々な分類器を訓練しよう。次に、それらの分類器をソフト投票分類やハード投票分類などのアンサンブルに結合し、検証セットを対象として個別のすべての分類器よりも性能の高いものを探そう。そのようなものが見つかったらテストセットで試してみよう。個別の分類器と比べてどれくらい高い性能が得られたか。
9.検証セットを対象として前問の個別の検証器で予測を行いその予測結果から新しい訓練セットを作りなさい。その訓練セットの個々の訓練インスタンスは、イメージに対してすべての分類器が返した予測をまとめたベクトルで、ターゲットはイメージのクラスである。ブレンダと分類器は全部まとめてスタッキングアンサンブルを形成している。では、テストセットを使ってアンサンブルを評価してみよう。テストセットに含まれる個々のイメージについて、すべての分類器で予測を行い、その結果をブレンダに送ってアンサンブルとしての予測を行う。前問で訓練した投票分類器と比較して性能はどうなっているか。

第八章
1.データセットを次元削減する主要な要因は何か。次元削減の主要な欠点は何か。
2.次元の呪いとは何か。
3.データセットを次元削減した後で、次元をもとに戻すことはできるか。できるならどのようにしてするのか。できないならなぜか。
4.PCAは、高次非線形データセットの次元削減に使えるか。
5.因子寄与率(explained variance ratio)を95%に設定して1000次元のデータセットにPCAを適用する場合、得られるデータセットの次元はどの程度になるか。
6.通常のPCA、逐次学習型PCA、ランダム化PCA、カーネルPCAはどのように使い分けるか。
7.データセットに対する次元削減アルゴリズムの性能はどのようにすれば評価できるか。
8.2つの異なる次元削減アルゴリズムを続けて使うことに意味はあるか。
9.MINISTデータセットをロードして、訓練セットとテストセットに分割しなさい。(最初の60000件を訓練用、10000件をテスト用にする)。このデータセットを使ってランダムフォレスト分類器を訓練し、かかった時間を測定してから、得られたモデルの性能をテストセットで評価する。次に、因子寄与率95のPCAで次元削減する。次元削減後のデータセットで新しいランダムフォレスト分類器を訓練し、かかった時間を計測して次元削減前の分類器の

訓練時間と比較しなさい。また、テストセットで分類器を評価し、前の分類器と性能を比較しなさい。

10.t-SNEを使って**MINIST**データセットを**2**次元に次元削減し、**Matplotlib**を使って結果をグラフにしなさい。**10**色で個々のイメージのターゲットクラスを表現する散布図を使うこと。個々のインスタンスの位置に色つきの数字を表示したり、スケールダウンした数字イメージ自体をプロットしたりしてもよい。数字のクラスタがはっきりと分かれた感じのグラフが得られるはずだ。**PCA**、**LLE**、**MDS**など、ほかの次元削減アルゴリズムも試して、得られたビジュアライゼーションを比較してみよう。

第九章

1.あなたならクラスタリングをどのように定義するか。また、クラスタリングアルゴリズムをいくつか挙げなさい。

2.クラスタリングアルゴリズムの主要な応用分野をいくつか挙げなさい。

3.K平均法を使う時に適切なクラスタ数を選択するための方法を**2**つ説明しなさい。

4.ラベル伝播とは何か。なぜそのようなものを実装するのか。またどうすれば実装できるか。説明しなさい。

5.大規模なデータセットに対するスケーラビリティが高いクラスタリングアルゴリズムを**2**つ挙げなさい。また、高密度の領域を探すクラスタリングアルゴリズムを**2**つ挙げなさい。

6.能動学習が役に立つユースケースを挙げなさい。また、どのように実装するか答えなさい。

7.異常検知と新規検知の違いは何か。

8.混合ガウスモデルとは何か。どのようなタスクで使えるか。

9.混合ガウスモデルを使っているときに適切なクラスタ数を見つけるための**2**つのテクニックとは何か。

10.古典的なオリベッティ顔画像データセットには、**40**人の顔写真が**10**回ずつ撮影されており、**64×64**ピクセルのグレイスケール画像が**400**個含まれている。個々の画像は**4096**要素の**1**次元ベクトルに平坦化されている。通常のタスクは、個々の写真が誰を撮ったものか予測することである。**sklearn.datasets.fetch_olivetti_faces()**関数を使ってデータセットをロードし、訓練セット、検証セット、テストセットに分割しなさい。データセットの規模が小さいので、階層化サンプリングを使って同一人物の写真が各セットに同数ずつ入るようにすべきだ。次に、**K**平均法を使って画像をクラスタリングしなさい。クラスタ数は、この章で説明したテクニックのどれかを使って妥当なものを選ぶようにすること。

最後にクラスタを可視化しなさい。各クラスタに同じ人の顔が含まれているだろうか。

11.引き続きオリベッティ顔画像データセットを使い、個々の写真が誰を撮ったものかを予測する分類器を訓練しなさい。検証セットで性能を評価すること。
次に、**K**平均法と次元削減ツールを使って、次元削減後のデータセットで分類器を訓練しなさい。分類器が最高性能を出せるクラスタ数を探してみよう。どの程度の性能が得られたか。次元削減後のデータセットの特徴量を元のデータセットに追加したらどうか。（ここでも最良のクラスタ数を探すこと。）
12.オリベッティ顔画像データセットで混合ガウスモデルを訓練しなさい。アルゴリズムを高速化するために、おそらくデータセットの次元削減が必要になる。（たとえば、分散の**99%**を維持しながら**PCA**を使ってみよう）。作ったモデルで新しい顔画像を生成し、（**sample()**メゾットを使う）、可視化しなさい（**PCA**を使った場合には、**PCA**の**inverse_transform()**メゾットを使う）。また、一部の画像に変更を加え（たとえば回転、反転、暗色化）、モデルが異常値を検知するかどうか試してみよう。（つまり正常な画像と異常にした画像で**score_samples()**メゾットの出力を比較する）。
13.次元削減テクニックの中には、異常検知にも使えるものがある。たとえば、分散の**99%**を維持しながら、オリベッティ顔画像データセットを**PCA**で次元削減し、個々の画像の再構築誤差を計算しなさい。
次に、前問で元の画像を変形して作った画像の一部を取り出し、その再構築誤差を計算しなさい。再構築誤差が大きくなったはずだ。再構築後のイメージをプロットすればその理由がわかる。再構築アルゴリズムは、正常な顔写真を再構築しようとしていたのである。