慶應義塾大学学術情報リポジトリ
Keio Associated Repository of Academic resouces

Title	消費者情報処理理論のDeep Learningモデルへの適用と構造化
Sub Title	
Author	朝岡, 恵一郎(Asaoka, Keiichirō) 井上, 哲浩(Inoue, Akihiro)
Publisher	慶應義塾大学大学院経営管理研究科
Publication year	2018
Jtitle	
JaLC DOI	
Abstract	
Notes	修士学位論文. 2018年度経営学 第3397号
Genre	Thesis or Dissertation
URL	https://koara.lib.keio.ac.jp/xoonips/modules/xoonips/detail.php?koara_id=KO40003001-00002018-3397

慶應義塾大学学術情報リポジトリ(KOARA)に掲載されているコンテンツの著作権は、それぞれの著作者、学会または出版社/発行者に帰属し、その権利は著作権法によって保護されています。引用にあたっては、著作権法を遵守してご利用ください。

The copyrights of content available on the KeiO Associated Repository of Academic resources (KOARA) belong to the respective authors, academic societies, or publishers/issuers, and these rights are protected by the Japanese Copyright Act. When quoting the content, please follow the Japanese copyright act.

慶應義塾大学大学院経営管理研究科修士課程

学位論文(　　2018　年度)

論文題名

消費者情報処理理論のＤｅｅｐ　Ｌｅａｒｎｉｎｇモデルへの適用と構造化

主　査	井上　哲浩　教授
副　査	余田　拓郎　教授
副　査	林　高樹　教授

氏　名	朝岡　恵一郎

論 文 要 旨

所属ゼミ	井上哲浩 研究室	氏名	朝岡　恵一郎

（論文題名）

消費者情報処理理論のＤｅｅｐ　Ｌｅａｒｎｉｎｇモデルへの適用と構造化

（内容の要旨）

本研究の目的は Deep Learning モデルにマーケティングにおける消費者の情報処理理論を組み込むことで、従来ブラックボックスであり解析不能とされていた Deep Learning モデルからマーケティング的に価値のある情報を抽出することである。

Deep Learning モデルはマーケティング分野の領域で使用され始めて久しく、その柔軟性と幅広い適用能力により数々のモデルが開発、適用されてきた。しかし、その使用用途は従来の統計モデルや機械学習モデルより高精度な予測を行うなどその出力のみに注目された非常に限定された使われ方しかしていないのが現状である。

そこで、本研究では従来の Deep Learning モデルの既存の適用例から一歩踏み込んで Deep Learning モデルから価値のある情報を抽出する方法を考察した。

Deep Learning モデルの構造は大量の行列演算であり、その行列演算を数理的に解析することは不可能である。そこで本研究では Deep Learning モデル内に消費者情報処理理論を構造として組み込むことで、その構造から Deep Learning モデルを解釈しマーケティング的に価値のある情報の抽出を試みた。

本研究で Deep Learning モデル内へ組み込みを行う消費者情報処理理論は精緻化見込みモデルと補償型ヒューリスティクスである。

本研究ではまず、Deep Learning モデル内にどのように精緻化見込みモデルの構造を反映させるか論じ、モデルの構築を行う。

モデル構築後は精緻化見込みモデルが正しく組み込まれているかを検証し、Deep Learning モデル内に理論構造を組み込むことの有効性について論ずる。

最後に Deep Learning モデル内に補償型ヒューリスティクスの組み込みを行い、そこからヒューリスティクスに関する情報の抽出と解釈を試みる。

結果としては Deep Learning モデルに精緻化見込みモデルと補償型ヒューリスティクスの理論構造の反映とそれに紐づくマーケティング情報の抽出が可能であることがわかった。

本研究は既存の Deep Learning とマーケティングに関する研究と比較すると全く新しい方法で Deep Learning モデルへのマーケティング理論の構造的反映と、マーケティング的に価値のある情報を抽出しており、今後のマーケティングと Deep Learning の融合によって発展の期待されるマーケティングサイエンスの分野における一種の嚆矢となる研究であると考えている。

目次

1. はじめに 1
2. Deep Learning について 3
2.1 Deep Learning とは 3
2.2 Deep Learning の仕組み 4
2.3 全結合層(DENSE 層) 10
2.3.1 数値データの処理 10
2.4 LSTM 層 12
2.4.1 テキストデータの処理 14
2.5 畳み込み層(CONV 層)と Pooling 層 18
2.5.1 画像データの処理 21
3. 先行研究のレビュー 23
4. 精緻化見込みモデルと補償型ヒューリスティクス 27
4.1 精緻化見込みモデル 27
4.1.1 周辺的ルート 28
4.1.2 中心的ルート 28
4.2 補償型ヒューリスティクス 28
5. データとシステム環境の概要 30
5.1 データ収集について 30
5.2 本研究に用いる入力データ(説明変数)と出力データ(被説明変数) 31
5.3 データのグループ化 33
5.4 データの分割 35
5.4.1 周辺的ルートデータと中心的ルートデータ 35
5.4.2 ホールドアウトデータ 36
5.5 システム環境の概要 37
6. 精緻化見込みモデル構築と評価 38
6.1 モデルの基本構造 38
6.1.1 情報入力部 39
6.1.2 情報認知部 39
6.1.3 情報処理部 40
6.1.4 出力部 40
6.2 モデルの深さと幅 41
6.2.1 モデルの深さ 41
6.2.2 モデルの幅 41
6.3 周辺的ルートモデル 43
6.4 中心的ルートモデル 44
6.5 モデルに学習させる意味 45

6.6　モデルの評価方法と仮説 46
6.7　モデルの選定 48
6.8　モデルに関する仮説に対する評価 52
6.9　Bootstrap 法を用いた検証 54
6.9.1　Bootstrap 法とは 54
6.9.2　MSE に対する Bootstrap 法の適用 54
6.9.3　Bootstrap の結果 56
7.　補償型ヒューリスティクスモデル構築と評価 57
7.1　補償型ヒューリスティクスの表現 57
7.2　補償型のモデルパターン 58
7.3　モデルの評価 60
7.4　出力部の重みの考察 61
7.5　Bootstrap 法の適用 65
7.5.1　MSE に対する Bootstrap 法の適用 65
7.5.2　出力部の重みに対する Bootstrap 法の適用 67
8.　課題とまとめ 70
8.1 課題 70
8.2 まとめ 71
参考文献 72
付録資料 77
周辺的ルートモデルにおける$\boldsymbol{MSE}$の可視化 77
中心的ルートモデルにおける$\boldsymbol{MSE}$の可視化 78
$\boldsymbol{MSEcP}$、MSEcCにおける z 値 79
プログラム（一部抜粋） 87
Model Creation 87
Model Training 90
Bootstrap 91

1. はじめに

近年、Deep Learning が様々な分野で注目を集めており、マーケティングの分野においてもその傾向が顕著にみられる。マーケティング AI や、マーケティングオートメーションといったワードが市民権を得つつある現代において、Deep Learning はマーケティングを研究するうえで外すことのできない一つのツールとなってきており、Deep Learning は以前、マーケティングの分野において統計や機械学習が担ってきた役割を代替する存在として活用が進んでいる。

なぜ、Deep Learning が統計や機械学習といった強力なツールが存在する状況にもかかわらずここまで成長してきたのか、それには様々な理由が考えられるが、その中でも大きな理由は 2 つの柔軟性と精度にあると考えられる。

Deep Learning の第 1 の柔軟性とは統計モデルのようにデータに分布の存在を仮定する必要がなく、非線形で柔軟な分析が可能であるという特徴を指している。以前と比較して多様かつ大量の情報が使用される現代において統計モデルや機械学習と比較すると Deep Learning は現代の環境に非常に適した特徴を有しており、この特徴が成長の大きな要因の 1 つであると考えられる。

第 2 の柔軟性はデータの取り扱い方に表れている。統計や機械学習において数値以外のデータ、例えば画像やテキストデータを取り扱うためにはデータにラベルを設定するなど、任意の変換をしたうえで、数値データ化して使用する必要があった。この変換によってデータはその情報を 100%保持できなくなってしまう。例えば画像にラベル付けをし、それをダミー変数化して入力とする場合は、そのラベル情報以外を画像が含んでいないことを意味するようになってしまう。しかし、Deep Learning では画像は画像情報として、テキストはテキスト情報として、任意の加工によって情報が失われることなく使用することができ、そうした数値以外のデータの重要性が高まっている現代において Deep Learning のこの特徴は研究と成長を後押しした 1 つの動因となっている。

また、Deep Learning はその精度の面において、統計モデルや機械学習のモデルを上回る性能を有する研究結果が多数報告されている。情報の複雑化、多様化によってノイズにあふれている現代ではモデルの高精度化はどの分野においても 1 つの大きな課題であり、他のモデルより精度の面において勝る Deep Learning の研究が進み、成長したことは時代の自然な流れの 1 つであったと考えられる。

こうして発展を遂げ、活用の進む Deep Learning ではあるが、ブラックボックス構造を含むという大きな問題を内包している。ブラックボックス構造とはそのモデルの中身を解釈することができないことを意味しており、統計モデルなどを始めとする他のマーケティングモデルと Deep Learning を用いたモデルはこの点において大きく異なる。これはつまり、Deep Learning はその結果に至る原因を分析、解釈できないことを意味している。

これはマーケティングにおいて非常に大きな問題点である。例えば売上を予測する回帰分析によって導出された線形モデルは、線形構造を持つがゆえに売上に対して何の因子がどの程度影響するかを理解することができる。つまり、この回帰式のような統計モデルは原因と結果を結ぶ因果を解釈する余地を我々に与え、我々はその因果関係から売上向上などに向けた戦略構築を行うことができることを意味している。

しかし、Deep Learning ではこういった解釈を行うことができない。様々なデータを使用し、精度の高い結果を算出することができたとしても、因果関係を分析できない以上、売上に影響する要因が特定できず、よって我々は Deep Learning モデルを使用しても統計モデルのように因果関係を特定したうえで、売上向上に向けた戦略を構築することができない。

こうした特徴を Deep Learning モデルが有するが故に、Deep Learning を用いたマーケティングモデルは予測精度の向上など、限定された目的に対してのみ使用されているのが現状である。つまりマーケティングに用いられている Deep Learning モデルはマーケットの現象を解釈するというマーケティングの本質とかけ離れた立場にいることを甘んじて受け入れているというのが現状であるとも言える。

本研究はこの現状に一石を投じることを目的としている。Deep Learning モデルが回帰分析やコンジョイント分析を始めとする統計モデルと異なり、事象を解釈できないのは Deep Learning モデルが数理的に理解不可能であり、事象に対する明確な構造が存在しないからであるということができる。数理的に理解不可能という問題は Deep Learning の仕組みそのものが大量の行列演算であることに起因しており、またその仕組みであるから故に構造的にも理解が難しい。

そこで本研究では数理的に Deep Learning モデルを理解するのではなく、マーケティングの理論構造を活用して Deep Learning モデルから解釈を得る方法の研究を行った。この方法はマーケティングの理論を用いて Deep Learning モデル内に任意のマーケティング理論による構造を作り上げることでその構造からマーケティング的に価値のある情報を抽出する方法であり、概念的に説明するのであればマーケティング理論を Deep Learning モデルに対する解釈のツールとして使用するような方法である。

本研究で Deep Learning モデルへの反映を行うマーケティング理論は精緻化見込みモデルと補償型ヒューリスティクスの構造であり、本研究ではまずこれらの理論が Deep Learning モデルに反映できたかを検証し、その後、それらの理論を用いて Deep Learning モデルの解釈を行い、Deep Learning モデルからマーケティング的に価値のある情報が抽出できるかどうかの考察を行った。

2. Deep Learning について

2.1 Deep Learning とは

Deep Learning は人間の脳を模した Neural Network が多層構造化したものを指し、1940 年代から研究が開始された(Hebb 1949、McCulloch and Pitts 1943、Rosenblat 1958)。当初、Neural Network はそれほど画期的かつ効率的な機械学習モデルではなかったが、後述する誤差逆伝播の発見や計算機の発展により、その研究はこれまで何回ものブームを経て今日に至っている。

その中でも大きなブームが 2 つ存在する。第 1 のブームは誤差逆伝播(Rumelhart and Mcclelland 1986)によって起こった。誤差逆伝播とは Neural Network の学習則に勾配法を適用した方法であり、この学習アルゴリズムの登場で研究は大きな広がりを見せたが、90 年代後半にはその広がりも終焉を迎えることとなった。誤差逆伝播によって大きく Deep Learning の研究に進展はあったものの、Deep Learning では次第に汎化性能の向上が見られなくなり、さらには過学習や勾配消失問題といった問題を解決する手法が当時は存在しておらず第 1 のブームは終焉を迎えた。現在では誤差逆伝播における過学習も活性化関数の工夫により解決されており、本研究ではその活性化関数を用いた誤差逆伝播によるモデルを構築した。

第 2 のブームは Deep Belief Network(Hinton et al. 2006)によって引き起こされた。Deep Belief Network は Deep Learning の各層を制約ボルツマンマシンと呼ばれる単層ネットワークに分解しながら学習を行う方法で、一般的な順伝播の Neural Network に転換することが可能でかつ学習も過学習に陥ることなく行えることが示された。さらに自己符号化(Bengio et al. 2006)によっても Deep Learning の学習が可能であることが解明され今日のブームにつながっている。

こうしたブームの原因には計算機の発展が寄与していることも欠かすことのできない背景として存在する。近年は CPU、GPU の発展により Deep Learning の計算速度も従来とは比較にならないほどの速度で発展してきており、またプログラミング言語の発展や Deep Learning の学習に欠かすことのできないデータの取得も容易になっている為、Deep Learning は研究機関のみならず民間企業や学術機関でも盛んに研究されるようになっている。

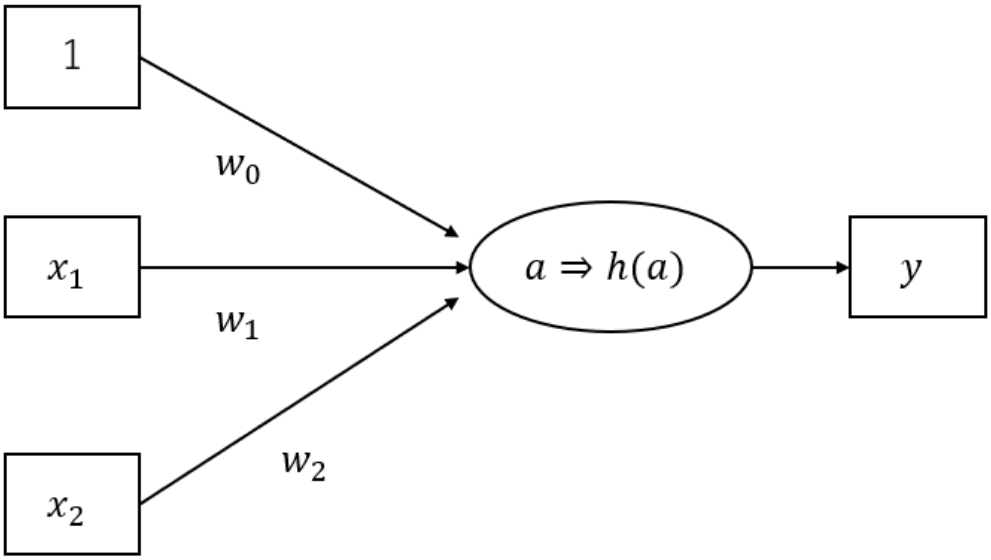
また、近年は Deep Learning の商業的利用も進んでおり、画像認識や音声認識に限定されることなく、機械の自動制御やロボットの開発など幅広い分野での使用が進んでいる。

2.2　Deep Learning の仕組み

Deep Learning は教師あり学習、教師なし学習の双方に適用することができるが、本研究で用いる Deep Learning モデルは教師あり学習である為、ここでは教師あり学習を行う Deep Learning の仕組みについて解説する。

Deep Learning の基礎となる Neural Network の構造は図表 2.1 のような構造となっている。

図表 2.1　Neural Network の構造



出典：伊藤（2018）を基に筆者作成。

図表 2.1 の四角はデータ、丸は各層におけるニューロンを示す。ここでは入力変数の数が 2 変数であるモデル例を示している。まず、入力（ $1, x_1, x_2$）はそれぞれに重み（w_0, w_1, w_2）に積和されて次の層へと伝わる。入力における 1 はバイアス項である。積和された値は a であり $a = w_0 + x_1 w_1 + x_2 w_2$ である。ニューロン内では積和された値を活性化関数と呼ばれる関数によって変換し、それを次の層へと伝えていく。

図表 2.1 は 1 層構造の Neural Network であり、活性化関数によって変換された値がこのモデルの出力 $y = h(a)$ である。

活性化関数は脳におけるニューロンが電気信号を次のニューロンへ伝える際の発火を模したものである。本研究で用いる Deep Learning モデルでは中間層の活性化関数に ReLU(Rectified Linear Unit)関数を用いる。ReLU 関数は以下のように定義される関数である。

$$\boldsymbol{f = max(0, x)} \tag{2.1}$$

ReLU 関数は微分すると以下のようになり、xがどんなに大きな値になっても 1 で返すので中間層にこの関数を用いることで学習時の勾配消失問題を回避することができ、tanh などの関数と比較してモデルが改善することが知られている(Glorot et al. 2011)。

$$f' = \begin{cases} 1 & (x > 0) \\ 0 & (x \leq 0) \end{cases} \tag{2.2}$$

Neural Network の学習は出力 y と教師信号 t との間の差を最小にすることを目的に行われる。この際に用いられる誤差逆伝播と呼ばれる学習アルゴリズムの基となっている考え方が勾配法(最急下降法)と呼ばれるものである。

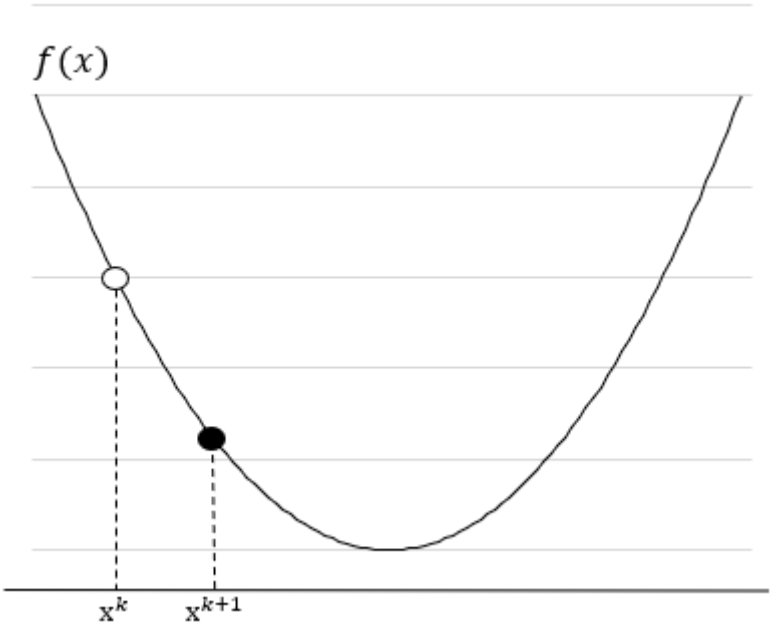
勾配法は関数の最小値または最大値を求める最適化問題を解くための手法として機械学習の分野においても広く活用されており、以下に説明する勾配法によって Neural Network の最適化を行うことを誤差逆伝播と呼ぶ。

今、入力を $\mathbf{x} = [x_1, x_2, \dots, x_n]$ とし出力を $\mathbf{y} = [y_1, y_2, \dots, y_n]$ とする。勾配法の試行回数を k で示すと勾配法による x の求め方は以下のようになる。なお、数式における正対の英数字はベクトルを表している。ηは係数学習率と呼ばれる任意の係数であり、勾配法による x の更新の度合いを調節する役割を担っている。

$$\mathbf{x}^{\mathbf{k+1}} = \mathbf{x}^{\boldsymbol{k}} - \eta \frac{\partial f(\mathbf{x}^{\boldsymbol{k}})}{\partial \mathbf{x}^{\boldsymbol{k}}} = \mathbf{x}^{\boldsymbol{k}} - \eta \begin{bmatrix} \frac{\partial f}{\partial x_1^k} \\ \frac{\partial f}{\partial x_2^k} \\ \vdots \\ \frac{\partial f}{\partial x_n^k} \end{bmatrix} \tag{2.3}$$

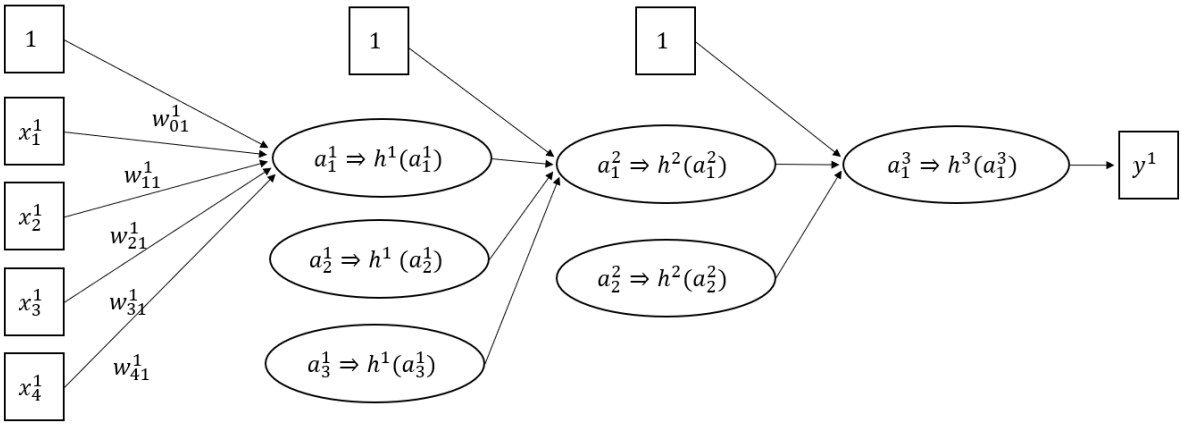
このアルゴリズムは $f(x)$ を下に凸の二次関数を想像すると理解しやすい。図表 2.2 のように関数 $f(x)$ の $\mathbf{x}^{\mathrm{k}}$ における傾きは負であり、勾配法によって更新される $\mathbf{x}^{\mathrm{k+1}}$ は関数の値が最小になるように更新される。このように勾配法は対象となる関数の微分式を用いて、最適解を導き出すアルゴリズムの 1 つとして知られている。

図表 2.2　下に凸の二次関数



この勾配法のアルゴリズムを使用して Deep Learning モデルでも学習が行われている。本研究で用いる Deep Learning モデルはこの Neural Network を多層構造にしたモデルであり図表 2.3 のような構造を持つ。

図表 2.3　多層構造をもつ Neural Network



出典：伊藤（2018）を基に筆者作成。

このモデルは 3 層の構造を持ち。第 1 層目のニューロン数は 3、第 2 層目のニューロン数は 2、第 3 層目のニューロン数は 1 のモデルである。入力が伝播し出力を形成する仕組みは先述の Neural Network と同様である。

図表 2.3 における中間出力 a の下付き文字は中間出力の番号を示し、上付き文字は中間層の番号を示す。例えば a_1^2 は 2 層目における 1 番目の中間出力である。 h は中間層における活性化関数を示しており、上付き文字は中間出力と同様に中間層の番号を示す。例えば h^3 は第 3 層目の活性化関数を示す。

入力は X であり X は $\mathrm{X} = [\mathbf{x^1}, \mathbf{x^2}, \dots, \mathbf{x^N}]$ とする。ここで $\mathbf{x^1} = [x_1^1, x_2^1, x_3^1, x_4^1]$ であり、X は全データ数 N 個、入力変数の数が 4 個のデータセットである。 w はそれぞれの入力または中間出力に掛かる重み係数であり下付き文字の左側は入力または中間出力の番号に対応し、右側は出力先の中間層への番号に対応し、上付き文字は重み係数の存在する中間層の番号を示す。出力 y の上付き文字は入力の番号に対応する。すなわち入力 $\mathbf{x^1}$ の出力は y^1 となる。

まず入力は先の例と同様にそれぞれに重みが積和され中間層 1 に伝播し、中間層に設定された活性化関数によって値が変換される。

$$\begin{bmatrix} 1 \\ x_1^1 \\ x_2^1 \\ x_3^1 \\ x_4^1 \end{bmatrix}^T \begin{bmatrix} w_{01}^1 & \cdots & w_{03}^1 \\ \vdots & \ddots & \vdots \\ w_{41}^1 & \cdots & w_{43}^1 \end{bmatrix} = \begin{bmatrix} a_1^1 \\ a_2^1 \\ a_3^1 \end{bmatrix} \Rightarrow \begin{bmatrix} h^1(a_1^1) \\ h^1(a_2^1) \\ h^1(a_3^1) \end{bmatrix} \tag{2.4}$$

次にその変換された値それぞれに重みが積和され中間層 2 へ伝播され中間層 2 においても同様に活性化関数によって値が変換され（式(2.5)）、同様のことが中間層 3 によっても行われる（式(2.6)）。

$$\begin{bmatrix} 1 \\ h^1(a_1^1) \\ h^1(a_2^1) \\ h^1(a_3^1) \end{bmatrix}^T \begin{bmatrix} w_{01}^2 & w_{01}^2 \\ w_{11}^2 & w_{12}^2 \\ w_{21}^2 & w_{22}^2 \\ w_{31}^2 & w_{31}^2 \end{bmatrix} = \begin{bmatrix} a_1^2 \\ a_2^2 \end{bmatrix} \Rightarrow \begin{bmatrix} h^2(a_1^2) \\ h^2(a_2^2) \end{bmatrix} \tag{2.5}$$

$$\begin{bmatrix} 1 \\ h^2(a_1^2) \\ h^2(a_2^2) \end{bmatrix}^T \begin{bmatrix} w_{01}^3 \\ w_{11}^3 \\ w_{21}^3 \\ w_{31}^3 \end{bmatrix} = [a_1^3] \Rightarrow [h^3(a_1^3)] \tag{2.6}$$

この場合出力は $y = h^3(a_1^3)$ であり、二乗誤差を誤差関数に設定すると誤差関数は以下のように定義される。

$$E = \frac{1}{2}\sum_{i=1}^{N}(y^1 - t^1)^2 \tag{2.7}$$

t は入力に対する教師信号であり入力 $\mathbf{x^1}$ の正解値は t^1 となる。

勾配法をこの誤差関数に適用していくと誤差関数から第 3 層への伝播は以下のようになる。

$$\begin{bmatrix} \frac{\partial E}{\partial w_{01}^3} \\ \frac{\partial E}{\partial w_{11}^3} \\ \frac{\partial E}{\partial w_{21}^3} \end{bmatrix} = \begin{bmatrix} 1 \\ h^2(a_1^2) \\ h^2(a_2^2) \end{bmatrix} [[h'^3(a_1^3)] \cdot [y^1 - t^1]^T \tag{2.8}$$

ここで $\delta^3 = [h'^3(a_1^3)] \cdot [y_1 - t_1]$ とすると、第 2 層への伝播は

$$\begin{bmatrix} \frac{\partial E}{\partial w_{01}^2} & \frac{\partial E}{\partial w_{11}^2} \\ \frac{\partial E}{\partial w_{11}^2} & \frac{\partial E}{\partial w_{12}^2} \\ \frac{\partial E}{\partial w_{21}^2} & \frac{\partial E}{\partial w_{22}^2} \\ \frac{\partial E}{\partial w_{31}^2} & \frac{\partial E}{\partial w_{32}^2} \end{bmatrix} = \begin{bmatrix} 1 \\ h^1(a_1^1) \\ h^1(a_2^1) \\ h^1(a_3^1) \end{bmatrix} \left[\begin{bmatrix} h'^2(a_1^2) \\ h'^2(a_2^2) \end{bmatrix} \cdot \begin{bmatrix} w_{11}^3 & w_{12}^3 \\ w_{21}^3 & w_{22}^3 \end{bmatrix} \delta^3 \right]^T \tag{2.9}$$

となり、$\delta^2 = \begin{bmatrix} h'^2(a_1^2) \\ h'^2(a_2^2) \end{bmatrix} \cdot \begin{bmatrix} w_{11}^3 & w_{12}^3 \\ w_{21}^3 & w_{22}^3 \end{bmatrix} \delta^3$ とすると、第 1 層への伝播は

$$\begin{bmatrix} \frac{\partial E}{\partial w_{01}^1} & \cdots & \frac{\partial E}{\partial w_{03}^1} \\ \vdots & \ddots & \vdots \\ \frac{\partial E}{\partial w_{41}^1} & \cdots & \frac{\partial E}{\partial w_{42}^1} \end{bmatrix} = \begin{bmatrix} 1 \\ x_1^1 \\ x_2^1 \\ x_3^1 \\ x_4^1 \end{bmatrix} \left[\begin{bmatrix} h'^1(a_1^1) \\ h'^1(a_2^1) \\ h'^1(a_3^1) \end{bmatrix} \cdot \left[\begin{bmatrix} w_{11}^2 & w_{12}^2 \\ w_{21}^2 & w_{22}^2 \\ w_{31}^2 & w_{32}^2 \end{bmatrix} \delta^2 \right] \right]^T \tag{2.10}$$

となる。このように始めの誤差 $[y^1 - t^1]$ が一つ層が戻る度に、重みと活性化関数の微分の値が付加されて逆へと伝播することで誤差関数を最小化するように Deep Learning は各層の重みを最適化していく。

これが Deep Learning が学習する仕組みであり、この学習が繰り返されることで Deep Learning は自身のパラメーターを最適化していく。

本研究で使用する Deep Learning モデルも上記と同様の学習アルゴリズムを使用してモデルの学習を行う。

また上記の数式で使用した T の上付き文字は転置行列を示し、・はアダマール積を示している。

上記の例で使用した中間層の種類は後述の全結合層と呼ばれる層であり、ほかにも様々な層が存在している。本研究では全結合層に加え、LSTM 層と畳み込み層を使用するので、次にそれらの層の詳細について記していく。

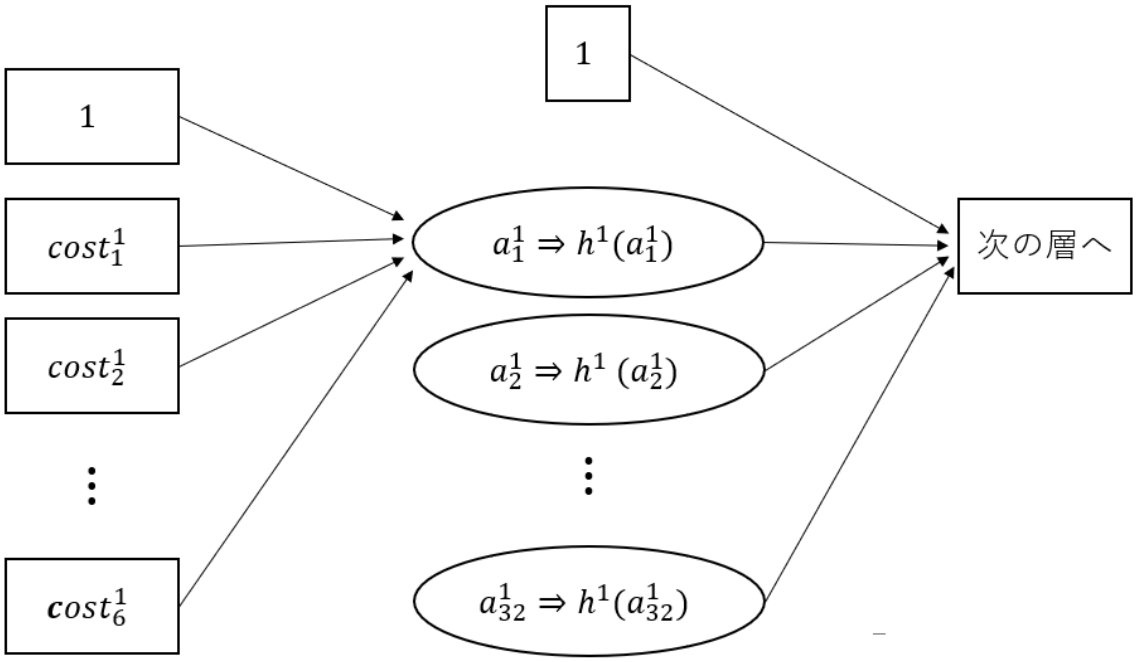
2.3　全結合層(DENSE 層)

全結合層は先の例で示したように全入力に重みを付加し積和を行う層を指す(式(2.4))。最もクラシカルな層であり主に数値データが入力となる場合に使用され、本研究においても数値データが入力に当たる際に使用している。

2.3.1　数値データの処理

本研究に用いるモデルの数値データ処理の例を以下に示す。データについての詳細は 5 章にて後述するが、本研究では車に関する数値データを使用する。具体的には価格や自動車の取得税、馬力などが該当する。図表 2.4 はコストに関する情報を全結合層によって処理している例である。

図表 2.4 コスト情報を全結合層で処理する概念図



出典：伊藤（2018）を基に筆者作成。

$cost$ はコストに関する数値データを示しており、先の例と同様に下付き文字は情報の種類の順番（価格、重量税など）を示しており、上付き文字は該当する車の番号を示している。

図表 2.4 のように本研究のモデルでは中間層に 32 個のニューロンを想定している。第 1 層における処理を数式化すると以下のようになる。

$$\begin{bmatrix} 1 \\ cost_1^1 \\ cost_2^1 \\ \vdots \\ cost_6^1 \end{bmatrix}^T \begin{bmatrix} w_{01}^1 & \cdots & w_{0(16)}^1 \\ \vdots & \ddots & \vdots \\ w_{61}^1 & \cdots & w_{6(32)}^1 \end{bmatrix} = \begin{bmatrix} a_1^1 \\ a_2^1 \\ \vdots \\ a_{32}^1 \end{bmatrix} \Rightarrow \begin{bmatrix} h^1(a_1^1) \\ h^1(a_2^1) \\ \vdots \\ h^1(a_{32}^1) \end{bmatrix} \tag{2.11}$$

第 5 章で詳述するが、本研究で数値処理の対象となる変数はグループ化された $maker, cost, power, size, usage$ の 5 つの変数郡が対象となる。それぞれは 3 層によって処理され、第 1 層のニューロン数は 32、第 2 層のニューロン数は 16、第 3 層のニューロン数は 16 である。

2.4　LSTM 層

LSTM 層は時系列データなどデータの並びに意味のあるデータに対して適用される層である（Hochreiter and Schmidhuber 1997)。再起型ニューラルネットワーク（リカレントニューラルネットワーク）と呼ばれるものが原型となっておりテキストデータや時系列データの分析においての成果が多数報告されている（Mikolov and Kombrink 2010)。

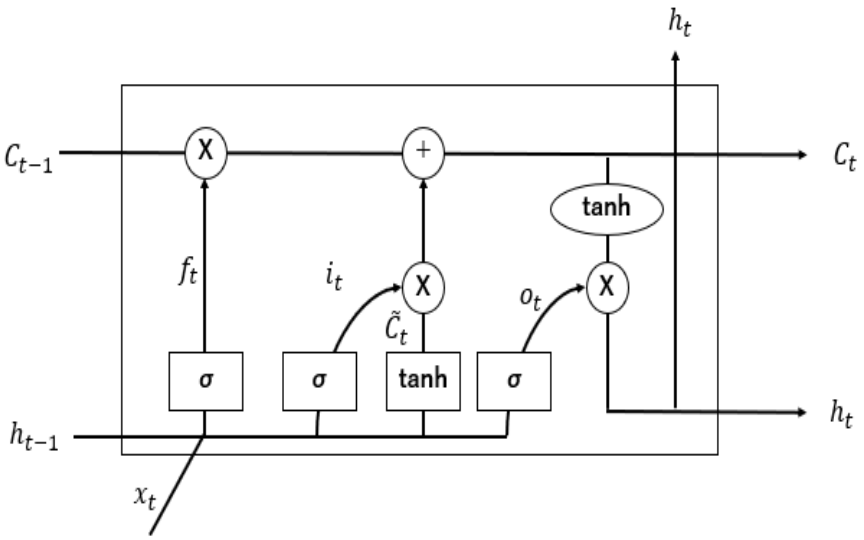
リカレントニューラルネットワークには長期依存性の問題と呼ばれる問題が存在する。この問題は短い文章の場合は単語の並びを Deep Learning モデルに理解させることは可能であるが、文章が長くなると単語間の結びつきの関係を理解できなくなる問題を指す。

この問題を解決するために開発されたものが LSTM 層となる。LSTM 層は人間の記憶に似た構造を層に持たせることで、離れた単語同士であっても記憶をたどることで相互間の関係を理解できるように設計されたものである。

LSTM 層は図表 2.5 のようセルと呼ばれるものが多数連なった構造を取っており、これらの相互作用により、単語間の記憶関係を模擬的に表現した層である。

LSTM 層は特別な方法で相互作用する 3 つのゲートと呼ばれるユニットをセルの中に内包しており、それらはそれぞれ忘却ゲート、入力ゲート、出力ゲートと呼ばれている。図表 2.5 は LSTM 層の概念図を示している。

図表 2.5　LSTM 層の概念図



出典：巣籠（2018）を基に筆者作成。

図表 2.5 は 1 つのセルを示しており、h_t は時刻 t におけるセルの値、x_t は入力、C_t は LSTM 層の状態 σ はシグモイド関数を、tanh は tanh 関数を示す。

まず、LSTM 層のセルは入力 x_t と同時に前のセルからの入力 h_{t-1} を受け取る。これらの入力はシグモイド層によって f_t へと変換される。

このシグモイド層は「忘却ゲート」と呼ばれ、前セルの状態 C_{t-1} と掛け合わせることで、前セルの状態をどの程度保持するかを調節する役割を担う。

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{2.12}$$

次に行われる処理は「入力ゲート」にて行われる。このシグモイド層は、どの値を更新するかを判定し、tanh 層でセル状態に加えられる新たなベクトルの候補値 $\widetilde{C_t}$ を作成する。

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2.13}$$

$$\widetilde{C_t} = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{2.14}$$

これらの計算結果を用いてセルの状態を C_{t-1} から C_t へと更新する。

$$C_t = f_t * C_{t-1} + i_t * \widetilde{C_t} \tag{2.15}$$

概念的に説明するならば f_t は前のセルの状態をどれくらい記憶しておくかの程度を表し、i_t は入力から得た新しい状態の候補値をどの程度セル状態に反映させるのかを調整していると考えられる。

最後は出力の判定を行う。この出力はセル状態に基づいて行われる。まず、シグモイド層によってセル状態のどの部分を出力するかを判定する。その後、出力の判定された部分のみを出力する為にセル状態にtanh関数を適用し、それにシグモイド層の出力を掛ける。

$$\mathrm{o_t} = \sigma(W_o[h_{t-1}, x_t] + b_o) \tag{2.16}$$

$$h_t = o_t * \tanh(C_t) \tag{2.17}$$

このように前のセル状態から現在の出力を行う工程を繰り返すことで、記憶と同様に、前セルの状態を反映することを LSTM 層は可能としている。

これらは主にテキスト分析において多くの業績を残しており、本研究で用いたモデルに関してもテキスト分析を行う部分に採用している。

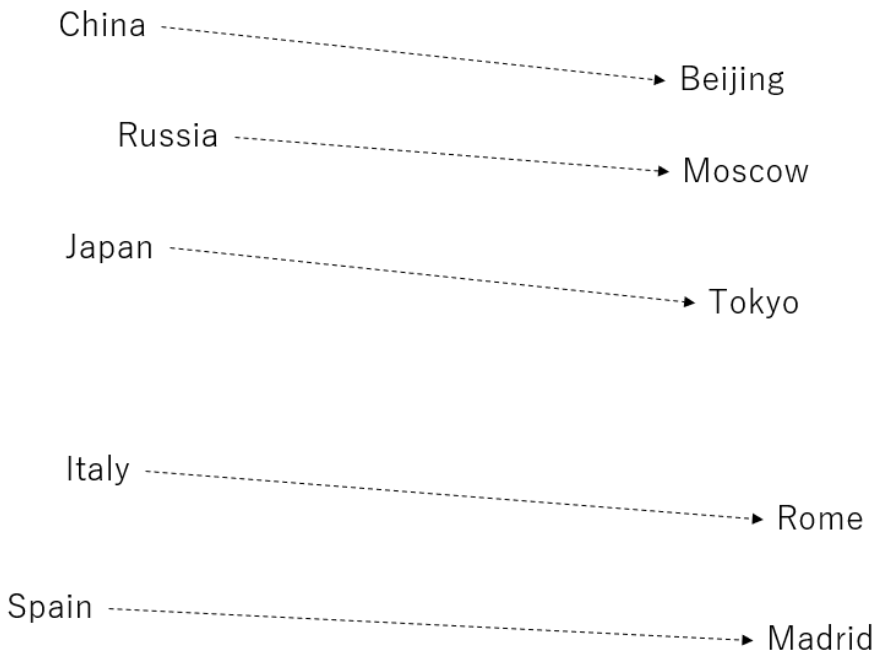
2.4.1　テキストデータの処理

本研究では LSTM 層にテキストデータを適用する為、Word2Vector (Mikolov et al. 2013, Goldberg and Levy2014)を用いて単語のベクトル空間を作り、それを LSTM 層に対応させることでテキストデータを意味のある数値の羅列に置き換えてから分析を行った。

この単語のベクトル空間は自然言語処理の分野で使用される分析手法であり単語間に意味のある関係をベクトルによって表現することを可能にする。

Word2Vector によって表現される単語空間は図表 2.6 のようなベクトル空間である。この点線で示されたベクトルは国名と首都との関係性を示しており、通常は 100～200 次元のベクトルによる表現となる。こうした空間を用いることで単語を意味のある数値データとして扱うことが可能となり、One-Hot-Coding とは違いテキスト情報の意味そのものを数値として Deep Learning モデルの入力として扱うことを可能とする。

図表 2.6　単語のベクトル空間



出典：Mikolov, Sutskever, Chen, Corrado, and Dean (2013) を基に筆者作成。

One-Hot-Coding などではこうした表現は不可能であり、自然言語処理の分野における文章分類の研究などで多数の成果が報告されている。

Word2Vector における 100～200 次元からなる単語空間とは Neural Network の層の重み空間を指す。この Neural Network は Skip-Gram と呼ばれる 2 層のニューラルネットワークであり、隠れ層は 1 つである。この Skip-Gram はある単語を入力した際、その周辺にどのような単語が表れやすいかを予測することをモデル化したものである。つまり、入力は特定の単語、出力は周辺単語の出現確率となる。

入力は One-Hot-Coding で示され、重み行列は単語数と任意の次元数で設定される。本研究で使用する Word2Vector の空間に関しては任意の行列数は 200 次元に設定しており、これはつまり 1 つの単語を 200 次元の空間のベクトルに置き換えて表現していることと同義である。

入力は図表 2.7 のように学習される。ここでは重み行列の各行が単語に対する意味ベクトルを表現していることが分かる。

例えば入力 $[0\ 1\ 0\ \cdots 0]$ が「マーケティング」という単語に対応している場合、図表 2.7 において「マーケティング」という単語の持つ意味は $[23\ \cdots 7]$ といった 200 次元の数値に変換される。

図表 2.7　Word2Vector による単語 X の表現

入力単語X　　単語Xのベクトル（1×200）

$$[0\quad 1\quad 0\quad \cdots\quad 0] * \begin{bmatrix} 15 & \dots & 264 \\ 23 & \dots & 7 \\ 2 & \cdots & 234 \\ 45 & \cdots & 56 \\ 98 & \cdots & 795 \end{bmatrix} = [23\quad \dots\quad 7]$$

重み行列（単語数×200）

出典：Goldberg and Levy (2014) を基に筆者作成。

このベクトル空間をテキストデータに反映させるため、本研究ではテキストデータを形態素解析した後に、対象の単語に該当する Word2Vector 空間をテキストデータに反映させ、この空間上の数値（200 次元のベクトル）でテキストデータを表現した。

この空間は各行がそれぞれ一つの単語を示しており、テキストデータが入力された際にテキストデータに含まれる単語がこの行の何行目に当たるのか参照されて数値へと変換される。LSTM 層ではこの仕組みを埋め込み行列と呼ぶ。

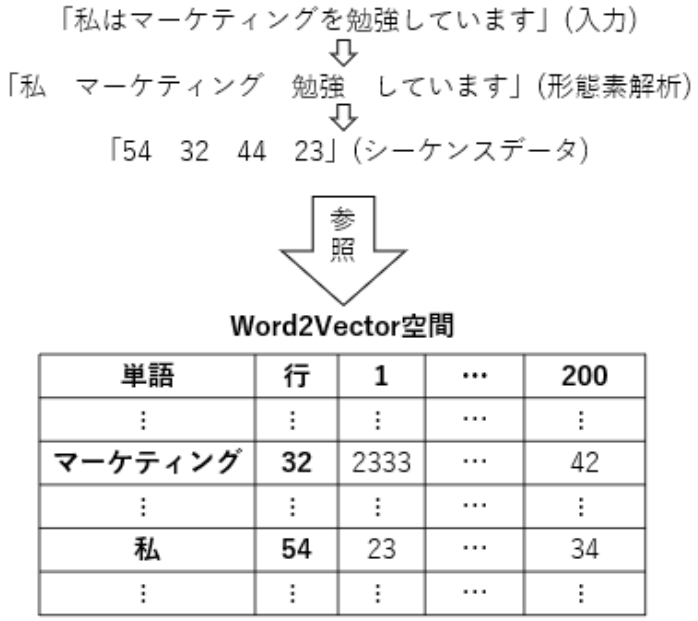
本研究で使用した Word2Vector の空間は日本語 Wikipedia の文章をもとに学習して造られたものであり、東北大学の乾教授らが作成、公開しているものを使用した。

テキストデータの入力はシーケンスデータに変換されて行われる。シーケンスデータとは入力されたテキストデータに含まれる単語が埋め込み行列の何行目に対応しているのかを示す数値データである。

例えば「私はマーケティングを勉強しています」というテキストデータが入力データである場合、まずこのテキストデータは形態素解析により単語ごとに分割され、「私、マーケティング、勉強、しています」といった形へ変換される。

次にこの分割された文章に含まれる各単語に Word2Vector の単語空間上の何行目のベクトルが対応しているのかを示すデータへと変換される。図表 2.8 の場合、「私」という単語は 54 行目、「マーケティング」という単語は Word2Vector を反映した埋め込み空間における 32 行目のベクトルであることを示しており、このように単語を埋め込み行列の行番号に変換したデータをシーケンスデータと呼ぶ。

図表 2.8　テキストデータの処理過程

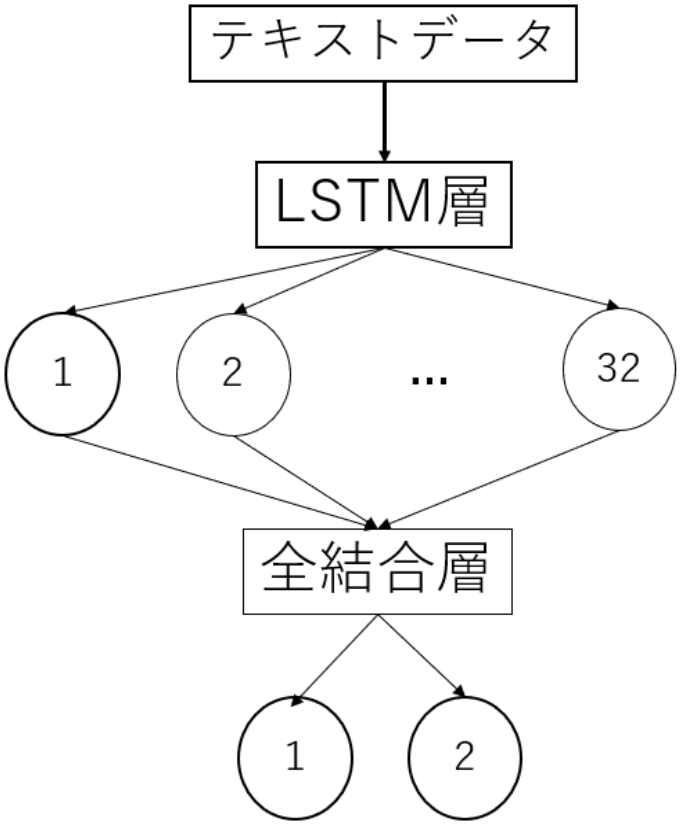


このようにシーケンスデータの数値を基に入力であるテキストデータは Word2Vector 空間の数値へと変換され、LSTM 層への入力として扱われることになる。この時の入力を $\mathbf{x}$ とすると、$\mathbf{x} = [x_1, x_2, x_3, x_4]$ と記すことができ、x_1 は「私」に対応する数値データ（[23, …,34] ）であり、x_2, x_3, x_4 も同様に「マーケティング」、「勉強」、「しています」に対応する 200 次元の数値に変換される。

本研究では以上のようにシーケンスデータに処理されたテキストデータを Word2Vector による埋め込み行列を有する LSTM 層に入力することでテキストデータの処理を行っている。

LSTM 層によって処理されたテキストデータはその出力を全結合層の入力とした処理がなされる。本研究で使用する LSTM 層の出力数は 32 個であり、全結合層によってこの出力は 2 個の出力へと変換される(図表 2.9)。

図表 2.9　本研究におけるテキストデータ処理の流れ

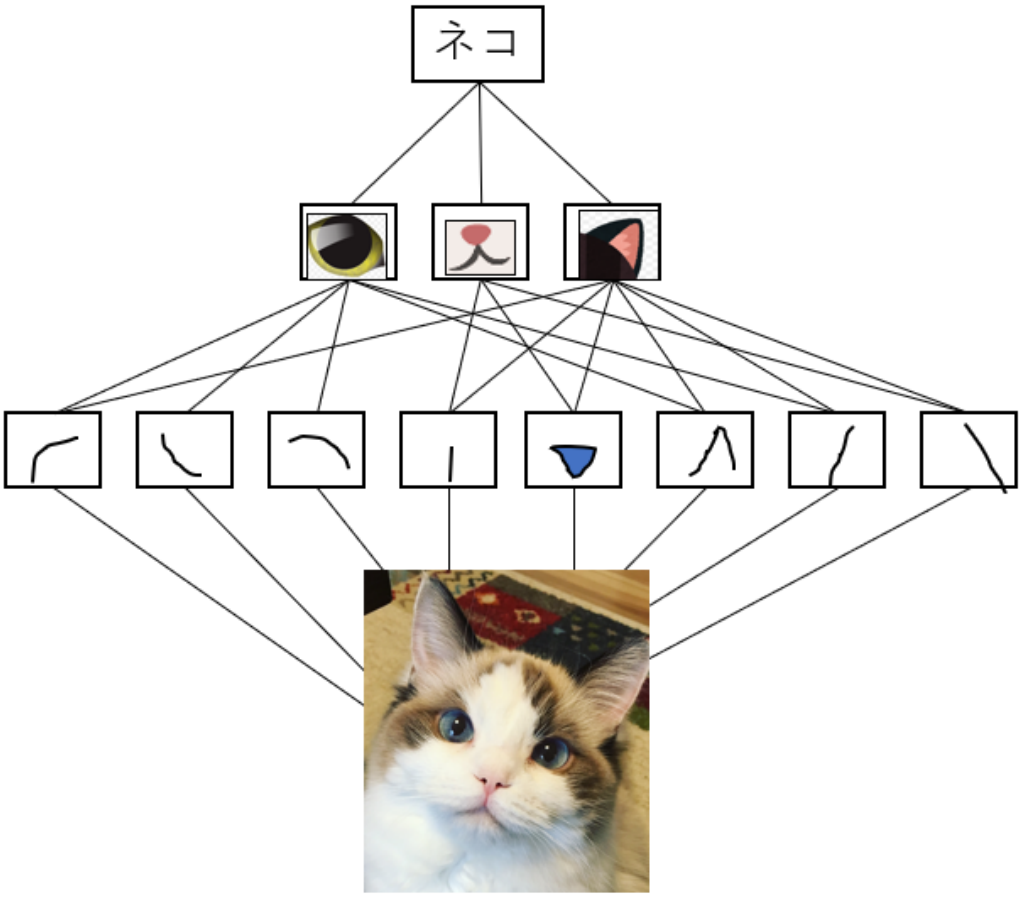


2.5　畳み込み層(CONV 層)と Pooling 層

畳み込み層とは画像認識などに主に用いられる層であり、局所的なパターンを学習することができる。畳み込み層を概念的に示したものが図表 2.10 となる。図表 2.10 はネコの画像を認識する過程を概念的に示したものである。

畳み込み層にはフィルターといわれるユニットが存在しており、このフィルターを画像に適用することで画像の特徴を抽出する(図表 2.10 のフィルターによる抽出)。次に、このフィルターが適用されることで変換されたデータから目や耳といった特徴を抽出し、その特徴からこの画像はネコであると認識するようになる。

図表 2.10　畳み込み層の概念図



出典：Chollet (2018) を基に筆者作成

このように畳み込み層は局所的なパターンを学習し、その局所的なパターンを組み合わせることで大局的なパターンを画像から見つけ出す機能を持っている。

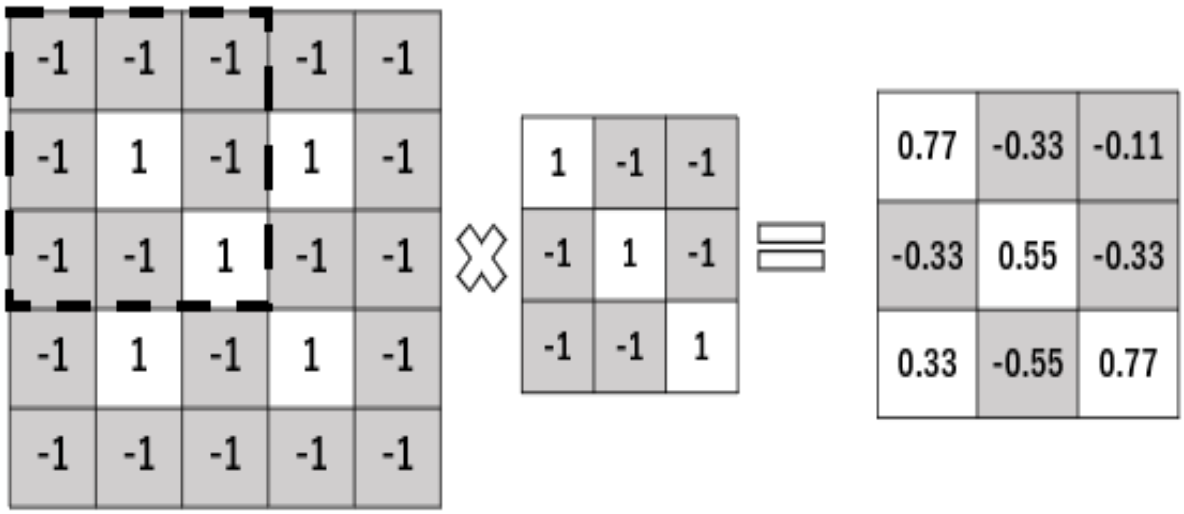
ここでのフィルターによる演算は畳み込み演算と呼ばれるものである。畳み込み演算は画像から特徴を抽出するための演算である。画像は RGB の 3 チャネルから構成される三次元データである。このデータのサイズは high, width, channel の 3 個のパラメーターによってあらわされ、例えば、28×28 の RGB 画像は high=28, width=28, channel=3 で示される。この三次元データはフィルターが適用されることで任意の次元を持つデータへと変換される。

例えばフィルターのサイズを(2×2)、フィルターの種類、つまりは抽出する特徴量の数を 32 とすると、画像はサイズが(26,26,32)のデータへと変換される。

この変換は元の画像(26×26)の右上から順番 1 ピクセルずつスライドさせながら(2×2)のフィルターによる変換を行っていることを意味している。

この工程を図で示したものが図表 2.11 となる。簡単の為、図表 2.11 では画像データを(5×5)、フィルターを(3×3)としている。

図表 2.11　画像に対するフィルターの適用



出典：巣籠（2018）を基に筆者作成。

このフィルターは右下がりの斜めを検知するフィルターとして解釈できる。フィルターが画像の左上部（点線四角）に適用されると画像の数値とフィルターの数値が掛け合わされた後にフィルター—のサイズ数で割られる。フィルター適用後の左上部の数値は以下のように計算される。

$$\frac{(1*1)*2+(-1*-1)*6+(-1*1)*1}{9}=0.77 \tag{2.18}$$

変換後のデータは右下がりの対角の数値が大きくなており、このようにフィルターが適用されることで画像に存在する右下がりの斜めを検知することができるようになる。

図からもわかるように畳み込み演算を適用していくということは、画像から特徴を抽出しながら画像を圧縮していく工程と考えることができる。

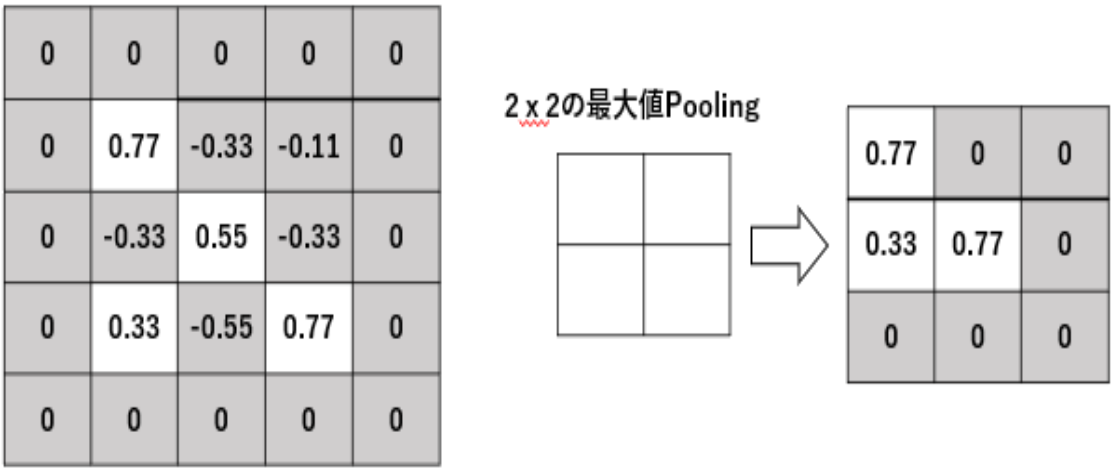
この圧縮しながら特徴を抽出していく工程を繰り返すことで、局所的特徴から画像に存在する大局的な特徴を抽出していく。これが畳み込み演算である。

また、ダウンサンプリングを目的とした Pooling と呼ばれる手法も存在する。ダウンサンプリングの目的はフィルターの係数を減らすことである。畳み込み層によって演算が行われた画像は最後の出力の段階で、1 次元化が行われる。この際にパラメーター数が多すぎると過学習の原因となるため、ダウンサンプリングを行い、ある程度パラメーター数を減らしてから 1 次元化を行う必要がある。

そうならないために、ダウンサンプリングによってパラメーター数を減少させる仕組みが Pooling である。

本研究で使用した Pooling の手法は最大値 Pooling であり、以下にその仕組みについて説明する。最大値 Pooling の工程はフィルター適用と同様の仕組みとして解釈することができる。フィルターの場合は元データにフィルターの係数が掛けられることで計算が行われていたが、最大値 Pooling ではその代わりにフィルター内の最大値をそのフィルターが適用された範囲における代表値として取り扱う。また Pooling ではフィルターの大きさと同じピクセル分だけスライドを行う。図表 2.12 では図表 2.11 でフィルターが適用された画像に最大値 Pooling を適用した例である。

図表 2.12　最大値 Pooling



出典：巣籠（2018）を基に筆者作成。

図表 2.12 のように 2×2 のサイズで最大値 Pooling を実行した場合、始めに左上の 2×2 の元データの中の最大値 0.77 が Pooling の結果として抽出される。

このように最大値 Pooling にはダウンサンプリングを行う効果があり、さらに畳み込み演算と同様に特徴を局所的から大局的に抽出する役割も担っている。

本研究で用いるモデルも上記の畳み込み演算とプーリングを行う畳み込み層を用いて画像の処理を行っている

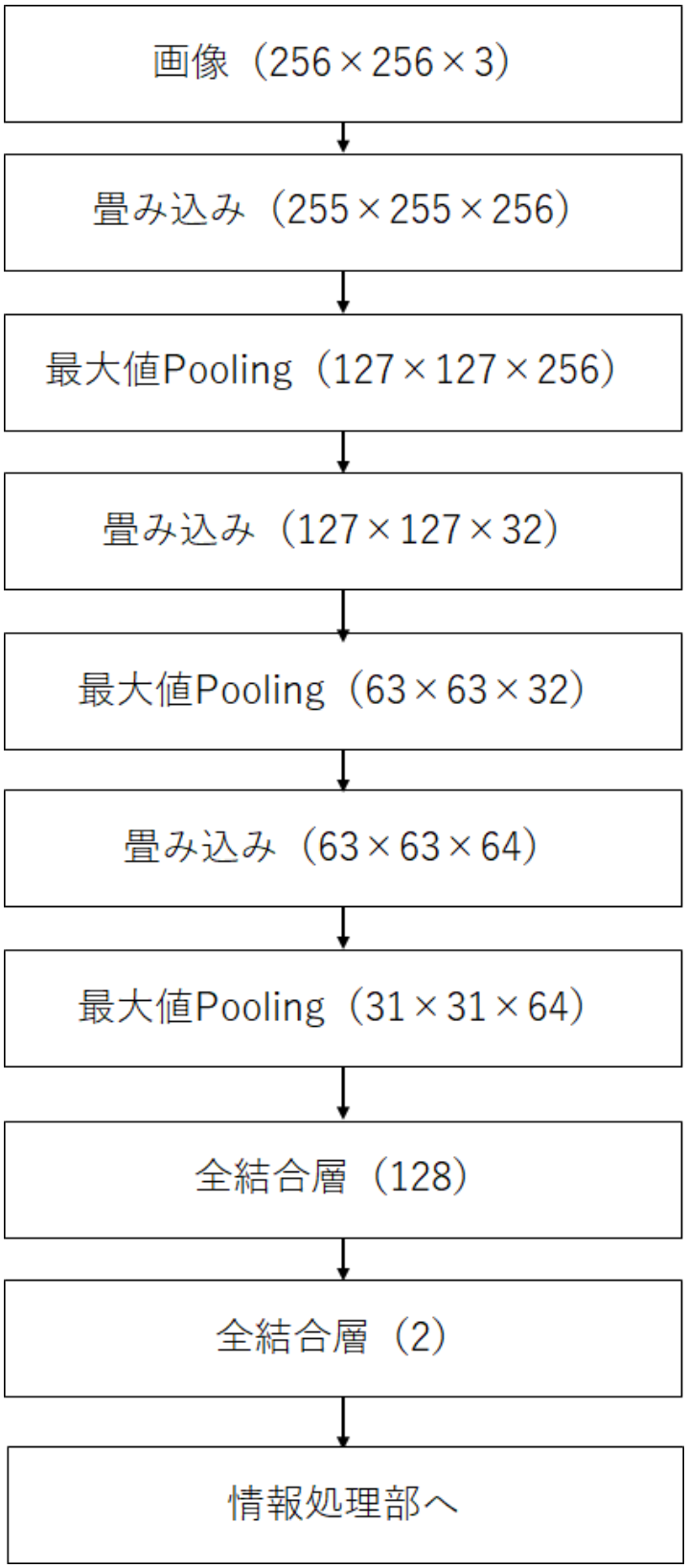
2.5.1　画像データの処理

本研究で用いる画像データは車の外観に関する画像データである。画像のサイズは(256×256×3)である。まず画像データを RGB の数値データに置き換える。この時、画像データの RGB それぞれの数値は 0 から 255 の数値で示される。

本研究で使用するモデルにおける画像処理は以下の構成となっている。畳み込み演算を行う層が合計 3 層、最大値 Pooling を行う層が合計 3 層の構造となっている。畳み込み演算のフィルターは 2×2 のサイズである。

まず画像は第 1 の畳み込み演算によって（256×256×3）から（255×255×256）へと変換される。この変換されたデータに対して最大値プーリングを行うことでデータサイズは（127×127×256）となる。この演算を合計 3 回繰り返すことでデータサイズは（31×31×64）のデータへと変更される。本研究においては平坦化された画像データは LSTM 層と同様に全結合層によって処理がなされる。全結合層は合計で 2 層存在し、第 1 層目の全結合層で出力数を 128、第 2 層目の全結合層の出力数を 2 としている図表 2.13。

図表 2.13　本研究における画像処理過程

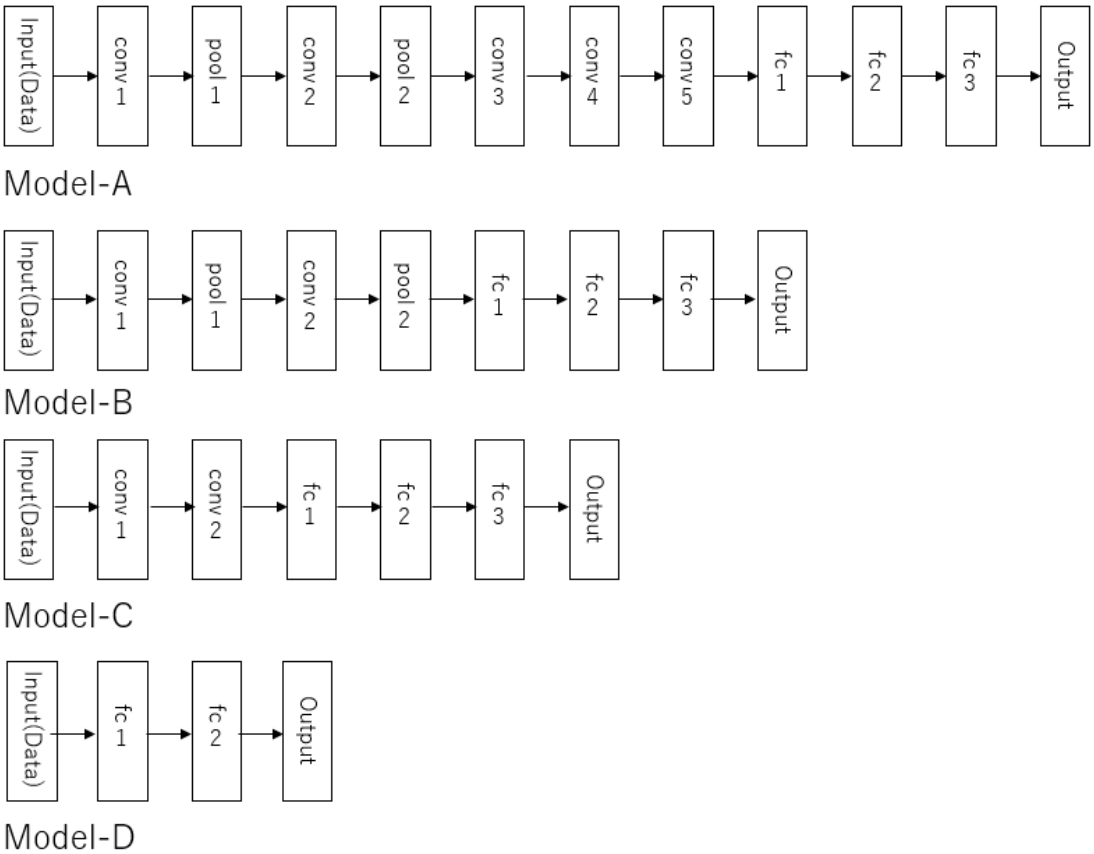


3. 先行研究のレビュー

Deep Learning モデルを用いたマーケティングに関する研究のレビューを行う。Deep Learning を用いたマーケティングに関する研究は近年盛んに行われており、売上予測、顧客管理分析、レビュー情報の分析、顧客の離脱率を分析した研究など様々な研究が行われており、どの研究においても分布を仮定せずに非線形で柔軟な分析が可能である Deep Learning を使用したモデルは既存手法のモデルより高精度かつ柔軟であると結論付けられている(Zaiyong 2011, Šťastný et al. 2011)。

若林らの研究（若林、生田目 2015）では Deep Learning モデルを用いて、予約のキャンセル予測を行うモデルを作成していた。ここでは全結合層、畳み込み層、プーリング層など様々な Deep Learning における層の種類を用いて様々なモデルが作成されており、その精度比較が行われていた。

図表 3.1　若林らの研究によって用いられていたモデル



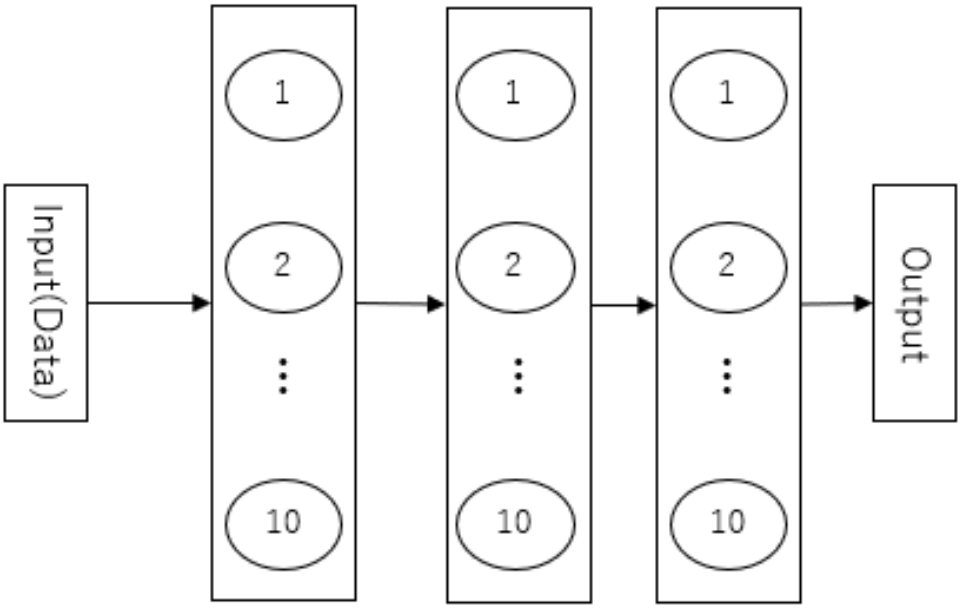
出典：若林、生田目（2015）を基に筆者作成。

彼らは図表 3.1 の使用層の数と種類の異なるモデルの精度比較を行っておりマーケティングの特徴を捉えるうえでは畳み込み層や Pooling 層を用いることが効果的であるとしていた。

新美らの研究（新美、星野 2017）では顧客行動の分析に多様性という概念を導入することの意義について研究しており、ここでも Deep Learning を用いたモデリングが行われていた。彼らは HHI（ハフィンダール・ハーシュマン指数）(Davis and Graces2009) を用いて顧客の多様性を定義し、それらを説明変数に加えることがモデルの精度向上に貢献するということを研究で明らかにした。

彼らの使用したモデルは Deep Learning の中でも代表的な Feed Forward Neural Network であり、3 層の隠れ層と各層に 10 個のニューロンを設定したモデルであった（図表 13）。

図表 3.2　新美らの研究に用いられている Deep Learning モデル



出典：新美、星野（2017）を基に筆者作成。

図表 3.2 のモデルの丸部はニューロンを示している。彼らの用いたモデルの層はすべて全結合層による計算によって行われており、各層における計算は以下の式で示すことができる。

$$h_j = tansign\left(b_j + \sum_{i=1}^{m} w_{ji} x_i\right) \tag{3.1}$$

$$tansign(x) = \frac{1}{1 + e^{-2x}} - 1 \tag{3.2}$$

h_i は全結合層によって変換された入力値の値で、m は入力 x_i の数であり、w_{ji} は重み、b_j はバイアスを示す。

彼らの研究では GLM に対する比較も行われており Deep Learning を用いたモデルのほうが GLM より高い精度を示す結果となっていた。

また、レビュー情報を分析した研究には Liu らの研究がある（Liu et al. 2017)。彼女らの研究ではインターネットの通販サイトにおけるレビュー情報から Conversion Rate を予測するモデルを作成しており、部分的に Deep Learning モデルを使用する Partial Deep Learning Model と Deep Learning モデルのみの Full Deep Learning Model を用いて研究を行っていた。

Partial Deep Learning Model はレビュー情報の含む 6 次元の特徴を抽出することを目的に行われおり、Conversion Rate は以下の式によって算出されていた。

$$u_{ikj} = \alpha_{ik} + \overrightarrow{\theta_k}\overrightarrow{Z_{it}} + \overrightarrow{\gamma_k}\overrightarrow{X_{jt}} + \zeta_j + \overrightarrow{\beta_k} * \overrightarrow{ReviewFeatures_{jt}} + \varepsilon_{ijkt} \tag{3.3}$$

$$\text{Conversion Rate}_{\text{ijkt}} = \frac{\exp(u_{ijkt})}{1 + \exp(u_{ijkt})} \tag{3.4}$$

i は消費者、k はデバイス、j は対象製品、t は時刻、u_{ijkt} はそれらによって算出される効用を示しており、α_{ik} は消費者の個人特性、$\overrightarrow{Z_{it}}$ は製品検索回数などの消費者毎によって異なる行動、$\overrightarrow{X_{jt}}$ は価格や平均レビュー得点などの製品の特性、ζ_j は製品の観測されていない特徴、ε_{ijkt} は誤差を示している。また $\overrightarrow{\beta_k} * \overrightarrow{ReviewFeatures_{jt}}$ は 6 つのレビューの含む特徴（Aesthetics, Conformance, Durability, Feature, Perceived-Quality, Price）にポジティブ、ネガティブを加えた 8 次元の特徴とその係数を示しており、以下の式によって定義されている。

$$\overrightarrow{\beta_k} * \overrightarrow{ReviewFeatures_{jt}} = \beta_k^p + PosReviews_{jt} + \beta_k^n * NegReviews_{jt} + \overrightarrow{\lambda_k}\overrightarrow{C_j} \quad (3.5)$$
$$+ \beta_k^a * SAesthetics_{jt} + \beta_k^c * SConformance_{jt}$$
$$+ \beta_k^d * SDurability_{jt} + \beta_k^f * SFeatures_{jt}$$
$$+\beta_k^{pq} * SPerceivedQualuty_{jt} + \beta_k^{pr} SPrice_{jt}$$

Partial Deep Learning Model はこれらのレビューの特徴をレビュー情報から抽出することを目的に使用されていた。

Full Deep Learning Model は Partial Deep Learning Model のようにレビューから一旦情報を抽出することなく、直接 Conversion Rate の予測を行っており、Deep Learning モデル自体に関する解釈は行われていなかった。

このように Deep Learning を使用したマーケティングに関する研究は数多く存在するが、どの研究も Deep Learning を予測や分類などの精度向上を目的に使用しており、Deep Learning のブラックボックス部分に触れたり、Deep Learning モデルの解釈を何らかの方法で行い、マーケティング的に意味のある情報の抽出を行っている研究は見かけられなかった。

本研究はこれらの先行研究とは異なり、Deep Learning モデルを単なる精度向上に用いるのではなく、そのモデルからマーケティング的に価値のある情報の抽出を行うものである。

本研究では態度を対象に Deep Learning モデルを作成する。そのモデルには第 4 章で触れる精緻化見込みモデルと補償型ヒューリスティクスの消費者情報処理理論の組み込みを行い、モデル内に構造を構築する。

構造の構築を行った後は、次にモデルに理論構造が反映できたかどうかの検証を行う。第 6 章で詳しく触れるが、この判定には MSE を指標として用いて、異なるモデル間の精度比較を行う。

そして、理論構造の反映を確認した後は、そのモデルから数値の部分的抽出を行うことで、ヒューリスティクスに関する情報の抽出を試みる。

4. 精緻化見込みモデルと補償型ヒューリスティクス

4.1 精緻化見込みモデル

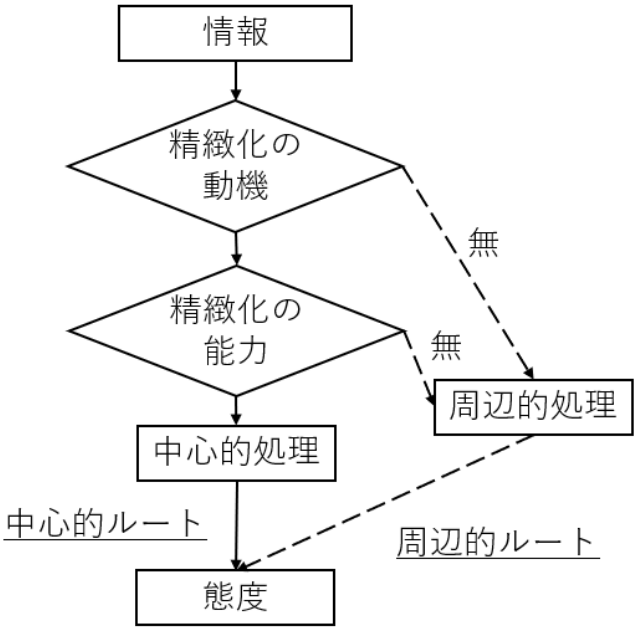
精緻化見込みモデル（Petty and Cacippo 1984a, 1984b）とは 1984 年に Petty と Cacippo が提案した、消費者の態度形成に関するモデルである。

精緻化見込みモデルは消費者の情報を精緻化する動機と能力によって情報処理の過程が異なることを示したモデルである。

精緻化見込みモデルは消費者が態度を形成する際に、画像などの周辺的手掛かり情報など簡単な情報をそれほど吟味せずに態度形成などの意思決定を行うのか、それとも製品の詳細情報などの情報を精緻に吟味して態度形成などの意思決定を行うかの 2 つの情報処理方略を仮定したモデルであり、それらはそれぞれ「周辺的ルート」、「中心的ルート」と呼ばれている（図表 4.1）。

これらのルートの選択は消費者個人の情報に対する精緻化の動機と能力の有無によって決定される。能力と動機付けがともに高い場合は、消費者は製品につて詳細な情報を吟味し態度形成を行うが、能力と動機付けのいずれか一方、または双方が低い場合は周辺的手掛かり情報によって態度形成が行われると考えられており、この仮説は多くの研究によって支持されている（Kohyama and Fujihara 1992、 Petty and Cacippo 1984a、Petty et al. 1993）。

図表 4.1 精緻化見込みモデル



出典：Petty and Cacioppo (1984) を基に筆者作成。

4.1.1　周辺的ルート

周辺的ルートは情報の精緻化に関する動機、能力のない消費者が通る態度形成に対する情報処理のルートである。周辺的ルートを通り、態度形成を行う消費者は周辺的手掛かり情報を用いて態度形成を行う。

周辺的手掛かり情報とは事前に製品に対する知識がなくても、その情報から何らかの意味や特徴を抽出することができる処理が簡単な情報であり、本研究では製品画像と、製品に対して書かれた一般人によるレビューを周辺的手掛かり情報と規定した。

こうした情報の精緻化に関する動機、能力のない消費者が周辺的手掛かり情報によってのみ態度形成を行う情報処理過程を周辺的ルートと呼び、このルートによってよって形成された態度は堅牢ではなく、比較的簡単に態度変容が起こることが分かっている。

4.1.2　中心的ルート

中心的ルートは情報の精緻化に関する動機、能力双方を持った消費者が通る態度形成に対する情報処理のルートである。中心的ルートを通る消費者は製品の価格、性能、品質などのスペックに関する細やかな情報を丁寧に吟味し、態度形成を行うとされている。

本研究ではそうした製品にかかわる詳細情報をメーカー、コスト、パワー、サイズ、利便性の 5 つのグループに分類された変数郡であるとした。これらの情報は複雑で知識がなければ情報処理を行うことが難しいと考えたためである。

周辺的ルートと異なり、中心的ルートによって形成された態度は堅牢であり、態度を変えることは比較的困難とされている。

本研究ではこれら 2 つの情報処理過程のルートを Deep Learning モデルの構造に反映させることを試み、各ルートが的確に反映されているか検証していく。

4.2　補償型ヒューリスティクス

消費者が態度形成を行う際、その決定方略には補償型(compensatory)、と非補償型(non compensatory)に分類して考察することがある。

補償型とはある情報の評価値や重要度が低い場合、その他の情報の評価値や重要性が高ければ、その高い評価値の情報によって、低い評価値の情報が補われて総合的な態度形成を行う決定方略である (図表 4.2)。これと対照的に非補償型では情報同士の補償関係が見られない場合を指す (Roberts and Lilien 1997)。

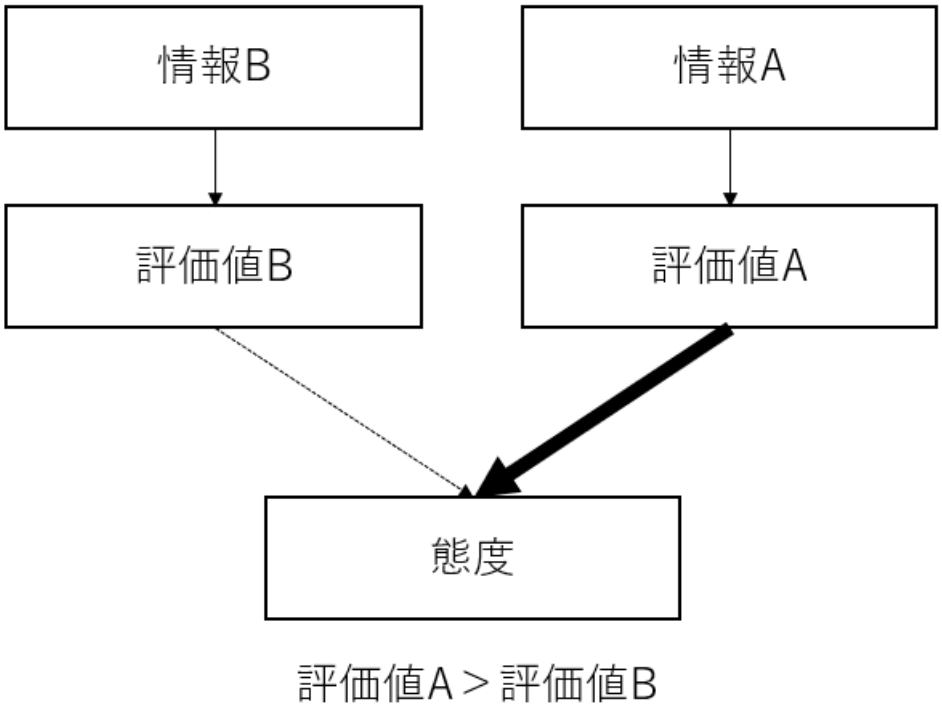
本研究では LAMBDA 層と呼ばれる特殊な Deep Learning の層を用いてこの補償型ヒューリスティクスを表現した。

また、複数の研究からは、選択肢や属性が増えると補償型の態度形成はなされないことが知られておりこれは処理すべき情報量が多すぎることが原因であるとされている。本研究で用いる Deep Learning モデルでは一度に学習するデータの数(Batch Size)を制限することで対応している。

本研究の Deep Learning モデルにもこの理論を反映させることで、どの情報がどの情報と補償関係にあるかを明らかにできるかどうかを検証していく。

図表 4.2 は消費者が情報 A、情報 B からそれぞれ評価値を算出し、補償型ヒューリスティクスによって態度形成を行っている過程を概念的に示したものである。この場合、消費者は情報 A について情報 B より高く、つまりは重要であると評価しており、この場合は情報 A から算出された評価値が情報 B の評価値を補う形で態度形成が行われる。

図表 4.2　補償型ヒューリスティクスの概念図



出典：Roberts and Lilien (1997)を基に筆者作成。

5. データとシステム環境の概要

5.1 データ収集について

本研究にはある Web サイトからデータを収集した。収集したデータは新車に関するデータであり、合計で 1123 車種分、21 種類を取得した。データの内訳は図表 5.1 に示す。なおレビュー情報に関しては最新の 5 件を 1 車種ごとに収集し、入力データとした。

図表 5.1 取得データ（2018 年 8 月 26 日取得）

	変数名	例	データ種
1	製品画像	製品画像	三次元行列
2	一般人レビュー	ボクサー５を…	テキストデータ
3	メーカー	ポルシェ	ダミー変数
4	価格	1239 万円	連続値
5	最大出力	257	連続値
6	最大トルク	390	連続値
7	車税	58,000 円	連続値
8	重量税	36,900 円	連続値
9	取得税	309,700 円	連続値
10	発売日からの経過日数	x 日	連続値
11	cc	3,436cc	連続値
12	全長	4,500mm	連続値
13	全幅	1,810mm	連続値
14	ホイール長	2,450mm	連続値
15	定員	4 名	連続値
16	ドア数	2 ドア	連続値
17	シート数	2 列	連続値
18	タンク容量	64L	連続値
19	ボディタイプ	クーペ	ダミー変数
20	駆動系	エンジン	ダミー変数
21	燃料	ハイオク	ダミー変数
22	お気に入り登録数	39	連続値

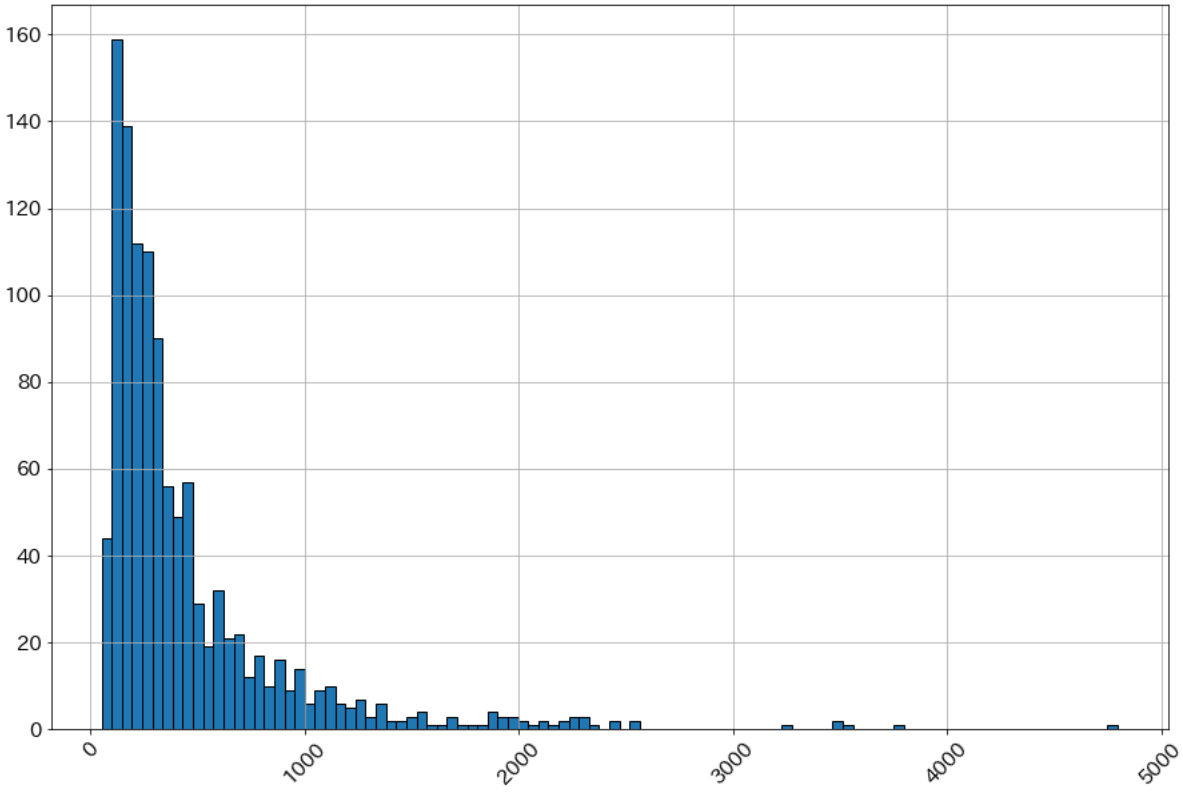
5.2　本研究に用いる入力データ(説明変数)と出力データ(被説明変数)

取得したデータの中で被説明変数に設定するのはお気に入り登録数であり、説明変数はそれ以外のデータとなる。

「お気に入り登録数」とは、消費者が当該 Web サイトを見た際に、気に入ったときや、購入を検討している際にクリックすることで加算されていくもので、当該車種がどの程度人気があり、好かれているのかを示す指標である。

被説明変数となる「お気に入り登録数」の分布は図表 5.2 のようになる。

図表 5.2　「お気に入り登録数」のヒストグラム



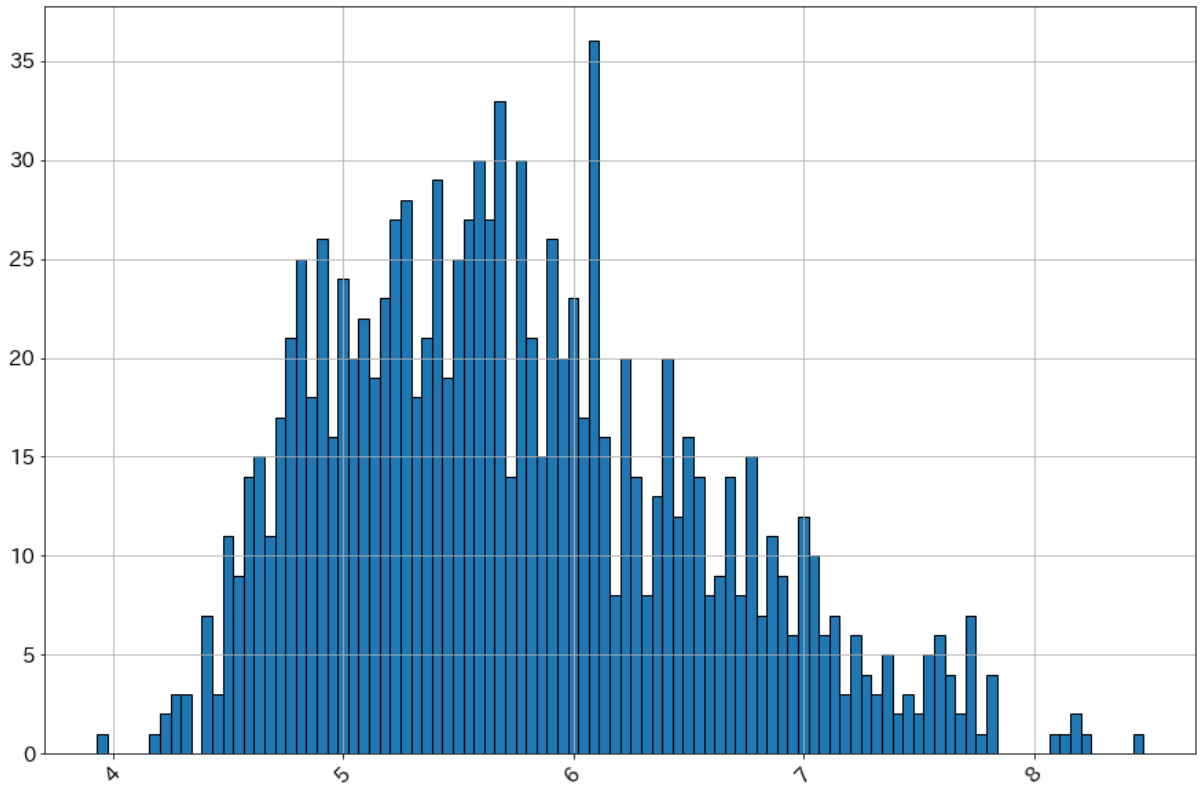
図表 5.2 のグラフから明らかなようにお気に入り登録数は非常に左に偏ったデータとなっている。通常の統計モデルと同様、偏りのあるデータでは正規分布と等分散性を仮定することができずに分析結果に大きなずれが発生してしまう恐れがあるため、本研究においてもそうした問題が発生しないよう、対数変換を行った。

対数変換を行った結果、「お気に入り登録数」の分布は図表 5.3 のようになり、正規分布に近い形の分布を得ることができた。

本研究では図表 5.3 のように対数変換された「お気に入り登録数」を被説明変数としてこの数値を説明変数から予測する Deep Learning モデルを構築する。

説明変数には価格や全長といった連続値のデータとメーカーやボディタイプなどの離散値データが存在する。これらの説明変数は連続値データの場合、学習を効率的に行う為、正規化を行い、離散値データの場合はダミー変数化して本研究のモデルの入力とした。

図表 5.3　対数変換後の「お気に入り登録数」のヒストグラム



5.3　データのグループ化

消費者は説明変数を同時に並列処理するとは考え難い。データ収集を行ったホームページにおいても、コストに関する情報、大きさに関する情報などは一纏めにされたり、近い場所に記述されていた。つまり消費者は属性の近い情報は同時に処理する傾向にあると考えられる。

本研究で用いる Deep Learning モデルにおいても消費者のこうした情報処理を反映させるため、データを属性に応じてグループ化して入力を行うように構築を行った。

つまり、コストに関する情報はコスト情報としてグループ化して入力し、そのグループ化された情報が処理された後にその他の情報と統合され態度形成が行われるモデルを作成した。

グループ化は図表 5.4 の通りである。グループ数は合計で「画像」、「一般人レビュー」、「メーカー」、「コスト」、「パワー」、「サイズ」、「利便性」の 7 グループとなった。

これらのグループ化された情報はそれぞれ、周辺的手掛かり情報と中心的ルートによって精緻に処理される情報に分けられる。

周辺的手掛かり情報とは周辺的ルートを通る情報処理を行う際に参照される情報のことで、事前知識や能力がなくても解釈が可能な情報とした。この場合は画像と一般人レビューが該当する。画像は車に関して詳しくない消費者でもデザインや色といった情報からその画像情報を処理することができ、またレビューも一般人によるものであるので、専門的な知識や知識がなくても処理できる情報と考え、本研究ではこれらの 2 つのグループ化された変数を周辺的手掛かり情報とし、後述の周辺的ルートモデルに対する入力情報とした。

周辺的手掛かり情報とは対照的に中心的ルートによって処理される情報とは処理するために知識と能力が必要となる情報を指し、本研究の場合は「メーカー」、「コスト」、「パワー」、「サイズ」、「利便性」が該当する。本研究ではこれらの情報を中心的情報と呼ぶ。

図表 5.4　変数のグループ化

グループ名	変数名	分類
画像	画像	周辺的手掛かり情報
レビュー	一般人レビュー	周辺的手掛かり情報
メーカー	メーカー	中心的情報
コスト	価格	中心的情報
	最大出力	
	車税	
	重量税	
	発売日からの経過日数	
パワー	cc	中心的情報
	最大出力	
	最大トルク	
	全長	
	全幅	
	ホイール長	
	ボディタイプ	
サイズ	定員	中心的情報
	ドア数	
	シート数	
	全長	
	全幅	
	ホイール長	
	ボディタイプ	
利便性	定員	中心的情報
	ドア数	
	全長	
	全幅	
	駆動系	
	燃料	

5.4　データの分割

次にデータの分割を行った。まずはデータを周辺的ルートデータと中心的ルートデータに分割した後に、Deep Learning モデルの評価を行うため、それらのデータからホールドアウトデータを抽出した。

5.4.1　周辺的ルートデータと中心的ルートデータ

周辺的ルートデータと中心的ルートデータとは、それぞれ周辺的ルート、中心的ルートによって態度形成がなされると考えられるデータである。

周辺的ルートデータは情報の精緻化が行われずに態度形成が行われると想定される車種を選定し、中心的ルートデータでは情報の精緻化が行われると想定される車種を選定した。

車種の選定には以下の 3 つの基準を用いた。

まず第 1 の基準は価格である。低価格の車種の場合、購入時のリスクが低く、またそういった車種は車について詳しくない消費者を想定して開発されるものが多いと考えられるため、価格を選定基準の一つとした。ここでは 300 万円以上の車種は中心的ルートデータ、300 万円未満を周辺的ルートデータの条件とした。

第 2 の基準はボディタイプである。ボディタイプとはセダン、ミニバン、クーペなどの車の車体の形状を指す。低価格であってもクーペやオープンカーなど移動手段としてではなく、車そのものに楽しみを見出し、購入を検討する場合、情報の精緻化は行われると考えることができる。ボディタイプがクーペ、またはオープンカーに分類されるものは中心的ルートデータ、その他のボディタイプの場合を周辺的ルートデータの条件とした。

第 3 の基準は輸入車か否かである。輸入車を購入する消費者はボディタイプと同様に車についてこだわりを持つ消費者であると考えられる。よって輸入車であることを中心的ルートデータ、国産車であることを周辺的ルートデータの条件とした。

上記 3 つの条件を鑑みて、中心的ルートデータの条件を一つでも満たす場合はその車種を中心的ルートデータの車種、その他の場合、つまり上記 3 つの周辺的ルートデータの条件をすべて満たす車種を周辺的ルートデータに分類した（図表 5.5）。

分割の結果、周辺的ルートデータは 348 件、中心的ルートデータは 775 件となった。

図表 5.5　周辺的ルートデータ・中心的ルートデータの条件

	周辺的ルートデータ	中心的ルートデータ
価格	300 万円未満	300 万円以上
ボディタイプ	その他	クーペ・オープンカー
輸入車	国産車	輸入車

5.4.2　ホールドアウトデータ

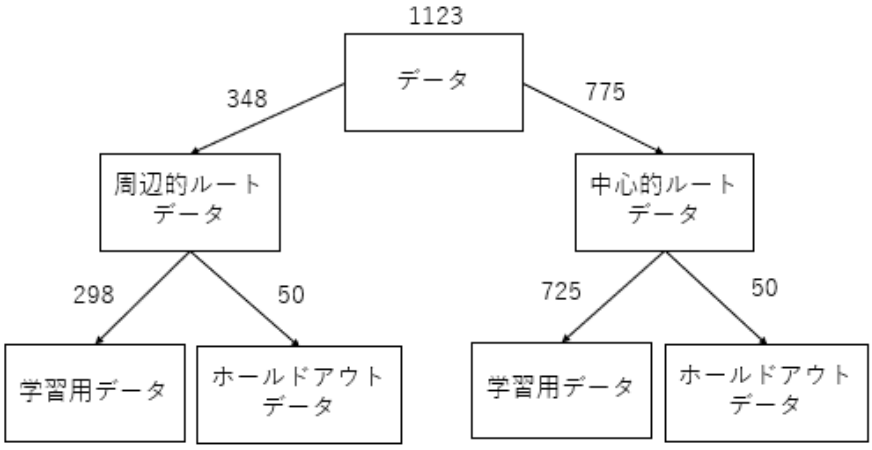
ホールドアウトデータとは構築、学習したモデルの評価を行うためのデータである。本研究ではモデルを作成後、学習に使用しなかったデータを用いてモデルの汎化性能の評価を行う。

ホールドアウトデータは周辺的ルートデータ、中心的ルートデータからランダムにそれぞれ 50 件ずつ抽出した。

データ分割についてまとめると図表 5.6 のようにまとめることができる。図表の数値はデータ数を示している。

以上のように本研究では取得したデータに対し、第 1 に全体の 1123 件のデータを周辺的ルートデータと中心的ルートデータに車種ごとに分割をおこない、第 2 に周辺的ルートデータ、中心的ルートデータの中からランダムに 50 件の車種データをホールドアウトデータとして抽出し、残りを学習用データとする分割操作を行った。ここでいう学習用データは第 6 章で構築するモデルの学習にのみ使用し、ホールドアウトデータは学習したモデルの精度を確認する為に使用するデータのことを指す。

図表 5.6　データ分割の全体図



5.5　システム環境の概要

本研究は Deep Learning モデルを構築するために Python を使用し、プログラムを作成した。

Python とはプログラミング言語の 1 つであり、元々は ABC 言語という教育用の言語にオブジェクト指向を適用させるために開発されたプログラミング言語である。現在では、Web アプリケーションの開発や統計分析から Deep Learning まで幅広く活用されており、現代では主流なプログラミング言語の一つとなっている（Python HP)。

また本研究で構築した Deep Learning モデルは TensorFlow をベースに開発されたオープンソースニューラルネットワークライブラリである Keras を使用して構築した。

プログラミングの実行環境は以下の図表 5.7 の通りである。OS は Windows10、CPU は i7、GPU は NVIDIA 社の GeForce GTX 1060、GeForce GTX 1070 である。計算には 2 台の PC と GPU を使用して計算を行った。

図表 5.7　システム環境

PC	OS	CPU	GPU
PC1	Windows10	Corei7	GeForceGTX 1060
PC2	Windows10	Corei7	GeForceGTX 1070

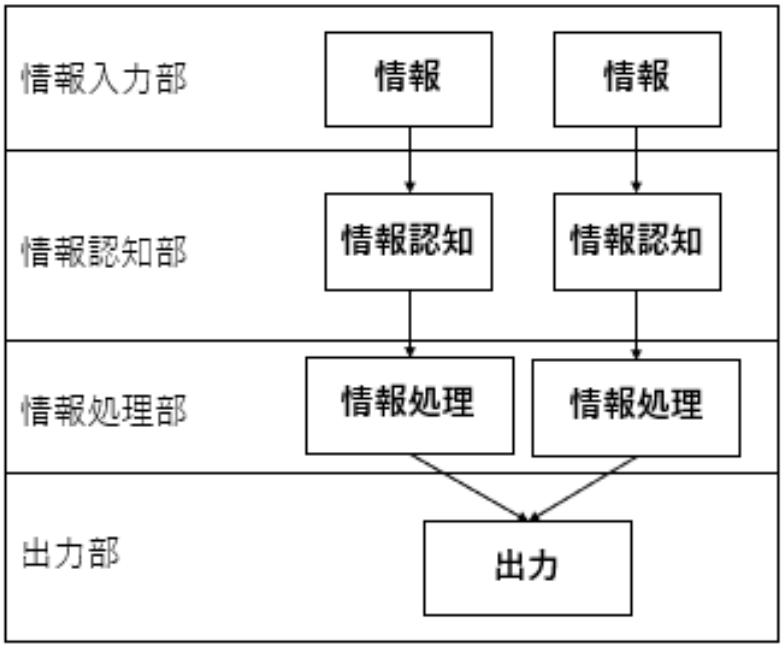
6.　精緻化見込みモデル構築と評価

6.1　モデルの基本構造

本研究ではまず、モデルの基本構造を規定し、それらをカスタマイズすることで様々な方法処理の過程を構造化してモデル化した。

モデルの基本構造は図表 6.1 のようになり、本研究で用いるモデルは情報入力部、情報認知部、情報処理部、出力部の 4 つのユニットから構成されるものとする。では次に、それらの構造を以下の項（6.1.1~6.1.4）で見ていくことにする。

図表 6.1　モデルの基本構造



6.1.1　情報入力部

情報入力部は情報の入力が行われる部分である。ここではグループ化された情報のうちいずれかを選択して当該モデルの説明変数となる情報の選択を行う。

詳しくは後述するが、例えば周辺的ルートモデルの場合は、周辺的手掛かり情報によってのみ態度形成が行われることを仮定している為、情報入力部では周辺的手掛かり情報に分類される、画像、一般人レビューが入力として選択されることになる。

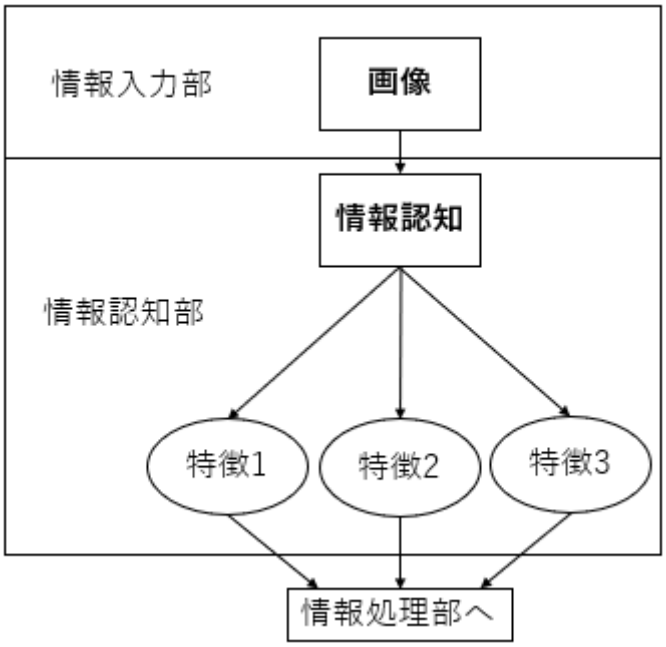
6.1.2　情報認知部

情報認知部はグループ化された情報を処理する部分である。画像データであれば畳み込み層、レビューデータであれば LSTM 層、連続値やダミー変数などであれば全結合層によって処理が行われる。

情報認知部からの出力は任意の数を設定することができる。この出力数は入力された情報に対して認知した特徴数と考えた。

例えば情報認知部の出力数を 3 と設定すれば、それは当該情報から 3 つの特徴を認知したと考えることができる。図表 6.2 では画像情報から情報認知部によって 3 つの特徴を抽出している概念図を示す。

図表 6.2　情報認知部



6.1.3　情報処理部

情報処理部では情報認知部からの出力を全結合層によって処理する部分である。

情報処理部の出力数は１であり、情報認知部によって認知された特徴からその情報に対する評価値を出力する役割を担っていると考えることができる。

このように以下では情報処理部からの出力を評価値と呼ぶ。

6.1.4　出力部

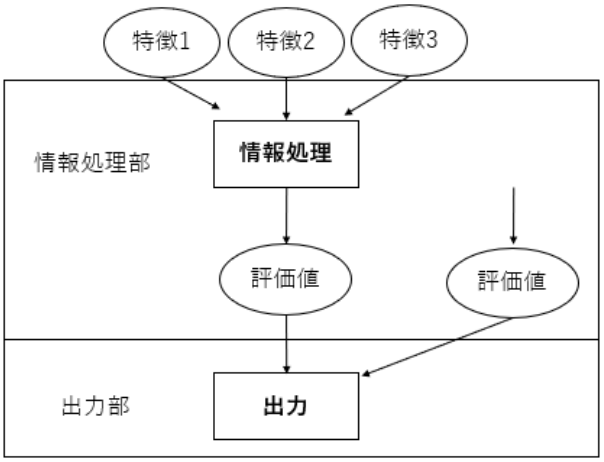
出力部では情報処理部からの出力（評価値）を用いて態度形成の予測値を出力する。

本研究で用いる出力値は「お気に入り登録数」の対数変換された値であり、出力部では出力である「お気に入り登録数」の対数変換された値を出力する。

正確に記すと情報認知部からの評価値を入力とした全結合層によって、評価値に任意の重みを掛け合わせて積和された値が「お気に入り登録数」の対数変換された予測値となる。情報 i の評価値を Ev_i 、情報 i の評価値に掛かる重みを w_i とすると出力 outは式(6.1)のように示すことができる。この式を見ると明らかなように出力部の全結合層における計算過程は多属性態度モデルと酷似しており、7.4 節にてこの構造に注目してヒューリスティクスに関する考察を行っていく。図表 6.3 に特徴から評価値が算出され出力が行われる様子を示した。

$$out = \sum_{i=1}^{N} w_i Ev_i \tag{6.1}$$

図表 6.3　情報処理部・出力部



6.2　モデルの深さと幅

モデルに周辺的ルートと中心的ルートを表現する為に、ここでモデルの深さと幅という概念を導入する。この概念は周辺的ルート、中心的ルートそれぞれを通って態度形成を行う消費者の情報処理過程をより区別をつけて反映させるための概念である。

6.2.1　モデルの深さ

モデルの深さとは、消費者の情報処理の精緻化の能力と動機を示す一つの概念である。情報の精緻化に対して能力と動機をもつ消費者は、それを持たない消費者と比較してより深く情報に対して考察を行う傾向にあると考えることができる。この消費者の情報処理過程の特徴を Deep Learning モデルに反映させるために必要な概念がモデルの深さという概念である。より深く考察を行うということは、それらの情報を処理する過程で様々な考察がなされると考えることができる。Deep Learning モデルもその層構造が多層構造になればなるほど、ある入力値はその他の入力値と複雑に結びつく。つまり、このより深く考察を行うという消費者の情報処理の特徴はモデルの層の数よって表現することができると本研究では考えた。

以上より、情報の精緻化に対する能力と動機を持つ消費者が行う態度形成に対する情報処理は、つまり中心的ルートは、そうでない消費者、つまり周辺的ルートと比較してより多い情報処理部における層の数を持つと考えた。

6.2.2　モデルの幅

次にモデルの幅について説明を行う。モデルの幅も消費者の情報に対する精緻化の能力と動機を表現する為に必要な概念である。

情報に対する精緻化の能力と動機を持つ消費者はそうでない消費者と比較して一つの情報からより多くの特徴を見出すと考えられる。

本研究で用いるモデルに置き換えると、情報認知部から情報処理部への出力、つまり抽出される特徴量がこの概念に該当する。本研究では中心的ルートは周辺的ルートと比較して情報認知部からの出力数が多くなるように設定し周辺的ルートと中心的ルートの表現を行った。

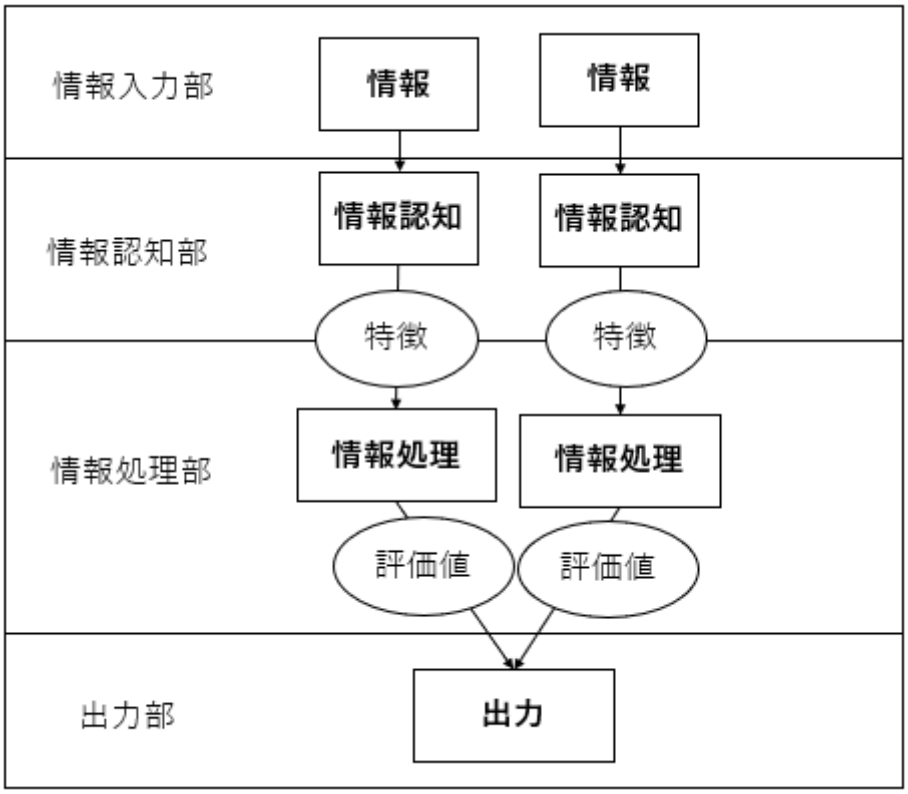
この 2 つの概念によって周辺的ルートと中心的ルートをモデルとして表現する際に、より差異を持たせて構築を行った。以上をまとめると図表 6.4 のようになる。

図表 6.4　周辺的ルートと中心的ルートのモデルの幅と深さ

	周辺的ルート	中心的ルート
深さ	浅い(2)	深い(6)
幅	少ない(3)	多い(5)

以上のように本研究ではモデル構築に関して四つのユニット（情報入力部、情報認知部、情報処理部、出力部）を設定し、さらに深さと幅という概念を導入し、精緻化見込みモデルと補償型ヒューリスティクスを反映した異なる構造を持つ様々なモデル構築を行っていく。各概念をまとめたものを図表 6.5 に示した。情報は情報認知部によって特徴が抽出され、その特徴が情報処理部によって評価値に変換される。そしてその評価値による全結合層による積和が出力値として算出される。

図表 6.5　モデルの構造



6.3　周辺的ルートモデル

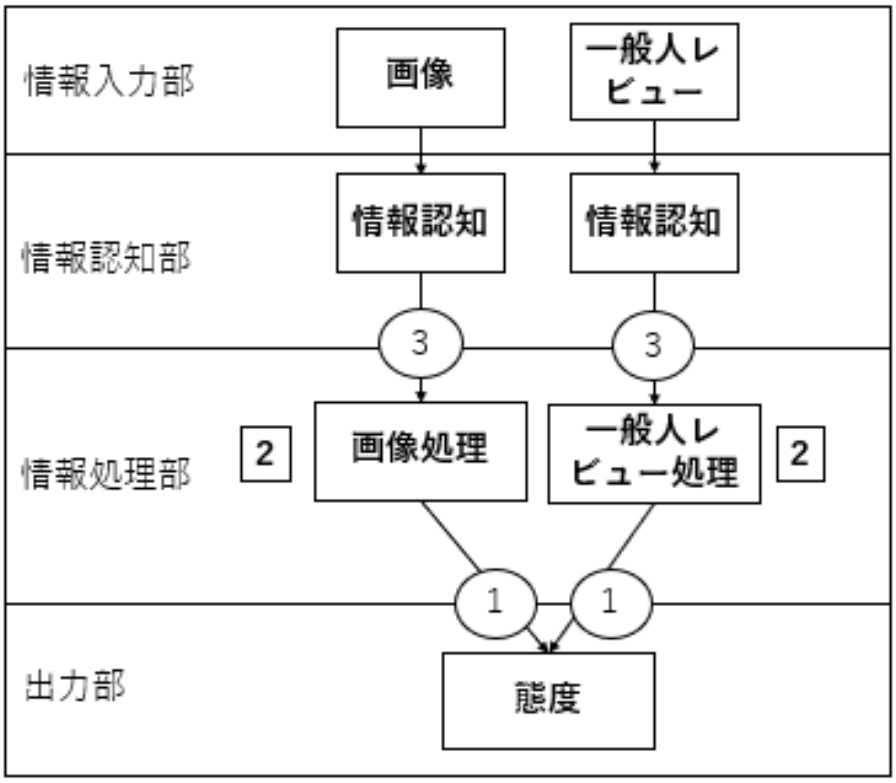
周辺的ルートモデルとは周辺的ルートを通る消費者の情報処理過程を Deep Learning モデルに構造として反映させたモデルである。

周辺的ルートによる態度形成は周辺的な情報によって行われると考えられる。その為、周辺的ルートモデルの入力は今回収集した情報の中で周辺的な情報と考えることができる「画像」と「一般人レビュー」に限定した。

また周辺的ルートモデルは中心的ルートモデルと比較し、情報認知部の出力数（情報から抽出する特徴量）と情報処理部の層の数が少なくなると想定することができるので、それぞれ情報認知部からの出力を 3、情報処理部の総数を 2 層に設定した。

図表 6.6 に周辺的ルートモデルの例を示した。図表 6.6 の丸に囲まれた数値は各ユニットからの出力数を、認知処理部のユニットの横に表示された四角に囲まれた数値は層の数を示している。

図表 6.6　周辺的ルートモデルの例



6.4　中心的ルートモデル

中心的ルートモデルとは中心的ルートを通る消費者の情報処理過程を Deep Learning に構造として反映させたモデルである。

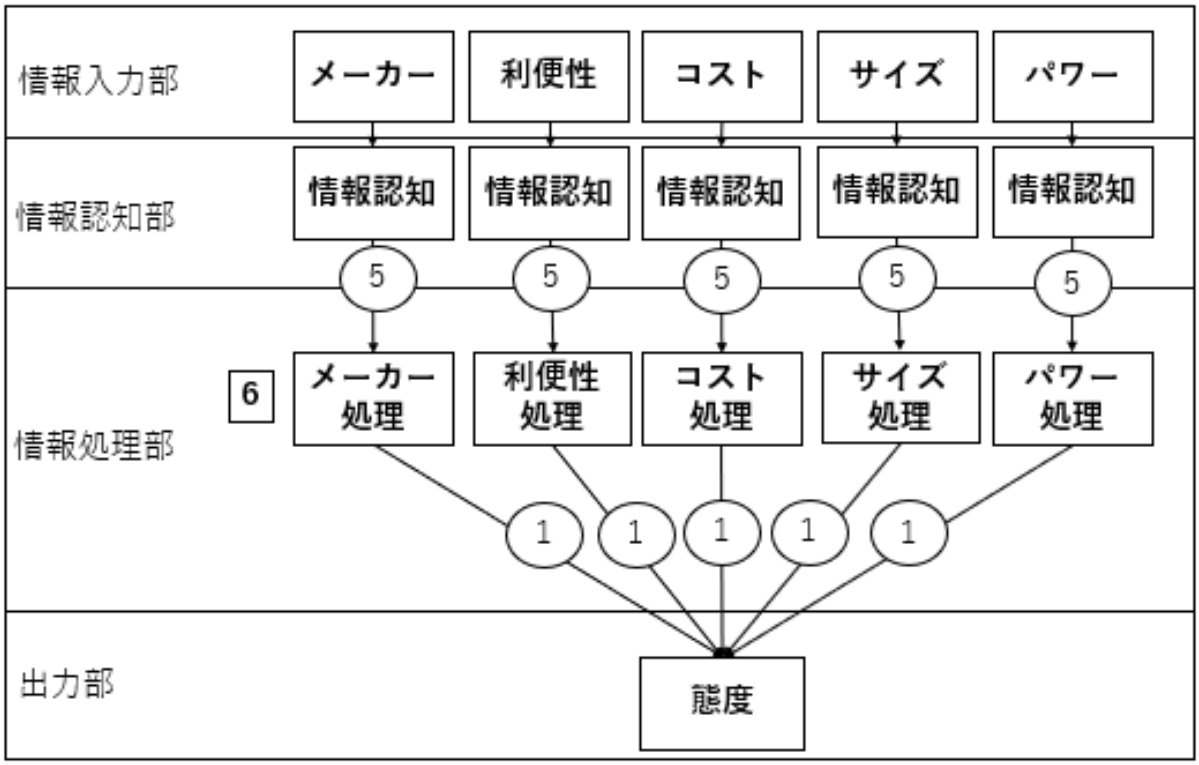
中心的ルートによる態度形成は中心的な情報によって行われると考えられる。その為、中心的ルートモデルの入力は今回収集した情報の中で中心的情報と考えることができる「メーカー」、「コスト」、「パワー」、「サイズ」、「利便性」に限定した。

また中心的ルートモデルは周辺的ルートモデルと比較し、情報認知部の出力数と情報処理部の層の数が多くなると想定することができるので、それぞれ情報認知部からの出力を 5、情報処理部の総数を 6 層に設定した。

図表 6.7 に中心的ルートモデルの例を示す。

以上のように消費者の態度形成に対する情報処理過程を Deep Learning モデルに構造として取り込み、本研究におけるモデルを構築した。

図表 6.7　中心的ルートモデルの例



6.5　モデルに学習させる意味

次にモデルを学習させることの意味を説明する。

モデルが学習するとはモデルに表現力を与えることと同義である。

例えば回帰分析を始めとする統計学のモデルでは最小二乗法や最尤法といったアルゴリズムにより説明変数から被説明変数を尤よく説明できる係数を算出していく。こうして各説明変数に係数が与えられることでモデルは説明変数から被説明変数を尤もらしく表現することができるようになる。

要するにモデルの学習は説明変数から被説明変数を予測する為の表現力をモデルに付与する工程であると考えられる。

この考え方は Deep Learning においても同様である。Deep Learning モデルを構築しただけではそのモデルは何も表現しておらず、学習を行うことで初めて表現力を獲得できる。

本研究において Deep Learning モデルに表現させたいのは情報を処理して態度形成を行う消費者である。上述の通り、情報処理には周辺的ルートと中心的ルートが存在し、それぞれを表現したモデルにそれぞれに適したデータを学習させる必要がある。

その為には周辺的ルートモデルには周辺的ルートデータ（すなわち周辺的ルートによって態度形成がなされると考えられる安価な国産車などのデータ）に対して尤もらしく振舞うようにモデルに学習させる必要があり、当然だが周辺的ルートモデルの学習には周辺的ルートデータが使用される。こうして学習させる前は単なる周辺的ルートの構造のみしか持っていなかった周辺的ルートモデルは周辺的ルートデータに対して尤もらしく振舞う消費者を模することができるようになる。この考えは中心的ルートモデルに関しても同様であり中心的ルートモデルは中心的ルートデータによって学習が行われる。

6.6　モデルの評価方法と仮説

モデルの評価にはホールドアウトデータによる予測によって計算される *MSE*（平均二乗誤差）を用いる。

学習済みの Deep Learning モデルに各ホールドアウトデータを適用することで予測値が算出され、そしてその予測値と各ホールドアウトデータに対する正解値から *MSE* が算出される。

MSE は以下の式によって計算される。

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(pred_i - true_i)^2 \tag{6.2}$$

上記の式における N はデータ数、$pred_i$ はホールドアウトデータの i 番目の車種データに対する「お気に入登録数」の対数変換された予測値、 $true_i$ は i 番目の車種データに対する「お気に入り登録数」の対数変換された正解値をそれぞれ示す。

本研究における *MSE* の表現は下付き文字と上付き文字を使い、以下のように表現する。*MSE* の下付き文字はモデルのタイプ、すなわち周辺的ルートモデルか中心的ルートモデルかを表す。この場合、p を周辺的ルートモデル、c を中心的ルートモデルとする。*MSE* の上付き文字は予測を行う際に適用したホールドアウトデータの種類を示す。P の場合は周辺的ルートデータ、C の場合は中心的ルートデータを示す。例えばMSE_p^Pは周辺的ルートモデルに対して周辺的ルートデータのホールドアウトデータを適用した際の *MSE* の値である。ホールドアウトデータについては、周辺的ルートデータに分類されるもの、中心的ルートデータに分類されるものがそれぞれ 50 件ずつ存在する。

予測時の入力は以下の通りである。例えば学習済みの周辺的ルートモデルに周辺的ルートデータに分類されるホールドアウトデータを適用しMSE_p^Pを計算する場合、周辺的ルートモデルの入力は画像と一般人レビューの 2 変数なので、周辺的ルートデータに分類される車種のホールドアウトデータのうち、画像と一般人レビューをモデルに入力する。

また周辺的ルートモデルに中心的ルートデータに分類されるホールドアウトデータを適用し、MSE_p^Cを計算する場合、周辺的ルートモデルの入力は画像と一般人レビューの 2 変数なので、中心的ルートデータに分類される車種のホールドアウトデータのうち、画像と一般人レビューをモデルに入力する。

中心的ルートデータにはメーカーやコスト情報のみでなく、中心的ルートデータに分類された車種の画像、一般人レビューの情報もあり、周辺的ルートモデルにも当該データが適用できるようにデータ分割を行っていることをここで確認しておく(周辺

的ルートデータについても同様)。

MSE は全部で 4 種類のものが算出される（図表 6.8)。MSE 列は各 *MSE* の名称、Model 列は *MSE* を算出する際に適用するモデルの種類、DATA 列は *MSE* を算出する際に適用するホールドアウトデータの種類を示している。

図表 6.8　各 MSE の定義

MSE	Model	DATA
MSE_p^P	周辺的ルートモデル	周辺的ルートデータ
MSE_p^C	周辺的ルートモデル	中心的ルートデータ
MSE_c^P	中心的ルートモデル	周辺的ルートデータ
MSE_c^C	中心的ルートモデル	中心的ルートデータ

ここで *MSE* を評価することの意味について考察を行う。本研究では各 *MSE* を比較することでモデルの評価を行っていく。

MSE の比較によって検証したいことは各モデルがそれぞれの情報処理過程を反映することができているのかどうかである。

もし上記のように構築した周辺的ルートの情報処理過程が正しく反映することができていれば、周辺的ルートモデルに周辺的ルートデータのホールドアウトデータを適用した *MSE* の方が、同モデルに中心的ルートデータのホールドアウトデータを適用した *MSE* より小さくなると考えられる。このような比較をするためには、同一モデルに異なるホールドアウトデータを適用した *MSE* 同士の比較を行う必要があり、つまりはMSE_p^PはMSE_p^Cと、MSE_c^PはMSE_c^Cと比較が必要となる。

上記のことから、以下の仮説が考えられる。

$$\text{仮説 1；} MSE_p^P < MSE_p^C \tag{6.3}$$

$$\text{仮説 2：} MSE_c^P > MSE_c^C \tag{6.4}$$

上述の通り、モデルとホールドアウトデータの種類が合致している場合は、合致していない場合より *MSE* の値が低くなるはずであり、式(6.3)は周辺的ルートモデルが周辺的ルートの情報処理を、式(6.4)は中心的ルートモデルが中心的ルートの情報処理を正しく反映している際に成立すると考えられる仮説である。この仮説については以下の 6.8 節においてこの仮説の検証を行う。

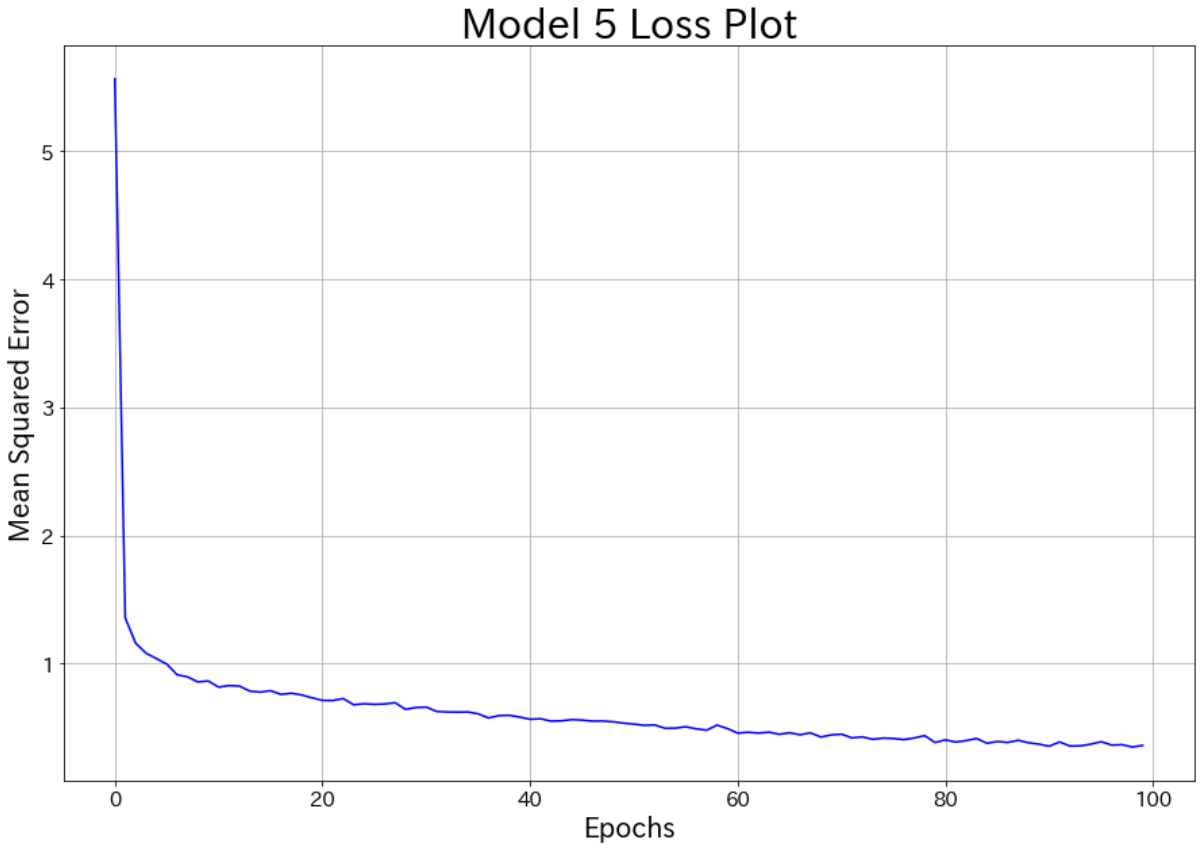
6.7　モデルの選定

Deep Learning はその学習過程で学習回数毎のモデルを生成する。例えば学習を 100 回行ったとすると学習回数毎に 100 個の異なるモデルが生成される。

この場合、必ずしも学習回数が多いモデルが最適であるとは限らない。学習回数を重ねるとその分、学習に使用したデータに対するモデルの当てはまりはよくなる場合が多いが、これは必ずしも学習に使用したデータ以外のデータに対するモデルの当てはまりの度合いとは比例しない。

図表 6.9 はモデルの学習過程における誤差関数の値の推移を示している。横軸は学習回数を示しており、学習が進むごとに誤差関数の値が減少しある一定の値の付近に収束していることが分かる。このモデルにおいても学習回数の最大値のモデルが未学習のデータに最適な振舞いをするとは限らず、ある一定の学習回数後は学習用データに対して過度に適合している可能性も考えられる。

図表 6.9　モデルの学習における誤差関数の推移(中心的ルートモデル)

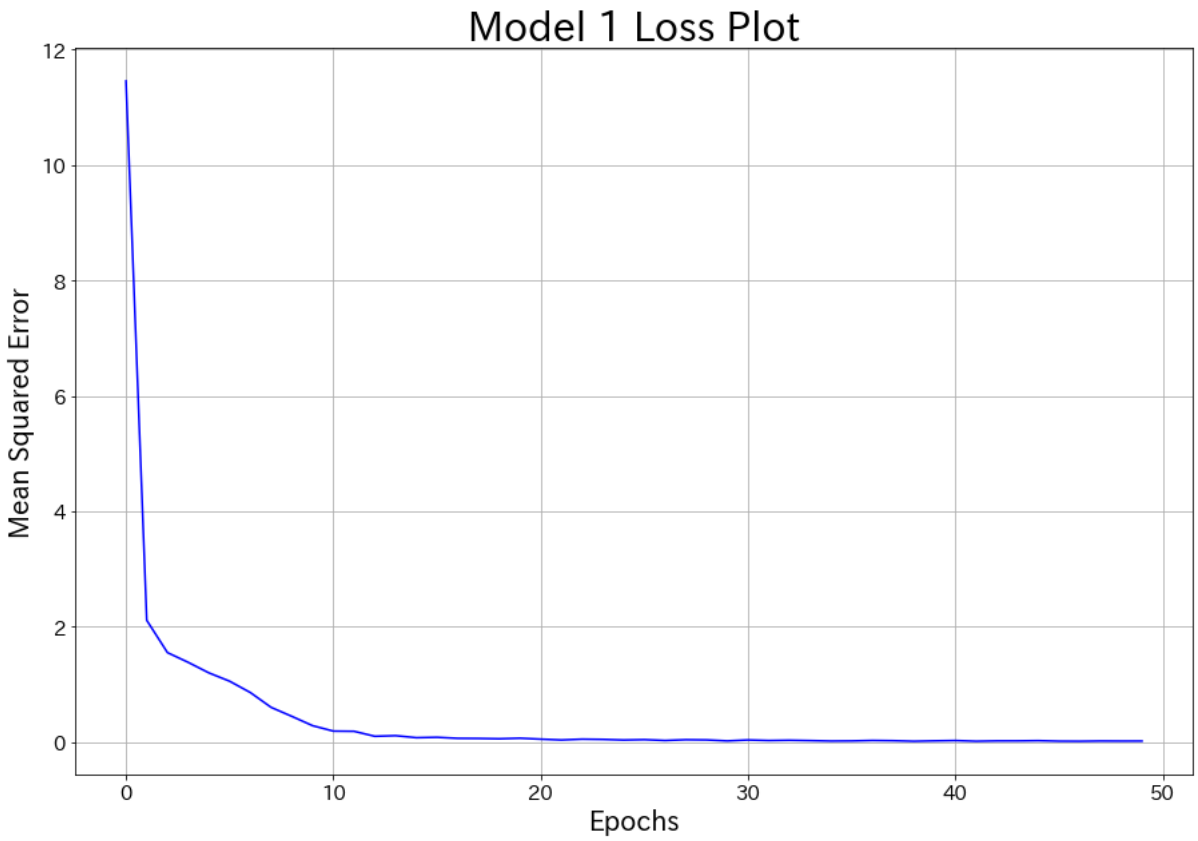


これは過学習と呼ばれる現象であり、機械学習の分野における重要な問題の一つであり本研究においても対策が必要である。

第 1 のこの問題に対する対策は学習回数の制限である。学習回数が多すぎる場合は過学習に陥る可能性が高く、学習しすぎないように周辺的ルートモデル、中心的モデル双方に学習回数の制限を設定した。

周辺的ルートモデルは中心的ルートモデルより誤差関数の収束が早く、学習回数 20 前後で安定しているように見えるので（図表 6.10）学習上限を 50 回、中心的ルートモデルは 70 回前後で安定しているように見えるので（図表 6.9）学習上限を 100 回に設定した。

図表 6.10　モデルの学習における誤差関数の推移(周辺的ルートモデル)

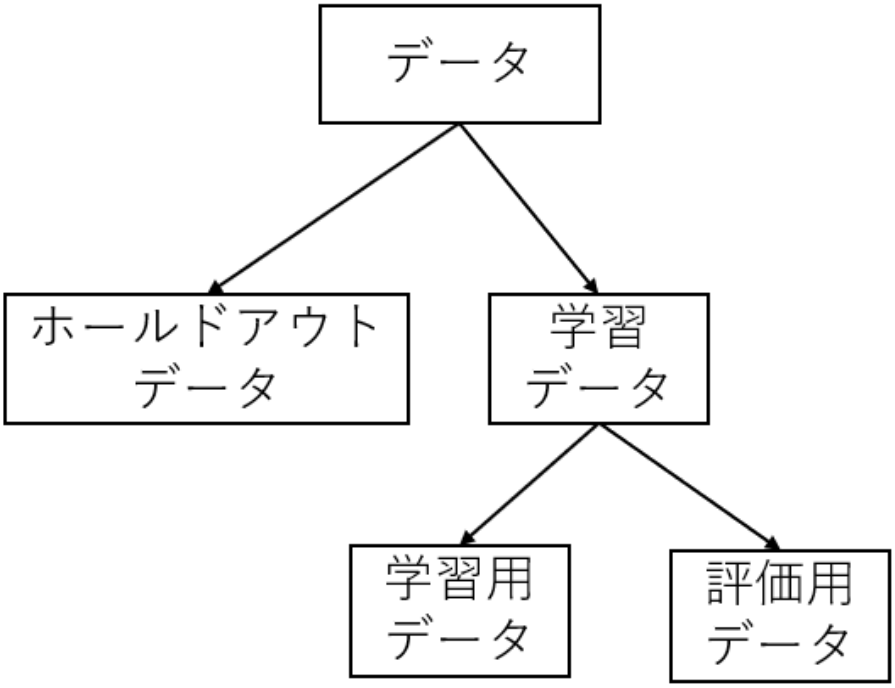


第 2 の対策として交差検証法のアルゴリズムを利用した（Geisser 1975)。この方法は機械学習の分野におけるモデル選定の方法としてよく使用される方法である。

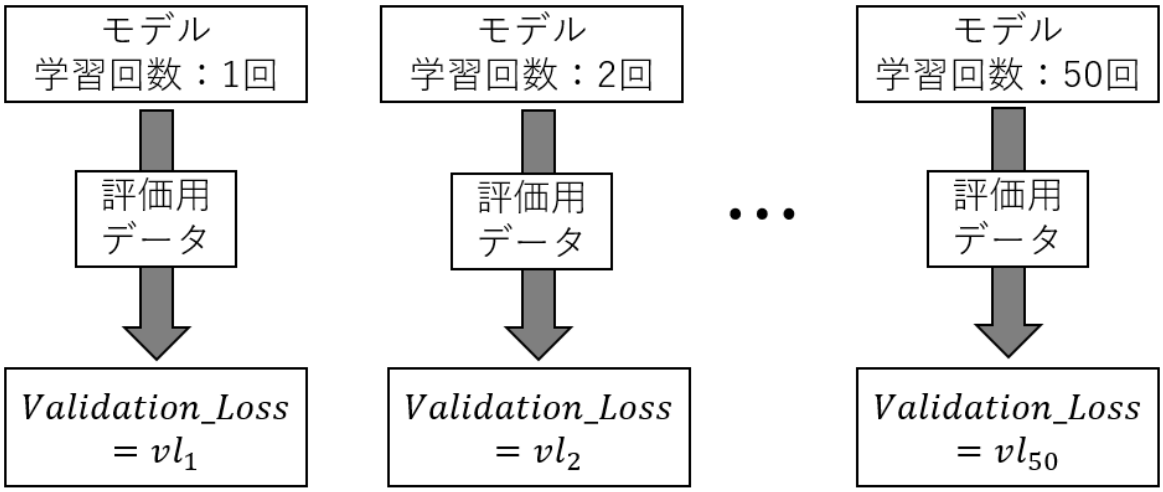
交差検証法とは学習データを学習用データと評価用データにさらに分割し、学習回数毎にその評価データを用いて予測精度を計算することでモデルの評価を行う手法である。つまり、学習毎に未学習データによる評価を行うことで過学習に陥っているかどうかを評価し、モデルの汎化性能を補償しようとする方法である。

本研究においてもモデルの選定にこのアルゴリズムを取り入れる。本研究ではまず学習データを学習用データと評価用データに分割する（図表 6.11)。そして、学習毎に評価用データに対する MSE を計算する。つまり、モデルの最大学習回数が 50 回とするとモデル 1 つに対して学習毎に評価用データを適用した 50 個の MSE が計算される。このように学習が所定の回数終了したモデルのみではなく、学習途中のモデルに関しても評価用データを適用して計算される MSE はモデルが過学習に陥らないための監視数値の役割を果たすことになる。モデルxにおいて、この評価用データから学習回数 i 回目に算出される MSE を vl_x^i （Validation Loss の意）と表現した場合、この過程を図で表現すると図表 6.12 のように表現できる。

図表 6.11　データ分割の過程



図表 6.12　Validation Loss の算出過程



本研究ではモデル毎に上記の過程によって算出された Validation Loss の値が最小の学習回数のモデルをそのモデルにおける最適モデルとして採用した。

このモデル選定のアルゴリズムを数式で示すと以下の式によって示すことができる。

$$idx = \underset{i}{\operatorname{argmin}}\left(vl_x^i\right) \tag{6.5}$$

$$model_x^* = model_x^{idx} \tag{6.6}$$

vl_x^i はモデル番号 x、学習回数 i 回目のモデルに評価用データを適用した際の *MSE* の値を示しており、 idx は学習回数を示している。

モデル選定の際は、まず第 1 に式(6.5)によって *Validation Loss* が最小である学習回数の数値を取得する。式(6.6)の$model^*$は採用されるモデル、$model$の下付き文字はモデルの番号、上付き文字はモデルの学習回数を示しており、この式によって *Validation Loss* が最小であるモデルが採用される。

6.8　モデルに関する仮説に対する評価

モデルの評価はホールドアウトデータに対する MSE を計算し、それを評価値として行った。ここで、6.6 節にて論じた仮説を再度確認する。

$$\text{仮説 1}：MSE_p^P < MSE_p^C \tag{6.7}$$

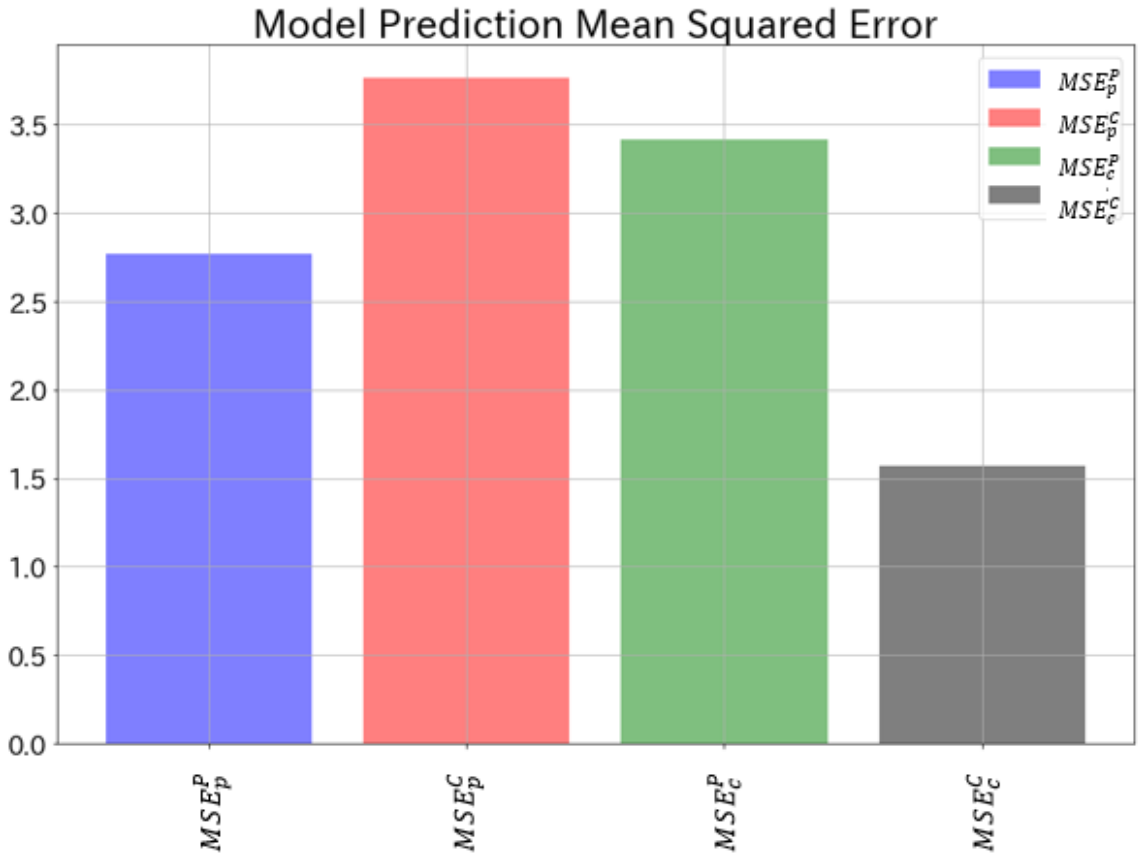
$$\text{仮説 2}：MSE_c^P > MSE_c^C \tag{6.8}$$

ここで考えられる仮説はモデルが周辺的ルート、中心的ルートをそれぞれ反映することができていれば、それぞれのルートに該当するホールドアウトデータを適用した MSE の値はそうでない場合の値より小さくなるという仮説である。つまり、MSE_p^P は MSE_p^C より、MSE_c^C は MSE_c^P よりそれぞれ値が低くなるということが考えられる。

式(6.7)は周辺的ルートモデルに関する仮説を示している。これはつまり周辺的ルートを反映したモデルを用いて周辺的ルートデータに分類される車種のホールドアウトデータを予測する場合は中心的ルートデータに分類される車種のホールドアウトデータを予測する場合より精度が良くなるということを意味している。もしこの仮説が成立すれば、モデルは正しく周辺的ルートによって情報処理を行い態度を形成する消費者の情報処理過程を表現できていると考える。なお中心的ルートモデルに関しても同様に考える（式(6.8)）。

図表 6.13 は周辺的ルートモデル、中心的ルートモデルの各学習後モデルに各ホールドアウトデータを適用した際の MSE の値を表している。

図表 6.13　モデルにホールドアウトデータを適用した際の MSE の値



図表 6.13 のグラフからもわかるように MSE を比較してみると$MSE_p^P < MSE_p^C$、$MSE_c^P > MSE_c^C$が成立しており仮説通りの結果となった。

これはつまり、Deep Learning モデルに精緻化見込みモデルにおける周辺的ルート、中心的ルートそれぞれをモデルに反映することができていると考えられる。

この結果は Deep Learning モデルに精緻化見込みモデルのような理論体系を構造として反映することの有効性を示しており、今まで Deep Learning モデルが使用されていたようにただ単に探索的にモデルを構築していく方法とは一線を画している。

6.9　Bootstrap 法を用いた検証

次にこの精緻化見込みモデルを Deep Learning モデルに反映することによって得られた結果、つまり仮説が統計的に有意な差を有しているかの検証を行っていく。

6.9.1　Bootstrap 法とは

Bootstrap 法とは再標本化とモンテカルロ法を用いて行う統計手法であり、分布の推定や仮説検証に用いられる統計手法である（Efron 1979, Efron and Tibshirani1993)。本研究では Bootstrap 法を用いて経験的に標準誤差を推定することで検定に用いる数値を算出した。

以下に本研究で行う Bootstrap 法についての説明を行う。本研究では 6.8 節で得られた *MSE* の差が統計的に有意なものであるかどうかの検証を行う為に Bootstrap 法を用いる。

この検定をするためには複数の *MSE* サンプルから標準誤差を算出する必要がある。複数の *MSE* サンプルを得るためには複数の学習データとホールドアウトデータが必要になるが、本研究の場合、対象が車であり複数の *MSE* を算出するために、複数の異なるデータ得ることは困難である。そこで本研究では Bootstrap 法を用いることで複数のデータを疑似的に生成し、その生成したデータでモデルを構築し複数の *MSE* サンプルを生成した。そしてその *MSE* サンプルから標準誤差を算出し、6.8 節で示した各 *MSE* 間に有意な差が存在するかどうかを検証した。

6.9.2　MSE に対する Bootstrap 法の適用

この *MSE* サンプルを生成する手順は以下に記載する手順によって作成する。

まず複数のデータを疑似的に生成する方法について述べる。始めに学習データの数と同数の自然数の乱数を重複ありの条件下で発生させる。この乱数の範囲は[1~学習データ数]である。

次にこの乱数をデータのインデックスとして使用し、学習データから疑似的に異なる学習データを生成する。例えば、学習データ数が 5 つの場合は図表 6.14 のような過程を経て疑似サンプルが生成される。学習データにはそれぞれのデータに対応するインデックス番号があるとする（図表 6.14：データ番号)。その番号の範囲で自然数の乱数を重複ありでデータの数と同数発生させ（図表 6.14：乱数)、その発生させた乱数に該当するインデックス番号のデータが復元抽出され、新しい学習データとして生成される(図表 6.14：右)。

図表 6.14　ブーストラップ法を用いたデータ生成の過程

データ番号	データ
1	$Data_1$
2	$Data_2$
3	$Data_3$
4	$Data_4$
5	$Data_5$

乱数
4
4
1
3
5

データ番号	データ
4	$Data_4$
4	$Data_4$
1	$Data_1$
3	$Data_3$
5	$Data_5$

次に複数の MSE サンプルの算出方法について述べる。

上記の過程を経て生成された学習データによってモデルの学習を行い、モデル選定のルールに則りモデルを選定する。この過程を N 回繰り返すことで N 個の学習データと N 個のモデルが生成され、この N 個のモデルに各ホールドアウトデータを適用することで複数の MSE サンプルを生成した。

次にこの算出した MSE 群から標準誤差を算出し、z 検定を行う。z 検定とは各数値の平均値と分散、サンプル数を用いて z 値を算出し、数値の平均値が任意の値であることを帰無仮説に設定して行う仮説検定である。

z 検定に用いる z 値と呼ばれる統計量は以下の式によって算出される。この式は MSE_p^P, MSE_p^C の間に有意差が存在しない、すなわち差が 0 であることを帰無仮説に設定した場合の数値である。

$$z = \frac{MSE_p^P - MSE_p^C}{\sqrt{\left(\frac{var.boot.MSE_p^P}{BST-1} + \frac{var.boot.MSE_p^C}{BST-1}\right)}} \tag{6.6}$$

$$var.boot.MSE_p^P = \frac{1}{BST}\sum_{i=1}^{BST}\left(boot_i.MSE_p^P - mean.boot.MSE_p^P\right)^2 \tag{6.7}$$

$$mean.boot.MSE_p^P = \frac{1}{BST}\sum_{i=1}^{BST} boot_i.MSE_p^P \tag{6.8}$$

式(6.6)の $var.boot.MSE_p^P$ は Bootstrap によって算出した MSE の分散であり、BSTは Bootstrap 法を適用した回数となる。$boot_i.MSE_p^P$ は i 回目の Bootstrap によって算出されたMSE_p^P を示し、$mean.boot.MSE_p^P$ は Bootstrap の全試行回数の MSE_p^P の平均値を示す。今回は Bootstrap 法を 100 回適用したので $BST = 100$である。

6.9.3 Bootstrap の結果

Bootstrap 法を用いて経験的に MSE の標準誤差を推定し、z 検定を行った結果は、図表 6.15 のようになった。

図表 6.15 各 MSE に Bootstrap 法を適用した際の z 値

z 値	MSE_p^C	MSE_c^C
MSE_p^P	-21.84	
MSE_c^P		12.17

この図表は各 MSE 間に z 検定を行った場合の z 値を表示しており、左上のセルは $\boldsymbol{MSE_p^P}$ と $\boldsymbol{MSE_p^C}$ に対して、右下のセルは $\boldsymbol{MSE_c^P}$ と $\boldsymbol{MSE_c^C}$ に対して両者の差が 0 であることを帰無仮説に設定して z 検定を行った結果である。

図表からわかるように両者の間には有意差が存在することが確認でき、有意に $MSE_p^P < MSE_p^C$、$MSE_c^P > MSE_c^C$が成立していることが分かる。

この結果は有意に各モデル間に差が存在することを示しており、本研究のモデルは精緻化見込みモデルによる周辺的ルートと中心的ルートの違いを確かに反映することができていると考えられる。

7. 補償型ヒューリスティクスモデル構築と評価

7.1 補償型ヒューリスティクスの表現

補償型ヒューリスティクスはある情報の評価値が小さい場合に、その他の情報の評価値や重要性が高ければ、その高い評価値の情報によって、低い評価値の情報が補われて総合的な態度形成を行うことを指す。

態度が出力される際、情報 i の評価値を Ev_i 、その情報に掛かる出力部の重みを w_iとすると本モデルでは以下の式(7.1)で表すことができる。

$$out = \sum Ev_i w_i \tag{7.1}$$

補償型ヒューリスティクスを表現したモデルを構築するためにはこの評価値に掛かる重みを任意に操作する必要がある。例えばある情報 A が他の情報を補償する役割を担う場合は、この評価値に掛かる重み w_A を従来より大きくする必要がある。

そこで本研究では式(7.1)で出力がなされるモデルを基準モデルとして置き、この重みに任意の操作を加えることでモデルに補償型ヒューリスティクスを反映した。

その任意の操作とは出力部に評価値を入力する際にその評価値に任意の値を掛ける操作を指す。例えば、情報 A の評価値を τ 倍して出力部に入力すれば、以下の式(7.2)のように間接的にその重みを τ 倍していることになる。

$$out = \begin{cases} \sum(\tau Ev_i)w_i = \sum Ev_i(\tau w_i), & i = A \\ \sum(Ev_i)w_i, & i \neq A \end{cases} \tag{7.2}$$

式(7.2)のように基本モデルの場合は $\tau = 1$ であるが補償型を想定したい場合は、補償の役割を担わせたい情報に対する評価値をτ 倍 $(\tau > 1)$ すれば間接的にその評価値に掛かる重みをτ 倍することができ、補償型ヒューリスティクスを仮定したモデルを作成できる。本研究ではこの構造を LAMBDA 層と呼ばれる Python における LAMBDA 関数と同等の働きをする層を用いて実現した。

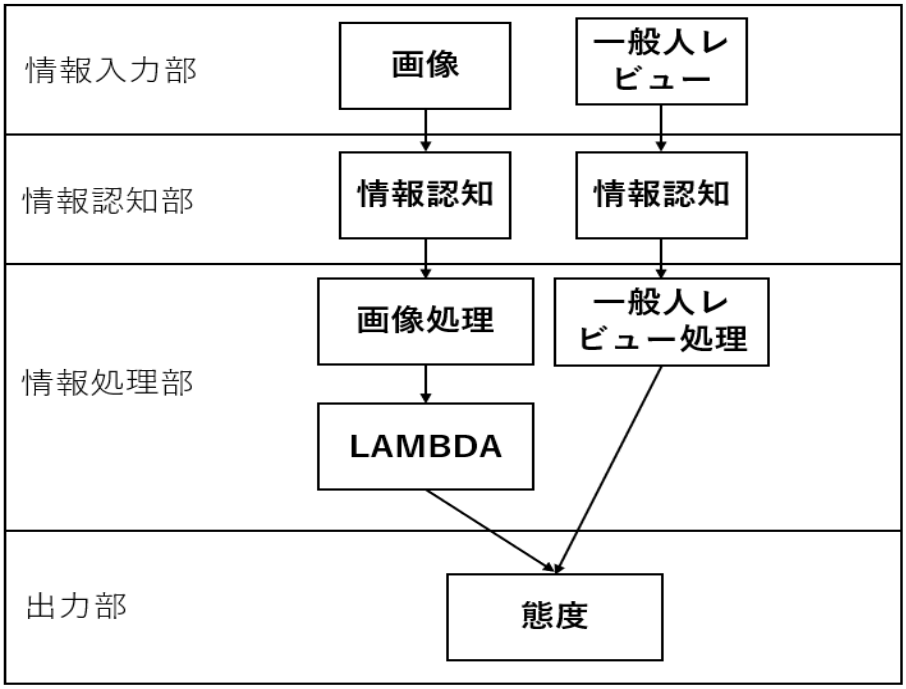
詳しくは後述するが、ここでいう出力部の重みは情報の評価値に対する信念、すなわちどの程度その情報を重視しているかを示す一種のヒューリスティクスを示していると考えることができる。このように本研究では任意の情報に対する評価値を τ 倍することで補償型ヒューリスティクスの表現を行った。なお本研究では $\tau = 2$としている。

7.2　補償型のモデルパターン

補償型を Deep Learning モデルに組み込むと全部でモデルのパターンは 36 パターン存在することになる。周辺的ルートモデルの場合、補償の役割を果たす情報として指定できる情報は「画像」と「一般人レビュー」の 2 つであり、それらの評価値に補償の役割を想定するか否かを考慮すると$2^2 = 4$通りの周辺的ルートモデルが構築できる。同様に中心的ルートモデルの場合は「メーカー」、「コスト」、「パワー」、「サイズ」、「利便性」の 5 つのグループ化された変数に適応するか否かを設定することができ$2^5 = 32$通りの中心的ルートモデルが構築できる。それらのモデルをまとめると図表 7.2 のようになる。図表 7.2 の行はモデルの番号、列は補償型に指定する情報を示す。セル内の数値 1 は補償型を適用する、すなわち評価値を 2 倍にすることを意味している。モデル番号の列は前部のアルファベットがモデルの種類を、後部の数字がモデルの番号を示している。アルファベットは P が周辺的ルートモデルであること、C が周辺的ルートモデルであることを示しており、例えば C2 は中心的ルートモデルの 2 番目のモデルであることを示している。

補償型ヒューリスティクスを考慮したモデルの例を図表 7.1 に示す。このモデルは P2 であり、画像が補償の役割を果たしているモデルとなる。

図表 7.1　P2 の構造



図表 7.2　補償型ヒューリスティクスを仮定したモデルのパターン

モデル番号	レビュー	画像	メーカー	コスト	パワー	サイズ	利便性
P1	0	0					
P2	1	0					
P3	0	1					
P4	1	1					
C1			0	0	0	0	0
C2			1	0	0	0	0
C3			0	1	0	0	0
C4			0	0	1	0	0
C5			0	0	0	1	0
C6			0	0	0	0	1
C7			1	1	0	0	0
C8			1	0	1	0	0
C9			1	0	0	1	0
C10			1	0	0	0	1
C11			0	1	1	0	0
C12			0	1	0	1	0
C13			0	1	0	0	1
C14			0	0	1	1	0
C15			0	0	1	0	1
C16			0	0	0	1	1
C17			1	1	1	0	0
C18			1	1	0	1	0
C19			1	1	0	0	1
C20			1	0	1	1	0
C21			1	0	1	0	1
C22			1	0	0	1	1
C23			0	1	1	1	0
C24			0	1	1	0	1
C25			0	1	0	1	1
C26			0	0	1	1	1
C27			1	1	1	1	0
C28			1	1	1	0	1
C29			1	1	0	1	1
C30			1	0	1	1	1
C31			0	1	1	1	1
C32			1	1	1	1	1

7.3　モデルの評価

モデルの評価は 6 章と同様の方法で行った。各モデルの学習と選定を行い、ホールドアウトデータを適用して *MSE* を計算した結果は以下の図表 7.3 のようになった。

図表 7.3 は後述する出力部の重みと各モデルによって算出された *MSE* の値を示したものである。

この図表からもわかるようにすべての周辺的ルートモデルにおいて$MSE_p^P < MSE_p^C$が成立しており、補償型ヒューリスティクスを考慮した場合の全ての周辺的ルートモデルにおいて、情報処理の構造を Deep Learning モデルに反映していると考えることができる。

また、同様に図表からすべての中心的ルートモデルのすべてのモデルにおいても、$MSE_c^C < MSE_p^P$が成立していることが分かる。これは周辺的ルートモデルと同様に中心的ルートモデルにおいても、情報処理の構造を Deep Learning モデルに反映していると考えることができる。

以上から 6 章と同様に補償型ヒューリスティクスを考慮した場合でも、周辺的ルート、中心的ルートの情報処理過程は各モデルに正しく反映することができていると考えられる。

また図表 7.3 の *MSE* を見ると、周辺的ルートモデルでは P4 が、中心的ルートモデルでは C3 がそれぞれ最も良い精度となっていることが分かる。

周辺的ルートモデルの P4 は画像情報とレビュー情報が相互に補償関係にあるモデルである。この結果から周辺的ルートによって態度形成を行う消費者は、画像情報、レビュー情報の双方の情報が補償関係にある情報処理を行っていると考えることが尤もらしいといえる。

中心的ルートモデルの C3 はコスト情報が他の情報に対して補償関係にあるモデルである。この結果から中心的ルートによって態度形成を行う消費者は、コスト情報がその他の情報を補償関係にある情報処理を行っていると考えることが尤もらしいといえる。

また各モデルの *MSE* を比較してみるとそこにも差があることが確認できる。これは精緻化見込みモデルの時と同様に、補償型ヒューリスティクスの理論体系を構造として Deep Learning モデルに組み込むことでそれぞれの差異をモデルが確かに表現していると考えることができ、ここからも Deep Learning モデルに理論体系を構造として反映することの有効性を示していると考えられる。

7.4　出力部の重みの考察

次に評価値に掛かる重みの考察を行う。評価値に掛かる重みとは情報処理部から算出された評価値に掛かる出力部における全結合層の重み行列のことであり 7.1 節で論じたように補償型ヒューリスティクスを表している部分である。

この重みは 6.1.4 項、7.1 節で示したように多属性態度モデルと酷似した構造になっている。7.1 節でこの重みをヒューリスティクスであるとして論じたが、この重みがどのようにヒューリスティクスを表しているか、多属性態度モデルの理論体系を踏まえながら今一度ここでその説明と重みに対する解釈を行う。

情報処理部から出力される評価値は出力部にある重みが掛けられ、その積和によって出力がなされる。情報 i の評価値に掛かる出力部の重みを w_i 、情報 i の評価値を Ev_i 、出力値を out とすると出力は以下の式によって表現される。

$$out = \sum_{i=1}^{N} Ev_i w_i \tag{7.3}$$

これは多属性態度モデルの式と酷似したものである。多属性態度モデルとは、社会心理学における Rosenberg の研究（Rosenberg 1956）や、対象に対する態度は属性に対する評価値と、対象がその属性を持つとする信念の強さによってあらわされるとする Fishbein の研究に基づいて開発された（Fishbein 1963）。フィッシュバイン型と呼ばれるモデルは対象に対する態度を式(7.3)で定義している。

$$A_j = \sum_{i=1}^{n} a_i b_{ij} \tag{7.4}$$

式(7.4)における A_j は対象 j に対する全体的態度、a_i は属性 i の評価値、b_{ij} は対象 j が属性 i を有することに対する信念の強さ、n は属性の数を示している。

式(7.3)と式(7.4)を比較すると明らかなように本研究で作成したモデルの出力部は式(7.4)に示されるフィッシュバイン型モデルと酷似している。Fishbein はこの信念を消費者が持つ主観的な対象物と価値や概念、属性に対して持つ確率的な関係性と定義しており、さらに人はこうした信念を持つがゆえに、特定の属性を有することや特定の行動をすることで特定の結果につながることを主観的に考えていると示している（Fishbein and Ajzen 1975）。すなわちこれは消費者が経験的に有する情報に対するヒューリスティクスであると解釈することができると本研究では考えた。

本研究では補償型ヒューリスティクスを表現する為に、この式を以下のように変換

した。

$$out = \sum_{i=1}^{N} \tau_i E v_i w_i \tag{7.5}$$

式(7.5)の　τ_i　は LAMBDA 層において評価値を任意倍する際の係数であり、本研究では $\tau_i = 1 \ or \ 2$ である。

LAMBDA 層の中では評価値を 2 倍しているが、7.1 節で説明したものと同様に式(7.5)を見るとこれは間接的に重みを 2 倍しているとみなすことも可能である。

評価値に掛かる重み、すなわち w_i は Deep Learning モデルの学習過程で決定される。本研究で用いたモデルは出力層で評価値と重みの積和によって態度が算出されるのでここの重みは評価値に対する信念であると解釈することができると考えた。

この重み、すなわち信念は評価値をどれだけ重視しているかと捉えることができ、この重視の傾向は情報に対するヒューリスティクスであると本研究では考えている。

この評価値に掛かる重みの数値は、図表 7.3 のようになった。

図表 7.3　モデル毎の評価値に掛かる重みの値と MSE の値

モデル番号	レビュー	画像	メーカー	コスト	パワー	サイズ	利便性	MSE_p^P	MSE_p^C	MSE_c^P	MSE_c^C
P1	-0.003	-1.188						2.77	3.76		
P2	-0.574	0.907						2.84	3.06		
P3	-0.66	0.717						2.76	3.26		
P4	1.046	1.386						2.55	3.01		
C1			0.138	-0.895	0.318	-0.286	0.328			3.41	1.57
C2			-0.812	-0.403	0.151	-0.274	-0.563			3.86	1.7
C3			0.683	-0.723	0.92	-0.055	-1.008			3.65	1.24
C4			0.743	-0.177	-0.128	-1.011	-0.579			3.58	1.6
C5			0.973	-0.27	0.153	-0.546	-0.957			4.09	1.78
C6			-0.581	-0.367	-0.942	0.319	-0.465			4.02	1.84
C7			0.958	0.269	0.954	-0.178	-0.692			4	1.42
C8			0.817	0.134	-0.169	0.364	-0.664			3.19	1.37
C9			0.716	0.424	0.513	0.495	0.868			3.27	1.61
C10			0.948	0.601	0.929	0.381	0.656			3.6	1.49
C11			0.529	0.375	-0.195	0.9	-0.662			3.28	2.01
C12			-0.169	0.631	-0.063	-0.552	-0.494			3.45	1.42
C13			0.587	0.478	0.986	0.768	-0.305			3.29	1.28
C14			0.249	0.808	0.592	0.248	-0.352			3.65	1.36
C15			-0.297	-0.107	0.284	0.486	0.058			3.48	1.74
C16			0.383	-0.339	-0.992	0.707	0.331			3.35	1.36
C17			0.26	0.146	0.601	-0.119	0.612			3.21	1.46
C18			-0.489	0.475	-0.814	0.313	-0.194			3.32	1.62
C19			0.447	0.441	-0.472	-0.304	-0.262			3.81	1.69
C20			0.291	-0.245	-0.138	0.266	0.814			4.12	1.43
C21			0.532	-0.086	0.153	-0.235	-0.976			2.77	1.47
C22			-0.748	0.764	0.708	-0.885	-0.878			3.37	1.6
C23			-0.985	0.256	0.847	0.319	0.263			3.12	1.38
C24			-0.629	-0.836	0.977	0.544	-0.226			3.89	1.66
C25			0.147	-0.943	0.143	-0.421	-0.259			4.47	1.72
C26			0.989	-0.286	0.873	-0.468	-0.642			3.77	1.92
C27			-0.505	0.251	-0.915	0.673	-0.857			3.38	1.79
C28			0.908	0.795	-0.135	0.746	-0.984			3.59	1.58
C29			0.205	-0.624	-0.769	-0.971	-0.991			3.12	1.56
C30			-0.397	0.133	0.446	0.215	-0.181			3.17	1.49
C31			-0.101	-0.431	-1.01	0.485	-0.377			3.21	1.48
C32			0.42	-0.142	0.549	0.671	0.316			3.27	1.59

周辺的ルートモデル、中心的ルートモデルで最も精度の良かったモデルはそれぞれP4、C3であった。ここではこの2モデルについての考察を行う。

周辺的ルートモデルであるP4は画像、一般人レビュー双方の評価値を2倍して出力部に入力しているモデルであり、相互が補償関係にあることを想定している。それぞれの評価値に掛かる重みを見てみると、画像の評価値に掛かる重みは一般人レビューに掛かる重みの約1.3倍の重みがあり、消費者の信念の強さに置き換えて考えると一般人レビューは画像の1.3倍の評価を行っていると考えることができる。これはヒューリスティクスの程度を示していると考えられる。また、バイアスに掛かる重みは0.06と非常に小さくなっており、画像と一般人レビューからの情報によってほとんどの態度形成がなされていると考えられる。

中心的ルートモデルであるC3はコストの評価値を2倍にして出力部に入力しているモデルである。つまり消費者はコスト情報を他の情報に対して補償の役割を担っている、モデルを想定している。重みを見てみるとメーカーはコストの約-0.47 倍、パワーはコストの約-0.63倍、サイズは約-0.04倍、利便性は約0.7倍となっている。コストを基準とすると上記のような程度でヒューリスティクスが働いていると考えられる。またこのモデルにおいてもバイアスに掛かる重みは 0.04 と非常に小さく、中心的ルートモデルから入力された 5 変数でほとんどの態度形成がなされていると考えらえる。

図表 7.4　P4の重み

画像	一般人レビュー	バイアス
1.05×2	1.39×2	0.07

図表 7.5　C3の重み

メーカー	コスト	パワー	サイズ	利便性	バイアス
0.68	-0.72×2	0.92	-0.06	-1.01	0.04

7.5　Bootstrap 法の適用

第 6 章と同様に補償型ヒューリスティクスを反映したモデルに対しても Bootstrap 法による検定を適用した。検定対象は *MSE* と評価値に掛かる重みである。

7.5.1　MSE に対する Bootstrap 法の適用

MSE に対する Bootstrap 法の適用は、第 6 章と全く同様である。

まず、学習データ数分の自然数からなる乱数を発生させ、その乱数もとに第 6 章と同様の方法で、再標本化による学習用のデータセットを作成する。次にその再標本化されたデータによりモデルの学習を行い、各学習回数毎のモデルを保存し、先のモデル選定と同様の方法で選定されたモデルによって *MSE* を計算する。これをモデル毎かつ、設定した Bootstrap の試行回数回繰り返すことで、各モデル毎に Bootstrap の試行回数個の *MSE* が算出される。この *MSE* から標準誤差を算出し、 z 検定に使用した。

算出する *MSE* は周辺的ルートモデルの場合、MSE_p^P, MSE_p^C で、中心的ルートモデルの場合、MSE_c^P, MSE_c^C である。算出された各 *MSE* から標準誤差を算出し、z 検定を行い、7.3 節にある各 *MSE* 間に有意な差が存在するかを検証していく。

まず、周辺的ルートモデルに対して Bootstrap を適用し、z 値を算出した結果を図表 7.6 に示す。

z 検定に用いる z 値は以下の式(7.3)のように計算される。これも 6 章の場合と同様であり、この計算を各モデル毎に行っていく。

式(7.13)の各数値の先頭にあるmdはモデルを示し、下付き文字の番号が *MSE* を計算するモデル番号を示している。式中の$md_{P1}.MSE_p^P$ P1 におけるMSE_p^Pを、$md_{P1}.var.boot.MSE_p^P$ は P1 に対して Bootstrap 法を適用した際に算出されたMSE_p^Pの分散を示している。また同様に$md_{(P1:P1)}.z$は P1 のMSE_p^Pと P1 のMSE_p^Cに Bootstrap 法を適用して算出された z 値を示している。 下付き文字のコロン(:)前の数値は MSE_p^P を計算するモデル番号を、MSE_p^C を計算するモデル番号を示している。例えば式(7.14)は P1 と P2 との間のMSE_p^P, MSE_p^Cに対する z 検定の z 値となる。ここでも同様に Bootstrap の回数は 100 回とした。

$$md_{(P1:P1)}.z = \frac{md_{P1}.MSE_p^P - md_{P1}.MSE_p^C}{\sqrt{\left(\frac{md_{P1}.var.boot.MSE_p^P}{BST-1} + \frac{md_{P1}.var.boot.MSE_p^C}{BST-1}\right)}} \tag{7.13}$$

$$md_{(P1:P2)}.z = \frac{md_{P1}.MSE_p^P - md_{P2}.MSE_p^C}{\sqrt{\left(\frac{md_{P1}.var.boot.MSE_p^P}{BST-1} + \frac{md_{P2}.var.boot.MSE_p^C}{BST-1}\right)}} \tag{7.14}$$

図表 7.6　周辺的ルートモデルの MSE に対する z 値

z 値	$md_{P1}.MSE_p^P$	$md_{P2}.MSE_p^P$	$md_{P3}.MSE_p^P$	$md_{P4}.MSE_p^P$
$md_{P1}.MSE_p^C$	-21.84	-19.58	-20.85	-23.36
$md_{P2}.MSE_p^C$	-6.53	-4.69	-6.30	-9.89
$md_{P3}.MSE_p^C$	-9.98	-8.21	-9.64	-12.80
$md_{P4}.MSE_p^C$	-4.81	-3.20	-4.71	-8.15

図表 7.6 からわかるようにすべてのモデル間におけるMSE_p^P, MSE_p^Cに有意に差が存在する結果となった。次に中心的ルートモデルに対して Bootstrap を適用し、z 値を算出した結果を図表 7.7 に示す。

図表 7.7　中心的ルートモデルの MSE に対する z 値（略図）

z 値	$md_{C1}.MSE_c^P$	$md_{C2}.MSE_c^P$	…	$md_{C31}.MSE_c^P$	$md_{C32}.MSE_c^P$
$md_{C1}.MSE_c^C$	12.17	8.56	…	9.34	12.08
$md_{C2}.MSE_c^C$	11.14	8.04	…	8.52	10.96
⋮			⋮		
$md_{C31}.MSE_c^C$	12.79	8.92	…	9.89	12.75
$md_{C32}.MSE_c^C$	12.01	8.50	…	9.23	11.91

図表 7.7 からわかるように中心的ルートモデルにおいても全ての各MSE_c^P, MSE_c^C間に有意差が存在することが確認できる。

以上の結果から周辺的ルートモデルにおいては、MSE_p^P, MSE_p^C間に、中心的ルートモデルにはMSE_c^P, MSE_c^C間に有意差が存在することが分かった。

これはつまり、補償型ヒューリスティクスを考慮した状態のモデルも 6 章と同様に精緻化見込みモデルの構造を反映することができていると考えられる。

7.5.2　出力部の重みに対する Bootstrap 法の適用

次に評価値に掛かる重みに関する考察を行う。

Bootstrap 法の適用手順はほぼ *MSE* に対するそれと同様の手順を踏む。まず Bootstrap 法によって新たな学習データを生成し、その学習データを用いてモデルの学習を行う。そして、モデルの学習回数毎に対象となる重み、すなわち評価値に掛かる重みをモデルから抽出する。これを設定した Bootstrap 回数行うことで、1 モデルにつき、学習回数回×Bootstrap 回数の重みが抽出される。今回は Bootstrap の回数を 100 回とした。

例えば P1 に適用する場合、まず Bootstrap 法の 1 回目の適用の際、学習回数個の一般人レビューの評価値に掛かる重み、画像の評価値に掛かる重みが抽出される(式(7.15))。

$$[md_{P1}.b_1.wt_{rev}^1, md_{P1}, b_1, wt_{rev}^2, \dots, md_{P1}.b_1.wt_{rev}^{50}] \tag{7.15}$$

式(7.15)の md_{P1} は P1 であることを、b_1 は Bootstrap 回数が 1 回目であることを示しており、wt_{rev}^1 は学習回数 1 回目のモデルにおける、一般人レビューの評価値に掛かる重みを示している。すなわち$md_{P1}.b_1.wt_{rev}^1$ は P1 モデルの Bootstrap 回数が 1 回目、学習回数 1 回目の一般人レビューの評価値に掛かる重みを示している。式(7.15)ではすべての重みが表示されているが、実際の計算は学習回数の初回 20%分をバーンインとして計算の対象外とするため、Bootstrap 法を 100 回、適用することによって生成されるデータは式(7.16)のようなである。

$$\begin{bmatrix} md_{P1}.b_1.wt_{rev}^{10} & \cdots & md_{P1}.b_1.wt_{rev}^{50} \\ \vdots & \ddots & \vdots \\ md_{P1}.b_{100}.wt_{rev}^{10} & \cdots & md_{P1}.b_{100}.wt_{rev}^{50} \end{bmatrix} \tag{7.16}$$

各行は Bootstrap 回数毎に生成された学習回数毎の一般人レビューの評価値に掛かる重みを示している。すなわちこの行列の大きさは 100×40 となる。

MSE の時と同様に、こうして生成されたデータから標準誤差を算出し、z 検定を行うことで、7.4 項で論じた考察の統計的有意性を論じる。

ここでの計算も 7.4 項と同様に各学習回数の 20%をバーンインとし計算の対象外とした。z 値は以下の式(7.17)によって算出される。

$$md_{P1}.z_{rev} = \frac{md_{P1}.mean_{rev} - 0}{\sqrt{\frac{boot.\sigma_{rev}^2}{BST}}} \tag{7.17}$$

式(7.17)の $md_{P1}.z_{rev}$ は P1 の一般人レビューの評価値に掛かる重みに関する z 値であることを示しており、$md_{P1}.mean_{rev}$ は 7.4 項で用いた重みから計算される平均値であり、今回の検定対象となる。$boot.\sigma_{rev}^2$は Bootstrap 法によって算出された一般人レビューの評価値に掛かる重みの分散を示している。それらの数値は以下の式によって計算される。

$$md_{P1}.mean = \frac{1}{n-10}\sum_{i=10}^{50} wt_{rev}^{i} \tag{7.18}$$

$$boot.\sigma_{rev}^2 = \frac{1}{BST}\sum_{i=1}^{BST}(b_i.mean.wt_{rev} - boot.mean.wt_{rev})^2 \tag{7.19}$$

$$b_1.mean.wt_{rev} = \frac{1}{n}\sum_{i=10}^{50} b_1.wt_{rev}^{i} \tag{7.20}$$

$$boot.mean.wt_{rev} = \frac{1}{BST}\sum_{i=1}^{BST} b_i.mean.wt_{rev}^{i} \tag{7.21}$$

このように経験的に推定した標準誤差を用いて z 検定を行うことで、統計的有意性を確認する。

z 検定の結果は図表 7.8 のようになった。

図表 7.8　出力部の重みに対する z 値

モデル番号	レビュー	画像	メーカー	コスト	パワー	サイズ	利便性
P1	-14885.81	-3.20					
P2	1095.44	-5904.29					
P3	546.40	-5486.78					
P4	4100.88	5371.89					
C1			1485.27	-101.92	796.40	-132.25	202.92
C2			-726.65	-86.06	-102.77	-40.42	-261.25
C3			1471.49	-134.61	93.69	-274.45	-601.48
C4			964.37	106.45	-941.11	-266.36	-384.25
C5			978.50	83.65	1035.72	108.00	-633.98
C6			1040.38	104.69	-155.80	-71.68	-308.65
C7			577.36	311.46	317.35	167.17	-400.77
C8			-215.31	280.80	573.83	251.00	-346.85
C9			1072.92	243.55	-88.06	407.77	307.53
C10			212.42	91.99	-47.01	421.40	311.24
C11			-537.52	750.74	272.81	-243.78	-386.90
C12			554.36	-29.13	557.98	462.12	-344.01
C13			302.77	-178.23	164.52	76.18	-128.78
C14			-962.53	29.49	-955.40	256.00	-201.92
C15			358.87	387.41	474.89	427.74	51.89
C16			376.18	164.83	-677.24	-54.12	133.84
C17			928.35	-107.92	-282.28	169.68	423.86
C18			-1026.43	-44.08	-51.83	-131.11	-103.69
C19			-1956.63	359.09	79.24	58.07	-211.53
C20			-1421.26	98.27	567.60	-103.13	445.20
C21			227.95	-625.66	572.47	-267.56	-548.40
C22			2298.19	-247.49	1039.05	149.55	-421.17
C23			-731.35	-126.06	101.92	339.55	98.57
C24			1409.14	139.52	358.29	-250.86	-123.68
C25			512.92	277.19	-598.84	-254.61	-197.87
C26			-844.17	-207.40	-50.16	506.72	-232.24
C27			-152.59	48.03	-425.69	324.10	-607.74
C28			226.37	-198.49	412.55	-638.33	-720.88
C29			-1485.81	-76.37	-826.53	61.81	-415.57
C30			-668.21	133.53	309.45	125.93	-146.76
C31			-1381.81	79.75	-233.41	393.77	-185.46
C32			-338.95	489.87	-98.45	93.74	210.34

z 値の結果を見るとすべての評価値に掛かる重みは 0 であるという仮説を棄却しており、すべての情報に対する評価値にヒューリスティクスの程度を仮定することができると考えた。つまり 7.4 節で行った評価値に掛かる重みからヒューリスティクスを解釈する試みは有意に有効であると考えることができる。

8. 課題とまとめ

8.1 課題

本研究の課題として下記の 4 点を挙げる。第 1 の課題は個人特性の反映である。本研究は Web サイトからのデータ収集を行い、分析を行った。そのため、顧客個人ごとのモデルを作成することは不可能であり、周辺的ルート、中心的ルート共にそれらを代表する消費者例として作成せざるを得なかった。しかし、個人ごとのでもグラフィックデータなどを収集することができればさらに多くの変数を用いてより詳細に顧客ごとの個人特性を反映した情報処理構造を構築することが可能となる。そうしたデータを収集することは非常に困難ではあるが、消費者の情報処理を研究するうえで個人特性は無視できるものではなく、今後の課題として個人特性を考慮したモデル構築が 1 つ挙げられる。

第 2 の課題は情報の分類である。本研究では周辺的手掛かり情報に画像と一般人レビューを中心的情報にメーカー、コスト、パワー、サイズ、利便性情報を用いた。しかし、財によってこの情報分類は異なると考えられる。例えば画像情報が中心的情報となる財も考えられる。本研究の枠組みをより汎化させるためには情報分類に関するアルゴリズムや理論が必要となり、これも今後の 1 つの課題として考えられる。

第 3 の課題は Deep Learning モデルそのものを詳細に理解するアプローチである。機械学習の分野でもこの手の研究が近年は進んでおり、Deep Learning モデルの出力を最大化する入力情報を作成し分析する Activation Maximization (Le et al. 2012) や入力に対する感度を分析する Sensitivity Analysis を利用した SMOOTHGRAD (Smilkov et al. 2017)、出力から入力までの経路を逆にたどることで探索的に Deep Learning モデルの分析を行う Deconvolution/LRP (Springenberg et al. 2014) など様々な手法が開発されている。こうした研究を本研究で使用したマーケティングを対象としたモデルに適用し、解釈を行うことができればより詳細に消費者の情報処理構造を解明できる可能性があると考えられ、こうした機械学習分野の先端研究の反映も 1 つの課題として考えられる。

第 4 の課題はパラメーター数の最適化である。本研究では周辺的ルートモデル、中心的ルートモデルのパラメーターをそれぞれのモデルの特徴を考慮したうえで差を持たせる目的で任意に設定した。しかし、本研究で設定したパラメーターが最適であるとは限らない。こうしたハイパーパラメーターの設定に関する議論は機械学習分野においても大きな課題の一つであり、最適化アルゴリズムを反映することができればより正確に消費者の情報処理過程が表現できると考えられる。

8.2 まとめ

本研究では消費者の情報処理過程を Deep Learning とマーケティング理論を用いてモデル化し、そこからマーケティングにとって有益な情報を抽出することができたと筆者は考えている。また、本研究が対象にしたヒューリスティクスは通常の手段では定量化しにくい人間の感覚的な部分の数値であることから、従来のモデルではなく Deep Learning という柔軟性の高い手段をもってしてその部分に対して切り込み、定量的にヒューリスティクスというものを算出し考察する術を提示できたことは本研究が今後のマーケティング研究に貢献できる部分ではないかと筆者は考えている。

情報の多様化が進み、その絶対量が増えている現代では従来の統計モデルによって処理できない種のデータも増えてきている。こう言った現状とモデルの精度面を考えても Deep Learning を用いたモデル化は今後必要不可欠であり、さらには精度のみでなく、本研究のようにそこから重要かつ有用な情報を抽出することが必要不可欠になってくると考えられる。

マーケティングが長い年月と歴史の中で培ってきた理論を近年の先端技術である Deep Learning と融合させることで、以前より高精度かつ詳細に分析を行うことは今後のマーケティングにおいて重要な研究課題であり、本研究がその課題に対するアプローチの一つとして貢献できることを願う。

参考文献

Bengio,Y., Lamblin,P., Popovici,D. and Larochelle (2006). “Greedy layer wise training of deep networks,” *In Proc. NIPS*.

Chollet,F. (2018)『Python と Keras によるディープラーニング』、マイナビ出版。

Davis,P. and Graces,E. (2009). “Quantitative Techniques for Competition and Antitrust Analysis,” *Princeton University Press*.

Efron,B. (1979). “Bootstrap Methods: Another Look at the Jackknife”. *The Annals of Statistics 7 (1): 1–26*.

Efron,B. and Tibshirani,R.J. (1993). *An Introduction to the Bootstrap*. Chapman & Hall/CRC Monographs on Statistics & Applied Probability.

Fishbein, M. (1963) “An investigation of the relationships between beliefs about an object and the attitude toward that object,” *Human Relation. 16 233-240.*

Fishbein, M. and Ajzen, I. (1975). *Belief, attitude, intention and behaviour: An introduction to theory and research.* Reading, MA:Addison-Wesley.

Geisser,S. (1975). “The predictive sample reuse method with applications,” *J. Amer. Statist. Assoc. 70:320–328.*

Glorot,X., Bordes,A. and Bengio.Y. (2011). “Deep sparse rectifier neural networks,” *In Proc. 14th International Conference on Artificial Intelligence and Statistics 315–323.*

Goldberg,Y. and Levy,O. (2014). “word2vec Explained: Deriving Mikolov et al.’s Negative-Sampling Word-Embedding Method,” *arXiv preprint arXiv:1402.3722.*

Hebb,D.O. (1949). *The Organization of Behavior* . Wiley.

Hinton,G.E., Osindero,S. and The,Y. (2006). “A fast learning algorithm for deep belief nets,” *Neural Computation 18:1527-544.*

Hochreiter,S. and Schmidhuber,J. (1997). “LONG SHORT-TERM MEMORY,” *Neural Computation 9(8):1735-1780.*

Kohyama,T. and Fujihara,T. (1992). “Effects of need for cognition and distraction on issuerelevant elaboration and attitude change,” *The Japanese Journal of Experimental Social Psychology. Vol. 31, pp. 222-230.*

Le,Q.V., Ranzato,M., Monga,R., Devin,M., Corrado,G., Chen,K., Dean,J., and Ng,A.Y. (2012). “Building high-level features using large scale unsupervised learning,” *CML'12 Proceedings of the 29th International Coference on International Conference on Machine Learning. 507-514.*

Liu,X., Lee,D., Srinivasan,K. (2017). “Large Scale Cross Category Analysis of Consumer Review Content on Sales Conversion Leveraging Deep Learning,” *NET Institute Working Paper No. 16-09.*

McCulloch,W.S. and Pitts,W. (1943). “A logical calculus of the ideas immanent in

nervous activity," *Bulletin of Mathematical Biophysics, 5(4):115-133.*
Mikolov,T., Chen,K., Corrado,G.S. and Dean,J. (2013). "Efficient Estimation of Word Representations in Vector Space," *ICLR Workshop.*
Goldberg Mikolov,T. and Kombrink,S. (2010). "Recurrent neural network based language model," *INTERSPEECH, 1045:-1048.*
Petty,R.E. and Cacioppo,J.T. (1984). "Source factors and the elaboration likelihood model of persuasion," *Advances in Consumer Research. 11: 668.*
Petty.R.E. and Cacioppo.J.T. (1984). "The effects of involvement on responses to argument quantity and quality : central and peripheral routes to persuasion," *Journal of Personality and Social Psychology. Vol. 46, pp. 69-81.*
Petty,R.E., Schumann,D.W., Richman,S.A. and Strathman,A.J. (1993). "Positive mood and persuasion : different roles for affect under high- and low-elaboration condition," *Journal of Personality and Social Psychology. Vol. 64 pp. 5-20.*
Roberts,J.H. and Lilien,G.L. (1997). 『マーケティングハンドブック』、朝倉書店。
Rosenberg, M.J (1956). "Cognitive structure and attitudinal affect," *Journal of Abnormal and Social Psychology. 53 362-372.*
Rosenblat,F.(1958). "The perceptron: A probabilistic model for information storage and organization in the brain," *Psychological Review 65(6):386-408.*
Rumelhart,D.E. and Mccelland,.J (1986). *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*. MIT Press.
Smilkov,D., Thorat,N. Kim,B., Viégas,F., and Wattenberg.M. (2017). "Smoothgrad: removing noise by adding noise," *arXiv preprint arXiv:1706.03825.*
Springenberg,J.T., Dosovitskiy,A., Brox,T., and RiedmillerM. (2014). "Striving for simplicity: The all convolutional net," *arXiv preprint arXiv:1412.6806.*
Šťastný,J., Turčínek,P., Motyčka,A. "Using Neural Networks for Marketing Research Data Classification," *In International WSEAS Conference on Mathematical Methods and Techniques in Engeneering & Environmental Science. Catania, Italy, WSEAS Press. 252-256. ISBN 978-1-61804-046-6.*
Zaiyong,.T. (2011)."Improving Direct Marketing Profitability with Neural Networks," *International Journal of Computer Applications, 29(5)13-18.*
伊藤誠（2018）『新しい機械学習の教科書』、翔泳社。
巣籠悠輔（2018）『詳解ディープラーニング』、マイナビ出版。
新美潤一郎、星野崇宏（2017）『顧客行動の多様性変数を利用した購買行動の予測』、人工知能学会論文誌、32 巻 2 号 pp.B-G63_1-9。
若林憲人、生田目崇（2015）『Deep Learning を用いたコンテンツマーケティングの評価に関する研究』、経営情報学会 2015 年秋季全国研究発表大会、 セッション ID：J1-3。

Keras HP：https://keras.io/
Python HP：https://www.python.org/
Tensor Flow HP：https://www.tensorflow.org/
乾研究室 HP：http://www.cl.ecei.tohoku.ac.jp/~m-suzuki/jawiki_vector/

謝辞

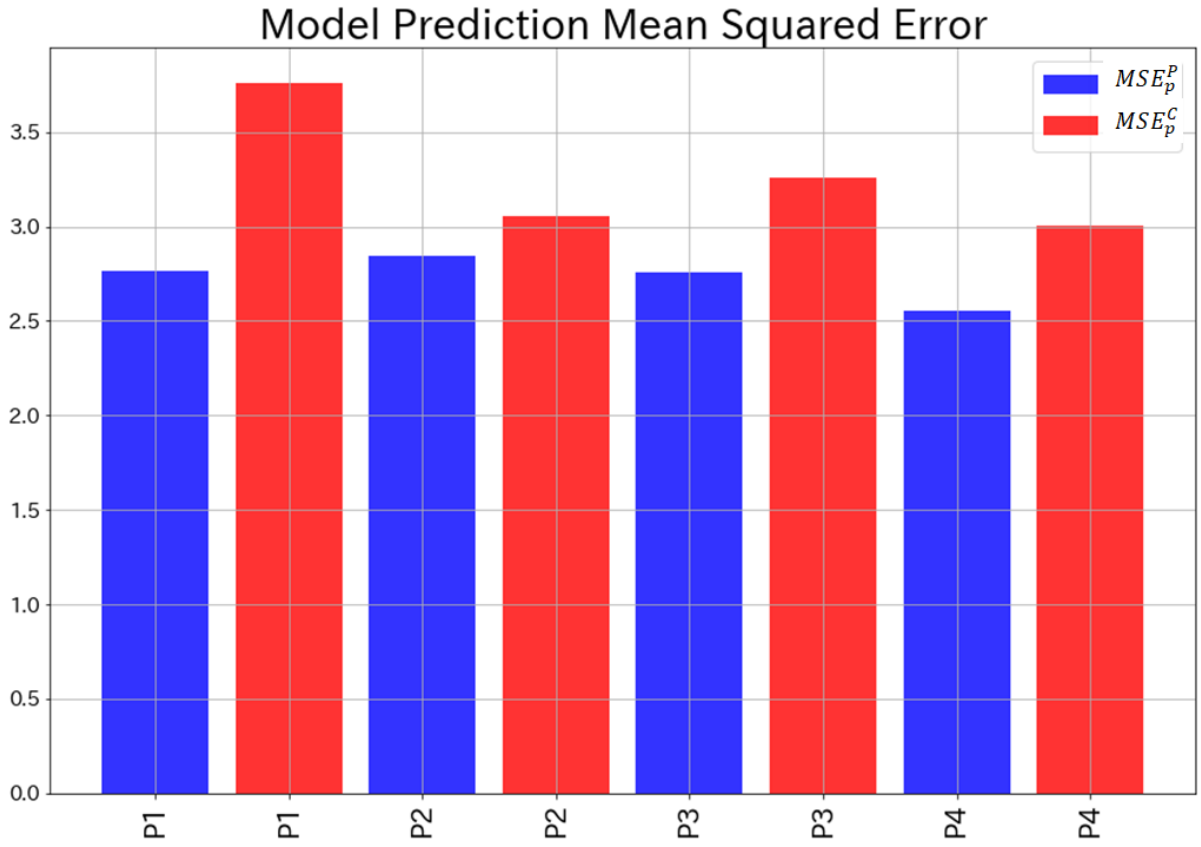
本論文は筆者が慶應義塾大学大学院経営管理研究科修士課程に在籍中の研究成果をまとめたものである。同研究科の井上哲浩教授には主査として論文に関する指導のみでなく、物事に対する考えかたや、姿勢といった思考の根幹となる部分を始め、多くのことをマーケティングを通じて学ばせていただいた。こうした思考の根幹部分を学び自己に還元することができたのは私が同研究科において学んだ最も大きなことであり、井上哲浩教授には感謝してもしきれない。ここに深謝の意を表する。同研究科の余田拓郎教授、林高樹教授には副査としてご助言をいただくとともに、本論文の指導をいただいた。ここに深謝の意を表する。

付録資料

周辺的ルートモデルにおける$\boldsymbol{MSE}$の可視化

7.3 節で論じた周辺的ルートモデルに関する MSE の値を可視化すると以下の図表 9.1 のようになる。青の棒グラフはMSE_p^P、赤の棒グラフはMSE_p^Cを示しており、すべてのモデルにおいて$MSE_p^P < MSE_p^C$が成立していることが分かる。

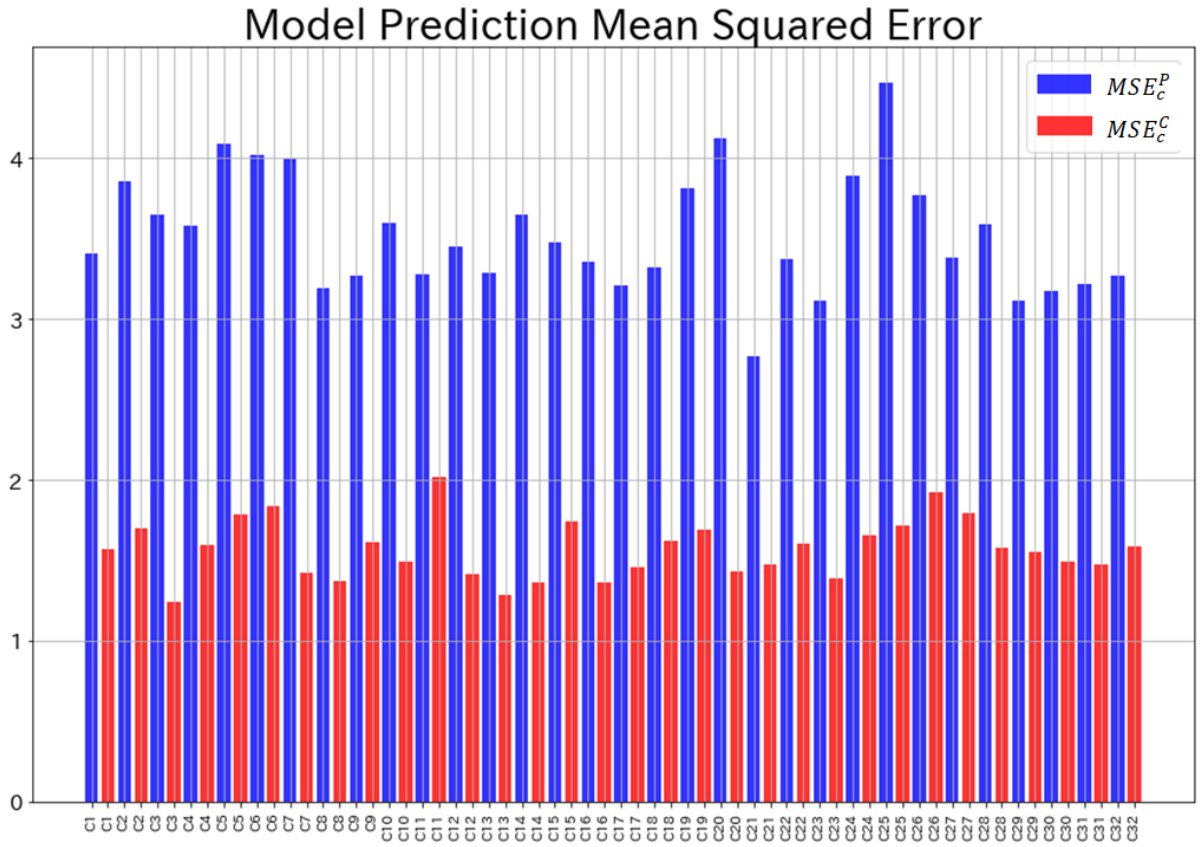
図表 9.1　周辺的ルートモデルにおける$\boldsymbol{MSE}$の可視化



中心的ルートモデルにおけるMSEの可視化

7.3 節で論じた中心的ルートモデルに関する MSE の値を可視化すると以下の図表 9.2 のようになる。青の棒グラフはMSE_c^P、赤の棒グラフはMSE_c^Cを示しており、すべてのモデルにおいて$MSE_c^P > MSE_c^C$が成立していることが分かる。

図表 9.2　中心的ルートモデルにおけるMSEの可視化



MSE_c^P、MSE_c^Cにおける z 値

7.5.1 項における図表 7.7 の全容を示す。この表は各中心的ルートモデル間のMSE_c^PとMSE_c^Cに関する z 検定の値を示している。行はMSE_c^C、列はMSE_c^Pを示しており、各セルが行と列の交差する MSE に関する z 検定の値である。行名、列名のMSEの前部にある mdx はモデル番号が x であることを示している。例えば 1 行 1 列目の 12.17 という値はモデル 5 のMSE_c^CとMSE_c^Pについて z 検定を行った結果であり、有意差があることが確認できる。表はサイズが大きいため、各列を 4 分割し、8 列ずつの記載とした。

図表 9.2 $\boldsymbol{md_{C1}}$~$\boldsymbol{md_{C8}.MSE_c^P}$と$\boldsymbol{MSE_c^C}$における z 値

	$md_{C1}.MSE_c^P$	$md_{C2}.MSE_c^P$	$md_{C3}.MSE_c^P$	$md_{C4}.MSE_c^P$	$md_{C5}.MSE_c^P$	$md_{C6}.MSE_c^P$	$md_{C7}.MSE_c^P$	$md_{C8}.MSE_c^P$
$md_{C1}.MSE_c^C$	12.17	8.56	8.42	11.07	13.31	9.53	17.38	7.04
$md_{C2}.MSE_c^C$	11.14	8.04	7.85	10.25	12.51	8.99	16.15	6.44
$md_{C3}.MSE_c^C$	14.36	9.81	9.76	12.89	15.07	10.82	19.74	8.48
$md_{C4}.MSE_c^C$	11.93	8.45	8.30	10.88	13.13	9.41	17.08	6.91
$md_{C5}.MSE_c^C$	10.80	7.78	7.57	9.92	12.22	8.72	15.90	6.13
$md_{C6}.MSE_c^C$	10.39	7.56	7.33	9.59	11.89	8.49	15.44	5.88
$md_{C7}.MSE_c^C$	13.19	9.14	9.04	11.92	14.13	10.13	18.50	7.71
$md_{C8}.MSE_c^C$	13.57	9.34	9.26	12.22	14.43	10.34	18.93	7.94
$md_{C9}.MSE_c^C$	11.76	8.38	8.21	10.75	13.00	9.33	16.86	6.82
$md_{C10}.MSE_c^C$	12.66	8.85	8.73	11.48	13.71	9.83	17.89	7.37
$md_{C11}.MSE_c^C$	9.19	6.89	6.61	8.59	10.94	7.80	14.12	5.10
$md_{C12}.MSE_c^C$	13.14	9.13	9.03	11.89	14.10	10.12	18.41	7.70
$md_{C13}.MSE_c^C$	14.17	9.67	9.62	12.72	14.90	10.68	19.58	8.32
$md_{C14}.MSE_c^C$	13.55	9.34	9.26	12.22	14.42	10.34	18.89	7.95
$md_{C15}.MSE_c^C$	11.08	7.93	7.73	10.15	12.44	8.88	16.23	6.30
$md_{C16}.MSE_c^C$	13.64	9.37	9.29	12.28	14.48	10.37	19.03	7.97

$md_{C17}.MSE_c^C$	12.94	9.00	8.89	11.71	13.93	9.98	18.21	7.54
$md_{C18}.MSE_c^C$	11.83	8.37	8.21	10.78	13.04	9.33	17.01	6.82
$md_{C19}.MSE_c^C$	11.41	8.13	7.95	10.43	12.71	9.08	16.56	6.54
$md_{C20}.MSE_c^C$	13.07	9.08	8.98	11.82	14.04	10.07	18.35	7.64
$md_{C21}.MSE_c^C$	12.80	8.92	8.81	11.59	13.82	9.91	18.05	7.46
$md_{C22}.MSE_c^C$	11.97	8.45	8.30	10.90	13.16	9.42	17.17	6.91
$md_{C23}.MSE_c^C$	13.40	9.26	9.18	12.09	14.30	10.26	18.70	7.85
$md_{C24}.MSE_c^C$	11.59	8.24	8.07	10.59	12.86	9.20	16.75	6.66
$md_{C25}.MSE_c^C$	11.12	8.00	7.81	10.21	12.48	8.95	16.18	6.39
$md_{C26}.MSE_c^C$	9.78	7.23	6.98	9.09	11.41	8.15	14.74	5.50
$md_{C27}.MSE_c^C$	10.69	7.74	7.52	9.84	12.13	8.67	15.75	6.08
$md_{C28}.MSE_c^C$	12.06	8.53	8.38	10.99	13.24	9.49	17.23	7.00
$md_{C29}.MSE_c^C$	12.23	8.61	8.47	11.13	13.37	9.58	17.42	7.10
$md_{C30}.MSE_c^C$	12.61	8.84	8.71	11.45	13.68	9.81	17.83	7.36
$md_{C31}.MSE_c^C$	12.79	8.92	8.80	11.59	13.81	9.90	18.04	7.45
$md_{C32}.MSE_c^C$	12.01	8.50	8.34	10.95	13.20	9.46	17.17	6.96

図表 9.3 md_{C9}~$md_{C16}.MSE_c^P$ と MSE_c^C における z 値

	$md_{C9}.MSE_c^P$	$md_{C10}.MSE_c^P$	$md_{C11}.MSE_c^P$	$md_{C12}.MSE_c^P$	$md_{C13}.MSE_c^P$	$md_{C14}.MSE_c^P$	$md_{C15}.MSE_c^P$	$md_{C16}.MSE_c^P$

$md_{C1}.MSE_c^C$	8.76	10.65	6.98	9.16	9.05	11.12	9.84	6.74
$md_{C2}.MSE_c^C$	8.02	9.88	6.42	8.47	8.30	10.33	9.09	6.22
$md_{C3}.MSE_c^C$	10.47	12.39	8.33	10.78	10.81	12.90	11.55	7.99
$md_{C4}.MSE_c^C$	8.60	10.48	6.86	9.01	8.89	10.95	9.68	6.63
$md_{C5}.MSE_c^C$	7.69	9.56	6.13	8.15	7.96	10.01	8.77	5.95
$md_{C6}.MSE_c^C$	7.38	9.24	5.89	7.86	7.64	9.69	8.46	5.73
$md_{C7}.MSE_c^C$	9.55	11.46	7.60	9.92	9.87	11.95	10.64	7.32
$md_{C8}.MSE_c^C$	9.83	11.75	7.82	10.18	10.16	12.25	10.92	7.52
$md_{C9}.MSE_c^C$	8.49	10.36	6.78	8.91	8.77	10.82	9.56	6.56
$md_{C10}.MSE_c^C$	9.15	11.05	7.29	9.54	9.46	11.53	10.23	7.03
$md_{C11}.MSE_c^C$	6.46	8.30	5.16	6.98	6.69	8.73	7.53	5.05
$md_{C12}.MSE_c^C$	9.53	11.44	7.60	9.90	9.85	11.92	10.62	7.31
$md_{C13}.MSE_c^C$	10.29	12.22	8.18	10.62	10.63	12.72	11.38	7.85
$md_{C14}.MSE_c^C$	9.84	11.75	7.83	10.18	10.16	12.24	10.92	7.53
$md_{C15}.MSE_c^C$	7.89	9.78	6.28	8.34	8.17	10.23	8.98	6.09
$md_{C16}.MSE_c^C$	9.88	11.80	7.85	10.22	10.20	12.30	10.97	7.55
$md_{C17}.MSE_c^C$	9.36	11.26	7.45	9.73	9.67	11.75	10.44	7.18
$md_{C18}.MSE_c^C$	8.50	10.38	6.77	8.91	8.78	10.85	9.58	6.54

$md_{C19}.MSE_c^C$	8.17	10.05	6.51	8.60	8.45	10.51	9.25	6.30
$md_{C20}.MSE_c^C$	9.47	11.37	7.54	9.84	9.78	11.86	10.55	7.26
$md_{C21}.MSE_c^C$	9.25	11.15	7.37	9.63	9.56	11.63	10.34	7.10
$md_{C22}.MSE_c^C$	8.61	10.50	6.85	9.02	8.90	10.97	9.69	6.62
$md_{C23}.MSE_c^C$	9.72	11.63	7.74	10.08	10.04	12.12	10.80	7.45
$md_{C24}.MSE_c^C$	8.31	10.20	6.63	8.74	8.60	10.66	9.40	6.41
$md_{C25}.MSE_c^C$	7.98	9.84	6.37	8.42	8.25	10.30	9.05	6.18
$md_{C26}.MSE_c^C$	6.92	8.77	5.53	7.43	7.17	9.21	8.00	5.40
$md_{C27}.MSE_c^C$	7.62	9.49	6.08	8.09	7.89	9.93	8.70	5.91
$md_{C28}.MSE_c^C$	8.70	10.59	6.94	9.11	9.00	11.05	9.78	6.71
$md_{C29}.MSE_c^C$	8.82	10.71	7.03	9.23	9.12	11.18	9.90	6.79
$md_{C30}.MSE_c^C$	9.13	11.02	7.28	9.52	9.43	11.50	10.21	7.02
$md_{C31}.MSE_c^C$	9.25	11.15	7.37	9.63	9.56	11.63	10.33	7.10
$md_{C32}.MSE_c^C$	8.66	10.54	6.91	9.07	8.95	11.01	9.74	6.67

図表 9.4 $\boldsymbol{md_{C17}}$~$\boldsymbol{md_{C24}.MSE_c^P}$ と $\boldsymbol{MSE_c^C}$ における z 値

	$md_{C17}.MSE_c^P$	$md_{C18}.MSE_c^P$	$md_{C19}.MSE_c^P$	$md_{C20}.MSE_c^P$	$md_{C21}.MSE_c^P$	$md_{C22}.MSE_c^P$	$md_{C23}.MSE_c^P$	$md_{C24}.MSE_c^P$
$md_{C1}.MSE_c^C$	6.48	9.09	7.45	12.35	5.16	8.13	9.72	12.16
$md_{C2}.MSE_c^C$	5.94	8.35	7.00	11.63	4.58	7.50	8.79	11.37
$md_{C3}.MSE_c^C$	7.80	10.82	8.56	13.95	6.60	9.63	11.81	13.89
$md_{C4}.MSE_c^C$	6.37	8.93	7.36	12.19	5.04	7.99	9.51	11.98
$md_{C5}.MSE_c^C$	5.66	8.01	6.76	11.34	4.26	7.19	8.41	11.07
$md_{C6}.MSE_c^C$	5.42	7.70	6.56	11.05	4.01	6.92	8.03	10.75
$md_{C7}.MSE_c^C$	7.09	9.90	7.96	13.10	5.82	8.82	10.70	12.97
$md_{C8}.MSE_c^C$	7.30	10.18	8.14	13.37	6.05	9.06	11.05	13.26
$md_{C9}.MSE_c^C$	6.29	8.82	7.29	12.08	4.96	7.90	9.37	11.86
$md_{C10}.MSE_c^C$	6.79	9.49	7.71	12.71	5.50	8.47	10.20	12.55
$md_{C11}.MSE_c^C$	4.72	6.77	5.97	10.18	3.24	6.11	6.90	9.81
$md_{C12}.MSE_c^C$	7.08	9.88	7.96	13.07	5.82	8.81	10.66	12.94
$md_{C13}.MSE_c^C$	7.65	10.64	8.43	13.80	6.43	9.47	11.61	13.73
$md_{C14}.MSE_c^C$	7.31	10.18	8.14	13.36	6.06	9.07	11.04	13.25
$md_{C15}.MSE_c^C$	5.81	8.22	6.89	11.54	4.43	7.36	8.67	11.29
$md_{C16}.MSE_c^C$	7.33	10.22	8.16	13.41	6.08	9.10	11.11	13.31
$md_{C17}.MSE_c^C$	6.94	9.70	7.84	12.91	5.66	8.65	10.46	12.77

$md_{C18}.MSE_c^C$	6.28	8.83	7.28	12.10	4.94	7.90	9.40	11.89
$md_{C19}.MSE_c^C$	6.02	8.50	7.07	11.79	4.66	7.61	9.00	11.56
$md_{C20}.MSE_c^C$	7.03	9.81	7.91	13.01	5.76	8.75	10.59	12.88
$md_{C21}.MSE_c^C$	6.86	9.59	7.77	12.81	5.58	8.56	10.32	12.66
$md_{C22}.MSE_c^C$	6.36	8.94	7.35	12.21	5.03	7.99	9.54	12.00
$md_{C23}.MSE_c^C$	7.22	10.07	8.07	13.25	5.97	8.97	10.90	13.13
$md_{C24}.MSE_c^C$	6.14	8.65	7.17	11.93	4.79	7.74	9.18	11.70
$md_{C25}.MSE_c^C$	5.89	8.30	6.96	11.60	4.53	7.45	8.75	11.34
$md_{C26}.MSE_c^C$	5.08	7.24	6.28	10.61	3.64	6.52	7.47	10.28
$md_{C27}.MSE_c^C$	5.61	7.94	6.72	11.27	4.22	7.13	8.32	10.99
$md_{C28}.MSE_c^C$	6.45	9.04	7.42	12.29	5.13	8.08	9.64	12.09
$md_{C29}.MSE_c^C$	6.54	9.16	7.50	12.40	5.22	8.19	9.79	12.21
$md_{C30}.MSE_c^C$	6.77	9.47	7.70	12.69	5.48	8.46	10.16	12.52
$md_{C31}.MSE_c^C$	6.86	9.59	7.77	12.81	5.58	8.56	10.32	12.65
$md_{C32}.MSE_c^C$	6.41	8.99	7.39	12.25	5.09	8.04	9.59	12.04

図表 9.5 $\boldsymbol{md_{C25}}$~$\boldsymbol{md_{C32}.MSE_c^P}$ と $\boldsymbol{MSE_c^C}$ における z 値

	$md_{C25}.MSE_c^P$	$md_{C26}.MSE_c^P$	$md_{C27}.MSE_c^P$	$md_{C28}.MSE_c^P$	$md_{C29}.MSE_c^P$	$md_{C30}.MSE_c^P$	$md_{C31}.MSE_c^P$	$md_{C32}.MSE_c^P$
$md_{C1}.MSE_c^C$	13.18	13.03	9.71	13.10	8.60	7.82	9.34	12.08
$md_{C2}.MSE_c^C$	12.51	12.12	8.93	12.08	7.80	7.13	8.52	10.96
$md_{C3}.MSE_c^C$	14.69	15.00	11.49	15.25	10.44	9.44	11.23	14.43
$md_{C4}.MSE_c^C$	13.03	12.82	9.54	12.86	8.42	7.67	9.16	11.82
$md_{C5}.MSE_c^C$	12.24	11.80	8.59	11.75	7.44	6.80	8.16	10.61
$md_{C6}.MSE_c^C$	11.96	11.44	8.27	11.35	7.11	6.51	7.82	10.17
$md_{C7}.MSE_c^C$	13.89	13.96	10.54	14.11	9.46	8.57	10.22	13.18
$md_{C8}.MSE_c^C$	14.14	14.29	10.83	14.48	9.76	8.84	10.53	13.59
$md_{C9}.MSE_c^C$	12.93	12.67	9.42	12.69	8.30	7.57	9.03	11.63
$md_{C10}.MSE_c^C$	13.53	13.47	10.12	13.58	9.02	8.19	9.77	12.60
$md_{C11}.MSE_c^C$	11.14	10.36	7.30	10.17	6.11	5.64	6.80	8.88
$md_{C12}.MSE_c^C$	13.86	13.91	10.51	14.06	9.43	8.56	10.20	13.12
$md_{C13}.MSE_c^C$	14.55	14.83	11.31	15.07	10.26	9.27	11.04	14.23
$md_{C14}.MSE_c^C$	14.14	14.28	10.83	14.46	9.76	8.84	10.53	13.57
$md_{C15}.MSE_c^C$	12.43	12.06	8.81	12.03	7.66	7.00	8.39	10.91
$md_{C16}.MSE_c^C$	14.18	14.35	10.88	14.55	9.81	8.88	10.59	13.67
$md_{C17}.MSE_c^C$	13.71	13.73	10.33	13.86	9.24	8.39	10.01	12.90

$md_{C18}.MSE_c^C$	12.95	12.73	9.43	12.77	8.31	7.57	9.05	11.71
$md_{C19}.MSE_c^C$	12.66	12.35	9.09	12.35	7.96	7.26	8.69	11.26
$md_{C20}.MSE_c^C$	13.81	13.85	10.45	13.99	9.36	8.49	10.13	13.05
$md_{C21}.MSE_c^C$	13.62	13.60	10.22	13.72	9.13	8.29	9.89	12.75
$md_{C22}.MSE_c^C$	13.05	12.86	9.55	12.91	8.43	7.67	9.18	11.87
$md_{C23}.MSE_c^C$	14.03	14.14	10.71	14.31	9.64	8.73	10.40	13.39
$md_{C24}.MSE_c^C$	12.79	12.52	9.25	12.53	8.12	7.40	8.85	11.46
$md_{C25}.MSE_c^C$	12.48	12.10	8.89	12.06	7.75	7.08	8.47	10.95
$md_{C26}.MSE_c^C$	11.55	10.89	7.79	10.74	6.62	6.09	7.31	9.51
$md_{C27}.MSE_c^C$	12.17	11.71	8.52	11.64	7.37	6.74	8.08	10.49
$md_{C28}.MSE_c^C$	13.13	12.94	9.65	13.00	8.54	7.77	9.28	11.96
$md_{C29}.MSE_c^C$	13.24	13.09	9.77	13.16	8.66	7.88	9.41	12.14
$md_{C30}.MSE_c^C$	13.50	13.44	10.09	13.54	8.99	8.17	9.75	12.55
$md_{C31}.MSE_c^C$	13.62	13.60	10.22	13.71	9.13	8.29	9.89	12.75
$md_{C32}.MSE_c^C$	13.09	12.90	9.60	12.94	8.49	7.73	9.23	11.91

プログラム（一部抜粋）

Model Creation

モデル作成はモデル作成専用の関数を自作し、モデルのパラメーターを設定するだけで自動生成ができるようにした。

モデルのパラメーターには以下のものがある。

図表 9.6　モデル作成用関数のパラメーター

パラメーター名	内容
input_variables	入力部の変数を指定
rev_out_dim	テキストの情報認知部の出力数
rev_embedding_mat	使用する埋め込み行列の指定
rev_pro_layers	テキスト情報の情報処理部における層数
rev_pro_dims	テキスト情報の情報処理部におけるニューロン数
img_size	画像サイズ（ピクセル）
img_out_dim	画像情報の情報認知部の出力数
img_layers	画像情報の情報認知部の層数
img_d_rate	画像情報の情報認知部における Drop 率（未使用）
filter_size	畳み込みフィルターのサイズ
maker_layer	メーカー情報の情報認知部の層数
maker_dims	メーカー情報の情報認知部におけるニューロン数
maker_out_dim	メーカー情報の情報認知部における出力数
maker_pro_layers	メーカー情報の情報処理部における層数
cost_layer	コスト情報の情報認知部の層数
cost_dims	コスト情報の情報認知部におけるニューロン数
cost_out_dim	コスト情報の情報認知部における出力数
cost_pro_layers	コスト情報の情報処理部における層数
power_layer	パワー情報の情報認知部の層数
power_dims	パワー情報の情報認知部におけるニューロン数
power_out_dim	パワー情報の情報認知部における出力数
power_pro_layers	パワー情報の情報処理部における層数
size_layer	サイズ情報の情報認知部の層数
size_dims	サイズ情報の情報認知部におけるニューロン数
size_out_dim	サイズ情報の情報認知部における出力数
size_pro_layers	サイズ情報の情報処理部における層数
usage_layer	利便性情報の情報認知部の層数
usage_dims	利便性情報の情報認知部におけるニューロン数
usage_out_dim	利便性情報の情報認知部における出力数
usage_pro_layers	利便性情報の情報処理部における層数
final	最終層のフラグ（未使用）
fin_list	最終層のリスト（未使用）
final_layers	最終層の層数（未使用）
final_dims	最終層のニューロン数（未使用）
emp_tg	補償型ヒューリスティクスの対象変数

emp_rate	補償型ヒューリスティクスの倍率
model_name	モデル名
model_save_dir	モデル保存用ディレクトリ
model_plot_dir	モデルプロットディレクトリ

図表 9.7　モデル作成用関数のプログラム画面

```
def Basic_Model_Maker(
                input_variables = ['power', 'cost', 'maker', 'size', 'usage'],
                rev_out_dim=5, rev_embedding_mat=rev_embedding_mat, # rev
                rev_pro_layers=3, rev_pro_dims = [16,16,16],
                img_size=256,img_out_dim=5, image_layers=3, img_d_rate=0, filter_size=2,# img
                img_pro_layers = 3, img_pro_dims = [16,16,16],
                maker_layers=4, maker_dims=[32,16,16], maker_out_dim=5, # maker
                maker_pro_layers=3, maker_pro_dims = [16,16,16],
                cost_layers=4, cost_dims=[32,16,16], cost_out_dim=5, # cost
                cost_pro_layers=3, cost_pro_dims = [16,16,16],
                power_layers=4, power_dims=[32,16,16], power_out_dim=5, #power
                power_pro_layers=3, power_pro_dims = [16,16,16],
                size_layers=4, size_dims=[32,16,16], size_out_dim=5, # size
                size_pro_layers=3, size_pro_dims = [16,16,16],
                usage_layers=4, usage_dims=[32,16,16], usage_out_dim=5, # usage
                usage_pro_layers=3, usage_pro_dims = [16,16,16],
                final=True,
                fin_list = [['power', 'cost', 'maker', 'size', 'usage']], final_layers=3, final_dims=[16,16,16],
                emp_tg=['power'], emp_rate=2, # emp
                model_name='test', model_save_dir=model_save_dir, model_plot_dir=model_plot_dir #model definition
               ):
```

Model Training

モデルの学習には Python の Keras に内蔵されている fit 関数を用いて学習を行っている。図表 9.8 は中心的ルートモデルの学習過程を示した図である。ここでは学習回数を 100 回とし、さらに各学習段階におけるモデルを ModelCheckpoint 関数を用いることで保存している。

図表 9.8　中心的ルートモデルにおける学習過程

```
model_idx_list = np.arange(5,37)
for mi in model_idx_list:
    checkpointer = ModelCheckpoint(filepath=model_save_dir + '/model_{0}/'.format(mi) +'model.{epoch:02d}-{val_loss:.2f}.h5',
                                   verbose=0, save_best_only=False)
    print('☆☆☆☆☆☆ Now Fitting Model == model_{0} ☆☆☆☆☆☆'.format(mi))
    tmp_model = load_model(model_load_dir + '/' + 'model_{0}.h5'.format(mi))
    tmp_hist = tmp_model.fit(input_list, log_high_okini_train, verbose=1, shuffle=True, batch_size=10, epochs=100,
                             validation_split=0.3, callbacks=[checkpointer])
    loss_saver(tmp_hist, save_dir=hist_save_dir + '/model_{0}'.format(mi), save_idx='model_{0}'.format(mi))

    del tmp_model
    del tmp_hist
    backend.clear_session()
```

```
☆☆☆☆☆☆ Now Fitting Model == model_5 ☆☆☆☆☆☆
Train on 507 samples, validate on 218 samples
Epoch 1/100
507/507 [==============================] - 2s 4ms/step - loss: 7.4989 - mean_absolute_error: 1.9924 - val_loss: 2.3108 - val_mean_absolute
_error: 1.2201
Epoch 2/100
507/507 [==============================] - 1s 1ms/step - loss: 1.1599 - mean_absolute_error: 0.7987 - val_loss: 2.2692 - val_mean_absolute
_error: 1.1858
Epoch 3/100
507/507 [==============================] - 1s 1ms/step - loss: 1.1244 - mean_absolute_error: 0.7605 - val_loss: 2.4278 - val_mean_absolute
_error: 1.2292
```

Bootstrap

Bootstrap 過程を自動で行う為のプログラムを作成し、実行した。Bootstrap 毎に乱数によって学習データから Bootstrap サンプルを作成し、そのサンプルによるモデル学習と保存を行っている。

図表 9.9　中心的ルートモデルの Bootstrap プログラム

```
from keras.callbacks import ModelCheckpoint
from keras.backend import tensorflow_backend as backend

high_n = high_iv_sc_cost_train.shape[0]

for md_idx in np.arange(5,37):

    for bst in np.arange(1,101): # set Bootstrap Index
        re_l = np.random.choice(high_n, high_n, replace=True)
        tmp_high_maker = high_iv_sc_maker_train[re_l]
        tmp_high_cost = high_iv_sc_cost_train[re_l]
        tmp_high_power = high_iv_sc_power_train[re_l]
        tmp_high_size = high_iv_sc_size_train[re_l]
        tmp_high_usage = high_iv_sc_usage_train[re_l]
        tmp_high_input = [tmp_high_maker, tmp_high_cost, tmp_high_power, tmp_high_size, tmp_high_usage]
        tmp_high_okini = log_high_okini_train[re_l]

        checkpointer = ModelCheckpoint(filepath=model_save_dir + '/model_{0}/bst_{1}/'.format(md_idx, bst) +'model.{epoch:02d}-{val_loss:.2f}.h5
                                       verbose=0, save_best_only=False)

        print('☆☆☆☆☆ Now Fitting Model == model_{0} == bst == {1} ☆☆☆☆☆'.format(md_idx, bst))
        tmp_model = load_model(model_load_dir + '/' + 'model_{0}.h5'.format(md_idx))
        tmp_hist = tmp_model.fit(tmp_high_input, tmp_high_okini, verbose=1, shuffle=True, batch_size=10, epochs=100,
                                 validation_split=0.3, callbacks=[checkpointer])
        loss_saver(tmp_hist, save_dir=hist_save_dir + '/model_{0}/bst_{1}'.format(md_idx, bst), save_idx=md_idx)
        del tmp_model
        del tmp_hist
        backend.clear_session()
```

```
☆☆☆☆☆ Now Fitting Model == model_5 == bst == 11 ☆☆☆☆☆
Train on 507 samples, validate on 218 samples
Epoch 1/100
507/507 [==============================] - 2s 4ms/step - loss: 5.1429 - mean_absolute_error: 1.7244 - val_loss: 2.1759 - val_mean_absolute
_error: 1.1690
Epoch 2/100
507/507 [==============================] - 1s 1ms/step - loss: 1.9847 - mean_absolute_error: 1.1035 - val_loss: 1.7000 - val_mean_absolute
_error: 1.0078
Epoch 3/100
507/507 [==============================] - 1s 1ms/step - loss: 1.7679 - mean_absolute_error: 1.0245 - val_loss: 1.6225 - val_mean_absolute
_error: 0.9921
Epoch 4/100
507/507 [==============================] - 1s 1ms/step - loss: 1.4802 - mean_absolute_error: 0.9407 - val_loss: 1.7607 - val_mean_absolute
_error: 1.0681
```

謝辞

本論文は筆者が慶應義塾大学大学院経営管理研究科修士課程に在籍中の研究成果をまとめたものである。同研究科の井上哲浩教授には主査として論文に関する指導のみでなく、物事に対する考えかたや、姿勢といった思考の根幹となる部分を始め、多くのことをマーケティングを通じて学ばせていただいた。こうした思考の根幹部分を学び自己に還元することができたのは私が同研究科において学んだ最も大きなことであり、井上哲浩教授には感謝してもしきれない。ここに深謝の意を表する。同研究科の余田拓郎教授、林高樹教授には副査としてご助言をいただくとともに、本論文の指導をいただいた。ここに深謝の意を表する。