# J4 情報システム実験実習 II　実験報告書

題目　　機械学習によるデータ解析

実施年月日　２０２５年１１月０７日

２０２５年１１月２１日

２０２５年１２月０５日

提出年月日　２０２５年　１２月１９日

共同実験者　０６班

宮本　嘉本　長尾　見山　アヌ

提出者

通し番号２１　学籍番号２２０５９　氏名　来間　空

１週目

1. 課題(1)

regression_analysis.py に R^2 と RMSE によるモデル評価をする関数を追加し，main 関数内で値を表示させるようにした．

また，TV，Radio，Newspaper の説明変数にそれぞれの相乗効果の説明変数を設計行列に追加し，実行した際に，説明変数を選択できるような機能を実行した．実行画面図 1 に示す．

```
表示する回帰係数を選択(スペース区切り)
1:TV, 2:Radio, 3:Newspaper, 4:TV * Radio, 5:TV
* Newspaper, 6:Radio * Newspaper
: 1 2 3 5

                    <省略>

モデル評価
R^2: 0.9034636281763918
RMSE: 1.6170217777658322
```

図 1　regression_analysis.py を改良したプログラムの実行画面

2. 考察(1)

まず TV，Radio，Newspaper の$R^2$の値を図 2 に示す．

| 設計行列 | R^2 |
|---|---|
| TV | 0.61187505 |
| Radio | 0.33203246 |
| Newspaper | 0.05212045 |

図 2　説明変数単体の$R^2$の値

図 2 より，Newspaper の$R^2$は他 2 つよりも著しく低い値を取っていることがわかる．決定係数$R^2$はモデルによって予測した値が実際の値をどの程度一致しているかを表す指標であり，$0 < R^2 < 1$の値域をとる．つまり，Newspaper は TV や Radio と比べて，広告費をあげても売上に影響しにくいということが考えられる．

以上を踏まえて，相乗効果の設計行列を追加し，各項を追加・削除した時の$R^2$の値を図 3 に示す．

| | 設計行列 | R^2 |
|---|---|---|
| ① | TV，Radio，Np | 0.89721064 |
| ② | TV，Radio，Np，TV*Radio | 0.96782224 |
| ③ | TV，Radio，Np，TV*Np | 0.90346363 |
| ④ | TV，Radio，Np，Radio*Np | 0.89745261 |
| ⑤ | TV，Radio，Np，TV*Radio，TV*Np，Radio*Np | 0.96863111 |
| ⑥ | TV，Radio，TV*Radio | 0.96779055 |

図 3　設計行列を追加削除した時の R^2 の値

図 3 より，①，②を比較すると，交互作用項 TV*Radio を追加したことによって②の$R^2$が 0.897 から 0.967 から大幅に向上した．一方で，①と③，④を比較すると，③，④での$R^2$の上昇はわずかであることがわかる．このことから，TV と Radio の広告費には強い相乗効果があることがわかり，Newspaper はやはり説明力のないノイズである．

また，⑤，⑥は②と同様に$R^2$が大幅な向上をしており，⑤と⑥を比較すると，値はほぼ同じであるが，説明変数の数が違うことがわかる．$R^2$は変数を増やすほど単調増加をするという性質を持つため，⑤の値が大きくなるのは自然であるが，説明力のない Newspaper の項が含まれていることを考慮すると，データの予測性能の低下を招く過学習のリスクの要因となる[1]．つまり，変数の数が少なく，$R^2$の値が高い⑥は適切なモデルであると考える．

1. 課題(2)

perceptron.py において学習データ x と教師データ t の正規化を行なった．未処理の正規化後における誤差と重みの結果を図 4 に示す．

| | | 誤差 | | | | | 重み | | |
|---|---|---|---|---|---|---|---|---|---|
| 処理方法 | 学習率 | 100回 | 200回 | 300回 | 400回 | 500回 | TV | Radio | Newspaper |
| 未処理 | 0.00001 | 69.8647 | 17.8556 | 6.4687 | 3.9752 | 3.4288 | 0.04996795 | 0.23425519 | -0.0035718 |
| 正規化 | 0.01 | 0.0232 | 0.004 | 0.0039 | 0.0038 | 0.0038 | 0.50236271 | 0.34841664 | -0.0111802 |

図 4　未処理と正規化の比較

2. 考察(2)

TV，Radio，Newspaper の入力データを比較すると，Newspaper と TV，Radio の数値が大きく乖離しているため，正規化をしなかった場合，学習率を非常に下げた状態でないと発散してしまう．そして，学習率が極小であると学習が進まず学習効率が悪くなってしまう[2]．perceptron.py を正規化したした場合，学習率を未処理の 100 倍の 0.01 に上げることができた．つまり，正規化をすれば学習効率が上がるということがわかる．図 4 より試行回数が同じ時，両者を比べても，誤差が大きく違うことがわかり，誤差は少なければ少ないほどいいので，正規化したプログラムが良い学習であると言える．また重みに注目すると，未処理では 3 つの重みに乖離があるが，正規化を行なったことにより，安定した重みをとっている．

このように，正規化をすれば，入力データにばらつきがあったとしても数値の安定し，学習効率を上げることができるため，正規化は必要である．

3 週目

1. 考察(3)

中間層と ReLU 関数を追加した時のモデルの変更前と変更後の正答率と誤差の比較を行う．中間層 100 と ReLU 関数を追加することにより，正答率の上昇と誤差の減少が期待できる．

まず，エポック数 10 回目のテストデータと学習データのそれぞれの正答率と誤差を図 5 に示す．

| | テストデータ | | 学習データ | |
|---|---|---|---|---|
| | 正答率[%] | 誤差 | 正答率[%] | 誤差 |
| 変更前 | 84.64 | 0.8007 | 83.49 | 0.8464 |
| 変更後 | 80.21 | 1.045 | 78.82 | 1.1187 |

図 5　エポック数 10 回目の正答率と誤差

次に，変更前と変更後の誤差の遷移を示したグラフを図 6 に示す．

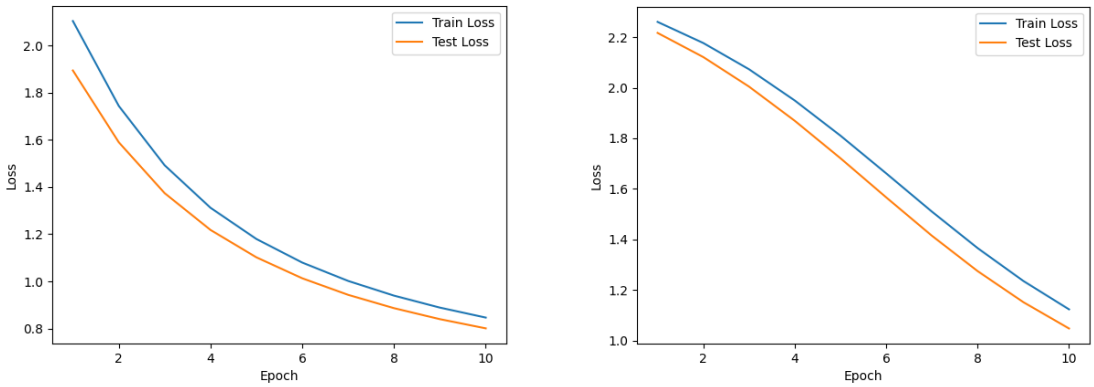


図 6　左：変更前の誤差遷移，右：変更後の誤差の遷移

図 5 より，通常では正答率は上昇し，誤差は減少しているはずである．しかし，実際には，正答率は減少，誤差は増加しており，反対の結果を示している．これは中間層を増やしたことにより，重みの数が増え，モデルが複雑になったことが関係している．実際に，変更前のパラメータ数は$784 \times 10 = 7840$個だったのに対し，変更後のパラメータ数は$(784 \times 100) + (100 \times 10) \approx 80000$個と 10 倍以上になっている．このことから，同じ 10 エポックという短い時間では，後者はモデルが複雑すぎることにより，学習途中で出力されたと考えられる．これは，図 6 の誤差の遷移からも確認できる．図 6 より，変更前においてエポック数 10 の時の誤差は収束に向かっていることが確認でき，学習が停滞しており，学習の終わりを示している．しかし，変更後は傾きが減少していないことから，誤差が収束しておらず学習途中であることが確認できる．また，期待した結果が出なかった要因として，学習率が 0.001 と変更がないため，変更前では適切な値であったが，変更後では学習進度が適切でないといことも挙げられる．

2.　考察(4)

変更後のモデルのエポック数を上げ正答率と誤差の値を比較する．

各エポック数におけるテストデータと学習データのそれぞれの正答率と誤差を図 7 に示す．

| | テストデータ | | 学習データ | |
|---|---|---|---|---|
| エポック数 | 正答率[%] | 誤差 | 正答率[%] | 誤差 |
| 10 | 80.21 | 1.045 | 78.82 | 1.1187 |
| 30 | 88.51 | 0.4657 | 87.68 | 0.4918 |
| 50 | 89.92 | 0.3704 | 89.34 | 0.3915 |
| 100 | 91.5 | 0.3024 | 90.91 | 0.3193 |
| 200 | 92.99 | 0.2506 | 92.72 | 0.2591 |

図 7　各エポック数における正答率と誤差

図 7 より，エポック数を 10 から 200 まで増やしたが，テストデータの正解率が下がり誤差が上昇してくような現象である過学習は確認できなかった．これは学習率が 0.001 と設定していたため，過学習が起きにくくなっていたと考える．学習率を 0.1 に上げ，エポック数 30 でもう一度実行する．エポック数 30 の時の正答率と誤差のグラフを図 8 に示す．

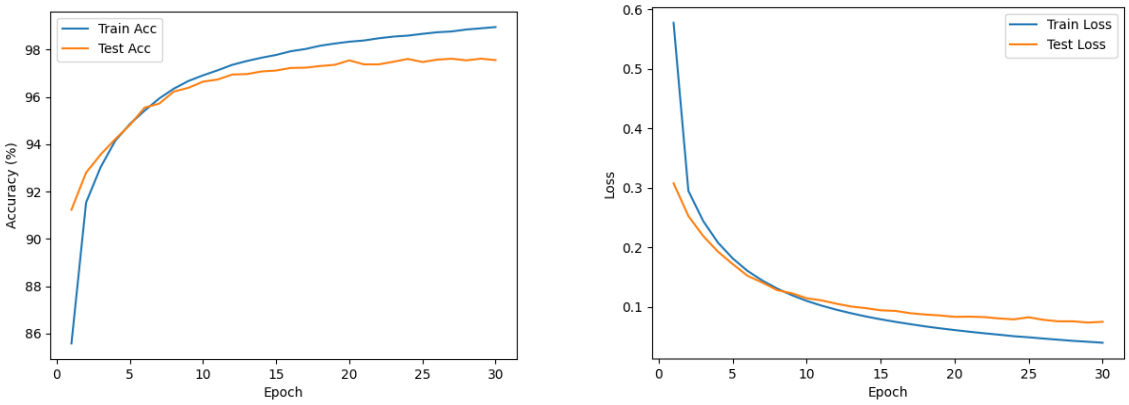


図 8　学習率=0.1，エポック数=30 の時の正答率と誤差

図 8 より，学習データと比べて，テストデータの正解率が下がり誤差が上昇していることが確認できる．これは，機械学習において汎化ギャップと呼ばれる一般的な挙動であると考える．その主な要因は，最適化のプロセスにある．モデルは学習時において，訓練データに対する誤差を最小化するようにパラメータを更新し続ける．その過程で，モデルは数字の一般的な形状だけでなく，訓練データに含まれる特有の筆跡の癖や微細なノイズまで過剰に学習してしまう傾向がある．

一方で，テストデータは学習に使用されていない未知のデータセットであるため，訓練データ特有のノイズに対する適合は機能せず，モデルが獲得した純粋な法則性のみがスコアに反映される．その結果，訓練データに対するスコアよりも，テストデータに対するスコアの方が必然的に低くなるということである．

## 3.　考察(5)

ニューラルネットワークで非線形な活性化関数を用いる理由を説明する．

ニューラルネットワークにおいて，1 つのニューロンが行っている計算は，出力を$y$，活性化関数を$h()$，バイアスを$b$，重みを$W$とすると以下の式で表される．

$$y = h(Wx + b)$$

活性化関数を通すことで，閾値以下のノイズを遮断したり（ReLU 関数など），出力値を一定の範囲に収めたり（Sigmoid 関数など）することで，次層へ伝える信号を制御し，計算の安定性や学習効率を高める役割を持つ．

また活性化関数の中でも，非線形の関数を使うことが必須である．1 層目の計算を$\mathrm{z} = \mathrm{W}_1\mathrm{x} + \mathrm{b}_1$，2 層目の計算を$\mathrm{y} = \mathrm{W}_2\mathrm{z} + \mathrm{b}_2$とする．ここで活性化関数を挟まずに，1 層目の出力をそのまま 2 層目に代入すると，以下のようになる．

$$\begin{aligned}\mathrm{y} &= \mathrm{W}_2(\mathrm{W}_1\mathrm{x} + \mathrm{b}_1) + \mathrm{b}_2 \\ &= \mathrm{W}_2\mathrm{W}_1\mathrm{x} + \mathrm{W}_2\mathrm{b}_1 + \mathrm{b}_2\end{aligned}$$

ここで，$W' = W_2W_1$，$b' = W_2b_1 + b_2$ とおくと，

$$y = W'x + b'$$

となる．この結果は、2 層のネットワークで行った計算が，数学的には 1 層の線形変換と全く等価であることを示している．つまり，活性化関数がない状態では，層をどれだけ深く重ねても表現力は向上せず，複雑な非線形問題を解くことができない．したがって，各層の間に ReLU などの非線形な活性化関数を挟むことで，層ごとの表現力の合成を防ぎ，ディープラーニング特有の複雑な境界線の学習を可能にしているのである．

感想

今回の実験は予想していた通り，他の実験よりも難しい実験であると感じた．特に 2 週目と 3 週目の内容のニューラルネットワークは思っていたよりも数学に近い概念で，理解に相当時間がかかった．授業内の説明では全く理解できなかったので，インターネットで別途調べたりもした．特に，YouTube の「3blue1broune」のニューラルネットワークの解説はとても面白く，理解の手助けとなった．インターンで強化学習をしたので，その知識も役に立った．総合的にみて，楽しく有益な実験であると感じた．

参考文献

[1] 「AIC（赤池情報量規準）ってなんだっけ？」, Qiita https://qiita.com/Ken_Sa/items/fcc33c7ec357896f6bd1, 2024 年 12 月 17 日更新.

[2] 「機械学習でなぜ正規化が必要なのか」, Qiita, https://qiita.com/yShig/items/dbeb98598abcc98e1a57, 2020 年 5 月 10 日更新.

[3] 「【活性化関数】01 活性化関数って？」, Zenn, https://zenn.dev/nekoallergy/articles/ml-basic-act-01, 2024 年 3 年 4 日公開.