?

4章　モデルの訓練

1. 勾配降下法
2. 悪影響を受けるアルゴリズムは勾配降下法で、損失関数を最小化するパラメータを探索するのに時間がかかるという影響がある。これに対処するには、scikit-learnのStanderdScalerクラスを使うなどして全ての特徴量が同じスケールになるようにする。
3. ロジスティック回帰の損失関数（Log Loss）は凸関数であり、局所的な最小値は存在しないので、当然そこから抜け出せなくなるようなこともない。
4. バッチ勾配降下法、確率的勾配降下法、ミニバッチ勾配降下法が紹介されているが、これらからは厳密には互いに異なるモデルが得られる。なぜなら、バッチ勾配降下法は真の最小値に収束する一方、残りの二つは十分なイテレーション数を経ても真の最小値付近でずっと変動し続けるからである。しかし、確率的勾配降下法やミニバッチ勾配降下法については適切な学習スケジュール（例えば、イテレーションが増えるごとに学習率を減らしていく）を用いれば真の最小値に辿り着ける。また、そうせずとも、3つのアルゴリズムからはほとんど同じモデルが得られる。
5. 学習率が大きすぎることで損失関数が発散していると考えられる。この場合、学習率を下げてからやり直せば良い。
6. ミニバッチ勾配降下法においては、イテレーション数に関する損失関数の値の推移が滑らかではないため、検証誤差が上がり出した直後の段階では最小値に達したかどうか判断できない。よって、これは良い方法ではない。検証誤差がしばらくの間上がり続けたことを確認した後、最小値となった時のパラメータにロールバックするのが良い。
7. 最適な解の近辺に最も速く到達するのは確率的勾配降下法である。このアルゴリズムは、そのままでは収束しないが、イテレーションごとに学習率を下げていくといった学習スケジュールを取れば収束する。その他のアルゴリズムについては、損失関数が凸関数の場合、バッチ勾配降下法は収束し、ミニバッチ勾配降下法は確率的勾配降下法と同様のことが言える。
8. 過学習が起きている。これに対処するには、
    a. 検証誤差が訓練誤差に達するまで訓練データを増やす
    b. 次数を下げる
    c. 早期打ち切りを行う

   などが挙げられる。
9. バイアスが高いと問題文のように過小適合しやすい。αの値は下げるべきである。
10. まず、線形回帰ではなく、Ridge回帰を使うべき理由を述べる。Ridge回帰では新たに正規化項が加えられたことによって、パラメータの重みをできるだけ小さくするようになる。このおかげで過学習が起こりにくいという強みを持つことになる。次に、Ridge回帰ではなくLasso回帰を使うべき理由を述べる。Lasso回帰の正規化項は、高次多項式特徴量のように不要なパラメータを0にする働きを持つ。このおかげで、必要最低限のパラメータを選択できるという強みを持つことになる。最後に、Lasso回帰よりもElastic Netを使うべき理由を述べる。Lasso回帰では、訓練インスタンスの数よりも特徴量の数の方が多い時や、複数の特徴量間に強い相関があるときに不規則な動きを示すことがある。Ridge回帰とLasso回帰の混合であるElastic Netではこの不規則性の影響を減らすことができるため、このアルゴリズムを用いた方が良い。
11. 2つのロジスティック回帰分類器を作るべきである。屋外/屋内、日中/夜間といったクラスは相互排他的ではないため、多出力ではないソフトマックス回帰は使えない。
12. ファイル「code_4_12.ipynb」参照のこと。