プログラミング練習4：

ニューラルネットワークの学習

機械学習

導入

この練習では、神経のバックプロパゲーションアルゴリズムを実装します
手書きの数字認識のタスクに適用します。前
プログラミング演習から始めて、
ビデオ講義を行い、関連トピックのレビュー質問を完了します。
エクササイズを開始するには、スターターをダウンロードする必要があります
コードを完成させたいディレクトリにその内容を解凍します
運動。必要に応じて、Octave / MATLABでcdコマンドを使用して変更する
この練習を始める前にこのディレクトリに入れてください。
Octave / MATLABのインストール手順については、「En-
コースのウェブサイトの「vironment Setup Instructions」を参照してください。

この練習に含まれるファイル

ex4.m - エクササイズの手順を示すOctave / MATLABスクリプト
ex4data1.mat - 手書き数字のトレーニングセット
ex4weights.mat - エクササイズ4のニューラルネットワークパラメータ
submit.m - ソリューションをGoogleのサーバーに送信する送信スクリプト
displayData.m - データセットを視覚化するための関数
fmincg.m - 関数最小化ルーチン（fminuncと同様）
シグモイドm - シグモイド関数
computeNumericalGradient.m - 数値的に勾配を計算する
checkNNGradients.m - グラデーションの確認に役立つ関数
debugInitializeWeights.m - ウェイトを初期化する関数
predict.m - ニューラルネットワーク予測関数
[*] sigmoidGradient.m - シグモイド関数の勾配を計算する
[*] randInitializeWeights.m - ランダムにウェイトを初期化する
[*] nnCostFunction.m - ニューラルネットワークコスト関数

*は、完了する必要があるファイルを示します

エクササイズでは、ex4.mというスクリプトを使用します。これらのスクリプト
問題のデータセットを設定し、あなたが望む関数を呼び出す
書きます。スクリプトを変更する必要はありません。あなたは変更する必要があります
この課題の指示に従うことによって、他のファイルの機能を使用することができます。

助けを得る場所

このコースの演習ではOctaveを使用しています [1]またはMATLAB、高レベルのプログ
数値計算によく適しています。あなたが持っていない場合
オクターブまたはMATLABがインストールされている場合は、
コースウェブサイトの「環境設定手順」を参照してください。
Octave / MATLABコマンドラインでhelpと入力してからfunc-
組み込み関数のドキュメントを表示します。たとえば、ヘルプ
プロットはプロットのヘルプ情報を表示します。の詳細なドキュメント
オクターブ機能はで見つけることができ オクターブドキュメントページ 。マット-
LABのドキュメントはで見つけることができます MATLABドキュメントのページ 。
また、オンラインディスカッションを使用して、
他の学生と一緒にercises。しかし、書かれたソースコードを見ないでください
あなたのソースコードを他の人と共有することができます。

1ニューラルネットワーク

前回の練習では、あなたはneu-
ralネットワークを使用し、それを使って手書き数字を予測しました。
提供されます。この演習では、バックプロパゲーションアルゴリズムを実装します
ニューラルネットワークのパラメータを学習する。
提供されたスクリプトex4.mは、このエクササイズを手助けします。

1.1データの可視化

ex4.mの最初の部分では、コードがデータをロードし、それを
2次元プロット（図 1 ）関数displayDataを呼び出すことによって、

1 オクターブは、MATLABへの無料の代替です。プログラミング演習では、あなたは無料です
OctaveまたはMATLABを使用します。

図1：データセットの例

これは、前の練習で使用したのと同じデータセットです。がある
ex3data1.matの5000のトレーニング例。各トレーニング例は
その桁の20画素×20画素のグレースケール画像。各画素は、
その位置でのグレースケール強度を示す浮動小数点数。
20×20ピクセルのグリッドは、400次元ベクトルに「展開」されます。各
これらのトレーニング例のうちの1つは、データ行列Xの1つの行になります。これは
私たちに5000×400の行列Xを与えます。各行は、すべての行が
手書き数字画像。

$$X = \begin{bmatrix} - (X^{(1)})^T - \\ - (X^{(2)})^T - \\ \vdots \\ - (X^{(M)})^T - \end{bmatrix}$$

トレーニングセットの第2の部分は、5000次元のベクトルyである。
トレーニングセットのラベルが含まれています。物事の互換性を高める
オクターブ/ MATLABインデックス作成、ゼロインデックスがない場合、マップされています
0から10までの数字。したがって、「0」桁は「10」とラベル付けされ、一方、
数字「1」～「9」は自然順序で「1」～「9」とラベル付けされている。

1.2モデル表現

我々のニューラルネットワークは、図に示されている 2 。それは3つの層を持っています - 入力層、
隠れ層と出力層を含む。入力はピクセル値であることを思い出してください

桁の画像の 画像は20×20の大きさなので、これは400の入力を与える
（+1を出力する余分なバイアスユニットはカウントしません）。ザ
トレーニングデータは、ex4.mスクリプトによって変数Xとyにロードされます。
あなたは、ネットワークパラメータ$(\Theta^{(1)}, \Theta^{(2)})$のセットで提供されています
すでに私たちによって訓練されています。これらはex4weights.matに格納され、
Theta1とTheta2にex4.mによってロードされます。パラメータには寸法があります
第2層に25ユニット、第2層に25ユニットを有するニューラルネットワークのためのサイズである。
出力単位（10桁のクラスに対応）。

```
%保存された行列をファイルからロードする
負荷（ 'ex4weights.mat'）;

%Theta1とTheta2の行列がワークスペースに追加されます
%Theta1のサイズは25 x 401です
%Theta2のサイズは10 x 26です
```

図2：ニューラルネットワークモデル

1.3フィードフォワードおよびコスト関数

今度は、ニューラルネットのコスト関数と勾配を実装します。
作業。まず、nnCostFunction.mでコードを完成させてコストを返します。

ニューラルネットワークのためのコスト関数（regulariza-

）は

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} \sum_{k=1}^{K} [-y_k^{(i)} \log((H_\theta(X^{(i)}))_k) - (1 - y_k^{(i)}) \log(1 - (H_\theta(X^{(i)}))_k)]$$

ここで、k）時間 $\theta^{(X^{(I)})}$ は 、図のように計算される 2及びK = 10、合計であります
可能なラベルの数。その時間 $\theta^{(X^{(i)})}$ を 、K = Aに注意してください超動（出力
k番目の出力部の出力値である。また、元のラベル
（変数y内）は1,2、...、10であり、神経を訓練する目的で
ネットワークでは、ラベルを値0または1だけを含むベクトルとして再コード化する必要があります
1となるので、

$$y = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \text{または} \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \text{または} \begin{bmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{bmatrix}$$

Xは、$^{(i)}$、次に数字5の画像である場合、例えば、対応します
Y（あなたはコスト関数で使用することを）$^{(i)}$ は 10次元でなければなりません
$Y_5 = 1$、および0に等しい他の要素を持つベクトル 。
あなたは時間 θ を 演算 する フィードフォワード計算を実装する必要があります $^{(X^{(i)})}$ を
すべての例iに対してコストを合計します。あなたのコードはすべきです
任意の数のラベルを使用して、任意のサイズのデータセットに対しても機能します（
少なくともK≧3ラベルがあると仮定することができる）。

実装上の注意：行列Xには行の例が含まれています
（すなわち、X（iは、:）'i番目の訓練例の $x^{(i)}$ は 、×1のように表現
ベクトル）。nnCostFunction.mでコードを完成すると、
1の列をX行列に追加する必要があります。それぞれのパラメータ
ユニットは、Theta1とTheta2に1つとして表されます
行。具体的には、Theta1の第1行は、
第2層のユニット。for-loopを使用すると、
コストを計算する。

完了したら、ex4.mはロードされたnnCostFunctionを使用してnnCostFunctionを呼び出します
Theta1とTheta2のパラメータセット。あなたは、コストが
約0.287629。

今すぐソリューションを提出する必要があります。

1.4正規化コスト関数

正則化されたニューラルネットワークのコスト関数は、

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}\left[-y_k^{(i)}\log\left(\left(H_\theta(X^{(i)})\right)_k\right) - \left(1-y_k^{(i)}\right)\log\left(1-\left(H_\theta(X^{(i)})\right)_k\right)\right] + \frac{\lambda}{2m}\left[\sum_{j=1}^{25}\sum_{k=1}^{400}\left(\Theta_{J,K}^{(1)}\right)^2 + \sum_{j=1}^{10}\sum_{k=1}^{25}\left(\Theta_{J,K}^{(2)}\right)^2\right]$$

ここで、k)はログ - (1 kyの ログ(1 - (Hθ(X(i))を)

あなたは、ニューラルネットワークが3つのレイヤーしか持たないと仮定することができます - 入力層、隠れ層、および出力層を含む。しかし、あなたのコードはうまくいくはずです
任意の数の入力単位に対して、隠れ単位と出力単位を指定できます。我々は
メモを行い、分かりやすくするために$\Theta^{(1)}$及び$\Theta^{(2)}$
にわたっては、上記の合計インデックスを明示的に列挙されていません
あなたのコードは、任意のサイズの$\Theta^{(1)}$及び$\Theta^{(2)}$を、一般的な作業でなければならないこと 。

それに対応する用語を正規化してはならないことに注意してください
バイアス。Theta1とTheta2の行列の場合、これは最初の
各行列の列。これで、コストに正規化を追加する必要があります
関数。まず、不規則コスト関数を計算することができます
Jを使用して、既存のnnCostFunction.mを使用してから、後で
正則化の用語。

完了したら、ex4.mはロードされたnnCostFunctionを使用してnnCostFunctionを呼び出します
Theta1とTheta2のパラメータセット、λ= 1です。
コストは約0.383770です。

今すぐソリューションを提出する必要があります。

2バックプロパゲーション

練習のこの部分では、バックプロパゲーションのalgo-
ニューラルネットワークコスト関数の勾配を計算する。君は
nnCostFunction.mを完了する必要があります。これにより、
卒業生のための値を食べました。グラデーションを計算したら、あなたは
コスト関数J(Θ)を最小化することによってニューラルネットワークを訓練する。
fmincgなどの高度なオプティマイザ

まず、バックプロパゲーションアルゴリズムを実装して、
（規則化されていない）ニューラルネットワークのパラメータの勾配を計算する。後

あなたは、正規化されていない場合の勾配計算
正則であるニューラルネットワークの勾配を実装します。

2.1シグモイド勾配

練習のこの部分を始めるのを手助けするために、まずあなたは
シグモイド勾配関数 シグモイド関数の勾配は、
として計算される

$$G_{/(Z)} = \frac{d}{dz}g(z) = g(z)(1-g(z))$$

どこで

$$\text{シグモイド}(z) = g(z) = \frac{1}{1 + E^{-z}}$$ 。

完了したら、sigmoidGradient（z）を呼び出していくつかの値をテストしてみてください。
Octave / MATLABコマンドラインで実行します。大きな値（正と負の両方
負である）zの場合、勾配は0に近くなければならない.z = 0の場合、
entは正確に0.25でなければなりません。あなたのコードはベクトルと
行列。行列の場合、関数はシグモイドグラデーションを実行する必要があります
すべての要素に対して機能します。

今すぐソリューションを提出する必要があります。

2.2ランダム初期化

ニューラルネットワークを訓練するときは、無作為に初期化することが重要です。
対称性を破壊するためのラマー。ランダム初期設定のための1つの効果的な戦略は、
ションは、Θのためにランダムに選択した値（リットル）均一の範囲内の[INIT-ε、ε のinit] です 。
あなたは 、initが 0.12を=εを使用する必要があります 。2値のこの範囲は、パラメータことを保証します
学習がより効率的になります。

あなたの仕事は、randInitializeWeights.mを完了してウェイトを初期化することです
Θについては、ファイルを変更し、次のコードを入力します。

%重みを小さな値にランダムに初期化する
εinit = 0.12;
W = rand（L out、1 + L in）* 2 *εinit - εinit

この演習のこの部分でコードを提出する必要はありません。

εの初期化を選択するための 2 つの効果的な戦略は、 内のユニットの数に、それをベースにあります
ネットワーク。εinitをの良い選択は、ε のinit =です$\frac{\sqrt{6}}{\sqrt{L_{\text{で}} + L_{\text{アウト}}}}$、= S ıとL で L アウト = sでのリットル+ 1がある場所
Θ（リットル） に隣接する層のユニットの数 。

2.3バックプロパゲーション

図3：バックプロパゲーションアップデート。

ここで、バックプロパゲーションアルゴリズムを実装します。思い出してください
逆伝播アルゴリズムの背後にある直観は以下の通りである。与えられた
訓練例 (X(t)は 、Y(t))を 、我々が最初に計算するために「往路」を実行します
ネットワーク全体のすべてのアクティベーション。
仮説時間 Θ(x)とします。次に、レイヤーlの各ノードjについて、
「誤差項」δ $\delta_j^{(1)}$そのノードがどれだけ「責任がある」かを測定します
私たちの出力に誤りがあるかどうか。

出力ノードでは、ノード間の差を直接測定することができます。
ネットワークの活性化と真の目標値を計算し、それを使ってδ $\delta_j^{(3)}$
（レイヤー3は出力レイヤーなので）。隠れユニットについては、
$\delta_j^{(1)}$層内のノードの誤差項の加重平均に基づいて
(l + 1)となる。

詳細には、ここでは、バックプロパゲーションアルゴリズム（図
3 ）。1つの例を処理するループに手順1～4を実装する必要があります
一度に。具体的には、t = 1 : mのforループを実装し、
場所は、forループ 番目の反復トンと、実行内側に以下の工程1-4
トレーニング例t 番目 (X(t)は 、Y(T)) の計算 。ステップ5では、
ニューラルネットワークの勾配を得るためにm累積された勾配
コスト関数。

1. t番目のトレーニング例のx(t)に入力層の値 ((1))を 設定します 。
フィードフォワードパスを実行します（図 2 ）、アクティベーションを計算する(Z(2)、(2)、Z(3)、(3))
レイヤー2とレイヤー3の場合は、+1タームを追加する必要があります。
レイヤのアクティベーションのベクターは、(1)及び(2)また、バイアスを含みます
単位。Octave / MATLABでは、1が列ベクトルの場合、
1 = [1; a 1]。

2.レイヤ3（出力レイヤ）の各出力ユニットkに対して、

$$\delta_k^{(3)} = (a_k^{(3)} - Y\text{は}_K)、$$

ここで、y kは ∈{0,1}は、現在の訓練例かどうかを示しBE-
クラスk（Yの K = 1） にlong値 、または、異なるクラス（YのK = 0）に属する場合。
この作業に役立つ論理的な配列を見つけることができます（事前に説明されています）
大規模なプログラミング演習）。

3.隠れ層l = 2の場合、

δ(2)=(Θ(2))T δ(3)。*グラム/(Z(2))

4.この例の勾配を次のfor-
mula。δをスキップまたは削除する必要があることに注意してください
δを取り除く デルタ2 =デルタ2(2:エンド)に対応する。

$$\Delta(L)=\Delta(L)+\delta(L+1)((L))T$$

5.ニューラルネットワークコスト関数の(不規則化された)勾配を取得する。
によって蓄積された勾配を分割することによりション :

$$\frac{\partial}{\partial\Theta_{ij}^{(1)}}J(\Theta)=D_{IJ=}^{(1)}\frac{1}{m}\Delta_{ij}^{(1)}$$

Octave / MATLABヒント:バックプロパゲーションを実装する必要があります
アルゴリズムは、フィードフォワードを正常に完了した後でなければなりません。
コスト関数。バックプロパゲーションアルゴリズムを実装している間は、
サイズ関数を使用してバリアントのサイズをプリントアウトするのに便利なことがよくあります。
ディメンションの不一致エラーが発生した場合の作業
(Octave / MATLABの "nonconformant arguments"エラー)。

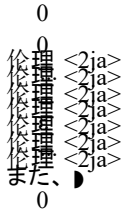
バックプロパゲーションアルゴリズムを実装した後、スクリプト
ex4.mは実装のグラディエントチェックを実行します。ザ
グラデーションチェックを使用すると、コードが
グラジエントを正しく計算します。

2.4勾配検査

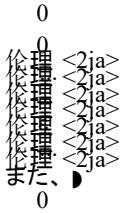
あなたのニューラルネットワークでは、コスト関数J(Θ)を最小にしています。に
あなたのパラメータのグラデーションチェックを実行すると、あなたは "アンロール"
パラメータ(1)、(2)長いベクトルθに。そうすることで、あなたは
コスト関数はJ(θ)であり、以下の勾配検査
手順。

あなたがうわさによれば計算 関数fがあると
あなたは 私が正しい微分値を出力しているFかどうかを確認したいと思います。 $\frac{\partial}{\partial\theta_{私}}J(\theta);$

(のi +)=θ+ みよう



そして、(I-)=θ -



そのi番目の要素だけインクリメントされた以外そこで、θは (私は+)、と同じです
εである。同様に 、(I-)は、番目の要素と対応するベクトルが減少します

εで表す。これで、数値ごとに、チェックすることにより、I(θ)の正しさfを確認することができます
私、それは:

$$F_{I(\theta)} \approx \frac{J(\theta(i\text{は}+)) - J(\theta(1-))}{2\varepsilon} \quad 。$$

これらの2つの値が互いに近似する程度は、
J.の詳細に依存しかし、あなたは、通常ことを見つけることができ、ε= 10 -4を 仮定
上記の左辺と右辺は、少なくとも4つの有意な
数字(そしてしばしばもっと多く)。
数値勾配を計算する関数を実装しました
あなたはcomputeNumericalGradient.mにいます。あなたは変更する必要はありませんが
このファイルを理解するには、コードを見てみることを強くお勧めします
使い方。
ex4.mの次のステップでは、指定された関数checkNNGradients.mを実行します。
小さなニューラルネットワークとデータセットが作成されます
あなたのグラデーションをチェックしてください。バックプロパゲーションの実装が正しい場合は、

あなたは1e-9より小さい相対的な差があるはずです。

実際的なヒント:グラディエントチェックを実行するときは、
比較的少数の小さなニューラルネットワークを使用することは効率的である
入力ユニットと隠れユニットの数が少なく、比較的少ない数
パラメータの θの各次元はコストの2つの評価を必要とする
これは高価になる可能性があります。機能チェックでは、
私たちのコードは小さなランダムなモデルとデータセットを作成します。
グラデーションチェックのためのcomputeNumericalGradient。さらに、
グラデーションの計算が正しいと確信している場合は、
学習アルゴリズムを実行する前にグラジェントチェックをオフにしてください。

実用的なヒント:グラディエントチェックは、あなたがいる場所のどの機能でも機能します
コストと勾配を計算する。具体的には、同じものを使用できます
computeNumericalGradient.m関数を使用して、
他の練習問題も正しいです(例:ロジスティック回帰
コスト関数)。

あなたのコスト関数が(非正規化された)
ニューラルネットワークのコスト関数、あなたはニューラルネットワークの勾配を提出する必要があります
(バックプロパゲーション)。

2.5正規化されたニューラルネットワーク

バックプロパゲーションアルゴリズムを正常に実装した後は、
グラデーションに正則化を追加します。正則化を説明するために、
これを追加用語として追加することができます
バックプロパゲーションを用いたグラジエント。
具体的には、Δを計算した後 (1)

正規化を追加する必要があります IJ あなたは、バックプロパゲーションを用いて、

$$\frac{\partial}{\partial\Theta_{ij}^{(1)}}J(\Theta) = D_{IJ}^{(1)} = \frac{1}{m}\Delta_{ij}^{(1)} \qquad j = 0\text{の場合}$$

$$\frac{\partial}{\partial\Theta_{ij}^{(1)}}J(\Theta) = D_{IJ}^{(1)} = \frac{1}{m}\Delta_{IJ}^{(1)} + \frac{\lambda}{m}\Theta_{ij}^{(1)} \qquad j\geq 1\text{の場合}$$

これはあなたが Θ(リットル) の最初の列を規則化すべきではないことに注意してください
バイアス項に使用されます。さらに、パラメータΘ $\Theta_{IJ}^{(1)}$、私はインデックス化され

1から始まり、jは0から始まってインデックス付けされる。したがって、

$$\Theta^{(L)} = \begin{bmatrix} \Theta_{1,0}^{(私)} & \Theta_{1,1}^{(1)} & \dots \\ \Theta_{2,0}^{(私)} & \Theta_{2,1}^{(1)} & \dots \end{bmatrix}$$

<2ja>

やや混乱して、Octave / MATLABのインデックスは1から始まります（for
iとjの両方）、したがってTheta1（2,1）は実際にΘ $\Theta_{2,0}^{(1)}$（すなわち、エントリ
2行目で、行列$\Theta^{(1)}$の最初の列 ）上に示し
今度は、nnCostFunctionのgradを計算するコードを修正してください
正則化のため。完了したら、ex4.mスクリプトは実行に移ります
実装のグラディエントチェック。コードが正しい場合は、
1e-9より小さい相対的な差があると予想します。
今すぐソリューションを提出する必要があります。

2.6 fmincgを使用した学習パラメータ

あなたが正常にニューラルネットワークのコスト関数を実装した後
およびグラジエント計算では、ex4.mスクリプトの次のステップでは、fmincg
良いセットパラメータを学ぶ。
トレーニングが完了すると、ex4.mスクリプトは
例のパーセンテージを計算することによって分類子の訓練の正確さ
それは正しいことを得た。実装が正しい場合は、レポートが表示されます
トレーニングの精度は約95.3％です（これは、
ランダム初期化）。より高い訓練精度を得ることが可能です。
より多くの反復のために神経ネットワークを訓練する。試してみることをおすすめします
より多くの反復のためにニューラルネットワークを訓練する（例えば、MaxIterを400に設定する）
また、正則化パラメータλを変化させる。適切な学習設定では、
ニューラルネットワークをトレーニングセットに完全に適合させることが可能である。

3隠れた層の可視化

あなたのニューラルネットワークが学んでいることを理解する1つの方法は、視覚化することです
どのような表現が隠されたユニットによって捕捉されたか。非公式には、
特定の隠された単位、それが計算するものを視覚化する1つの方法は、
それを起動させる（すなわち、起動値を持つ）入力x
$(a_i^{(1)})$
i）1に近いです）。あなたが訓練されたニューラルネットワークの場合、気付くそのi番目の行
$\Theta^{(1)}$は、i 番目のパラメータを表し、401次元のベクトルであります

隠しユニット。バイアス項を捨てると、400次元のベクトル
各入力ピクセルから隠れた単位への重みを表す。
したがって、隠された人によってキャプチャされた「表現」を視覚化する1つの方法
ユニットは、この400次元ベクトルを20×20画像に再形成することであり、
それを表示するexample.mの次のステップでは、displayData
機能とそれが図のように（あなたのイメージが表示されます 4 ）を25単位で、
それぞれがネットワーク内の1つの隠れユニットに対応しています。
あなたの訓練を受けたネットワークでは、隠れたユニットは、
ストロークやその他のパターンを探す検出器に
入力。

図4：隠しユニットの可視化

3.1オプションの（格付けのない）練習

この練習の部分では、さまざまな学習設定を試してみることになります
ニューラルネットワークがどのようにニューラルネットワークの性能を見て
は、正規化パラメータλおよびトレーニングステップの数（
fmincg使用時のMaxIterオプション）。
ニューラルネットワークは非常に複雑な形を形成することができる非常に強力なモデルです
決定境界。正則化がなければ、ニューラルネットは、
トレーニングセットを「オーバーフィット（overfit）」して、100％近い精度を得るようにします。
トレーニングは設定されていますが、それが見られていない新しい例はありません
前。正則化λを小さな値に設定し、MaxIter
より多くの反復回数にパラメータを設定して、自分自身で確認してください。

3 これが最高の活性化を与える入力を求めることと等価であることが判明します

隠れユニットのために、入力の「ノルム」制約（すなわち、$\|x\| \leq 1$）与えられました 。

ページ14

また、視覚化の変更を自分で確認することもできます
学習パラメータλとMaxIterを変更すると隠れユニットの数が増えます。

このオプション（格付けなし）のソリューションを提出する必要はありません。
運動。

提出と格付け

課題のさまざまな部分を完了したら、必ず提出する
お客様のソリューションを当社のサーバーに提出することができます。以下は、
この練習の各部分がどのように採点されたかの内訳。

部	提出されたファイル	ポイント
フィードフォワードおよびコスト関数	nnCostFunction.m	30ポイント
正規化コスト関数	nnCostFunction.m	15ポイント
シグモイド勾配	sigmoidGradient.m	5ポイント
ニューラルネット勾配関数（バックプロパゲーション）	nnCostFunction.m	40ポイント
規則的な勾配	nnCostFunction.m	10ポイント
合計ポイント		100ポイント

あなたはあなたのソリューションを複数回提出することができます。
最高のスコアだけを考慮に入れます。