＜ 修 士 論 文 ＞

テレマティクスデータを用いた
ドライバー分類/認識モデルの構築

滋 賀 大 学 大 学 院

デ ー タ サ イ エ ン ス 研 究 科

デ ー タ サ イ エ ン ス 専 攻

修了年度　：２０２１年度

学籍番号　：6020107

氏　　名　：佐藤　晴紀

指導教員　：田中　琢真

提出年月日　：２０２２年1月１１日

目 次

1　はじめに

1.1　研究の背景

損害保険会社は従来、事故発生時の経済的損失への補償を提供してきた。しかし近年、損害保険会社は、少子高齢化などによる国内保険市場の縮小に直面している。縮小しつつある市場では、顧客ごとにパーソナライズしたサービスを提供することで、保険契約に付加価値を与えることが求められる。損害保険種目の中で特に重要度が高いのが自動車保険である。自動車保険の特徴は、一台の契約車両に複数名のドライバーがいる可能性があることである。運転特徴の異なる複数名のドライバーを区分しない場合には、適切にパーソナライズされないおそれがある。そのため、顧客ごとにパーソナライズしたサービスを提供するためには、ドライバーを同定する必要がある。

ドライバーを同定するための仕組みを有する商品を提供している保険会社も存在する。あいおいニッセイ同和損害保険株式会社が提供する法人向け自動車保険商品である“ささえるNAVI「Lite」”では、顔認証システムを搭載した専用のドライブレコーダーを提供している[1]。また、個人向けのテレマティクス自動車保険商品においては、ドライバー情報を設定する機能を搭載する専用アプリを提供している。他にも、東京海上日動火災保険株式会社が提供する“ドライブエージェント パーソナル(DAP)”では、契約者が自身で運転を始めるときにドライブレコーダーのパネルからドライバーを選択する方法をとっている[2]。しかしながら、今後ドライブレコーダーの普及率がさらに上昇し、標準搭載となる可能性もある。そのため、専用のドライブレコーダーを前提とする商品の需要が減少することが予測される。また、契約者自身でドライバー情報を設定する方法の場合には、契約者に手間が発生する。そのため、ドライバー情報を適切に設定しない契約者も存在することが懸念される。さらに、適切にドライバー情報を設定しているのかを保険会社が確認することができないという課題がある。ドライバー情報が適切に設定されていない場合には、適切なサービスを提供できない可能性がある。加えて、保険料率の算出にドライバーごとの運転挙動を反映させている場合には、合理的な料率とならないおそれがある。

いくつかの保険会社では、契約者に対して自動車に搭載する通信機能付き車載器を提供している。これにより、契約者の走行データを取得し、当該データを活用した様々なサービスを提供している。具体的には、契約者ごとの運転挙動から安全運転アドバイスを行うことや安全運転スコアを算出して保険料に反映させる仕組みを有する商品を販売している。ドライバーを同定して契約者単位からドライバー単位に細分化することにより、これらの

サービスをより適正かつ効果的に行うことができる。なお、通信機能付き車載器から取得したデータはテレマティクスデータと呼ばれている。ここで、テレマティクス (telematics) とは “telecommunication” と “informatics” を組み合わせた造語で、自動車などの移動体に通信システムを搭載することにより、さまざまな情報を送受信できるシステムのことを言う。テレマティクスデータから、ドライバーを同定することができれば、ドライバーを同定するために追加の機器やアプリの機能を必要としない。そのため、本研究においては、テレマティクスデータを活用して、ドライバーを同定することを考える。

1.2 先行研究

ドライバーごとの運転挙動の違いについてはこれまでも研究されてきた。文献 [3] では、アクセル開度率 (%) とブレーキ圧 (Mpa) のセンサ時系列データに対して、距離関数として DTW（動的時間伸縮法）を用いた ward 法によるクラスタリングを行った。クラスタリングの結果からドライバーまたは道路種別ごとにクラスタの割合には違いがあることを明らかにした。これにより、アクセルおよびブレーキの踏み方には、ドライバーまたは道路種別ごとに差があることを確認した。文献 [4] では、テレマティクスデータからドライバーごとの v-a ヒートマップを作成した。そして、v-a ヒートマップに対してクラスタリングを行うことで、類似する運転挙動を有するドライバーごとにグルーピングを行った。

ドライバーごとの運転特徴の把握やドライバーのグルーピングを行うだけでなく、ドライバーを推定する研究も存在する。文献 [5] では、テレマティクスデータから抽出されるドライバーの運転特徴量として、速度、加速度、進行方向の変化という 3 変数の時系列情報を用いた CNN を実装した。この研究では、3 人のドライバーを分類する教師あり学習モデルを構築した。この研究により、ラベル付きの少人数のドライバーを分類することは可能となったが、実用を考える際には、次の課題がある。現実的には学習データに含まれていないドライバーが運転することもある。しかしながら、CNN のような通常の教師あり分類モデルでは事前に想定していないドライバーを判別することはできない。また、教師あり学習であるためモデルの学習時に分類するクラスのラベル付きデータが必要となる。しかしながら、ドライバーのラベルを取得可能なドライバーから取得した走行データに対して、ドライバーを推定することが求められるケースはあまり多くない。そのため、ドライバーの教師あり学習モデルの活用先は限定的だと考えられる。

1.3 方法

本研究では、先行研究における課題を解決した、テレマティクスデータを用いたドライバー分類モデルの構築を行った。具体的には、顔認証タスクにおいて開発された深層距離学習モデルである ArcFace および AdaCos を用いた深層距離学習モデルを使用した。このことにより、ドライバーの分類だけでなく学習データに存在しないドライバーも検出可能であるドライバー認識モデルの構築を行った。さらに、大規模データセットで学習した深層学習モデルを特徴量ベクトル生成器として使用した。そして、生成した特徴量ベクトルに対して、ArcFace や AdaCos のロス関数と整合性のあるコサイン距離を用いた k-means 法をベースとする、球面 k-measn++法による教師なし分類を行った。このようにして、より現実的な状況設定に適合したドライバーの分類モデルの構築を行った。

1.4 本論文の構成

本論文の構成は以下の通りである。まず、第 2 章では使用するデータの概要、データの前処理方法およびモデルの構築に使用する変数について説明する。第 3 章では，分析に使用する 3 種類のモデルの概要をそれぞれ説明し、教師なし分類モデルの構築に用いる手法である球面 k-means++法の説明を行う。第 4 章では、第 3 章で説明したモデルおよび手法を使用して行った 4 つの分析内容を説明し、それぞれの分析結果および結果の考察を記述する。最後に、第 5 章では分析を通じて得られた結論を整理し、今後の展望を述べる。

2　データの概要

2.1　使用データ

本研究では、ある日本の民間企業の社有車に取り付けられた車載器から取得したテレマティクスデータを用いてモデルの構築・評価を行った。走行開始から運転停止までを1単位（以下ではトリップと呼ぶ）としてデータが分かれている。取得される走行情報としては、観測時点ごとのGPS速度、3軸方向の加速度、GPS進行方向、走行時間、走行距離、GPS位置情報などが測定される。このうち3軸方向の加速度、走行時間、走行距離は15 Hzという高頻度で観測される。GPS速度、GPS進行方向、GPS位置情報は1 Hzで観測され、マップマッチとスムージング処理が行われ、15 Hzにアップサンプリングされている。定性的な情報としては、トリップに対応するドライバー情報が付与されている。分析には2018年4月から2018年10月までの7カ月間に取得したデータを使用する。ただし、これらのデータはデータ送信のためのアプリの起動時のみ取得される。そのため、対象ドライバーの7カ月間の全ての走行データを取得しているわけではない。また、取得したデータの地理的な走行範囲について、多くのドライバーが狭い範囲に集中しているのではなく、日本全土に広く分布している。

当該データに存在するドライバー数は2,365人、トリップ総数は791,608であり、容量は約1.9 TBというビッグデータである。ドライバーごとのトリップ回数、走行距離、走行時間の平均値と五数要約は表1の通りである。また、ドライバーごとのトリップ回数、走行距離、走行時間のヒストグラムは図1の通りである。深層学習モデルの学習においては十分なデータ数が必要とされるため、ドライバー分類モデルの構築において、ドライバーごとのトリップ数は特に重要である。

表1: ドライバーごとのトリップ回数、走行距離、走行時間の平均値と五数要約

	トリップ回数	走行距離 (km)	走行時間 (時間)
Mean	335	3105.2	91.1
Min.	1	0.3	0
1st Qu.	104	899.3	29.9
Median	309	2575.0	82.0
3rd Qu.	518	4579.6	138.5
Max.	1621	22963.1	510.9

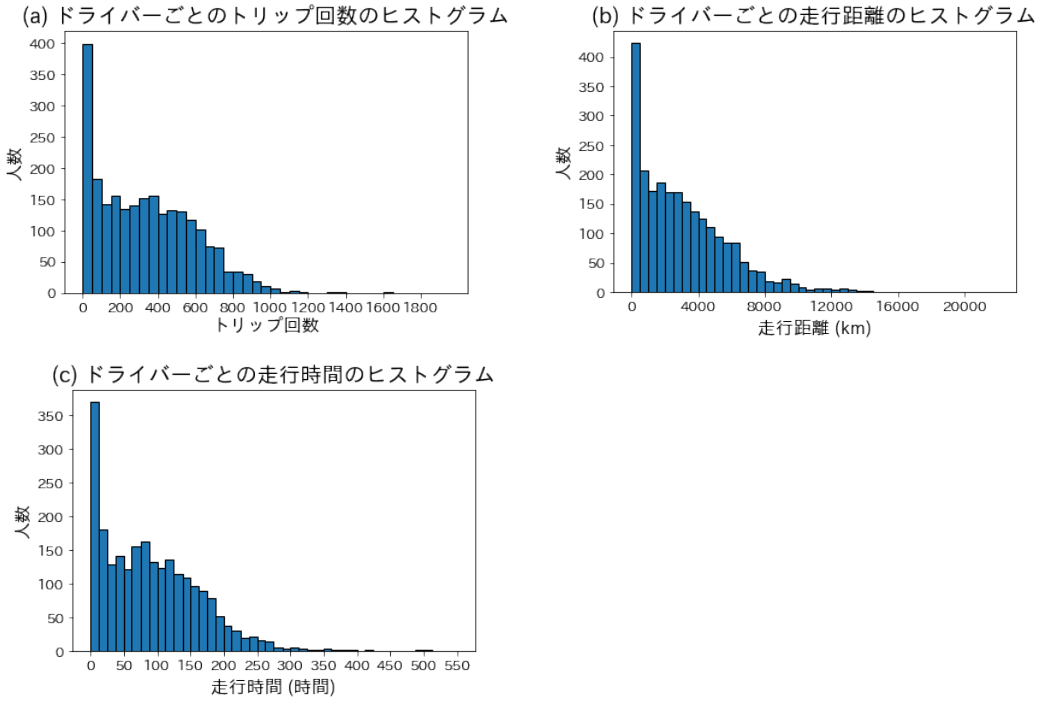

図 1: ドライバーごとのトリップ回数、走行距離、走行時間のヒストグラム。(a) はトリップ回数、(b) は走行距離、(c) は走行時間のヒストグラムを表す。

2.2 前処理および使用する変数

テレマティクスデータとして観測される情報の中で、ドライバー i、トリップ k の観測時間を $T_{i,k}$（秒）、GPS 速度を $\boldsymbol{v_0}^{i,k} = (v_{0,t}^{i,k}) \in \mathbb{R}^{15T_{i,k}}$ (m/s)、進行方向の加速度を $\boldsymbol{a_0}^{i,k} = (a_{0,t}^{i,k}) \in \mathbb{R}^{15T_{i,k}}$ (m/s^2)、GPS 進行方向を $\boldsymbol{\varphi}^{i,k} = (\varphi_t^{i,k}) \in \mathbb{R}^{15T_{i,k}}$（度）とおく。分類モデルに使用する変数として、$\boldsymbol{\delta}_0^{i,k} = (\delta_{0,t}^{i,k}) \in \mathbb{R}^{15T_{i,k}}$、$\boldsymbol{j}_0^{i,k} = (j_{0,t}^{i,k}) \in \mathbb{R}^{15T_{i,k}}$ (m/s^3) を以下で定義する。

$$\delta_{0,t}^{i,k} = \left|\sin\left(\frac{\pi}{180}(\varphi_t^{i,k} - \varphi_{t-1}^{i,k})\right)\right|$$
$$j_{0,t}^{i,k} = 15(a_{0,t}^{i,k} - a_{0,t-1}^{i,k})$$

$\delta_{0,t}^{i,k}$ は進行方向の変化を変換したものである。$\varphi_t^{i,k}$ は 0 度から 360 度の値を取る。そのため、ラジアンに変換して sin を取ることで 0 度と 360 度を同一視している。さらに絶対値をとることで、回転の向きによる影響を取り除いている。ここで、角速度ではなく進行方向の変化を用いているため、観測間隔が異なるデータを使用する際には問題となる可能性がある。sin に入れる値のスケールを変えると、スケールが小さければほぼ差そのものになり、スケールが大きければ非線形性が出てくる。観測間隔で除した角速度を用いなかった理由としては、非線形性は深層学習モデルで捉えればよく、スケールを大きくしてゆがませることを避けるべきと判断したためである。$j_{0,t}^{i,k}$ は加速度の差分を観測間隔で割って算出されており、加

加速度（ジャークもしくは躍度）と呼ばれる物理量である。加加速度は運転の乗り心地に影響を与えることが知られている。

分析においては、入力データの次元を統一するために、トリップごとの観測時間および観測間隔を揃えたデータを使用することとする。トリップごとに統一された観測時間をTおよび観測間隔Δを定め、モデルへの入力データを作成するため、$(\boldsymbol{v}_0^{i,k}, \boldsymbol{a}_0^{i,k}, \boldsymbol{\delta}_0^{i,k}, \boldsymbol{j}_0^{i,k}) \in \mathbb{R}^{15T_{i,k}\times 4}$に対して以下の前処理を行った。

- 停車時の走行データを取り除くことおよび道路種別を揃える目的から、速度は$[2, 50]$ km/hのデータを使用し、平常時の走行データとするため、加速度は$[-3, 3]$ m/s^{2}の範囲のデータに限定した。また、制限した後のドライバーi、トリップkの観測時間を$\tilde{T}_{i,k}$とおく。
- $\tilde{T}_{i,k} \geq T$以上のトリップのみに絞り、走行時間$\tilde{T}_{i,k}$の中間時点から前後$\frac{T}{2}$秒ずつを取得する。
- $\frac{1}{15}$秒間隔でデータが存在している$(\boldsymbol{v}_0^{i,k}, \boldsymbol{a}_0^{i,k}, \boldsymbol{\delta}_0^{i,k}, \boldsymbol{j}_0^{i,k})$に対して、データを間引くことにより、観測間隔が$\Delta$のデータとなるようにする。

上記の前処理により取得された4変数に対して、さらに標準化したものを$\boldsymbol{v}^{i,k}, \boldsymbol{a}^{i,k}, \boldsymbol{\delta}^{i,k}, \boldsymbol{j}^{i,k}$とおき、モデルへの入力を

$$\boldsymbol{x}^{i,k} = (\boldsymbol{v}^{i,k}, \boldsymbol{a}^{i,k}, \boldsymbol{\delta}^{i,k}, \boldsymbol{j}^{i,k}) \in \mathbb{R}^{\frac{T}{\Delta}\times 4}$$

とする。なお、モデルの学習に用いるドライバーと当該モデルを用いて分類を行うドライバーが異なる場合において、分類対象のトリップを標準化する際には、分類対象トリップの平均と標準偏差の代わりに、学習データの作成時の標準化に使用した平均と標準偏差を使用する。

3　分析手法

この章ではドライバー分類/認識問題で使用する深層学習モデルである ResNet、ArcFace および AdaCos の説明を行い、加えて教師なし学習で使用するクラスタリング手法である球面 k-means++法の説明を行う。

3.1　ResNet

ResNet は文献 [6] において提案された。ショートカットコネクションを用いた残差学習の枠組みを提案することにより、勾配劣化問題の解決を図ったモデルである。これによって非常に深い層の深層学習モデルの学習が可能になった。ResNet は ILSVRC 2015 の分類タスクで 1 位を獲得したこともあり、画像認識分野で使用されていることが多いモデルである。しかしそれにとどまらず、文献 [7] において、ResNet は幅広いドメイン、様々な長さの時系列データおよび様々なデータ量である、多様な時系列データセットの分類問題においても、高い精度が得られることが確認されている。そこで、本研究では 3 つのモデルを構築するが、全体が ResNet からなる、もしくは ResNet をベースとしたモデルを構築する。以下では、分析にて使用した ResNet の構造を説明する。

本論文においては、以下の構造を持つ深層学習モデルを ResNet と呼ぶことにする。分析に使用した ResNet は 3 個の残差ブロックから構成され、その後にグローバル・アベレージ・プーリング層（GAP）とクラス数と同じサイズのニューロン数を有する全結合層およびソフトマックス分類器へと続く構造をしている。各残差ブロックは 3 つの畳み込み層で構成され、その出力値に残差ブロックの入力値を加算したものを残差ブロックの出力値とする。ただし、畳み込み演算により入力からサイズが変わる場合には、サイズが揃うように入力値の代わりに、入力値にカーネル数が 1 の畳み込み演算を施したものを加算することとする。なお、残差ブロックに用いられる畳み込み層は畳み込み演算、バッチ正規化、活性化関数である ReLU からなる。畳み込み演算に用いるフィルター数は全て 64 とし、それぞれの畳み込み演算におけるカーネル幅はそれぞれ 8、5、3 で、ストライドは全て 1 とし、ゼロパディングを行うこととする。この設定とすることで中間層においても入力時系列データの長さが保たれることとなる。このことは分類に特に効いている入力時系列の時点を可視化する際に重要となる。また、GAP 層の前の層の畳み込み層に使用するフィルター数については、次節で説明する ArcFace などの特徴量ベクトルの次元ともなるため、特に重要なパラメータである。

次にモデルの学習時の設定を記す。最適化手法としては、パラメータ $\beta_1 = 0.9$、$\beta_2 = 0.999$、$\epsilon = 10^{-7}$ とした Adam を用いた。エポック数は 200、バッチサイズは 10、学習率は 10^{-3}、weight decay の係数を 10^{-6} とした。そして、バリデーションデータに対してロスを最小にするパラメータを保存し、推論時に使用する。これらの設定については、次節以降で説明するモデルである ArcFace および AdaCos の学習時においても同様の設定を行う。

3.2 ArcFace

ArcFace モデルは、[8] において提案された。深層距離学習モデルの 1 つであり、その名称の通り、顔認識タスクにおいて提案されたモデルである。ArcFace は ResNet のような出力層が全結合層およびソフトマックス関数を用いている分類モデルにおいて、全結合層の重みとロス関数を修正することで構築されるモデルである。本研究では前節で導入した ResNet をもとにした ArcFace を用いることとし、以下では当該モデルの構造に関する説明を行う。

ResNet の出力層のソフトマックスとクロスエントロピーロスは、データ数 N、クラス数 C、クラス c に対応する全結合層の重み $\boldsymbol{W}_c$ およびバイアス項 b_c、データ i に対応する GAP 層の出力ベクトルを $\boldsymbol{z}_i$ および正解ラベル y_i を用いて以下のように表せる。

$$L_1 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{\boldsymbol{W}_{y_i}^T\boldsymbol{z}_i+b_{y_i}}}{\sum_{c=1}^{C}e^{\boldsymbol{W}_c^T\boldsymbol{z}_i+b_c}} \tag{1}$$

式 (1) において、全ての i, c に対して、バイアス項 $b_c = 0$、$||\boldsymbol{W}_c||_2 = ||\boldsymbol{z}_i||_2 = 1$ を仮定すると、$\boldsymbol{W}_c^T\boldsymbol{z}_i$ は $\boldsymbol{W}_c$ と $\boldsymbol{z}_i$ のなす角 $\theta_{i,c}$ を用いて、$\boldsymbol{W}_c^T\boldsymbol{z}_i = \cos\theta_{i,c}$ と表現できる。したがって、L_1 は、

$$L_1 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{\cos\theta_{i,y_i}}}{e^{\cos\theta_{i,y_i}}+\sum_{c\neq y_i}e^{\cos\theta_{i,c}}} \tag{2}$$

と表せる。

次に、式 (2) のロス関数において、マージンパラメータ m および再スケーリングパラメータ s を以下のように導入する。

$$L_2 = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{s\cos(\theta_{i,y_i}+m)}}{e^{s\cos(\theta_{i,y_i}+m)}+\sum_{c\neq y_i}e^{s\cos\theta_{i,c}}} \tag{3}$$

ここで、マージンパラメータ m は正解ラベルに対して罰則を与えることでより正解ラベルに対するコサイン類似度がより大きくなるように、すなわちコサイン距離がより近くなるように学習することを目的としたパラメータである。また、$\boldsymbol{W}_c$ および $\boldsymbol{z}_i$ を L^2 正規化したことにより、ソフトマックス関数への入力値が –1 から 1 という狭い範囲に制限されてしまう。

このことにより学習が進まなくなることを防ぐために導入されたパラメータが再スケーリングパラメータ s である。これらのパラメータはハイパーパラメータである。したがって、学習により最適な値を獲得するものではなく、利用者が最適だと思われる値を設定する必要がある。

上述の通り、ArcFace は ResNet をベースとして構成している。推論時には GAP 層の出力ベクトルを特徴量ベクトルとして、特徴量ベクトルをもとにクラスを推定する。具体的には、学習データから事前に生成した各クラスの特徴量ベクトルとテストデータから生成された特徴量ベクトルとの間のコサイン類似度を計算し、類似度が閾値以上の一番高いクラスと予測する。なお、各クラスとのコサイン類似度の最大値が閾値未満の場合には、学習データに存在しないクラスのデータとみなす。ArcFace は同一クラスの特徴量ベクトル間の距離を小さくし、異なるクラスの特徴量ベクトル間の距離が大きくなるように学習する。この特徴から、ResNet のように各クラスに属する確率の予測値を出力するのではなく、学習時に存在しないクラスを含めて認識することができるという特徴を持つ。

なお、各クラスの特徴量ベクトルの生成方法は複数考えられる。本研究においては各クラスの全ての学習データから生成された特徴量ベクトルの中央値を当該クラスの特徴量ベクトルとみなす。他にも中央値ではなく平均値を使用することや、学習データの特徴量ベクトルの集合において最近傍のデータと比較することも考えられる。学習時に推定誤りをしたデータによる影響を取り除く目的から中央値が適していると考え、当該手法を採用した。

3.3 AdaCos

前節で説明した通り、ArcFace は 2 つのハイパーパラメータを持ち、これらの最適な値を見つけることは容易ではない。そこで、文献 [9] で提案された AdaCos は ArcFace におけるハイパーパラメータを自動で決定する方法を提案するモデルである。提案論文では、ArcFace における m の値による影響と s の値による影響を確認し、両者を比較して s の値の方が重要性が高いことを示した。この結果から、AdaCos モデルでは $m = 0$ とおき、s の値を最適化問題を解くことにより決定することを提案している。さらに、s の値を固定された定数とする方法に加えて、学習効率が高まるように s の値をイテレーションごとに変えていく方法を提案している。本研究においては後者の手法を用いることとする。

次に s を決定する方針を簡単に説明する。式 (3) の log の中の式である予測確率を P_{i,y_i} と

おくと、$B_i = \sum_{c \neq y_i} e^{s \cos \theta_{i,c}}$ を用いて、

$$
\begin{aligned}
P_{i,y_i} &= \frac{e^{s \cos \theta_{i,y_i}}}{e^{s \cos \theta_{i,y_i}} + \sum_{c \neq y_i} e^{s \cos \theta_{i,c}}} \\
&= \frac{e^{s \cos \theta_{i,y_i}}}{e^{s \cos \theta_{i,y_i}} + B_i}
\end{aligned}
$$

と表現される。最適化の方針として、イテレーションごとに θ_{i,y_i} の変化に対して予測確率 P_{i,y_i} が大きく変化するような s を設定することを考える、すなわち、$\left|\frac{\partial P_{i,y_i}}{\partial \theta_{i,y_i}}\right|$ が最大となる s を近似的に求める。

上記の方針により、s は具体的には以下で設定される。

$$
\tilde{s}^{(t)} = \begin{cases} \sqrt{2}\log(C-1) & t = 0 \\ \frac{\log B_{\text{avg}}^{(t)}}{\cos(\min(\frac{\pi}{4}, \theta_{\text{med}}^{(t)}))} & t \geq 1 \end{cases}
$$

ここで、$B_{\text{avg}}^{(t)}$ および $\theta_{\text{med}}^{(t)}$ は、$\mathcal{N}^{(t)}$ を t 番目のイテレーションにおけるミニバッチインデックスの集合として、以下で定まる変数である。

$$
B_{\text{avg}}^{(t)} = \frac{1}{|\mathcal{N}^{(t)}|} \sum_{i \in \mathcal{N}^{(t)}} B_i^{(t)} = \frac{1}{|\mathcal{N}^{(t)}|} \sum_{i \in \mathcal{N}^{(t)}} \sum_{c \neq y_i} e^{\tilde{s}^{(t-1)} \cos \theta_{i,c}},
$$

$$
\theta_{\text{med}}^{(t)} = \underset{i \in \mathcal{N}^{(t)}}{\text{median}}(\theta_{i,y_i}^{(t)})
$$

以下、前節の ResNet モデルをベースとして AdaCos を組み込んだモデルを AdaCos と呼ぶことにする。

3.4 球面 k-means++法

4 章において、走行データから特徴量ベクトルを生成し、これら生成された特徴量ベクトルをクラスタリングすることにより教師なし分類を行う。クラスタリング手法としては、文献 [10] で提案された球面 k-means++法を用いる。単位球面上に存在するデータに対するクラスタリング手法としては球面 k-means 法がある。しかしながら、球面 k-means 法は初期値依存性があることが知られている。球面 k-means++法とは、k-means++法と類似した手順により初期クラスタの選ぶことにより、初期値依存性を改良したクラスタリング手法である。k-means 法において距離関数としてコサイン距離を使用する際に有効な手法であると考えられる。

まず、球面 k-means++法の説明をするために、必要となる記号の定義を行う。d を自然数、$S^{d-1} = \{\boldsymbol{s} \in \mathbb{R}^d | \, ||\boldsymbol{s}||_2 = 1\}$、クラスタリングの対象データ集合を $Z \subset S^{d-1}$ とする。また、

$\boldsymbol{s}_1 \in S^{d-1}$ と $\boldsymbol{s}_2 \in S^{d-1}$ の近さを測る尺度として関数 d_α を

$$d_\alpha(\boldsymbol{s}_1, \boldsymbol{s}_2) = \alpha - \langle \boldsymbol{s}_1, \boldsymbol{s}_2 \rangle$$

と定義する。

注目すべき事項として、d_1 はコサイン距離と呼ばれている近さを測る尺度であるが、三角不等式を満たさないため厳密には距離関数ではない。一方で、k-means++法の初期値依存性の改善に関する理論的な性質は、距離関数が三角不等式を満たすことを使用して導かれたものである。そのため、コサイン距離にそのまま適用することができない。そこで、d_α が $\alpha \geq \frac{3}{2}$ のときに、三角不等式を満たす性質を利用して、d_α, $\alpha \geq \frac{3}{2}$ を用いてクラスタ初期値の選択を行うことを考える。$\boldsymbol{z} \in Z$、$U \subset S^{d-1}$ に対して、点と集合の間の距離 $D(\boldsymbol{z}, U)$ を

$$D(\boldsymbol{z}, U) = \inf_{\boldsymbol{u} \in U} d_\alpha(\boldsymbol{z}, \boldsymbol{u})$$

とする。さらに、集合 $C^{\boldsymbol{u}}$ を $C^{\boldsymbol{u}} = \{\boldsymbol{z} |\, d(\boldsymbol{z}, \boldsymbol{u}) = d(\boldsymbol{z}, U)\}$ と定め、目的関数を

$$J(U) = \sum_{\boldsymbol{u} \in U} \sum_{\boldsymbol{z} \in C^{\boldsymbol{u}}} d_\alpha(\boldsymbol{z}, \boldsymbol{u})$$

とする。以下ではクラスタ数を k とする。

上記設定において、球面 k-means++法のアルゴリズムは Algorithm 1 の通りである。当該アルゴリズムに従い取得したクラスタ中心を U とする。目的関数 $J(U)$ の期待値は、最適な目的関数 J^* を用いて、次のように上から抑えられることが知られている。

$$\mathbb{E}[J(U)] \leq 4(lnk + 2)J^*$$

Algorithm 1 球面 k-means++(SKM++)

SKM++1. 以下の手順で初期クラスタ中心の集合 U を与える。

SKM++1-1. 反復回数 t=1 とする。$\boldsymbol{u} \in Z$ をランダムに選択し、クラスタ中心の初期集合 $U^t = \{\boldsymbol{u}\}$ とする。

SKM++1-2. $D(\boldsymbol{z}, U^t)$ に比例的な確率で選択された $\boldsymbol{z}$ を新たなクラスタ中心とし、$U^{t+1} = U^t \cup \{\boldsymbol{z}\}$ とする。

SKM++1-3. t:=t+1 とする。t=k ならば終了。そうでなければ SKM++1-2. に戻る。

SKM++2. 各 $\boldsymbol{z} \in Z$ を $\boldsymbol{z}$ に最も近いクラスタ中心 $\boldsymbol{u}$ が代表するクラスタ $C^{\boldsymbol{u}}$ に割り当てる。

$$C^{\boldsymbol{u}} = \{\boldsymbol{z} |\, d(\boldsymbol{z}, \boldsymbol{u}) = d(\boldsymbol{z}, U)\}$$

SKM++3. 以下に従って各クラスタ中心を更新する。

$$\boldsymbol{u} = \mathrm{mean}(c^{\boldsymbol{u}}) = \frac{\sum_{\boldsymbol{z} \in C^{\boldsymbol{u}}} \boldsymbol{z}}{||\sum_{\boldsymbol{z} \in C^{\boldsymbol{u}}} \boldsymbol{z}||_2}$$

SKM++4. アルゴリズムの終了条件を満たせば終了。そうでなければ SKM++2 に戻る。

4　分析

4.1　観測時間および観測間隔と正答率の関係

本研究にて使用するテレマティクスデータは、観測頻度が15 Hzという高頻度観測データであることから分かる通り、データ量は膨大となる。そのため、全データを用いてモデルの構築を行うことは現実的ではない。そこで、1つ目の分析として、モデルへの入力データの観測時間Tおよび観測間隔Δを変えて、正答率との関係を調べる。この結果をもとに次節以降で行う分析に使用する観測期間と観測間隔の決定方針を策定する。分類対象としては、360秒以上のトリップが300件以上存在するドライバー402人の中からランダムに5人を選択する。当該5人のドライバーの300件ずつのトリップを用いた教師あり分類を行う。

分析に使用する観測時間および観測間隔の組み合わせとしては、観測時間については120秒、180秒、240秒、300秒および360秒の5通り、観測間隔については1/15秒、2/15秒、$\cdots$、15/15秒の15通りとし、全各組み合わせである75通りを確認する。それぞれの観測時間と観測間隔の組み合わせに応じて処理されたデータを用いたモデルのテスト誤差を調べる。分析にはResNetを使用し、各ドライバーのトリップである300トリップのうち、6割を学習データ、2割をバリデーションデータ、2割をテストデータに分割する。

分析結果は表2としてまとめており、結果としては想定通り観測時間が長くなるほどもしくは観測間隔が短いほど、正答率は高くなる傾向が見られた。また、正答率が最も低くなった「観測時間120秒・観測間隔が1秒」の組み合わせにおいても、正答率が80.0%を超えることが分かった。反対に正答率が最大になる「観測時間が300秒・観測間隔が2/15秒」または「観測時間が360秒・観測間隔が3/15秒」の場合には、正答率は99.0%となることが分かった。

表3には、観測時間および観測間隔の組み合わせごとの入力時系列のデータ長さを並べている。使用する時系列は4変数あるため、入力データの容量は表のデータ長にある数値を4倍にした値に依存すると考えられる。また、図2はデータ長と正答率の散布図である。乱数により正答率にばらつきが生じているため、必ずしも成り立たないが、データ容量が同じであるという条件下では、観測時間の値が大きい方が正答率が高い傾向にある。データ長はT/Δで計算されるため、データ長が等しい場合には、観測時間の値が大きいということは観測間隔の値が大きいということである。そのため、データ容量に制限を設けた場合には、観測間隔を短くするよりも観測時間を長く取る方が正答率が高くなることが期待できる。したがって、以降の分析では観測間隔として1秒を採用する。そして、観測時間はド

ライバー数も考慮しながら可能な限り長い時間を選択することとする。

表 2: 観測時間および観測間隔と正答率との関係

T＼Δ	1/15	2/15	3/15	4/15	5/15	6/15	7/15	8/15	9/15	10/15	11/15	12/15	13/15	14/15	15/15	平均
120	95.3%	93.3%	92.7%	92.0%	93.3%	93.0%	91.3%	90.7%	88.3%	87.7%	86.3%	82.7%	83.3%	88.3%	81.3%	89.3%
180	97.0%	97.3%	97.3%	97.0%	95.0%	96.3%	92.7%	96.3%	94.3%	91.7%	91.3%	92.0%	89.3%	85.7%	86.0%	93.3%
240	98.7%	96.0%	97.7%	98.0%	96.0%	93.0%	96.3%	93.7%	91.7%	96.7%	96.3%	94.3%	94.3%	92.0%	94.3%	95.3%
300	98.3%	99.0%	96.0%	96.0%	97.0%	96.7%	95.0%	97.3%	94.7%	94.7%	95.7%	95.3%	94.3%	92.0%	95.3%	95.8%
360	98.3%	98.3%	99.0%	98.0%	98.0%	98.7%	96.7%	96.0%	98.0%	95.7%	95.3%	96.0%	95.3%	96.3%	94.3%	96.9%
平均	97.5%	96.8%	96.5%	96.2%	95.9%	95.5%	94.4%	94.8%	93.4%	93.3%	93.0%	92.1%	91.3%	90.9%	90.3%	94.1%

表 3: 観測時間および観測間隔と入力データの長さの関係

T＼Δ	1/15	2/15	3/15	4/15	5/15	6/15	7/15	8/15	9/15	10/15	11/15	12/15	13/15	14/15	15/15
120	1,800	900	600	450	360	300	257	225	200	180	164	150	138	129	120
180	2,700	1,350	900	675	540	450	386	338	300	270	245	225	208	193	180
240	3,600	1,800	1,200	900	720	600	514	450	400	360	327	300	277	257	240
300	4,500	2,250	1,500	1,125	900	750	643	563	500	450	409	375	346	321	300
360	5,400	2,700	1,800	1,350	1,080	900	771	675	600	540	491	450	415	386	360

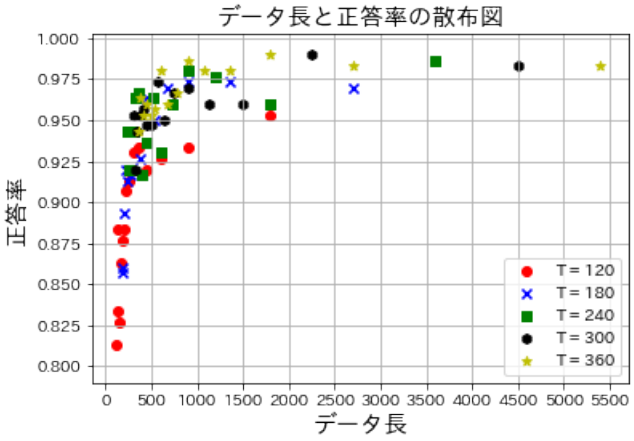

図 2: モデルへの入力データのデータ長と正答率の散布図。T の値ごとにシンボルの色および形を変えている。$T = 120$ に対応するシンボルは赤色の丸、$T = 180$ に対応するシンボルは青色のバツ、$T = 240$ に対応するシンボルは緑色の四角、$T = 300$ に対応するシンボルは黒色の六角形、$T = 360$ に対応するシンボルは黄色の星で表している。

4.2 1,000 人規模の分析

2 つ目の分析として、3 つの深層学習モデルの性能の限界確認および性能比較をするために、1,000 人規模のドライバーの分類精度を検証する。モデルの学習には分類対象となる各ドライバーごとに十分なデータ量が必要である。したがって、分析に使用する各ドライバーのトリップ数を 200 とする。

まず、分析に使用するデータの観測時間 T を決定する。観測時間としては、180 秒、240 秒、300 秒および 360 秒を考える。それぞれの観測時間について、当該観測時間以上のトリップ数が 200 トリップ以上存在するドライバーの人数を把握する。

観測時間とドライバー数の関係は、表 4 の通りである。この結果から 1,000 人を超える最長の観測時間として、$T = 300$ 秒を採用する。また、4.4 節で使用するドライバー 2 人を除いた 1,033 人のドライバーを対象として、モデルの構築・評価を行う。

表 4: 観測時間とドライバー数の関係

観測時間	200 トリップ以上のドライバー数
180	1,283
240	1,174
300	1,035
360	887

アンダーサンプリングの観点およびデータ量削減の観点から、各ドライバーのトリップをランダムに 200 トリップずつ選ぶ。そして、6 割を学習データ、2 割をバリデーションデータ、2 割をテストデータに分割する。なお、参考までに、1,033 人で各ドライバー 200 トリップであるため、トリップ総数は 206,600 トリップある。データ長が 300 の場合のデータ容量は 1.84 GB となる。使用可能なメモリの容量によっては、メモリ容量を考慮しながら、表 3 の値を参考に観測間隔を短くすることも可能である。また、ArcFace におけるハイパーパラメータの設定としては、$s = 30$、$m = 0.05$ を使用する。

各モデルにおけるテストデータに対する正答率は、表 5 の通りである。ResNet、AdaCos、ArcFace の順で正答率は高く、全てのモデルにおいて 50%を超える結果となった。1,000 クラス分類であれば、ドライバーをランダムに選択する場合には正答率は平均で 0.1%となるため、50%を超える結果は驚異的である。この結果から、運転挙動には個人差が存在し、個人ごとの特徴がたった 5 分間の走行データからでも抽出できることが判明した。また、各モデルの結果の比較として、ArcFace と AdaCos を比較すると ArcFace の方が正答率が低い。この

理由としては、ハイパーパラメータの設定が最適な値となっておらず、学習が効率的に行われていないことが考えられる。一方で、ResNet と AdaCos の結果を比べると、ResNet の方が正答率は高い。そのため、ResNet の方が良いモデルのように見えるかもしれない。しかしながら、AdaCos においては同じクラスの特徴量ベクトルが集まるように学習しているため、優れた内部表現が得られていることが期待できる。このことは、4.4.2 節において確認する。

表 5: モデルごとの正答率

モデル	正答率
ResNet	56.9%
ArcFace	51.9%
AdaCos	54.8%

図 3 にはそれぞれのモデルにおけるドライバーごとの正答率のヒストグラムを図示している。この図から正答率にはドライバーごとの個人差が大きいことが分かる。注目すべき点として、ほとんどのデータを判別できているドライバーも存在する。言い換えると、他とは異なる特徴的な運転を行っているドライバーが存在することが分かる。一方で、ほとんど見分けられていないドライバーが存在することも分かった。

次に AdaCos の結果を代表して、ドライバーごとの正答率が高いドライバーと低いドライバーの特徴を調べる。具体的には、対象ドライバーが各変数の平均値および標準偏差において特異な特徴を持つのかを確認する。各変数について、ドライバー i のトリップごとの平均値の平均値を $(\overline{v}_i, \overline{a}_i, \overline{\delta}_i, \overline{j}_i)$、トリップごとの標準偏差の平均値を $(\overline{\sigma}_{v_i}, \overline{\sigma}_{a_i}, \overline{\sigma}_{\delta_i}, \overline{\sigma}_{j_i})$ とおく。ドライバーごとの各変数の平均値の平均値と標準偏差の平均値の散布図は図 4 の通りである。正答率が高いドライバーに着目すると、加速度の散布図において比較的疎な点にプロットされているドライバーが多く見える。また、δ の散布図を見ると、平均値および標準偏差がともに小さい部分にプロットされているドライバーが多い。したがって、曲道を運転するデータではなく、直進しているデータを取得しているドライバーに多いことが分かる。

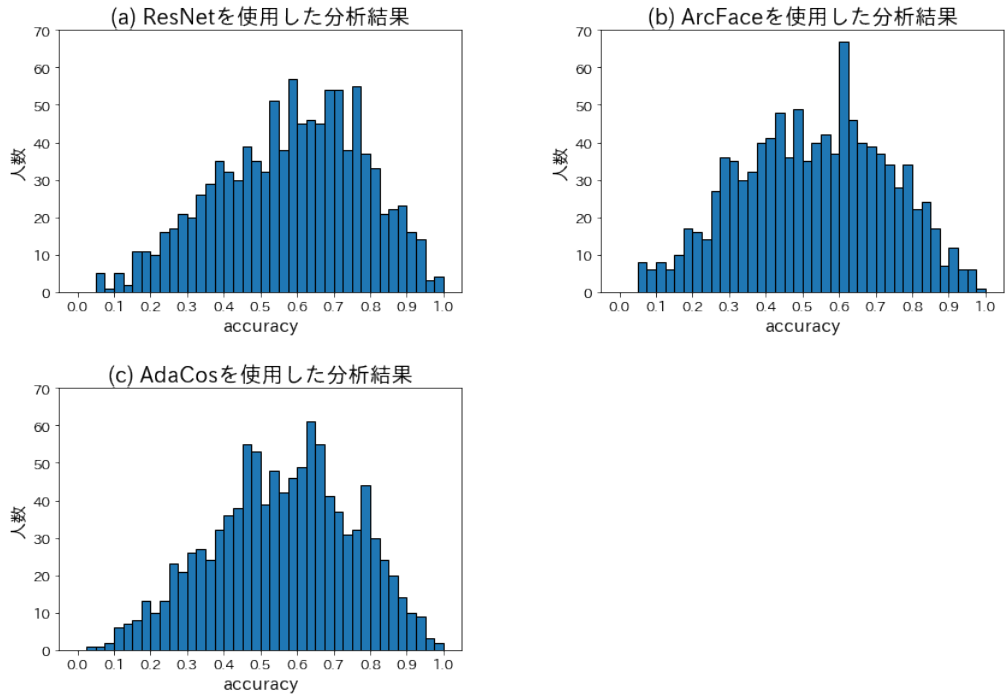

図 3: 3 つのモデルぞれぞれでのドライバーごとの正答率のヒストグラム。(a) は ResNet、(b) は ArcFace、(c) は AdaCos を使用した分析結果である。

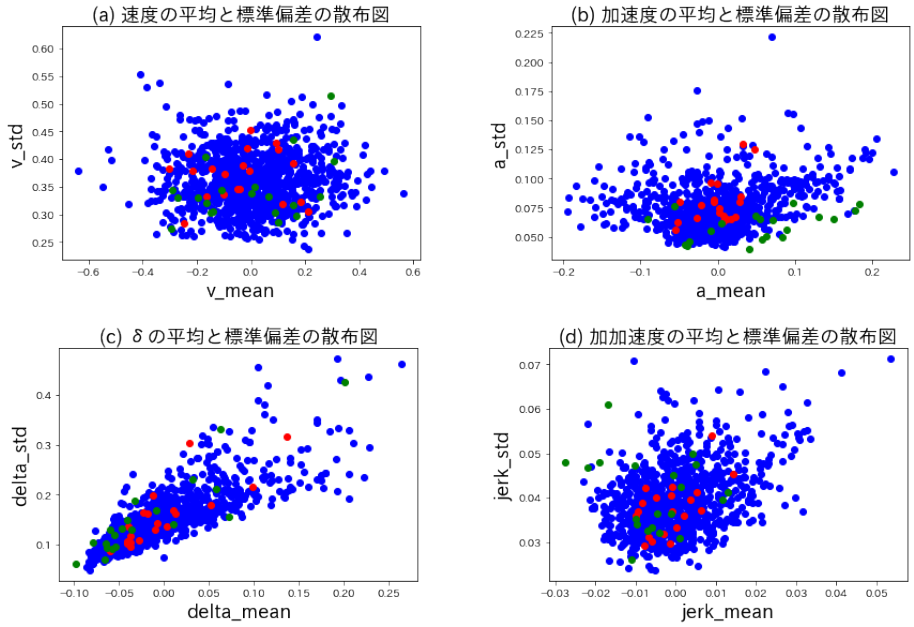

図 4: ドライバーごとの各変数に対して、トリップごとの平均値の平均値と標準偏差の平均値の散布図。各図における変数として (a) は速度、(b) は加速度、(c) は δ、(d) は加加速度を用いた。赤色のシンボルは正答率の低い順に 20 人のドライバー、緑色のシンボルは正答率の高い順に 20 人のドライバー、青色のシンボルはそれ以外のドライバーを表す。

4.3 ドライバー認識可能性

4.2 節で学習した AdaCos を用いて、ドライバーの認識が可能であるかを確認する。学習データに存在する n 人のドライバーのテストデータに学習データに存在しないドライバーのトリップ 40 個を加える。この n 人のドライバーおよびそれ以外のドライバーという $n+1$ クラス分類問題を考える。コサイン類似度の閾値と $n+1$ クラス分類としての精度との関係を調べることで、ドライバーの認識が可能であるかを確認する。

学習データに存在するドライバーは 1,033 人からランダムに選択する。また、学習データに存在しないドライバーのトリップは 107,895 トリップから 40 トリップをランダムに取得することとする。このようにランダムに選ばれたドライバーに対する $n+1$ クラス分類問題を 100 回繰り返す。なお、n としては 2 および 3 の 2 つのケースを考え、閾値としては 0 から 1 までの 0.05 刻みの 21 通りを確認する。

結果として、図 5 は 3 クラス分類と 4 クラス分類のそれぞれに対して、閾値ごとの正答率の平均値を図示している。どちらの場合にも適切な閾値を定めることで、学習データに存在していないドライバーのデータを含めても分類が出来ていることが分かる。3 クラス分類においては、閾値が 0.35 のときに正答率は最大の 90.0%となった。そして、4 クラス分類においては、閾値が 0.3 のときに最大値となり、87.6%であった。

上述した分析結果はデータに依らない閾値を定めた場合の結果である。そのため、ドライバーの組み合わせによっては適切な閾値となっていない可能性が高い。そこで、仮にドライバーの組み合わせごとに適切な閾値を設定できると仮定した場合の正答率の最大値、最小値、平均値、標準偏差を確認する。結果は表 6 の通りである。閾値を固定した場合の正答率の平均値とデータごとに最適な閾値を設定した際の正答率の平均値に大差がないことが分かった。そのため、最適な閾値はドライバーの組み合わせに大きくは依存していないことが分かる。また、3 人の場合の最小値は 76.2%であり、認識が難しいドライバーの組み合わせが存在することが判明した。

表 6: 3 クラス分類および 4 クラス分類についてそれぞれを 100 回行い、試行ごとに最適な閾値を設定した場合の正答率の最大値、最小値、平均値、標準偏差の表

	3 クラス分類	4 クラス分類
最大値	97.5%	96.3%
最小値	80.0%	76.2%
平均値	91.0%	88.5%
標準偏差	3.7%	4.3%

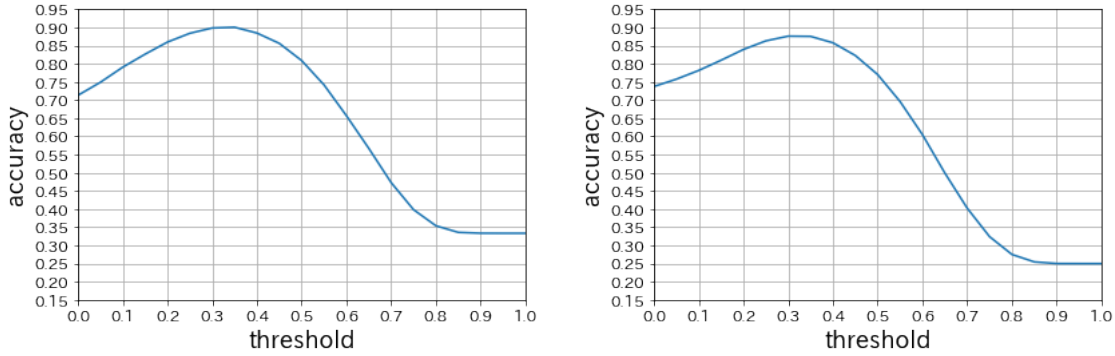

図 5: 3 クラス分類（左）と 4 クラス分類（右）に対する閾値ごとの正答率の平均値。

4.4　同一車両のドライバー分類

この節では、4.2 節で学習したモデルを特徴量生成器とみなす。モデル構築に使用していないドライバーのトリップデータに対して、教師なし分類を行う。教師なし分類を行う対象である 2 人のドライバーとしては、より現実的な状況に合致させるため、同じ車両を使用しているドライバーのデータを用いることとする。

4.4.1　教師あり分類

教師なし分類を行う前に、まずは教師あり学習モデルとしての精度を確認する。前節までと同様に対象 2 人の走行データのみを用いて ArcFace モデルを学習する。ここで、AdaCos はクラス数が 2 の場合には、s を適切に設定することができない。そのため、AdaCos ではなく ArcFace を用いる。ハイパーパラメータとしては、$s = 10$、$m = 0.05$ を採用する。この分析の目的としては、教師なし分類モデルの精度を教師あり分類モデルの精度と比較するためである。また、ドライバーの分類においては走行場所や使用した車両の違いによる影響を受けていることが考えられる。これまでの分析においてもこれらの情報をもとに分類している懸念がある。そのため、これらの条件を揃えたデータに対しても分類可能であるか、すなわち意図した通りにドライバーの運転挙動をもとにドライバーを分類することが出来ているかどうかを確認することができる。

分析対象の 2 人のドライバーを区別するため、それぞれドライバー 1 およびドライバー 2 と呼ぶことにする。観測時間として 300 秒を採用すると、ドライバー 1 のトリップ総数は 198 トリップ、ドライバー 2 のトリップ総数は 132 トリップとなった。ここで、2 人のドライバーのトリップのうち同じ車両を使用している場合のトリップデータのみに絞っている。そのた

め、2 人のドライバーは表 4 の人数に含まれているが、当該分析においては 200 トリップ未満となっている。観測間隔について、教師あり分類においてはデータ数の観点から $\Delta = 1/15$ 秒とすることも可能である。しかしながら、比較対象である教師なし分類モデルにおいては 4.2 節のモデルを特徴量生成器として使用するため、$\Delta = 1$ 秒とする。これらの 2 人のトリップデータに対して、ドライバー数の割合を保ったまま、6 割を学習データ、2 割をバリデーションデータ、2 割をテストデータに分割し、モデルの構築およびテストデータに対する分類精度を確認する。

分類結果としては、テストデータ 66 データのうち、65 データが正しく分類でき、正答率は 98.5%という結果となった。このことにより、同一の車両であってもドライバーの運転挙動には個人差があり、個人差を適切に抽出して分類できることが確認された。

4.4.2 教師なし分類

次に 4.2 節において 1,000 人規模のドライバーのトリップを用いて学習した各モデルを特徴量生成器とみなす。2 人のトリップデータから作成した特徴量ベクトルに対して、球面 k-means++法を用いた教師なし分類モデルの評価を行う。ここで、2 種類の分析を行った。1 つ目の分析では、ドライバーのトリップ全体を使用したクラスタリングによる分類の精度を確認する。2 つ目の分析では、第一段階として前項のテストデータを除いたデータでクラスタリングを行う。第二段階としてテストデータの特徴量ベクトルと第一段階で作られたクラスタ代表値とのコサイン距離をもとに所属するクラスタを決定する。この方法により、テストデータを分類し、精度を教師あり分類の精度と比較する。実用時の運用方法を考えると、まずはクラスタの情報がない状態でのクラスタリングを行う。その後過去のクラスタは固定された状態で新しく得られたデータが属するクラスタを決定することになる。二段階の分析はそれぞれに対応している。

1 つ目の分析として、ドライバー 1 とドライバー 2 の全データである 330 トリップ (=198+132) を対象にクラスタリングを行う。精度を測る評価指標について、球面 k-means++法はクラスタリング手法なのでドライバーを推定する手法ではない。しかしながら、結果として各ドライバーのトリップが属するクラスタに偏りが発生するため、各ドライバーのトリップが属する数が多い方のクラスタに属するトリップを当該ドライバーのトリップであると予測する。この予測結果を用いて、ドライバーごとの正答率を評価する。また、クラスタリングに用いるクラスタ数は、事前に対象ドライバー数が既知のものとして $k = 2$ と設定する。

ResNet、ArcFace、AdaCos をそれぞれ特徴量生成器と見なした場合の球面 k-menas++法によるクラスタリングの結果は、それぞれ表 7、表 8 および表 9 の通りであり、ResNet よりも ArcFace および AdaCos の方が良い分類結果となった。これは ArcFace および AdaCos がコサイン類似度をベースとしたロス関数を用いているため、コサイン距離をベースとしている球面 k-means++法と整合性があるためであると考えられる。特に学習モデルに含まれるドライバーのトリップに対して球面 k-means++法を行う場合には、代表特徴量ベクトルは異なるものとなるが、教師あり学習モデルを行う場合と同等の結果となることが予測される。

表 7: ResNet を用いた SKM++

ResNet	クラスタ 0 の個数	クラスタ 1 の個数	正答率
ドライバー 1	182	16	91.9%
ドライバー 2	3	129	97.7%

表 8: ArcFace を用いた SKM++

ResNet	クラスタ 0 の個数	クラスタ 1 の個数	正答率
ドライバー 1	192	6	97.0%
ドライバー 2	3	129	97.7%

表 9: AdaCos を用いた SKM++

ResNet	クラスタ 0 の個数	クラスタ 1 の個数	正答率
ドライバー 1	193	5	97.5%
ドライバー 2	3	129	97.7%

次に、教師あり学習モデルと比較するため、2 つ目の分析を行う。まず、テストデータ 66 トリップを除いた 264 トリップでクラスタリングを行う。そして、テストデータから作成された各特徴量ベクトルに対して、クラスタ代表値とのコサイン距離を測る。各テストデータについて、コサイン距離が近いクラスタに属するとして各トリップのドライバーを推定する。モデルとしては、全トリップを用いた場合の分析で一番精度の高かった AdaCos を使用する。

結果としては、表 10 の通りであり、教師ありモデルと同等の結果となった。

表 10: テストデータに対する球面 k-means++の結果

ResNet	クラスタ 0 の個数	クラスタ 1 の個数	正答率
ドライバー 1	39	1	97.5%
ドライバー 2	0	26	100%

4.4.3 結果の可視化

前節の 2 つ目の分析において、クラスタリングの結果に大きい影響を与える入力時系列データの時点を確認する。文献 [11] で提案された手法である CAM（class activation mapping）の考え方を準用した説明可能性を付与する手法を提案する。

テストデータに属するドライバー i、トリップ k のトリップから作成された特徴量ベクトルを $\boldsymbol{z}^{i,k} \in \mathcal{R}^d$、GAP 層への入力値を $\boldsymbol{y}^{i,k} \in \mathcal{R}^{\frac{T}{\Delta} \times d}$ とおく。以下ではドライバーおよびトリップを 1 つ固定して考え、$\boldsymbol{z} = \boldsymbol{z}^{i,k}$、$\boldsymbol{y} = \boldsymbol{y}^{i,k}$ のようにインデックスを略記することとする。GAP の計算方法より、以下の式が成立する。

$$\boldsymbol{z} = \frac{1}{T/\Delta} \sum_{t=1}^{T/\Delta} \boldsymbol{y}_t$$

テストデータの所属するクラスタを決定するためには、$\boldsymbol{z}$ とクラス c のクラスタ中心 $\boldsymbol{u}^c \in \mathcal{R}^d$ とのコサイン距離を計算し、値が最小となるクラスタのラベルを付与する。コサイン距離が最小となることとコサイン類似度が最大になることは同値であるため、コサイン類似度を用いて比較を行うことを考える。

$\boldsymbol{z}$ と $\boldsymbol{u}^c$ とのコサイン類似度は

$$\begin{aligned}\frac{1}{||\boldsymbol{z}||_2} \sum_{l=1}^{d} z_l u_l^c &= \frac{1}{||\boldsymbol{z}||_2} \sum_{l=1}^{d} \frac{1}{T/\Delta} \sum_{t=1}^{T/\Delta} y_{t,l} u_l^c \\ &= \frac{1}{||\boldsymbol{z}||_2} \frac{1}{T/\Delta} \sum_{t=1}^{T/\Delta} \sum_{l=1}^{d} y_{t,l} u_l^c\end{aligned}$$

という表現が可能である。したがって、CAM^* を

$$CAM_c^*(t) = \sum_{l=1}^{d} y_{t,l} u_l^c$$

と定めると、コサイン類似度は $CAM_c^*(t)$ を時点 t について総和を取ったものを正規化した値である。正規化項はクラス c および時刻 t に依存しない。したがって、$CAM_c^*(t)$ の値は、クラスタ c のラベルであるという予測結果に対する、入力時系列の時点 t の値の寄与度を表すと考えることができる。

テストデータに対してCAM^*を計算し、予測結果のクラスにおけるCAM^*を入力時系列と合わせて可視化したものが図6である。画像解析や自然言語処理分野のタスクにおいては、入力データの重要な領域を可視化することで、人間にも直接的に解釈することができるようになることが多いと考えられる。しかしながら、時系列データの場合は、その結果を解釈することは容易ではなく、理解するために専門的な知識が必要になることもある。直観的な理解までは行かないが、以下に挙げるように、可視化された図から読み取れる事項もある。これにより、モデルの信頼性確保、変数の重要性の確認や特徴量の設計方針の検討などに一定程度寄与することができる。

- 同じ時点の情報をもとに分類しているというような意図しない情報を基に分類しているわけではない。また角度の変化とCAM^*の関係から、ドライバーごとに道を曲がるか否かという走行場所の違いからドライバーを判別しているわけではないことが読み取れる。これはモデルに一定の妥当性があることを示唆している。
- 加速度および加加速度が変動している部分が結果への寄与度が大きいと解釈できる。
- 速度の変動は他の変数に比べて結果への寄与度は低いと解釈できる。

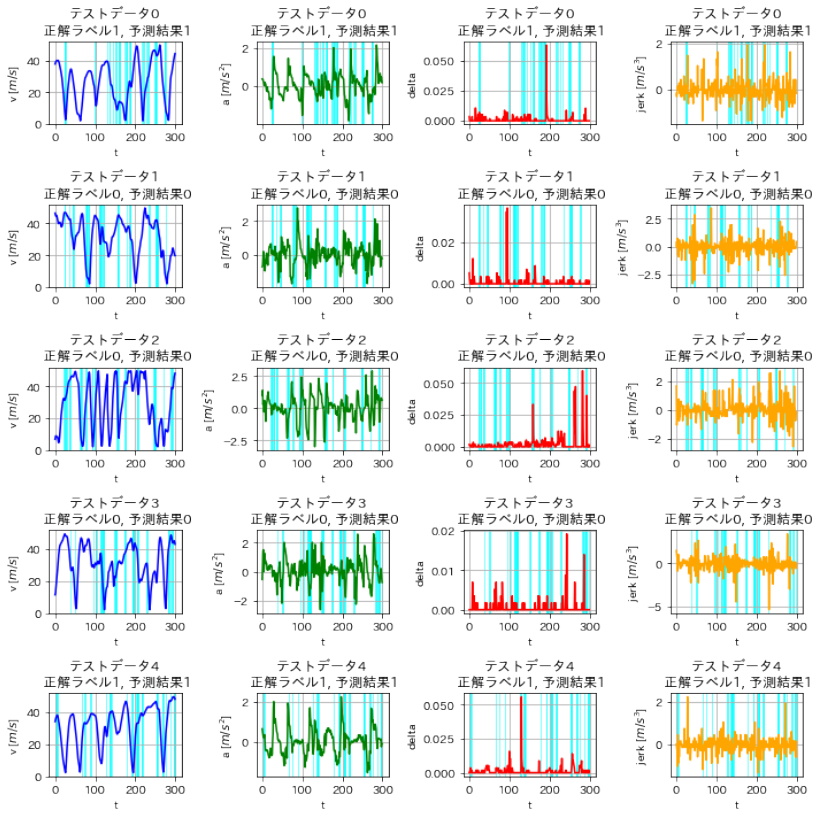

図 6: テストデータに対する CAM^* による可視化：各行の画像は同一のテストデータに対応するものであり、縦軸はそれぞれ速度（青色）、加速度（緑色）、角度の変化（赤色）、加加速度（黄色）を図示しており、横軸は観測時点（秒）を表している。さらに、予測結果の CAM^* の値の上位 10%を重要な入力時点と見なして、背景をシアン色に塗っている。

5　結論と今後の展望

ドライバーごとの運転挙動として、速度、加速度、進行方向の変化、加加速度という4変数の時系列情報を用いて、ドライバー分類モデルの構築を行った。1,000人規模の分類タスクを通じて、走行データには1,000人規模のドライバーを判別できるほどの個人差に関する情報が含まれている事が判明した。さらに、たった数分間の走行データからでも、ドライバーの運転特徴を十分に抽出可能であることが確認された。また、顔認証で用いられる深層距離学習モデルを用いることで、ドライバー認識モデルの構築を行った。これにより、多くのドライバーの組み合わせに対して、ドライバーの認識が可能であることを確認した。さらに、ArcFaceおよびAdaCosという深層距離学習モデルを特徴量ベクトル生成器として使用し、これらのモデルのロス関数と整合的な尺度であるコサイン距離によるクラスタリングを行った。この方法により、高精度の教師なし分類モデルを構築することができた。2人のドライバー分類に対しては、教師あり学習モデルと同等の水準の精度となることを確認した。

今後の展望としては以下の六点が挙げられる。

第一に、ドライバー分類モデルのビジネスでの実用可能性を判断するうえで重要な要素を確認する。4.2節の分析に用いたモデルの学習にはGe Force RTX 2060 SuperというGPUを使用した。モデルの学習時間は1エポックあたり85秒から90秒程度であり、200エポックで5時間ほど必要とする。また、推論時間については、テストデータ100サンプルに対して0.3秒程度で分類ができ、高速に推論が可能である。さらに、ドライバー分類モデルをアプリに搭載することを考える際にはモデルの容量も重要となる。各モデルの容量は、ResNetは1,035 KBであり、ArcFace、AdaCosはどちらも756 KBである。

第二に、ドライバー認識モデルとドライバーの教師なし分類モデルのそれぞれを個別に構築した。しかしながら、教師なし分類モデルにおいてもドライバー認識モデルと同様に、閾値を定めることで想定外のドライバーも含めて判別するモデルを構築することが考えられる。この観点からは、クラスタ内のデータが集まるとともに、異なるクラスタとの距離が十分に離れていることが望ましい。そこで、4.4.2項の全データのクラスタリングにおいて、球面k-means++法の目的関数の値および所属していないクラスタの代表値との距離の総和を算出し、モデルごとの比較を行う。結果は表11の通りであり、ResNetを特徴量生成器とした球面k-means++法はクラスタ内のデータが集まる傾向にあるが、異なるクラスタとの距離が近いことが分かる。これはResNetが異なるクラスのデータの距離が離れるよう

に学習していないためだと考えられる。ドライバー認識の観点からは、クラスタ内の平均コサイン類似度と異なるクラスタとの平均コサイン類似度の差が一番大きい ArcFace が一番良いモデルとなる可能性が高いと予測される。

表 11: モデルごとの球面 k-means++の目的関数および異なるクラスタの代表値との距離の総和とそれらの差およびそれらの数値から算出されるコサイン類似度の平均値

	ResNet	ArcFace	AdaCos
目的関数の値 (J)	286	296	290
異なるクラスタの代表値との距離 ($\tilde{J}$)	417	465	431
上記の差 ($J - \tilde{J}$)	132	169	141
クラスタ内のコサイン類似度の平均値	0.63	0.60	0.62
異なるクラスタの代表値とのコサイン類似度の平均値	0.23	0.09	0.19

第三に、本研究においては、走行データとドライバーという情報のみを用いて分類を行った。しかしながら、運転挙動は走行場所という地理的な要因、天候や時間などの環境的要因、さらには自動車の車種やドライバー個人の心理的な状態などの要因にも影響を受けることが考えられる。4.4 節の分析により、同じ車両を使用する異なるドライバーを判別することはできたが、反対に同一のドライバーで使用する車両を変更した際に、同一ドライバーであることを認識できるかも実務において重要であると考えられる。そのため、これら運転挙動に影響を与えると考えられる定性的な要因を変化させた状態での走行データをテストデータとして用いた場合にも、適切に判別が可能であるのかに興味がある。また、ドライバーの認識を目的とするものではないが、ドライバーの運転挙動の変化を捉えることにより、例えばロードレージの防止、老化などによる運動能力の低下による運転挙動の変化の検知、飲酒運転・酒気帯び運転の検知に活かすことができる可能性がある。

第四に、モデルの構造について考える。各モデルに使用するフィルター数を 64 から 128 に増やすことで、4.2 節の分析の正答率は向上することが確認された。具体的には、ResNet が 62.9%、ArcFace が 59.0%、AdaCos が 57.5%となった。しかしながら、4.3 節のドライバー認識モデルや 4.4 節の教師なし分類モデルでは精度の向上は確認されなかった。さらに、フィルター数を増やすことにより、モデルの容量は 3 倍以上になることが判明した。また、分析には ResNet および ResNet をベースとしたモデルを構築したが、ResNet は 2015 年に発表されたモデルである。ResNet の発表以降も画像解析分野において精度が改善されたモデルが複数提案されている。具体的には、文献 [12] で提案された SENet や文献 [13] で提案された CBAM 構造を組み込んだモデルなどがある。時系列データに対しての精度比較はなされて

いないが、これらのモデルを用いることで、さらに精度が向上できる可能性がある。また、ArcFace や AdaCos はベースとなる深層学習モデルにシンプルな修正を加えることで構築されるため、比較的理解しやすくかつ実装が比較的容易であるという利点がある。しかしながら、顔認識分野においても、例えば文献 [14] で提案された BioMetricNet など、ArcFace や AdaCos の発表以降に精度の改善が報告されたモデルが存在する。それらのモデルを用いることにより、認識精度の向上が期待できる。

第五に、入力データにおける観測時点の結果への寄与度については可視化をすることができたが、変数ごとの結果への寄与度にも興味がある。今回使用した ResNet においては、初めの畳み込み層において変数ごとの影響をミックスしてしまうため、CAM ベースの手法により各変数の結果への寄与度を確認することができなかった。例えば、各変数の影響を出力層まで残すようなモデルを構築することや、CAM ベースだけでなく変数ごとの重要性を測る説明化手法を合わせて使用することが考えられる。

最後に、テレマティクスデータを用いた研究としては、保険会社において重要な保険料率の算定に直結する自動車事故頻度予測への応用が盛んに行われている。テレマティクスデータを用いた事故頻度モデリングに関する研究を紹介する。文献 [15] では、速度と加速度の組み合わせごとの頻度を表す v-a ヒートマップを次元削減したものを説明変数に加えた GAM（一般化加法モデル）を用いて事故頻度モデルを構築した。文献 [16] では、年齢や地域などの従来から料率算定に使用されている指標を説明変数とした GAM モデルに対して、v-a ヒートマップを入力とする CNN を構築することによりブースティングを行い、予測精度を向上させる方法を提案した。文献 [17] では、急ブレーキ発生率の違いに対して、混合分布モデルを用いたクラスタリングによる特徴量定義方法を提案し、当該手法で作られた特徴量を用いたポアソン回帰により急ブレーキの発生頻度と事故との相関を調べた。また、文献 [18] では、走行距離、走行距離に対するアクセル回数、ブレーキ回数およびスピード超過回数を説明変数とするポアソン回帰モデルを構築することにより、AIC を基準として、それぞれの変数の有用性を示した。文献 [19] では、事故頻度に対する走行距離および保険期間という暴露量の影響が線形ではないことを GAM を用いて示し、さらに GAM を用いて構築された事故頻度モデルから実務に適した形の料率表を作成する方法を提案している。

これらの研究を踏まえて、本研究にてたった数分程度の走行データからでもドライバーごとの運転挙動に関する個人差を抽出することが可能であることが判明したため、この情報を自動車事故頻度の推定に活用することが考えられる。上記の事故頻度予測にドライバー

の運転挙動を反映する先行研究においては、特徴量として速度や加速度に対して定めた閾値を超えた回数や v-a ヒートマップを用いている。これらは観測期間が短いと特徴量が大きく変動し、安定しないことや、要約する段階で時系列的な情報が欠落するという欠点を有する。数分程度の走行データから運転者の運転挙動に関する個人差を抽出し、事故頻度の予測を行うモデルを構築することでこれらの欠点を改善した特徴量が作成されることが期待できる。特徴量作成方法としては、本研究で学習したモデルにおいて、個人差に関する情報を有する特徴量ベクトルを次元削減し、説明変数とすることが考えられる。より直接的な方法も存在する。目的変数をドライバーごとの事故回数とし、ResNet の出力層の次元を 1 次元にし、出力層の活性化関数として指数関数を用いるように修正する。さらに、ロス関数としてポアソンデビアンス残差を採用することで、入力データを時系列データとしたまま事故頻度の推定を行うことができる。後者の方法は、特徴量エンジニアリングを必要としないことや、より事故と相関のある特徴が抽出されるため、予測の観点からは望ましいかもしれない。課題としては、多くの事故は低速度帯で発生していることが知られており、通常の走行時に起こることは少ない。そのため、通常時の走行データではなく、信号での発進時、停車時または道を曲がる際などの危険挙動が発生する可能性がある時点の時系列データを入力とするなどの工夫が必要となると推測される。

謝辞

本研究を進めるにあたり、多くの方々にご協力いただきました。指導教員である田中琢真准教授には、研究の進め方から論文の執筆まで多くのご指導をいただき、深く感謝いたします。副指導教員を引き受けてくださった笛田薫教授には、主にテレマティクスデータを用いた事故頻度モデリングの研究に関してご教授いただき感謝申し上げます。また、同じ研究室に所属する学生の皆様、共にデータサイエンスの勉強に励んだ大学院の同級生の皆様に感謝いたします。

そして、社会人派遣学生として送り出してくださった、あいおいニッセイ同和損害保険株式会社の皆様に深く感謝し、お礼を申し上げます。特に、滋賀大学への派遣にあたりご支援いただいた人財革新グループの皆様、研究環境の整備やビジネス目線での助言などのお力添えいただいたデータソリューション室の皆様には厚く感謝申し上げます。また、社会人派遣生として滋賀大学に通われた会社の先輩方には研究のことから日常生活の困りごとまで広くご助力いただきました。心から御礼申し上げます。

参考文献

[1] あいおいニッセイ同和損害保険株式会社. “ささえる NAVI「Lite」.” https://www.ad-sasaerunavi.com (2021 年 1 月 5 日).

[2] 東京海上日動火災保険株式会社. “ドライブエージェント　パーソナル.” https://www.tokiomarine-nichido.co.jp/service/auto/total-assist/shohin/dap.html (2021 年 1 月 5 日).

[3] 岡田 将吾, 人見 謙太郎, ナイワラ P. チャンドラシリ et al. (2012). “車載センサログの時系列データマイニングに基づく運転挙動の分析.” IEICE Conferences Archives. The Institute of Electronics, Information and Communication Engineers, 2012.

[4] Mario V. Wüthrich (2017). “Covariate selection from telematics car driving data.” European Actuarial Journal 7.1 : 89–108.

[5] Guangyuan Gao and Mario V. Wüthrich. (2019). “Convolutional neural network classification of telematics car driving data.” Risks 7.1

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren et al. (2016). “Deep residual learning for image recognition.” Proceedings of the IEEE conference on computer vision and pattern recognition.

[7] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber et al. (2019). “Deep learning for time series classification: a review.” Data mining and knowledge discovery 33.4: 917–963.

[8] Jiankang Deng, Jia Guo, Niannan Xue et al. (2019). “Arcface: Additive angular margin loss for deep face recognition.” Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.

[9] Xiao Zhang, Rui Zhao, Yu Qiao et al. (2019). “AdaCos: Adaptively scaling cosine logits for effectively learning deep face representations.” Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition.

[10] 遠藤靖典, 徳山晴紀, 宮本定明 (2015). “球面 k-means++ 法について.” 日本知能情報ファジィ学会 ファジィ システム シンポジウム 講演論文集 第 31 回ファジィシステムシンポジウム. 日本知能情報ファジィ学会.

[11] Bolei Zhou, Aditya Khosla, Agata Lapedriza et al. (2016). “Learning deep features for discriminative localization.” Proceedings of the IEEE conference on computer vision and pattern recognition.

[12] Jie Hu, Li Shen and Gang Sun. (2018). “Squeeze-and-excitation networks.” Proceedings of the IEEE conference on computer vision and pattern recognition.

[13] Sanghyun Woo, Jongchan Park and Joon-Young Lee et al. (2018). “CBAM: Convolutional block attention module.” Proceedings of the European conference on computer vision (ECCV).

[14] Arslan Ali, Matteo Testa, Tiziano Bianchi et al. (2020). “BioMetricNet: deep unconstrained face verification through learning of metrics regularized onto Gaussian distributions.” European Conference on Computer Vision. Springer, Cham.

[15] Guangyuan Gao, Shengwang Meng, and Mario V. Wüthrich. (2019). “Claims frequency modeling using telematics car driving data.” Scandinavian Actuarial Journal 2019.2 : 143–162.

[16] Guangyuan Gao, He Wang, and Mario V. Wüthrich. (2021). “Boosting poisson regression models with telematics car driving data.” Machine Learning : 1–30.

[17] 荒井 隆, 笛田 薫 (2019). “加速度時系列データを用いた自動車事故と運転挙動との相関性分析.” 第 36 回応用経済時系列研究会報告集.

[18] 渋谷雄平 (2021). “テレマティクスデータを用いた自動車事故リスク評価に関する考察.” 滋賀大学大学院データサイエンス研究科修士論文.

[19] Jean-Philippe Boucher, Steven Côté, and Montserrat Guillen. (2017). “Exposure as duration and distance in telematics motor insurance using generalized additive models.” Risks 5.4.