JAIST Repository
https://dspace.jaist.ac.jp/

Title	オンラインゲームDota2の勝敗予測のためのチーム特徴に関する研究
Author(s)	熊, 能
Citation	
Issue Date	2022-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/17641
Rights	
Description	Supervisor:由井薗　隆也, 先端科学技術研究科, 修士（情報科学）

JAIST
JAPAN
ADVANCED INSTITUTE OF
SCIENCE AND TECHNOLOGY

Japan Advanced Institute of Science and Technology

修士論文

オンラインゲーム Dota2 の勝敗予測のためのチーム特徴に関する研究

Xiong, Neng

主指導教員　　由井薗　隆也教授

北陸先端科学技術大学院大学
先端科学技術研究科
（情報科学）

令和 4 年 3 月

Abstract

In recent years, online games have developed rapidly. One of the most famous online games type among them is MOBA.

Multiplayer online battle arena (MOBA) is a kind of Action Real-Time Strategy game(ARTS). Each player controls a hero with unique abilities in the game. The hero grows by killing neutral creatures or enemy heroes, and the team that destroys the enemy base first wins.

Dota2 is a MOBA game with the highest prize money in the world. There are ten players participating in one dota2 game. These ten players are divided into two teams to fight each other. The team that destroys the opponent's base first wins.

In dota2 competitions, predicting the outcome of the game is very important. With accurate win-loss predictions, professional players can train more efficiently.

In past outcome win-loss prediction studies, researchers have focused on hero combinations. But didn't get good results. In the dota2 game, each player has heroes that are good or bad, so players and teams should be considered when predicting the outcome. In this paper, we focus on the features of teams, and propose a method for generating features of teams.

The proposed feature generation method is divided into two parts. The first part is the feature generation method based on the player's performance, and the second part is the team economic feature generation method based on the player economy. Based on the features generated by these two methods, we used three machine learning models of Naive Bayes, Random Forest and XGBoost for training. The result is 82% for Naive Bayes, 95% for Random Forest, and 98% for XGBoost. The results allow us to predict the outcome of a match with a fairly accurate degree of player and team performance.

目次

第 1 章 はじめに 1

1.1 研究背景 1

1.2 研究目的 2

1.3 本論文の構成 3

第 2 章 関連研究 4

2.1 緒言 4

2.2 Dota2 とは 4

2.3 先行研究 6

2.3.1 ヒーロー構成に基づいた勝負予測の先行研究 7

2.3.2 プレイヤー、チームに関する先行研究 8

2.3.3 機械学習アルゴリズムが Dota2 の勝負予測研究における性能の先行研究 9

2.4 結言 9

第 3 章 チームに関する特徴作成 10

3.1 緒言 10

3.2 問題点 10

3.3 特徴の作成手法 11

3.3.1 試合前の特徴作成 11

3.3.2 選手パフォーマンスに着目したチーム特徴作成手法 12

3.3.3 ゴールドに着目したチーム特徴の作成手法 17

3.4 結言 23

第 4 章 評価実験 24

4.1 緒言 24

4.2 データ収集 24

4.2.1 OPENDOTA 24

4.2.2 ウェブクローラー 25

4.3 データの抽出と処理 26

4.4 機械学習モデルの構築 28

4.5 結言 33

第 5 章 実験結果と考察 34

5.1 緒言 34

5.2 実験結果 34

5.2.1 全体説明 34

5.2.2 ヒーロー構成と APM に基づいたベースラインの作成.....35

5.2.3 選手パフォーマンスに着目したチーム特徴を追加した実験結果.....40

5.2.4 ゴールドに着目したチーム特徴を追加する 44

5.2.5 両方を追加する .....**错误!未定义书签。**

5.3 考察 52

5.4 結言 56

第 6 章 おわりに 57

6.1 結論 57

6.2 将来の展望 58

図目次

図 1 Dota2 の地図と役割の配置 4

図 2 選手パフォーマンスに基づいた特徴の作成概念 14

図 3 チーム特徴の分析 18

図 4 チームのゴールド特徴の計算 20

図 5 正規化処理 22

図 6 ウェブクローラーのフローチャート 26

図 7 ヒーロー選択時の順番 27

図 8 ベイズ公式 29

図 9 バギング思想 30

図 10 ランダムフォレスト 31

図 11 データの割合 33

図 12 四つ実験のそれぞれの精度比較 35

図 13 ヒーロー構成と選手 APM に注目したベースラインナイーブベイズの訓練結果 37

図 14 ヒーロー構成と選手 APM に注目したベースラインのランダムフォレ

ストの訓練結果.....38
図 15 ヒーロー構成と選手 APM に注目したベースライン XGBoost の訓練結果.....39
図 16 ヒーロー構成と選手 APM に注目したベースライン訓練精度の比較.....40
図 17 選手パフォーマンスに着目したチーム特徴を追加したナイーブベイズの訓練結果.....41
図 18 選手パフォーマンスに着目したチーム特徴を追加したランダムフォレストの訓練結果.....42
図 19 選手パフォーマンスに着目したチーム特徴を追加した XGBoost の訓練結果.....43
図 20 選手パフォーマンスに着目したチーム特徴を追加して訓練精度の比較.....44
図 21 ゴールドに着目したチーム特徴を追加したナイーブベイズの訓練結果.....45
図 22 ゴールドに着目したチーム特徴を追加したランダムフォレストの訓練結果.....46
図 23 ゴールドに着目したチーム特徴を追加した XGBoost の訓練結果....47

図 24 ゴールドに着目したチーム特徴を追加して訓練精度の比較.....48

図 25 チーム特徴全体を追加したナイーブベイズの訓練結果 49

図 26 チーム特徴全体を追加したランダムフォレストの訓練結果.....50

図 27 チーム特徴全体を追加した XGBoost の訓練結果.....51

図 28 作成したチーム特徴全部追加して訓練精度の比較.....52

表目次

表 1 抽出した選手の APM（一部） 12

表 2 picks_bans と選手の APM（一部） 12

表 3 抽出した選手 K、D、A（一部） 13

表 4 選手パフォーマンスに基づいた特徴作成 15

表 5 選手パフォーマンスに着目したチーム特徴..... 17

表 6 抽出した倒した対象の数（一部） 18

表 7 構造したプレイヤーのゴールド表 19

表 8 チームのゴールド総量 20

表 9 処理したチームのゴールド総量..... 21

表 10 ゴールドに着目したチーム特徴..... 23

表 11 抽出した picks_bans..... 28

表 12 picks_bans を処理したデータセット（一部） 28

表 13 本研究の訓練時間の比較（チーム特徴全体追加した場合） 53

表 14 先行研究との比較 54

第1章 はじめに

1.1 研究背景

インターネットの発展と資本の流入と共に、e スポーツは近年急速に発展しており、すでに重要なスポーツ競技の一つとなっている。近年、世界中の e スポーツ視聴者は 3 億 8500 万人以上とされる[1]。伝統的なスポーツ競技と比べて、e スポーツ業界のデータ量は大きく、アクセスしやすいという特徴がある。e スポーツの中で、今最も人気があるゲームの一つは Dota2 で、世界最大級のゲームプラットフォーム Steam でのプレイヤー数は常に一位か二位である（https://store.steampowered.com/stats/? 2021 年 1 月 15 日）。

サッカーなど伝統的なスポーツ競技と違い、e スポーツはゲームプログラムに基づいて構築されている。これにより、選手に関する情報をより正確に収集することができる。

Dota2 において、122 個の異なるヒーローがいる（2022 年 1 月まで）。従来の勝負予測研究はヒーローの Ban&Pick（BP）に注目し、ヒーローの構成を通して試合の結果を予測していた[6]。ヒーローの組み合わせに注目する場合、それぞれのヒーローは一回のみ選択できることで、122 個ヒーローの場合は約

$$C_{122}^{5} \times C_{117}^{5} \times \frac{1}{2} \approx 1.73 \times 10^{16}$$

種違った組み合わせがある。また、それぞれのヒーローは独特な能力があって、それに基づいたヒーローの構成は違ったメリットやデメリットが存在している。故に、ヒーローの Ban&Pick だけに注目した勝負予測は不十分である。プロの世界で、トップレベルの選手でもすべてのヒーローをうまくマスターすることは不可能なことである。その上に、選手はそれぞれの得意と苦手なヒーローがいる。得意なヒーローを使うことで、試合がよりスムーズに進むと考えられる。

つまり、e スポーツの Dota2 において、ヒーロー構成のみに注目した試合勝負予測研究には限界がある。本研究では、試合勝負予測の精度を上げるため、チームに関する情報に着目する。そのために、機械学習でチームに関する特徴量を提案し、それが e スポーツの試合の予測に対する影響を調べる。

1.2 研究目的

E スポーツにおいて、同じヒーローの構成でも、選手が違うことで試合の結果も変わる。選手と選手が組んだチームを把握することで、試合の結果をより正確に予測できる。しかし、今までの勝負予測研究はチーム特徴を考慮していなかった。故に、チーム特徴を考慮する必要があると想定している。

しかしながら、現在の e スポーツ世界ではアクセスできる試合データにはチーム特徴は存在しない。

そこで、本研究で Dota2 のプロ試合に着目し、選手の試合パフォーマンスに

基づき、チーム特徴を作成する。それに基づいて、Dota2 の多くの特徴の中で、チーム特徴が Dota2 の試合勝負予測を向上できるかどうか明らかにする。

1.3 本論文の構成

本論文の構成は以下の通りである。第 1 章では本研究の研究背景と研究目的を述べた。第 2 章では関連知識、関連情報と先行研究を紹介した後、従来の研究の問題点を述べる。第 3 章では選手とチーム特徴の提案手法を説明し、機械学習モデルの構築を述べる。第 4 章では実験設計と実験結果説明する。第 5 章では実験の評価とその考察する。第 6 章の終わりにでは、実験の結果に基づき、結論をまとめる。最後に、本研究の不足と今後の研究展望を説明する。

第2章 関連研究

2.1 緒言

本章では、最初は関連知識として、Dota2 について述べる。その後は先行研究とその問題点について述べる。

2.2 Dota2 とは

Defense of the Ancients 2（Dota2）は Valve が発行された Multiplayer Online Battle Arena（MOBA）ゲームである。10 人のプレイヤーが地図の左下にある陣営（Radiant）と右上の陣営にある（Dire）二つチームに分かれる。

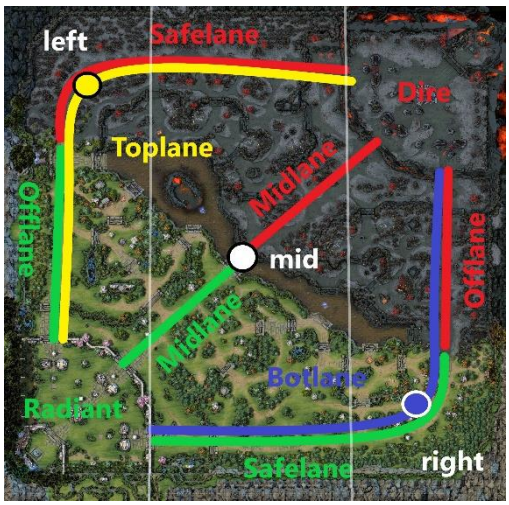


図 1 Dota2 の地図と役割の配置

(https://steamcommunity.com/)

(2022 年 1 月 15 日)

Dota2 では、それぞれのプレイヤーが一つのヒーローを操作する。ヒーローはゲーム内のゴールドを使ってアイテムを購入することと敵を倒して経験値を得ることで強くなれる。ヒーローが倒された場合、ゴールドを失い、ある時間の後で復活する。失ったゴールドと復活の時間はヒーローが倒された時点のレベルに関係がある。プレイヤーは敵ヒーローを倒すと、大量なゴールドと経験値を得られて、試合が味方に有利な方向に進める。そのために、両方のプレイヤーは敵ヒーローを倒すことに優先な目標としている。ゲーム内、最後の一撃で敵ヒーローを倒すことは Kill（K)、敵ヒーローに倒されたことは Death(D)、味方ヒーローを援助し、敵ヒーローを倒すことは Assist(A)と呼ばれている。K/D/A はある程度でプレイヤーの試合パフォーマンスを反映できる。

両陣営は上、中、下三つの道路に結ばれて、右下と左上の一番奥はエンシェントと呼ばれる本陣がある。試合の目的は敵のエンシェントを壊すことだが、道路で敵を自動に攻撃するユニットと塔が存在している。敵のユニットを倒すことは主なゴールド源である。また、道路と道路の中で中立ユニットも存在している。中立ユニットを倒すことでゴールド、経験値やアイテムをもらえる。

Dota2 は、サッカーやバスケットなどの伝統的なスポーツと違って、スコアのような客観的なリード指標は存在しない。それは、本陣が破壊された時点で試合が終了となるから。

2.3 先行研究

様々の種類のスポーツの勝者を予測するため、多くの研究が行われている。Shuo らはテニスと StarCraft II の試合結果を予測するため、コンテキスト情報を使用する確率的フレームワーク生成手法を提案した[2]。Anthony らはフットボールの勝敗予測するについて研究した[3]。

しかし、MOBA ゲームである Dota2 は他のゲームより複雑である。一つ試合で 100 以上選択できるヒーローがあって、それぞれのヒーローはまた独特な能力を持つ、プレイヤーはゲーム内のゴールドや経験値を得るためはあらゆる予測困難な行動を取る可能性がある[4]。ヒーローも違う役割を分担している[5]。試合に勝つため、プレイヤーはチームワークを重視しなければならない。また、試合の勝負を予測する際に、プレイヤーがコントロールしたユニットやコンピューターがコントロールしたユニットの増加することで、予測の複雑さが上がる。例えば、このユニットはプレイヤーのヒーローを攻撃し、ヒーローを倒すことは可能、ヒーローに倒された場合はゴールドと経験値提供する。また、MOBA ゲームには様々なアイテムがある。アイテムはヒーローにメリットやデメリットを提供できる。本質的に、MOBA ゲームは幅広い可能性が持っており、ゲームの勝負を予測することは困難なタスクになっている。

2.3.1 ヒーロー構成に基づいた勝負予測の先行研究

Song らはプレイヤーがすべてのヒーローをうまくプレイできることを仮定し、ヒーローの組み合わせに注目して試合の勝負予測について検討していた。彼らは、ヒーローの組み合わせだけに基づいて勝負を予測することは良い結果を得られず、勝負予測する際に選手の状況を考慮することは今後の課題にした[6]。

Wang らは単純ベイズ分類器を用い、ヒーロー構成に基づいて試合勝負を予測する実験を行った。彼らは単純ベイズ分類器におけるデータを用いた Dota2 勝負予測の可能性を検証し、単純ベイズ分類器は試合勝負予測ミッションにおいては高効率なモデルを示した[7]。

Kalyanaraman の研究では改進したロジスティック回帰アルゴリズムを使い、ヒーロー構成により試合の勝負予測を行い、従来のヒーロー構成に注目した勝負予測研究より高精度(74%)で予測をできた。また、彼らはチームに関する要素を加えることで試合の結果をより正確に予測できる可能性を予想した[8]。

Akhmedov の研究では Dota2 においてすべての特徴（164 個）を使い、Dota2 の勝負予測研究をしていた。彼らは LSTM（Long Short Term Memory）を使い、93%の精度で試合の勝負を予測できた。しかし、モデルの訓練の時は 100 試合を訓練するだけで 100 時間以上にかかった。彼らは特徴の選別により

勝負予測の精度とスピードが上がると討論した[9]。

以上の先行研究では、ヒーローの構成に基づいて試合の勝負を予測した研究である。彼らはヒーローの構成は試合の結果に影響があることを示している。しかし、予測の精度と選手、チームが試合の結果への影響を取り扱っていない。

2.3.2 プレイヤー、チームに関する先行研究

Arvyn の研究では SAW 法を使い、プレイヤーの KDA、毎分平均ゴールド収入（Gold per minute）を基づいて Dota2 の初心者と経験者とのデータの違いを示した[10]。

Anders らはプロ選手と初心者がゲーム内で取る行動の違いをチームレベルで示す、プロ試合ではより高いプレイヤーの連携を求め、プレイヤーの行動は試合の結果に影響があることを判明した[11]。

Grutzik の研究では機械学習を用いて、プレイヤーが過去 10 試合の毎分平均ゴールド収入、毎分平均経験値得た量（Exp per minute）毎分敵ヒーローを倒した数（Kill per minute)、ゴールドを得る効率(Lane efficiency)、個人競技ランキング（Solo competitive rank）のデータ使い、試合の勝負を予測していた。しかし、彼らの研究により、BP で試合勝負を予測した精度は 58%で、これに基づいたプレイヤーのデータを加えると、精度が 55%に下がった（SVM)。ニューラルネットワークを使った場合は 68%と 53%でした。彼らは、プレイ

ヤーに関するデータを作成したが、過去 10 試合のデータでプレイヤーに関する特徴作成は不十分であり、作成法の改進が必要なことを予想した。[12]

以上の研究ではプレイヤー、チームのデータによる試合の勝負を予測する可能性を示した。

2.3.3 機械学習アルゴリズムが Dota2 の勝負予測研究における性能の先行研究

Aleksandr の研究ではヒーローの構成により、プレイヤーのスキルレベルをノーマル、ハイ、ヴェリィハイに分けて、単純ベイズ分類器、ロジスティック回帰、XGBoost、決定木四つの機械学習アルゴリズムを使って性能を比較した。彼らは、XGBoost はこの四つのアルゴリズムの中で性能が一番高いことを示し、プレイヤーの実力が強いほど、試合の結果を正確に予測することは難しいになることを判明した。その原因は、彼らはハイレベルのプレイヤーはチームメイトへの連携より重視していることを予想した。そこで、ハイレベルの試合を正確に予測することが問題になっている[13]。

2.4 結言

本章では、Dota2 の勝負予測に関する先行研究とプレイヤーに関する先行研究を紹介した。また、先行研究の不足を討論した。

第3章 チーム特徴作成

3.1 緒言

本章では従来の研究の問題点を説明し、本研究で提案する特徴作成手法を説明する。

3.2 問題点

先行研究においては、ヒーローの構成に注目し、機械学習を用いて Dota2 の試合勝負予測研究をしていたが、チーム特徴を考慮しなかった[15]。しかし、MOBA ゲームでは同じヒーローの構成でも、選手やチームが違って試合の結果も変わる。選手とチームの特徴は試合に大きく影響を与えている。故に、MOBA ゲームの試合勝負をより正確に予測するため、選手とチームの要素を考えるべきである。また、選手とチームはどれぐらい試合の勝負に影響を与えたかを明らかにするべきである。

従来研究の問題点を解決するため、本研究では OPENDOTA で収集したプロ試合のデータを利用し、選手パフォーマンスに着目したチーム特徴とゴールドに着目したチーム特徴の作成手法を提案する。それに基づいて、試合の勝負を予測し、選手、チーム特徴と試合勝負との関連性を分析する。

3.3 特徴の作成手法

3.3.1 試合前の特徴作成

ヒーローの構成は試合の本番の前に決定したものである。ヒーローの構成だけで試合の結果を予測の正確さは低いことは先行研究ですでに検証された[6]。人間の運動能力はそれぞれである。ある運動に対して、人が訓練すればするほど運動に上手になれる。例えば、人が 100 メートルに専念し、訓練すると、100 メートルを走る時間が早くなれる。しかし、人が必ず自分の限界があって、毎回 100 メートルを走った時間は大体一緒である。この運動能力をゲームに対応すると、それは Action Per Minute（一分間にプレイヤーが行えるアクションの数）である。選手が毎回のゲーム中の APM は大体一緒であることを仮定し、選手の APM を試合前の特徴として作成し、学習データとしてデータベースに保存した（表 1)。

表 1 抽出した選手の APM（一部）

actions_per_min_1	actions_per_min_2	actions_per_min_3	actions_per_min_4	actions_per_min_5
142	179	146	165	131
112	162	126	133	164
150	137	143	154	132
153	174	211	156	143
102	173	191	143	193
164	126	185	210	150
103	143	194	146	180
142	458	145	144	142
175	165	162	133	349
123	207	159	155	170
152	145	159	177	163

本研究は先行研究のヒーロー構成に基づいて勝負を予測した研究に、選手の APM を試合前の特徴として、訓練特徴追加した。ベースラインに入力した特徴はヒーローの構成と選手の APM である（表 2）。

表 2 picks_bans と選手の APM（一部）

	target	is_pack_1	hero_id_1	team_1	is_pack_2	hero_id_2	team_2	is_pack_3	hero_id_3	team_3	...	actions_per_min_1	actions_per_min_2
0	False	False	111	1	False	32	0	False	9	1	...	142	179
1	True	None	-1	-1	None	-1	-1	None	-1	-1	...	112	162
2	True	False	28	1	False	99	0	False	98	1	...	150	137
3	True	False	103	1	False	38	0	False	29	1	...	153	174
4	False	False	2	1	False	98	0	False	93	1	...	102	173
...	...	...	...	...	...	...	...	...	...	...	...	...	...
73944	True	False	76	0	False	71	1	False	3	0	...	321	326
73945	True	False	23	1	False	91	0	False	128	1	...	435	256
73946	True	False	112	1	False	105	0	False	34	1	...	466	327
73947	True	False	15	0	False	63	1	False	91	0	...	199	246
73948	False	False	63	0	False	34	1	False	128	0	...	315	275

73949 rows × 83 columns

3.3.2 選手パフォーマンスに着目したチーム特徴作成手法

本研究では、チーム特徴が Dota2 の試合勝負予測を向上できるかどうか明らかにするため、選手のパフォーマンスに着目した特徴量を提案する。e スポーツの一種である MOBA ゲームでは、一つ試合でそれぞれのプレイヤー自身の

スコアを表示することは KDA である。試合内の KDA は本試合プレイヤーのパフォーマンスをある程度で表現できる。KDA の中で、選手の Kill と Assist が多い方は試合が有利になる。また、選手の Death が多い場合は本試合でのパフォーマンスが悪い可能性が大きい。学習全データから、十人の選手からそれぞれの Kill 、Death 、Assist を抽出した（表 3）。

表 3 抽出した選手 K、D、A（一部）

k_1	d_1	a_1	k_2	d_2	a_2	k_3	d_3	a_3
2	8	0	0	5	5	2	4	3
21	4	17	10	0	14	7	9	15
6	11	22	14	12	8	5	17	22
8	2	9	2	10	9	3	4	13
0	3	4	0	5	5	4	3	1
1	4	5	2	5	6	7	2	6
8	4	9	2	2	12	8	0	6
2	1	13	5	0	1	5	2	5
6	16	23	5	10	26	26	11	16
3	3	17	8	0	8	3	3	13
2	5	3	2	1	3	0	7	4
18	2	7	5	7	7	10	6	22

そこで、本実験における選手のパフォーマンスを KDA に基づいて定義した。作成したデータセットを利用して、選手が試合の勝利に対する Kill、Death 比率と Kill、Assist 比率を定義した（図 2）

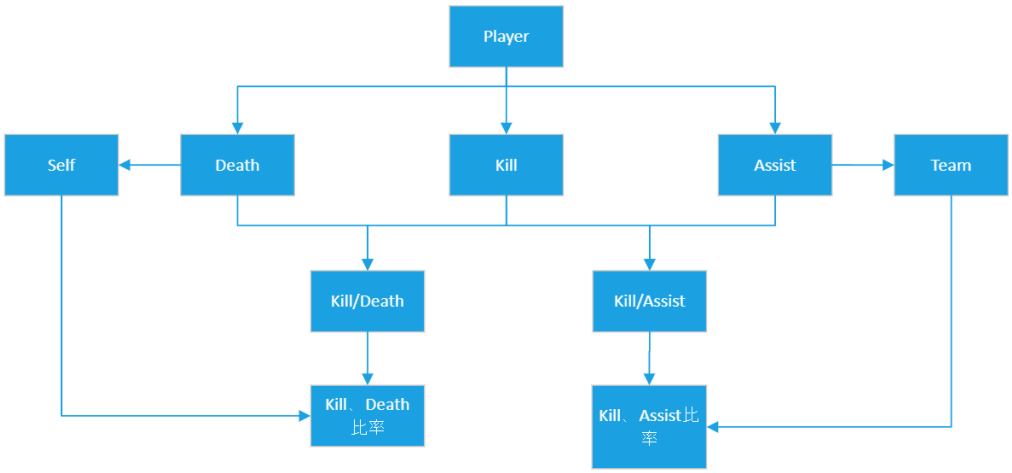


図 2 選手パフォーマンスに基づいた特徴の作成概念

以上の定義に基づいて、選手の試合でのパフォーマンスを表現できる。選手は試合での Death と Assist は 0 になる可能性があり、ゼロ除算を避けるため、それぞれの分母は 1 を足した。全体の被除数は同じ処理を行い、大きな偏差は発生しない。故に、一つ試合で、すべてのプレイヤーに対し、以下の新特徴を作成した（表 4)。

表 4 選手パフォーマンスに基づいた特徴作成

	Kill・Death 比率	Kill・Assist 比率
プレイヤー1	k1/(d1+1)	k1/(a1+1)
プレイヤー2	k2/(d2+1)	k2/(a2+1)
プレイヤー3	k3/(d3+1)	k3/(a3+1)
プレイヤー4	k4/(d4+1)	k4/(a4+1)
プレイヤー5	k5/(d5+1)	k5/(a5+1)
プレイヤー6	k6/(d6+1)	k6/(a6+1)
プレイヤー7	k7/(d7+1)	k7/(a7+1)
プレイヤー8	k8/(d8+1)	k8/(a8+1)
プレイヤー9	k9/(d9+1)	k9/(a9+1)
プレイヤー10	k10/(d10+1)	k10/(a10+1)

以上の処理を行い、データベースに保存した。また、それぞれのプレイヤーの k/d、k/a を再抽出した。

選手の KDA に注目し、チームの Kill、Assist 平均が表 5 で作成した micro 特徴を式 1 の通りに作成した。

$$micro_k/d_team1 = \frac{player1_k1 + player2_k2 + ...player5_k5}{player1_d1 + player2_d2 + ...player5_d5}$$
$$micro_k/d_team2 = \frac{player6_k6 + player_k7 + ...player_k10}{player6_d6 + player_d7 + ...player_d10}$$

$$micro_k/a_team1 = \frac{player1_k1 + player2_k2 + ...player5_k5}{player1_a1 + player2_a2 + ...player5_a5}$$
$$micro_k/a_team2 = \frac{player6_k6 + player_k7 + ...player_k10}{player6_a6 + player_a7 + ...player_a10}$$

（式 1）

チームに注目し、個人パフォーマンスの平均が表 5 で作成した macro 特徴を

式２の通りに作成した。

$$macro_k/d_team1 = \frac{player1_k1/d1 + player2_k2/d2 + \ldots player5_k5/d5}{5}$$

$$macro_k/d_team2 = \frac{player6_k6/d6 + player7_k7/d7 + \ldots player10_k10/d10}{5}$$

$$macro_k/a_team1 = \frac{player1_k1/a1 + player2_k2/a2 + \ldots player5_k5/a5}{5}$$

$$macro_k/a_team2 = \frac{player6_k6/a6 + player7_k7/a7 + \ldots player10_k10/a10}{5}$$

（式 2）

上記の計算を行い、選手パフォーマンスに着目したチーム特徴を作成した。また、チームの平均ダメージと平均経験値を計算し、ベースラインに追加して訓練した特徴は表 5 で示す。

表 5 選手パフォーマンスに着目したチーム特徴

特徴名
teamfight_time
average_death
average_t1_exp
average_t2_exp
average_t1_damage
average_t2_damage
micro_k/a_1
micro_k/d_1
micro_k/a_2
micro_k/d_2
macro_k/a_1
macro_k/d_1
macro_k/a_2
macro_k/d_2

3.3.3 ゴールドに着目したチーム特徴の作成手法

MOBA ゲームでは、チームが倒す敵が多ければ多いほど、試合が有利になる。そこで、どうすれば敵を倒すことを容易になるのか、プレイヤーは自分の操作を通して敵を倒すことは、本質的には敵に与えたダメージで求められると考えられる。敵に与えたダメージが多いチームが試合に勝ちやすいと考えられる。チームのダメージ量は、ヒーローのレベル、アイテムと強く関係があると推測した。そこで、ヒーローのレベルとアイテムの本質は経験値（Experience

point）とゲーム内のゴールド（Gold）である（図 3）。

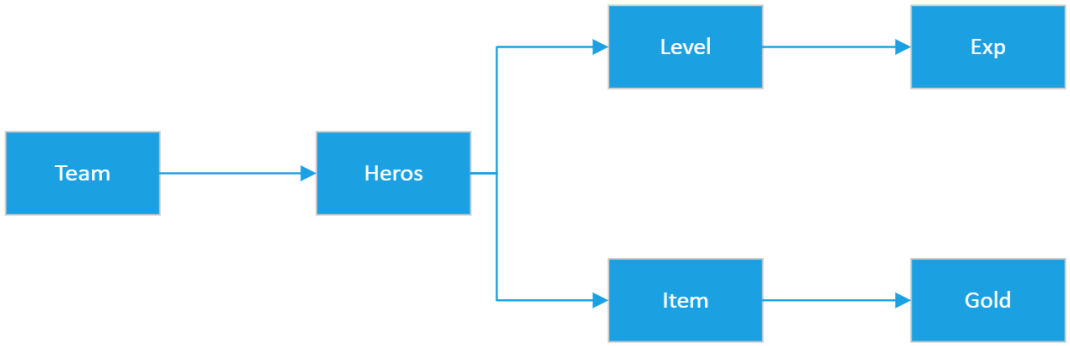


図 3 チーム特徴の分析

Dota2 のゴールドは、敵やユニットを倒すことで得られる。しかし、倒した相手によって、得られたゴールドの数が違う。Dota2 で、敵を倒してゴールドを手に入れることができる対象はトータル 9 種類がある。そこで、それぞれのプレイヤーが倒した対象を抽出し、データベースに保存した。(表 6)。

表 6 抽出した倒した対象の数（一部）

neutral_kills_1	tower_kills_1	courier_kills_1	lane_kills_1	hero_kills_1	observer_kills_1	sentry_kills_1	roshan_kills_1
0	0	1	25	2	1	0	0
9	5	0	270	21	4	2	1
8	0	0	91	6	0	2	0
51	0	0	163	9	0	0	0
6	0	0	21	0	0	0	0
12	0	0	15	1	0	0	0
108	0	0	47	8	0	0	0
1	0	0	16	2	0	2	0
1	0	1	24	6	1	1	0
5	0	0	14	3	3	0	0
33	0	0	42	1	2	0	0
15	3	0	126	17	0	0	2
41	1	0	225	5	0	0	1
71	4	0	170	8	1	0	1
4	0	0	13	3	0	0	0

本研究では、抽出した特徴に基づいて、プレイヤーのゴールド表を作成した。

Dota2 では、倒した相手によって、数が違うゴールドを得られる。プレイヤーのゴールド表は、プレイヤーが倒した対象とその数に基づく、プレイヤーが試合内で得たゴールドの数とそのソースを明らかにすることができる（表 7）。実際のゲームでは、得られるゴールドはある範囲で変化する。本研究では、その範囲の中央値を使用する。

表 7 構造したプレイヤーのゴールド表

対象名	ゴールドの数	説明
neutral	100	ジャングルにある中立ユニット
tower	300	敵のタワー
courier	200	敵のアイテムを運ぶユニット
lane	100	敵のクリープ
hero	500	敵ヒーロー
observer ward	50	設置すると、周囲の視野が得られる
sentry ward	30	設置すると、不可視なユニットが見えるようになる
roshan	200	ゲーム内の BOSS、倒すと経験値、ゴールドとアイテムが得られる
necronomicon	150	アイテムが召喚したユニット

構造したゴールド表に基づいて、それぞれのプレイヤーが試合内で得られたゴールドを計算した。また、それぞれのプレイヤーが得られたゴールドを計算し、所属するチームを計算した（図 4）（表 8）

```
for i in [1, 2]:
    data['team{}_economics'.format(i)] = data['team_neutral{}'.format(i)] * neutral + \
                                         data['team_tower{}'.format(i)] * tower + \
                                         data['team_courier{}'.format(i)] * courier + \
                                         data['team_lane{}'.format(i)] * lane + \
                                         data['team_hero{}'.format(i)] * hero + \
                                         data['team_observer{}'.format(i)] * observer + \
                                         data['team_sentry{}'.format(i)] * sentry + \
                                         data['team_roshan{}'.format(i)] * roshan + \
                                         data['team_necronomicon{}'.format(i)] * necronomicon
```

図 4 チームのゴールド特徴の計算

表 8 チームのゴールド総量

	team1_economics	team2_economics
0	44750	62630
1	135230	98330
2	132780	169920
3	76950	72200
4	17730	29850
...	...	...
73944	97650	83860
73945	149230	143470
73946	35730	23080
73947	163980	135480
73948	54540	80070

73949 rows × 2 columns

以上の処理を行い、構造したゴールド表に基づいて、データそれぞれの試合のチームゴールド総量を計算した。しかし、試合によって、両チームのゴールドの差は大きくなる可能性がある。そこで、機械学習のモデルが訓練しやすいため、表 8 のデータは式 3 の処理を行い、表 9 のように作成した。

$$data[team1_economics]=np.\log_2 data([team2_economics]+1)$$

$$data[team2_economics]=np.\log_2 data([team1_economics]+1)$$

（式 3）

ゼロ除算を避けるため、[team_economics]は＋1 の処理を行った。Team1 と Team2 は同じ処理を行い、大きな偏差は発生しない。処理したデータを表 9 に示す。

表 9 処理したチームのゴールド総量

	team1_economics	team2_economics
0	15.449632	15.934589
1	17.045066	16.585359
2	17.018689	17.374505
3	16.231652	16.139731
4	14.113986	14.865492
...	...	...
73944	16.575347	16.355712
73945	17.187188	17.130400
73946	15.124889	14.494418
73947	17.323169	17.047731
73948	15.735054	16.288992

73949 rows × 2 columns

それにより、データに対して図 5 の正規化処理を行う。正規化を通して、デ

ータは[0, 1]の区間内にすることができる。これがモデルの収束スピードを上がることができる。

$$\frac{X_i - X_{min}}{X_{max} - X_{min}}$$

図 5 正規化処理

ゴールドに着目したチーム特徴を作成した。ベースラインに追加して訓練した特徴は表 10 で示す。

表 10 ゴールドに着目したチーム特徴

team_neutral1
team_tower1
team_courier1
team_lane1
team_hero1
team_observer1
team_roshan1
team_necronomicon1
team_neutral2
team_tower2
team_courier2
team_lane2
team_hero2
team_observer2
team_roshan2
team_necronomicon2
team1_economics
team2_economics

3.4 結言

本章では、選手のパフォーマンスに基づいて作成したチーム特徴とゴールドに着目して作成したチーム特徴の作成方法を紹介した。

第4章 評価実験

4.1 緒言

本章では最初にデータ収集と処理について紹介する。それからモデル構築ついて述べる。

4.2 データ収集

4.2.1 OPENDOTA

選手、チームの特徴と試合勝負との関連性を研究する時に、本研究では大量なプロ試合のデータが必要とする。プロ試合に注目する理由として、ハイレベルの試合でプレイヤーはヒーロー間の連携をより重視している[14]。手動収集データはほぼ不可能なことなので、データを収集する時に利用するのはオープンソースなデータプラットフォームの OpenDota である。OpenDota は過去の試合のリプレイをアクセスできる API を提供している。本研究はプロ試合だけに注目し、一般の試合は不要とする。しかし、OpenDota の API ではプロ試合の ID のみ提供し、API で直接にプロ試合のリプレイをダウンロードできないという制約がある。故に、ウェブクローラー技術を利用し、プロ試合のリプレイデータをダウンロードして、データベースに保存する。

4.2.2 ウェブクローラー

ウェブクローラーはウェブ上のデータを自動的に収集し、データベースに保存できるプログラムである。それが自動化することで、データ収集の際の負担を軽減でき、大量の時間を節約できる。以下はウェブクローラーの動作原理とプロセスを紹介した後、本研究に使うウェブクローラーを紹介する。

ウェブクローラーのポイントはリクエスト、解析、自動化である。ユーザーが収集したいデータのリクエストを決め、ウェブクローラーがターゲットウェブサイトにリクエストを発送する。ターゲットウェブサイトがリクエストを処理し、ユーザーがウェッブサイトから発送したレスポンスをもらえる。ユーザーはレスポンスを解析と処理を行い、収集したいデータがデータベースに保存する。

OPENDOTA の API はプロ試合の ID と試合の基本情報しか提供していない。故に、プロ試合のリプレイをダウンロードできない。本研究に使うウェブクローラーはいか三つにプロセスがある。

1.最近で行われたプロ試合 ID、とその URL を特定し、記録する。

2.1 で特定した ID を順番に後ろにプロ試合の ID を特定し、記録する。

3.それぞれの試合 ID を通して、試合データをダウンロードする。

4.データベースに保存する。

図 6 に本研究のウェブクローラーのプロセスを説明するフローチャートを示す。

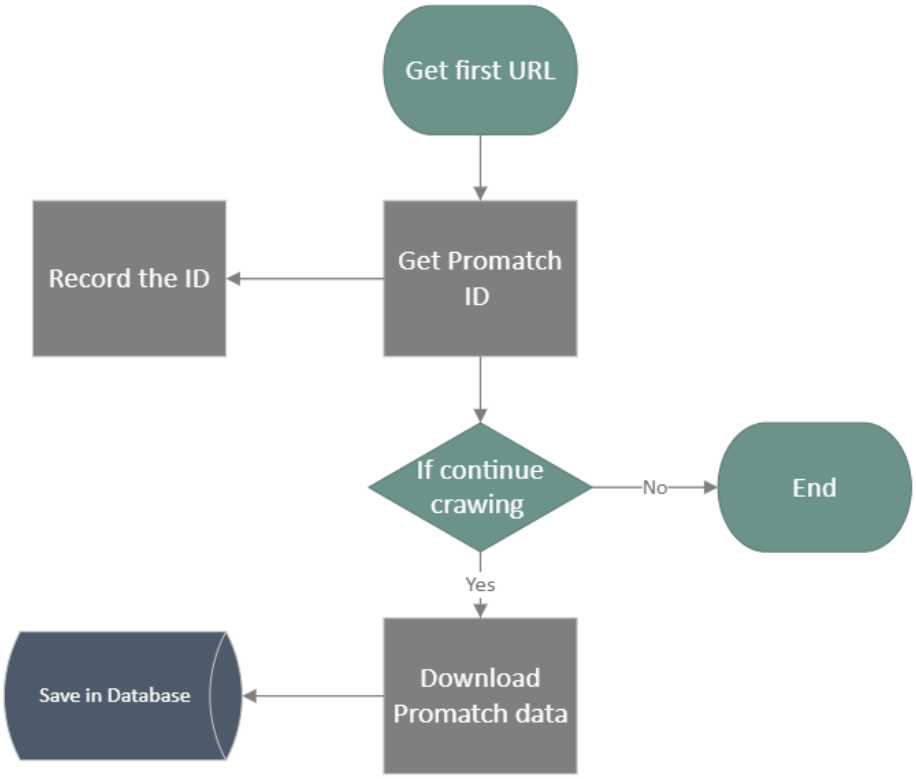


図 6 ウェブクローラーのフローチャート

4.3 データの抽出と処理

本研究では、ウェブクローラーを利用し、OPENDOTA から 73949 試合のデータをダウンロードした。

Dota2 試合で、ヒーローを選択する時に計 24 回の選択を行う（図 7）。

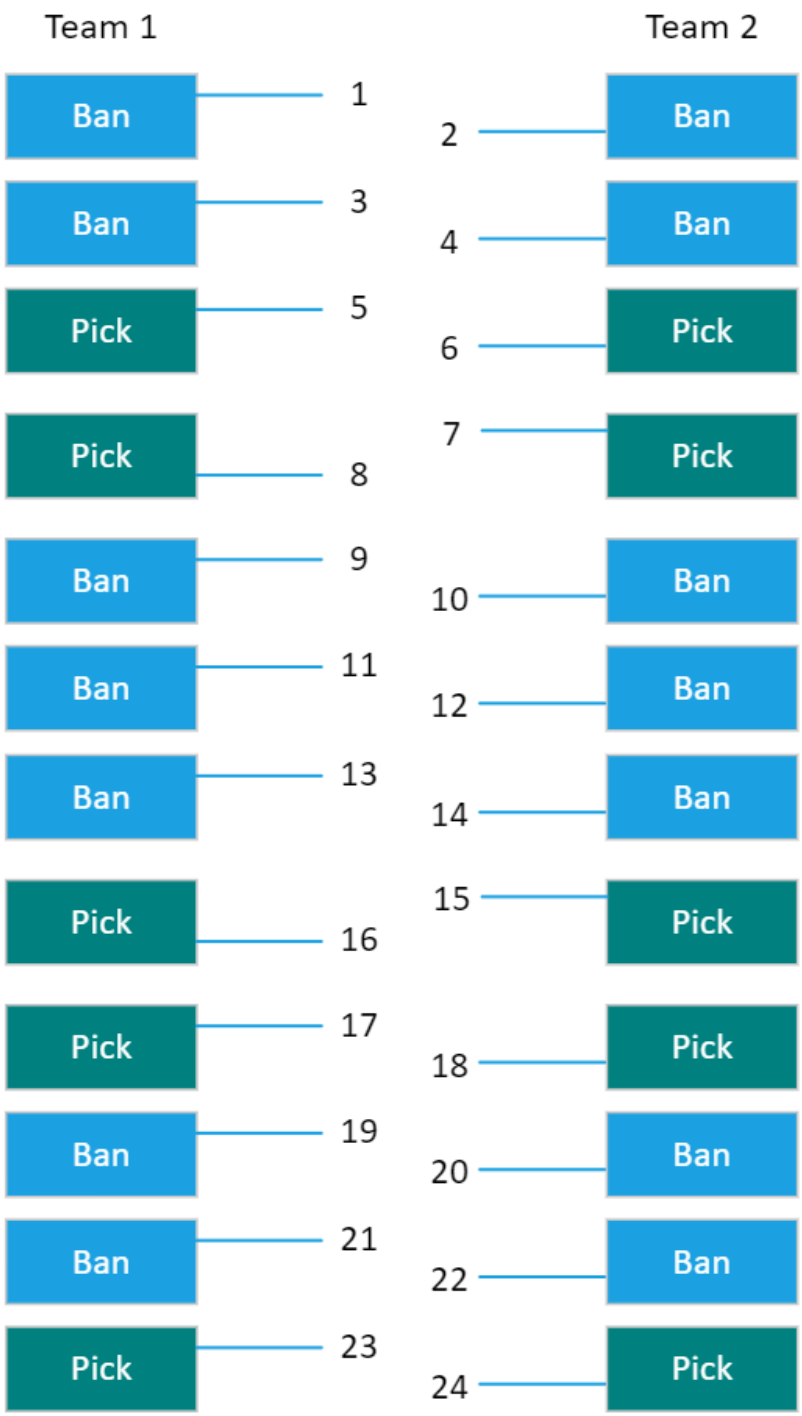


図 7 ヒーロー選択時の順番

ヒーロー選択のインフォメーションは picks_bans に保存している。このデー

タを処理し、一回の選択は三つの特徴は表 11 のように作成した。

表 11 抽出した picks_bans

特徴名	説明	値
is_pick	True は Pick、False は Ban	Boolean
hero_id	ヒーローの ID を表示する	int
team	該当するチーム	int

表 11 に基づき、一つ試合の picks_bans において、ヒーロー構成に関する特徴を作成した。試合の結果と試合の ID に合わせて抽出し、データセットを作成した。また、無効のデータ（中身は有効値がないデータ、赤枠中のデータ）は削除する（表 12）。

表 12 picks_bans を処理したデータセット（一部）

target	match_id	is_pack_1	hero_id_1	team_1	is_pack_2	hero_id_2	team_2	is_pack_3	hero_id_3	team_3
FALSE	2506261838	FALSE	111	1	FALSE	32	0	FALSE	9	1
TRUE	2506189203	None	-1	-1	None	-1	-1	None	-1	-1
TRUE	2506014084	FALSE	28	1	FALSE	99	0	FALSE	98	1
TRUE	2505962245	FALSE	103	1	FALSE	38	0	FALSE	29	1
FALSE	2505883130	FALSE	2	1	FALSE	98	0	FALSE	93	1
FALSE	2505881672	FALSE	32	0	FALSE	73	1	FALSE	98	0
TRUE	2505864413	FALSE	98	0	FALSE	87	1	FALSE	10	0
TRUE	2505855958	FALSE	38	0	FALSE	34	1	FALSE	32	0
TRUE	2505844649	FALSE	67	0	FALSE	80	1	FALSE	28	0
TRUE	2505779988	FALSE	29	1	FALSE	103	0	FALSE	79	1
FALSE	2505773200	FALSE	73	1	FALSE	41	0	FALSE	74	1
TRUE	2505723978	FALSE	55	0	FALSE	74	1	FALSE	28	0
FALSE	2505711783	FALSE	41	1	FALSE	28	0	FALSE	87	1
TRUE	2505565622	FALSE	41	0	FALSE	54	1	FALSE	98	0
FALSE	2505484651	FALSE	107	1	FALSE	103	0	FALSE	20	1
FALSE	2505063600	FALSE	32	1	FALSE	107	0	FALSE	41	1
TRUE	2504983361	FALSE	89	1	FALSE	32	0	FALSE	65	1
TRUE	2504779798	FALSE	6	0	FALSE	103	1	FALSE	32	0
TRUE	2504746331	None	-1	-1	None	-1	-1	None	-1	-1

4.4 機械学習モデルの構築

先行研究で試合勝負を予測する時は、主にナイーブベイズ、ランダムフォレストを使っていた。XGBoost は Aleksandr の研究で Dota2 勝敗予測において

性能が高いモデルとされている[13]。実験結果を評価するため、本研究では、ナイーブベイズ、ランダムフォレスト、XGBoost を使い、試合の勝負予測実験を行う。

最初はナイーブベイズアルゴリズムを紹介する。ナイーブベイズはベイズの定理をもとにした、教師あり学習アルゴリズムである[15]（図 8）。

$$P(Y_k|X) = \frac{P(X|Y_k)P(Y_k)}{\sum_k P(X|Y = Y_k)P(Y_k)}$$

図 8 ベイズ公式

本研究では、トレーニングセットはm個サンプルと n 次元と仮定する。

$(x(1)1, x(1)2, \ldots x(1)n, y1), (x(2)1, x(2)2, \ldots x(2)n, y2), \ldots (x(m)1, x(m)2, \ldots x(m)n, ym)$

試合の勝負において、全部 2 種類の出力があり、それぞれはC_1, C_2。それぞれの特徴出力サンプル個数は$m_1, m_2, \ldots, m_K$、第k個特徴は離散特徴であれば、特徴X_jの値はm_{kjl}で、その中のlは$1, 2, \ldots$で、Sjは特徴jの値である。アルゴリズムの手順は：

1.Yの事前確率がない場合、Yのk個事前確率を計算する：

$P(Y = Ck) = (mk + \lambda) / (m + K\lambda)$、それ以外は$P(Y = C_k)$が入力の事前確率。

2.特徴は離散値であるため、k類のj次元特徴のl個の値は

$P(Xj = xjl \mid Y = Ck) = \dfrac{mkjl + \lambda}{mk + Sj\lambda}$ （$\lambda \geqslant 0$）

3.サンプル $X^{(test)}$ に対し、以下の公式で計算する。

$$P(Y=C_k)\prod_{j=1}^{n}P(X_j=x_j^{(test)}\,|\,Y=C_k)$$

4. サンプル $X^{(test)}$ の分類 C_{result} を決定する。

$$C_{result}=\underbrace{\arg\max}_{C_k}P(Y=C_k)\prod_{j=1}^{n}P(X_j=x_j^{(test)}\,|\,Y=C_k)$$

以上の計算は、微分や行列乗算がないため、アルゴリズムの効率が高い。ナイーブベイズを通して、安定性が高い分類結果を得られる。

次にランダムフォレストアルゴリズムを紹介する。ランダムフォレストは、バギング思想（図 9）に基づいたアルゴリズムである[16]。

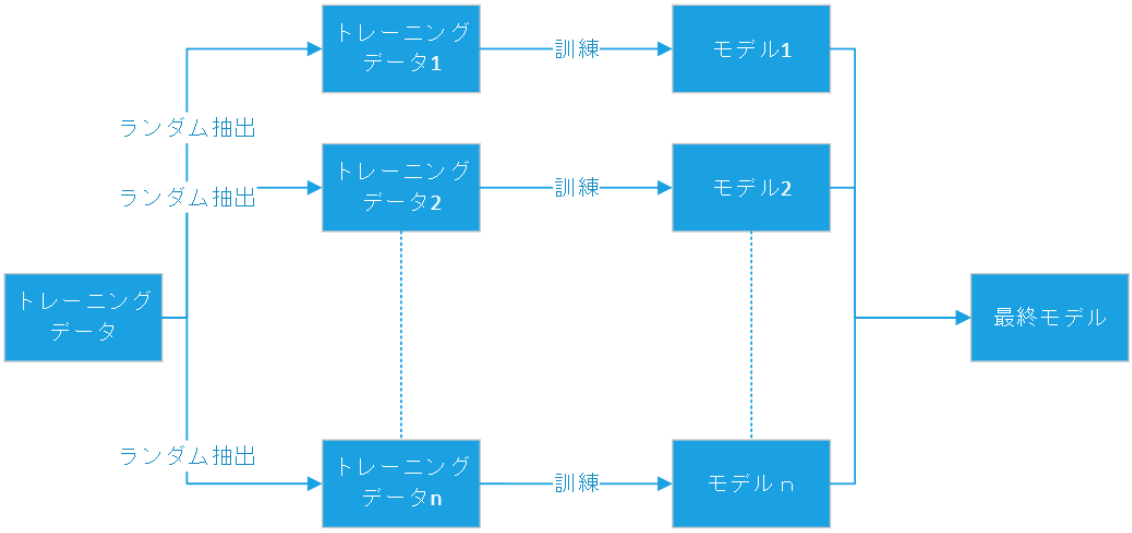


図 9 バギング思想

バギングは、トレーニングデータをランダム抽出し、トレーニングデータを

複数のデータセットに生成する。その一部のトレーニングデータを利用し、複数のモデルを作成してアンサンブルする方法である。そこで、ランダムフォレストはトレーニングデータごとにランダムに K 個の特徴量を選択し、決定木を N 個作成する。最後に N 個の決定木の結果をアンサンブルにより結合する（図 10）

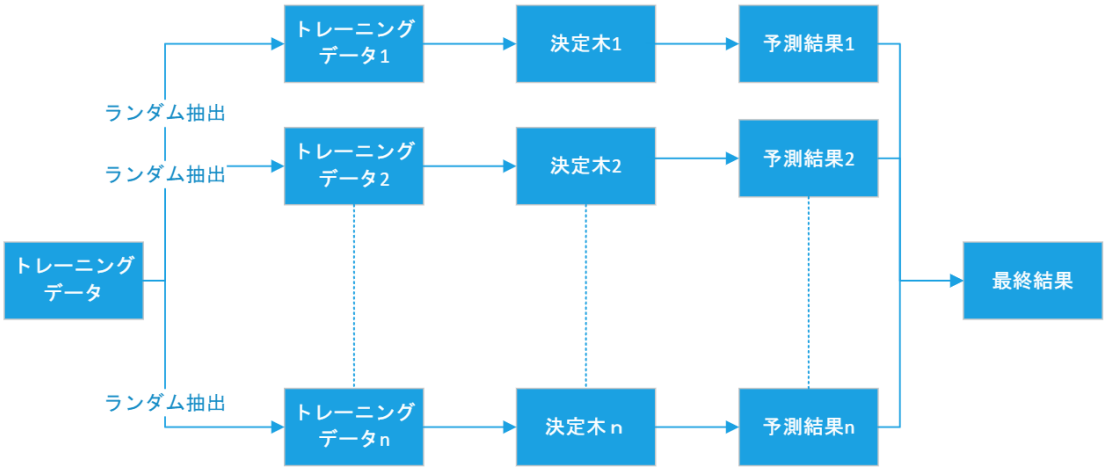


図 10 ランダムフォレスト

ランダムフォレストのメリットは、予測精度が高い、ビッグデータで複数のツリーの並列処理が可能、訓練のスピードが速い。

XGBoost は GBDT(Gradient Boosting Decison Tree)をベースとして考案されたアルゴリズムである[17]。GBDT は三つの部分がある：Decison Tree 、Boosting と Gradient Boosting である。Decison Tree（決定木）はツリーによってデータを分析する手法である。Boosting はいくつかの弱い分類器を使って、一つ性能が高い分類器を得る手法である。Gradient Boosting は勾配インフォ

メーションを使い、分類器に加重平均を行う手法である。GBDT の手順は次の通りである。

1. 初期化

2. 残差を求める

3.決定木を構造する

4. 葉ノードの重みを求める

5.出力を更新する

XGBoost は GBDT により、主にアルゴリズム最適化、効率最適化と安定性の向上三つの方面から改進した。

以上三つのモデルにより、ヒーロー構成と選手の APM をベースラインとして作成する。それに基づいて、選手パフォーマンスに着目したチーム特徴、ゴールドに着目したチーム特徴とその両方をそれぞれモデルに追加して、学習させる。訓練したモデルの精度と速度を先行研究と比較して本研究の結果を評価する。

また、収集した 73949 の試合データは図 11 の割合に分けた。

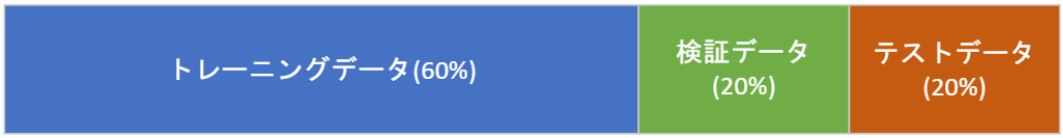


図 11 データの割合

4.5 結言

本章では、本研究のデータ収集、モデル構築と実験方法ついて述べた。

第5章 実験結果と考察

5.1 緒言

本章では、実験結果を示し、考察する。

5.2 実験結果

5.2.1 全体説明

本研究では、ヒーロー構成と選手 APM を機械学習のベースラインとして作成した。そこで、選手パフォーマンスに着目したチーム特徴とゴールドに着目したチーム特徴をそれぞれベースラインに追加した。最後に、両方の特徴をベースラインに追加した。実験の結果を図 12 に示す（付録 B を参照)。

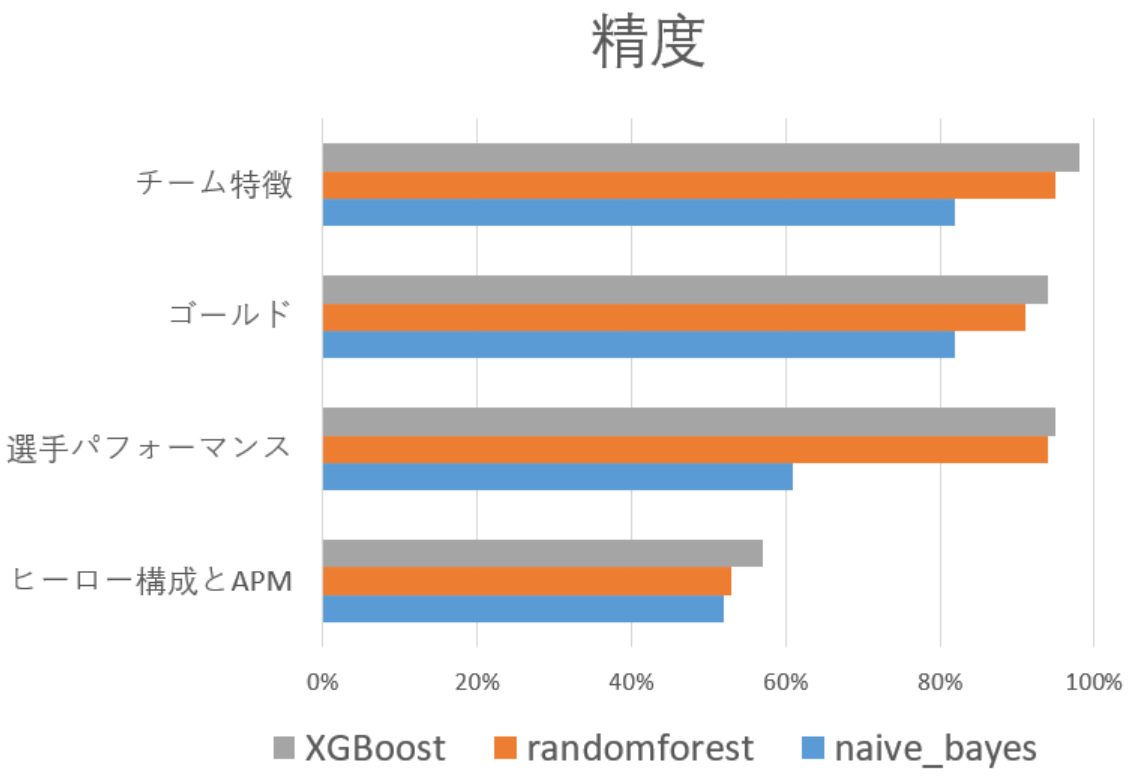


図 12 四つ実験のそれぞれの精度比較

実験の結果により、ナイーブベイズは三つのアルゴリズムの中で一番予測精度が低いアルゴリズムであった。選手パフォーマンスに着目したチーム特徴はゴールドに着目したチーム特徴よりナイーブベイズに与えた影響が低い。XGBoost は四つの実験では予測精度が一番高いモデルである。ゴールドが高いチームがプレイヤーの KDA より勝利に繋がっていることを示していた。

5.2.2 ヒーロー構成と APM に基づいたベースラインの作成

最初はヒーロー構成のみに注目し、ベースラインを作った。ナイーブベイズ、ランダムフォレスト、XGBoost 三つの方法を使い訓練してそれぞれの精度は

47%、50%、52%である。ベースラインの精度を上がるため、データベースに基づき、ヒーロー構成と選手の APM を試合前の特徴としてナイーブベイズ、ランダムフォレスト、XGBoost 三つの方法を使い訓練した。訓練した結果を図 13、図 14 と図 15 に示す。三つの方法を比較した結果を図 16 に示す。

検証データ

	precision	recall	f1-score	support
False	0.50	0.18	0.27	7117
True	0.52	0.83	0.64	7673
accuracy			0.52	14790
macro avg	0.51	0.51	0.45	14790
weighted avg	0.51	0.52	0.46	14790

****

テストデータ

	precision	recall	f1-score	support
False	0.49	0.18	0.27	7110
True	0.52	0.82	0.64	7680
accuracy			0.52	14790
macro avg	0.51	0.50	0.45	14790
weighted avg	0.51	0.52	0.46	14790

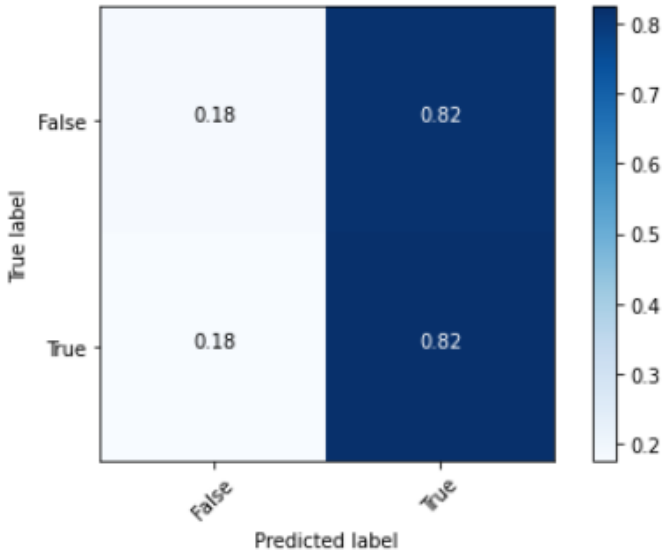


図 13 ヒーロー構成と選手 APM に注目したベースラインナイーブベイズの訓練結果

検証データ

	precision	recall	f1-score	support
False	0.59	0.08	0.14	7117
True	0.53	0.95	0.68	7673
accuracy			0.53	14790
macro avg	0.56	0.51	0.41	14790
weighted avg	0.56	0.53	0.42	14790

****

テストデータ

	precision	recall	f1-score	support
False	0.59	0.08	0.14	7110
True	0.53	0.95	0.68	7680
accuracy			0.53	14790
macro avg	0.56	0.51	0.41	14790
weighted avg	0.56	0.53	0.42	14790

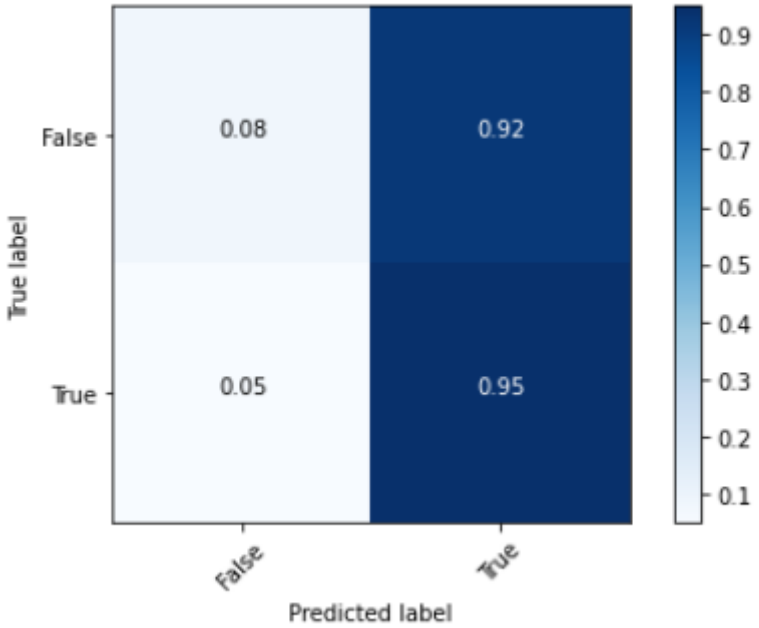


図 14 ヒーロー構成と選手 APM に注目したベースラインのランダムフォレストの訓練結果

検証データ

	precision	recall	f1-score	support
False	0.60	0.44	0.50	7117
True	0.58	0.73	0.65	7673
accuracy			0.59	14790
macro avg	0.59	0.58	0.57	14790
weighted avg	0.59	0.59	0.58	14790

****

テストデータ

	precision	recall	f1-score	support
False	0.59	0.43	0.50	7110
True	0.58	0.72	0.64	7680
accuracy			0.58	14790
macro avg	0.59	0.58	0.57	14790
weighted avg	0.59	0.58	0.57	14790

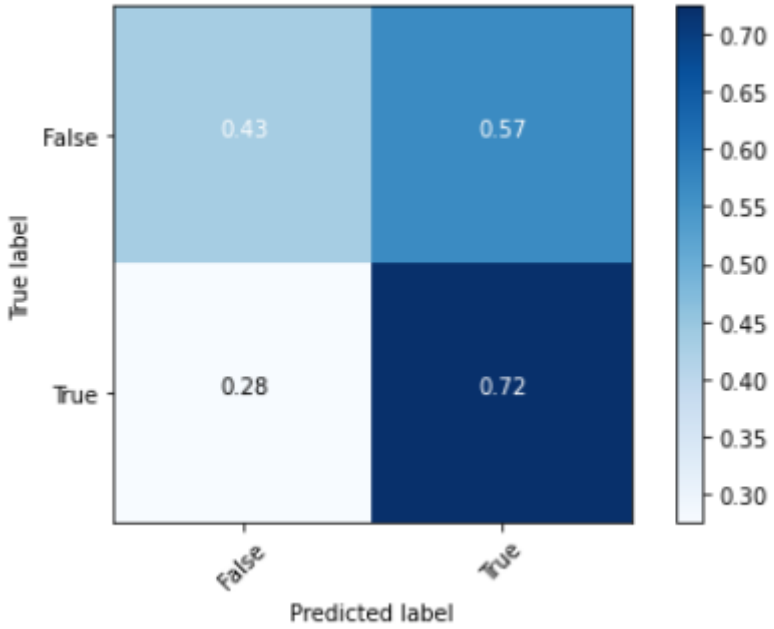


図 15 ヒーロー構成と選手 APM に注目したベースライン XGBoost の訓練結果

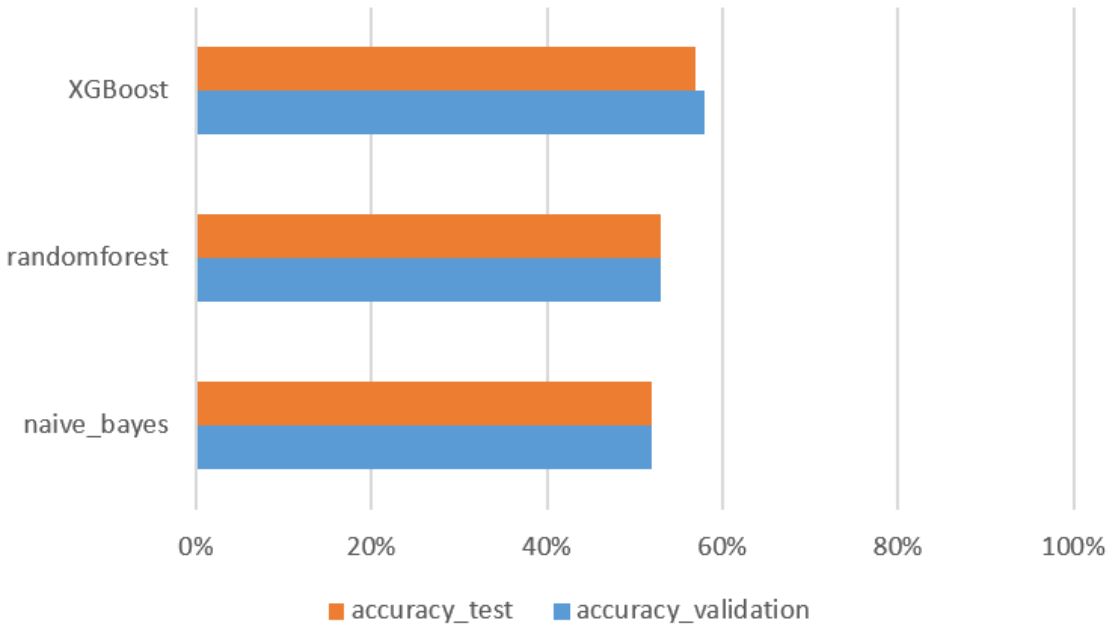


図 16 ヒーロー構成と選手 APM に注目したベースライン訓練精度の比較

試合前の特徴としての選手の APM を追加することで、XGBoost の精度は 52%から 58%に上がった。Pu らの研究の結論と似ている、試合前の特徴を使って試合の結果を予測することはヒーローの構成に基づいて訓練した結果は良いではない[14]。ベースラインにより、選手の APM は訓練の精度を上がることができたが、その効果ははっきりではない。また、新特徴の追加が必要とする。

5.2.3 選手パフォーマンスに着目したチーム特徴を追加した実験結果

ベースラインに基づき、3.3.2 で説明した選手パフォーマンスに着目したチーム特徴をモデルに学習させた。その結果を図 17、図 18、図 19 に示す。

検証データ

	precision	recall	f1-score	support
False	0.55	0.54	0.55	7256
True	0.57	0.57	0.57	7534
accuracy			0.56	14790
macro avg	0.56	0.56	0.56	14790
weighted avg	0.56	0.56	0.56	14790

****

テストデータ

	precision	recall	f1-score	support
False	0.56	0.75	0.65	7001
True	0.68	0.48	0.56	7789
accuracy			0.61	14790
macro avg	0.62	0.61	0.60	14790
weighted avg	0.63	0.61	0.60	14790

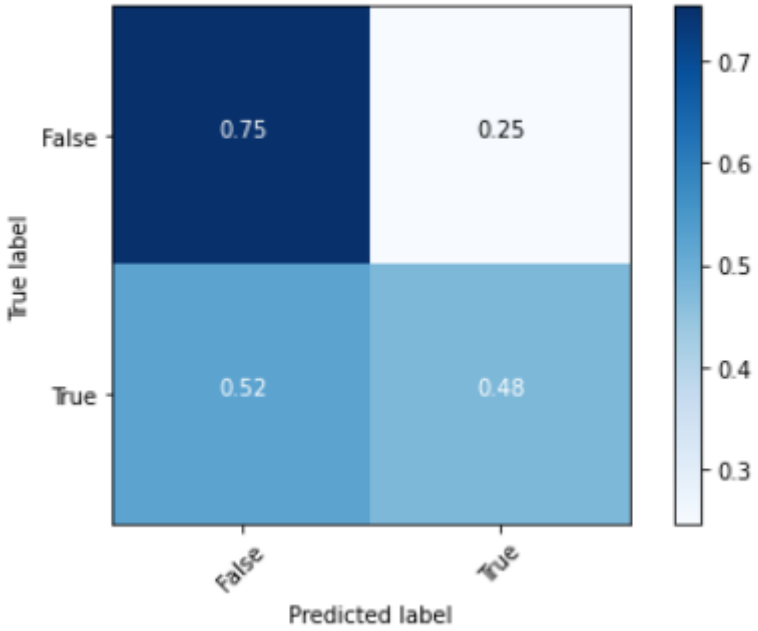


図 17 選手パフォーマンスに着目したチーム特徴を追加したナイーブベイズの訓練結果

検証データ

	precision	recall	f1-score	support
False	0.95	0.90	0.93	7256
True	0.91	0.95	0.93	7534
accuracy			0.93	14790
macro avg	0.93	0.93	0.93	14790
weighted avg	0.93	0.93	0.93	14790

****

テストデータ

	precision	recall	f1-score	support
False	0.96	0.92	0.94	7001
True	0.93	0.96	0.95	7789
accuracy			0.94	14790
macro avg	0.94	0.94	0.94	14790
weighted avg	0.94	0.94	0.94	14790

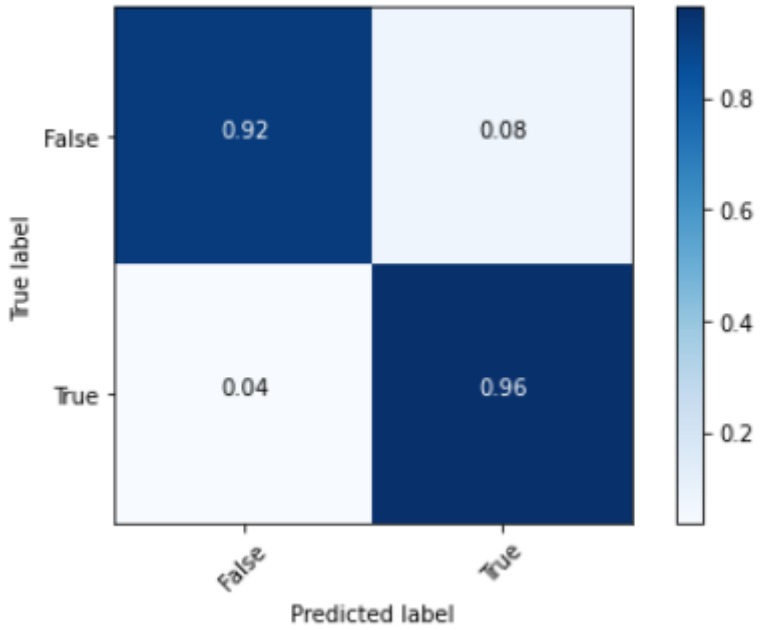


図 18 選手パフォーマンスに着目したチーム特徴を追加したランダムフォレストの訓練結果

検証データ

	precision	recall	f1-score	support
False	0.94	0.93	0.94	7256
True	0.94	0.94	0.94	7534
accuracy			0.94	14790
macro avg	0.94	0.94	0.94	14790
weighted avg	0.94	0.94	0.94	14790

****

テストデータ

	precision	recall	f1-score	support
False	0.95	0.95	0.95	7001
True	0.95	0.96	0.95	7789
accuracy			0.95	14790
macro avg	0.95	0.95	0.95	14790
weighted avg	0.95	0.95	0.95	14790

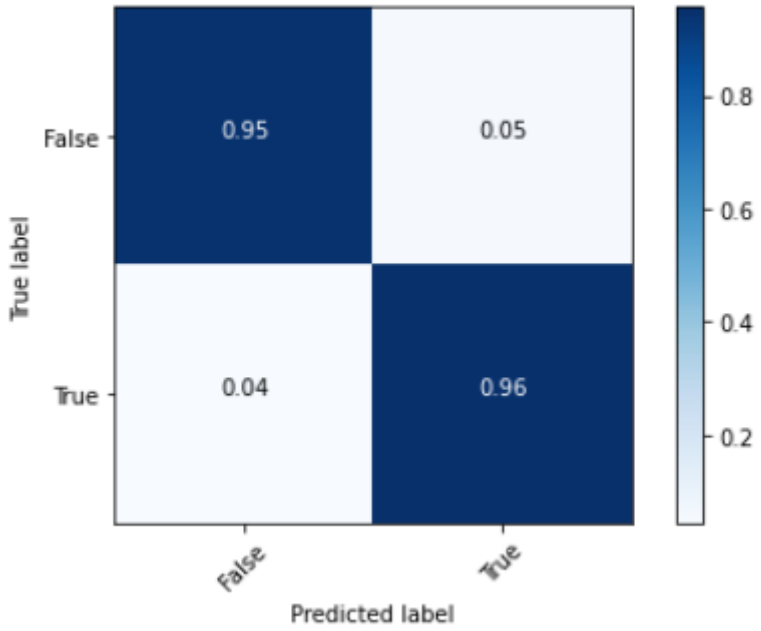


図 19 選手パフォーマンスに着目したチーム特徴を追加した XGBoost の訓練結果

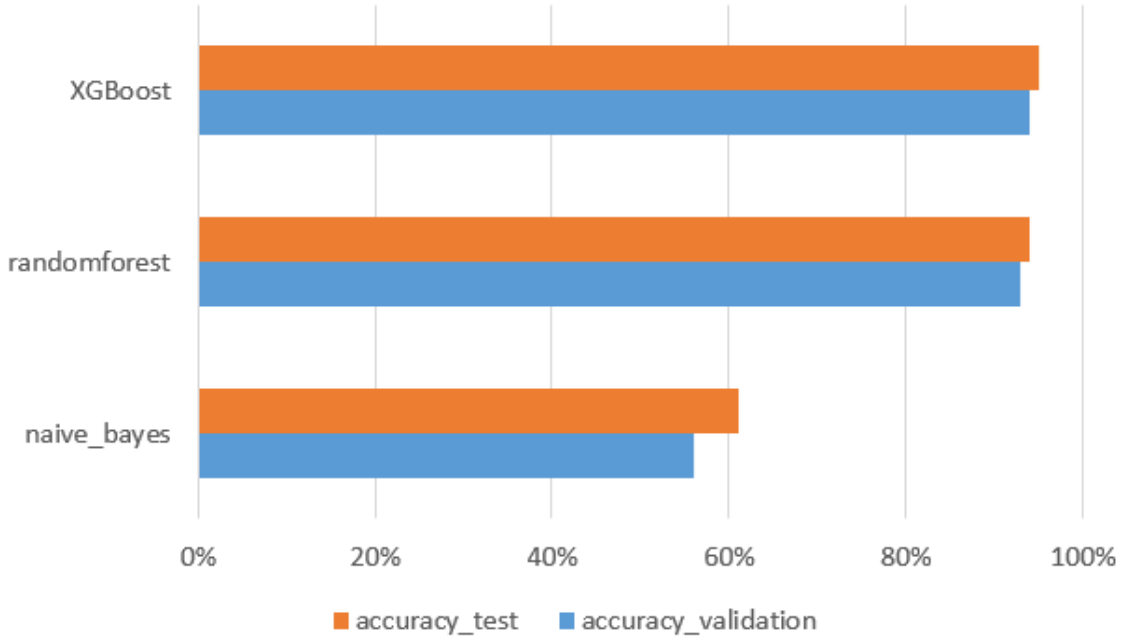


図 20 選手パフォーマンスに着目したチーム特徴を追加して訓練精度の比較

訓練した結果を図 20 に比較する。ナイーブベイズの精度とランダムフォレスト、XGBoost の精度の違いが出た。XGBoost とランダムフォレストの精度は 95%、94%であり、ナイーブベイズの精度は 61%である。

5.2.4 ゴールドに着目したチーム特徴を追加する

3.3.3 で構成したゴールド表により、ゴールドに着目したチーム特徴を追加特徴としてモデルに学習させた結果を図 21、図 22 と図 23 に示す。

検証データ

	precision	recall	f1-score	support
False	0.77	0.77	0.77	7256
True	0.78	0.78	0.78	7534
accuracy			0.77	14790
macro avg	0.77	0.77	0.77	14790
weighted avg	0.77	0.77	0.77	14790

****

テストデータ

	precision	recall	f1-score	support
False	0.78	0.87	0.82	7001
True	0.87	0.78	0.82	7789
accuracy			0.82	14790
macro avg	0.82	0.82	0.82	14790
weighted avg	0.83	0.82	0.82	14790

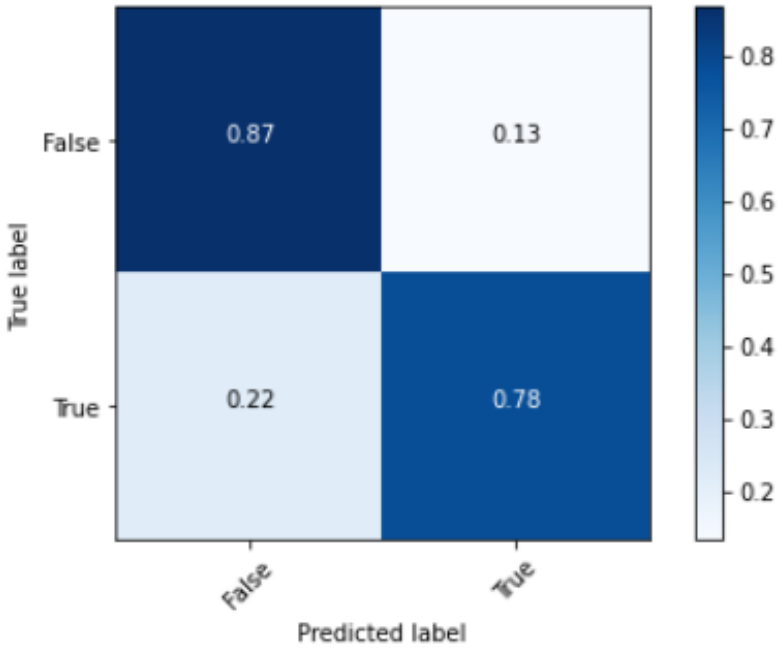


図 21 ゴールドに着目したチーム特徴を追加したナイーブベイズの訓練結果

検証データ

	precision	recall	f1-score	support
False	0.88	0.87	0.87	7256
True	0.87	0.89	0.88	7534
accuracy			0.88	14790
macro avg	0.88	0.88	0.88	14790
weighted avg	0.88	0.88	0.88	14790

****

テストデータ

	precision	recall	f1-score	support
False	0.94	0.87	0.90	7001
True	0.89	0.95	0.92	7789
accuracy			0.91	14790
macro avg	0.91	0.91	0.91	14790
weighted avg	0.91	0.91	0.91	14790

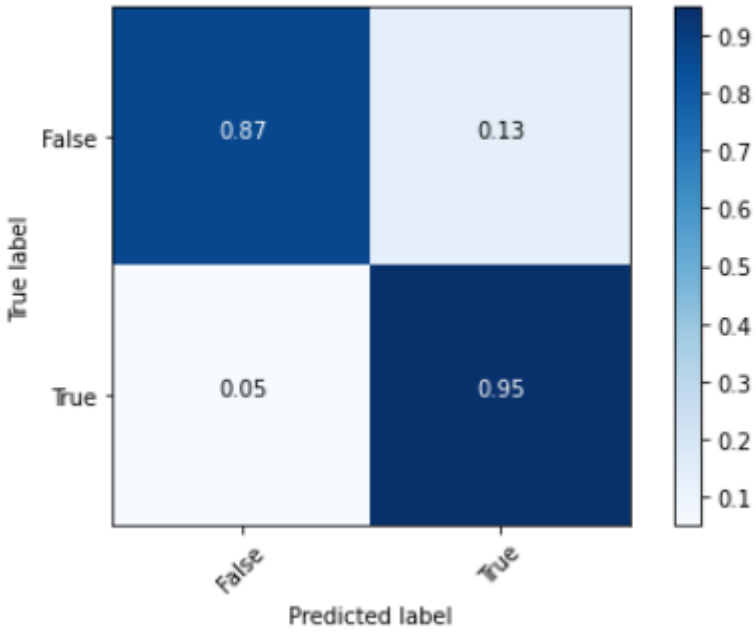


図 22 ゴールドに着目したチーム特徴を追加したランダムフォレストの訓練結果

検証データ

	precision	recall	f1-score	support
False	0.92	0.92	0.92	7256
True	0.93	0.92	0.92	7534
accuracy			0.92	14790
macro avg	0.92	0.92	0.92	14790
weighted avg	0.92	0.92	0.92	14790

****

テストデータ

	precision	recall	f1-score	support
False	0.92	0.96	0.94	7001
True	0.96	0.93	0.95	7789
accuracy			0.94	14790
macro avg	0.94	0.94	0.94	14790
weighted avg	0.94	0.94	0.94	14790

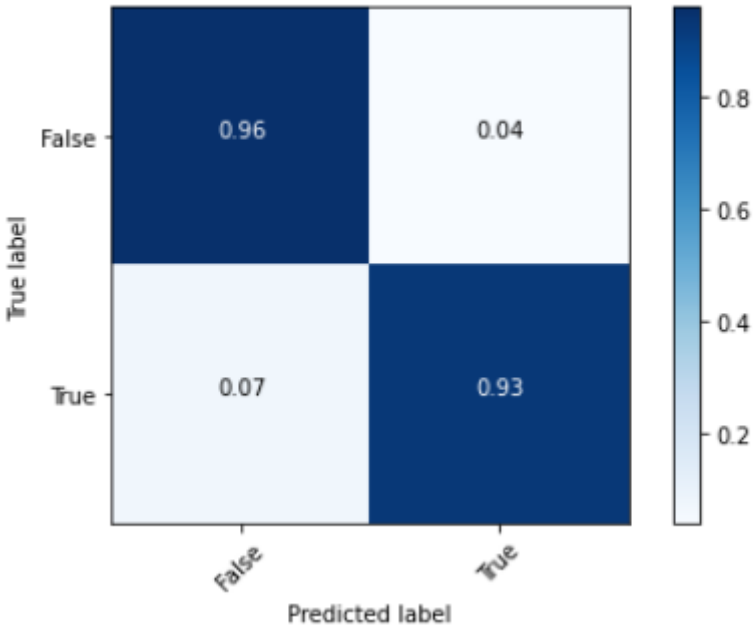


図 23 ゴールドに着目したチーム特徴を追加した XGBoost の訓練結果

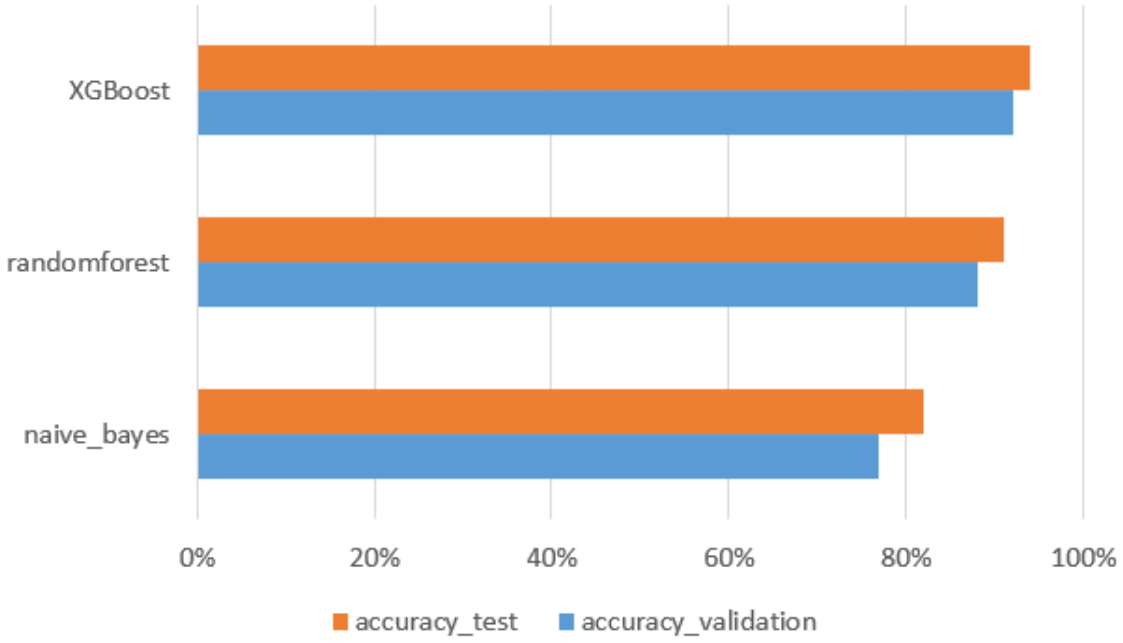


図 24 ゴールドに着目したチーム特徴を追加して訓練精度の比較

作成した特徴を追加し、訓練した結果を図 24 に示す。ナイーブベイズの精度は選手のパフォーマンス特徴より良い正確に予測できた。XGBoost とランダムフォレストの精度は実験 2 より低いが、94%の精度があり、チームのゴールドが試合の結果予測に貢献したことが分かった。

5.2.5 両特徴を追加した結果

最後は、3.3.2 と 3.3.3 で作成した選手パフォーマンスに着目したチーム特徴とゴールドに着目したチーム特徴の両方を追加特徴としてモデルに学習させた結果を図 25、図 26 と図 27 に示す。

検証データ

	precision	recall	f1-score	support
False	0.77	0.77	0.77	7256
True	0.78	0.78	0.78	7534
accuracy			0.77	14790
macro avg	0.77	0.77	0.77	14790
weighted avg	0.77	0.77	0.77	14790

****

テストデータ

	precision	recall	f1-score	support
False	0.78	0.87	0.82	7001
True	0.87	0.78	0.82	7789
accuracy			0.82	14790
macro avg	0.82	0.82	0.82	14790
weighted avg	0.83	0.82	0.82	14790

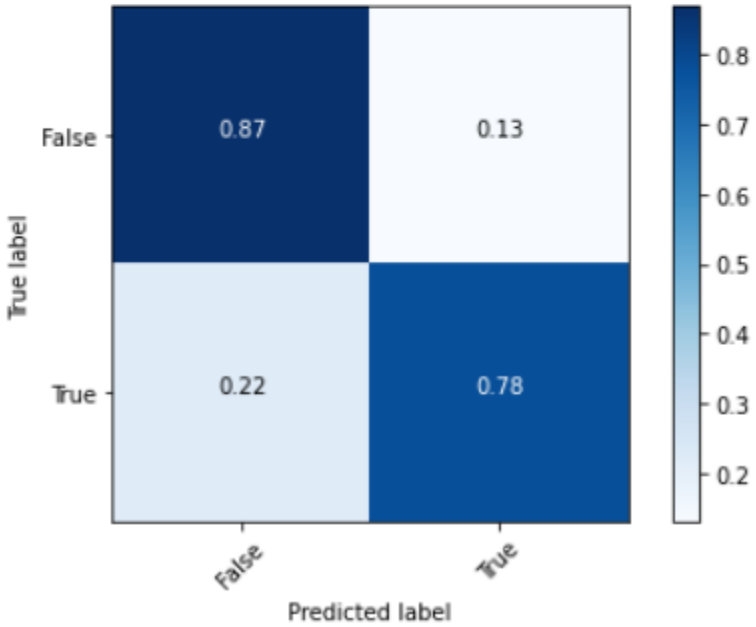


図 25 チーム特徴全体を追加したナイーブベイズの訓練結果

検証データ

	precision	recall	f1-score	support
False	0.94	0.92	0.93	7256
True	0.93	0.94	0.93	7534
accuracy			0.93	14790
macro avg	0.93	0.93	0.93	14790
weighted avg	0.93	0.93	0.93	14790

****

テストデータ

	precision	recall	f1-score	support
False	0.94	0.94	0.94	7001
True	0.95	0.95	0.95	7789
accuracy			0.95	14790
macro avg	0.95	0.95	0.95	14790
weighted avg	0.95	0.95	0.95	14790

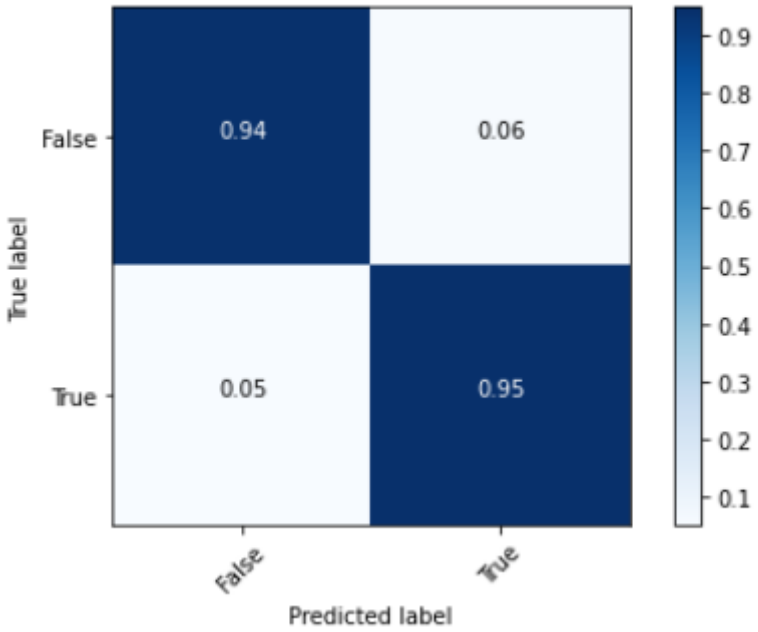


図 26 チーム特徴全体を追加したランダムフォレストの訓練結果

検証データ

	precision	recall	f1-score	support
False	0.97	0.96	0.96	7256
True	0.96	0.97	0.96	7534
accuracy			0.96	14790
macro avg	0.96	0.96	0.96	14790
weighted avg	0.96	0.96	0.96	14790

****

テストデータ

	precision	recall	f1-score	support
False	0.97	0.98	0.97	7001
True	0.98	0.98	0.98	7789
accuracy			0.98	14790
macro avg	0.98	0.98	0.98	14790
weighted avg	0.98	0.98	0.98	14790

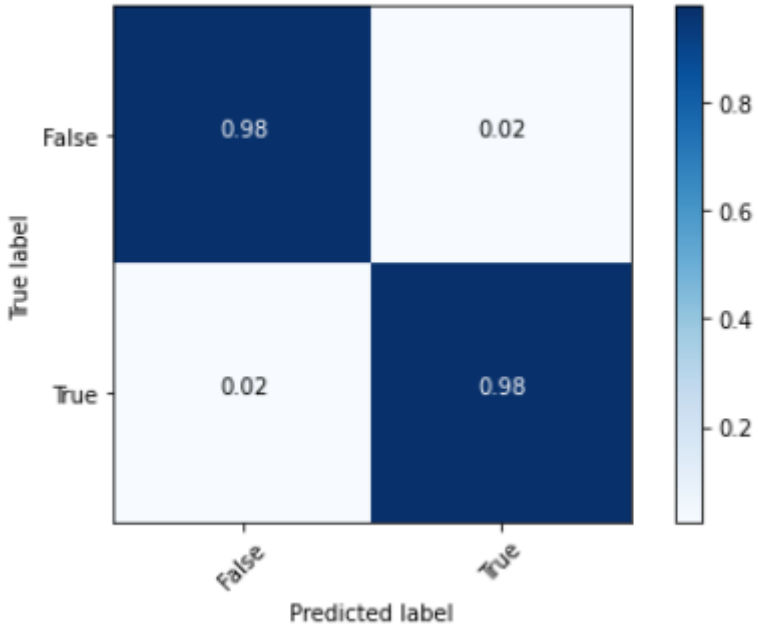


図 27 チーム特徴全体を追加した XGBoost の訓練結果

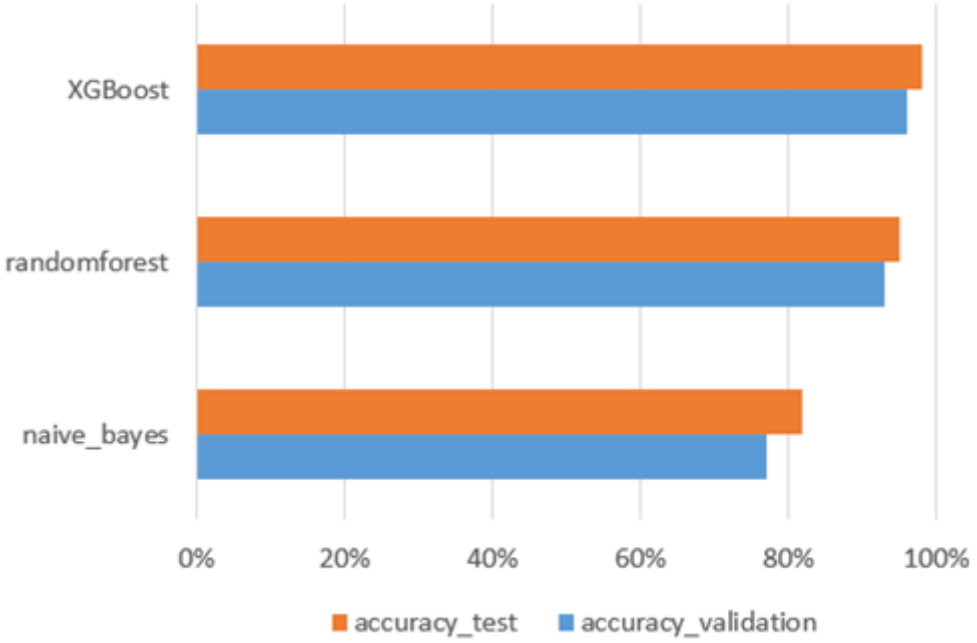


図 28 作成したチーム特徴全部追加して訓練精度の比較

三つのモデルを使い、訓練した結果を図 28 に示す。XGBoost の精度は 5.5.3 と 5.5.4 の実験より更に 98%に上がった。

5.3 考察

本研究では、Dota2 の勝敗予測ためのチーム特徴の作成手法を提案し、作成した特徴を機械学習のナイーブベイズ、ランダムフォレスト、XGBoost を使い、それぞれの効果を分析した。最初は Song の研究に基づいて[6]、ヒーローの構成を基に選手の APM 特徴を抽出し、モデルを訓練させた。Song の研究より精度が上がったが、効果は明らかではない。次に、選手が Dota2 の試合に影響が出るデータを分析し、選手の KDA に着目し、選手のパフォーマンスに基づいたチーム特徴の作成手法を紹介した。また、ゲーム内のゴールドにより、プレ

イヤーのゴールド表を作り、チームのゴールド特徴を作成した。本研究に対し、Akhmedov の研究では Dota2 のすべての特徴（164 個）を使い、試合の勝負予測研究を行なわれていた。彼らの実験では 93%の精度に至ったが（LSTM)、100 試合を訓練するだけで 100 時間以上かかった[9]。本研究で作成したチーム特徴を使い、訓練した時間を表 13 に示す。Akhmedov の研究と比べ、モデルの訓練時間が 788ms に短縮した。(XGBoost)。また、彼らの研究では Dota2 の試合勝負に影響が高い特徴を判明できなかった。本研究は、チーム特徴をモデルに訓練し、予測精度が上がったことで、チーム特徴は試合の勝負に大きい影響を与えていることが分かった。

表 13 本研究の訓練時間の比較（チーム特徴全体追加した場合）

	訓練時間
ナイーブベイズ	284ms
ランダムフォレスト	439ms
XGBoost	788ms

ナイーブベイズがかかった時間は三つのモデルで一番少ない同時に、予測の精度も一番低いである。XGBoost は三つのモデルで最適な Dota2 の試合勝負予測モデルである。これが Aleksandr の研究の結論と同じである[13]。先行研究との比較を表 14 に示す。

表 14 先行研究との比較

	本研究	本研究 (baseline)	本研究 (total)	Song[6]	Aleksandr[13]	Wang[7]	Akhmedov[9]
モデル	XGBoost	XGBoost	XGBoost	Logistic Regression	XGBoost	Naive Bayes	LSTM
特徴	ヒーロー構成	ヒーロー構成と APM	プラスチーム特徴	ヒーロー構成	ヒーロー構成	ヒーロー構成	全 164 個ゲーム内の特徴
入力数	72	82	116	224	226	113	不明
データ数	73949	73949	73949	3000	5071858	92649	100
試合レベル	プロ試合	プロ試合	プロ試合	普通試合	普通試合	普通試合	普通試合
予測精度	52%	58%	98%	49%	65%	59%	93%
訓練時間	388ms	390ms	788ms	不明	不明	不明	100 時間以上

本研究でヒーロー構成に注目した勝敗予測は Aleksandr ら[13]と Wang ら[7]の研究より精度が下がった。原因の一つは本研究が使用した訓練データはプロ試合のデータである。選手の実力が強いほど試合の勝敗を予測することが難しくなると Aleksandr ら[13]は予想していた。ヒーロー構成に基づき、選手 APM を試合前の特徴として追加した結果、予測精度は 52%から 58%に上がった。本研究の貢献は、チーム特徴を作成し、モデルに追加することで、試合結果予測の精度が 98%上がった同時に、訓練の時間を 788ms に短縮したことである。ただし、結果に関係する特徴

量を用いているという課題はある。また、訓練データはプロ試合に注目することで、プレイヤー間の連携は重視された試合でチーム特徴は試合の結果に大きく貢献していることが分かった。Dota2 の試合では、敵ヒーローを倒す時に様々なメリットがある。選手の KDA 以外、敵ヒーローを倒すことをある程度で表現できるのはチームのダメージ量である。従って、本研究はヒーロー構成に合わせて、チームのダメージ量を抽出してモデルに訓練させた。ナイーブベイズ、ランダムフォレスト、XGBoost それぞれの精度は 47%、82%、78%である。チームのダメージ量は予測の精度を上がることができるが、選手のパフォーマンスとゴールドより試合結果に対する影響が弱いである。

Dota2 は他の MOBA ゲームより多くのシステムが存在している。例えば、ルーン（ゲームの偶数時間でマップの特定な場所で現す、ヒーローが拾った時は様々なプラス効果を得る)、クーリエ（それぞれのヒーローが持っている、アイテムを運送できるユニット）の存在や味方のクリープを倒すことができるなど複雑なシステムが存在している。Dota2 を例として、選手、ゴールドに注目し、作成したチーム特徴を研究することで、MOBA ゲームで違った選手、チームが試合に対する影響を判明することに役に立つ。本研究では、ヒーロー構成で試合の勝負を予測することと比べて、選手パフォーマンスに着目し、チームの特徴を作成した。実験の結果はチーム特徴が試合の結果に影響している

ことが分かった。4.5.3 と 4.5.4 の研究で、チームのゴールドは選手の KDA より試合の勝負を影響していることが分かった。Dota2 において、プレイヤーが試合中の KDA に注目するより、ゴールドを得ることを優先にすることは試合に勝ちやすいことが考えられる。これがプロ選手の個人訓練やプロチームの戦略に役に立つ。将来の研究では、選手の APM だけではなく、細かい操作まで分析することは期待できる。これは AI が人間の操作を理解することに役に立つと考えられる。

5.4 結言

本章では、本研究の結果とそれに対する考察を討論した。

第6章 おわりに

6.1 結論

過去の Dota2 の試合勝負予測研究はヒーローの構成に注目していた。Song らがヒーロー構成に基づいた試合の結果予測研究は 49%であった[6]。Wang らがヒーロー構成に基づいた試合の結果予測研究は 58%であった[7]。本研究では、選手とチームに関するインフォメーションが Dota2 の試合勝負予測するときに考慮すべき特徴として、選手のパフォーマンスに着目して作成したチーム特徴量とゴールドに着目し、作成したチーム特徴量を提案した。作成した特徴に基づいて、比較実験を行った。その結果、一番性能が高い XGBoost は予測の精度が 98%に至った。また、Akhmedov らの研究より、100 試合を訓練した時間が 100 時間かかったより[9]、73949 試合を訓練した時間は 788ms に短縮した。これはチーム特徴が試合の勝利に強い関連性があって、ゴールド特徴は選手の KDA より試合の勝利に強く貢献していることが分かった。

本研究で提案した手法は、Dota2 の試合結果予測のスピードと精度に貢献した。将来の Dota2 の試合を予測研究は選手、チームと特徴を考えるべきである。なお、本研究で使用した特徴量は結果に直接関係するものがあるという課題が

ある。

6.2 将来の展望

Dota2 の試合では、それぞれヒーローが独特な能力を持つ、その能力の使う回数や対象などが本研究では考慮しなかった。選手とチーム以外、ヒーローの行動データは試合の結果にどれぐらいの影響があることは検討する予定である。それに基づいて、初心者向けのヒーローマスターシステムを作ることは将来性があると考えられる。

現在、OPENDOTA の API からアクセスできるデータの制限があり、ある選手、チームに関するデータを収集することができない。本研究では、事前特徴として選手の APM しか収集できなかった。より正確に試合の結果を予測するためより詳細な選手のデータが求められる。また、トップレベル選手の習慣に対するモデリング手法は有意義な研究と考えられる。

参考文献

[1] J.Hamari and M.Sjoblom, "What is esports and why do people watch it? " Internet research, vol. 27, no. 2, pp. 211–232, 2017

[2] S.Chen and T.Joachims. "Predicting matchups and preferences in context" Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining pp. 775–784,2016

[3] A.C.Constantinou, N.E.Fenton, and M.Neil. pi-football: "A bayesian network model for forecasting association football match outcomes" Knowledge-Based Systems, 36, 2012.

[4] C.Eggert, M.Herrlich, J.Smeddinck and R.Malaka. "Classification of Player Roles in the Team-Based Multi-player Game Dota 2" 14th International Conference on Entertainment Computing(ICEC) , pp.112-125, Sep 2015

[5] H.Yi, D.Sunil and P.Mark. "Player behavior and optimal team composition for online multiplayer games" arXiv preprint arXiv:1503.02230, 2015.

[6] K.Song, T.Zhang, and C.Ma."Predicting the winning side of dota2," Course project, Stanford University, 2015.

[7] K. Wang and W. Shang, "Outcome prediction of DOTA2 based on Naïve Bayes classifier" IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS), pp. 591-593, 2017

[8] K. Kalyanaraman "To win or not to win? A prediction model to determine the outcome of a DotA2 match" Technical report, University of California San Diego,2014

[9] K. Akhmedov and A. Phan "Machine learning models for DOTA 2 outcomes prediction" IEEE Transactions on Games,2021

[10] A. Dila Wijaya, F. Baskara, M. R. Kusuma, R. S. Bernhard and R. Senjaya, "Ease of use analysis of in-game interface design in DotA 2 for beginners using SAW method," International Conference on Information & Communication Technology and Systems (ICTS), pp. 123-126, 2015

[11] A. Drachen et al., "Skill-based differences in spatio-temporal team behaviour in defence of the Ancients 2 (DotA 2)," IEEE Games Media Entertainment, , pp. 1-8, 2014

[12] P.Grutzik, J.Higgins, and L.Tran. "Predicting outcomes of professional DotA2 matches " Technical Report.Stanford University,2017

[13] A.Semenov, P.Romov, S.Korolev, D.Yashkov,and K.Neklyudov, "Performance of Machine Learning Algorithms in Predicting Game Outcome from Drafts in Dota 2",Communications in Computer and Information Science 661,2016

[14] P. Yang, B.E.Harrison,and D. Roberts, "Identifying patterns in combat that are predictive of success in MOBA games" Proceedings of the Foundations of Digital Games 2014 Conference(FDG),2014

[15] Z. Zhihua, "Machine Learning", Tsinghua University Press,vol 7,pp147-162,2016

[16] G.Louppe, "Understanding Random Forests From Theory to Practice", University of Liège, PhD dissertation,2014

[17] C.Tianqi and G.Carlos "XGBoost: A Scalable Tree Boosting System" In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16). Association for Computing Machinery, New York, NY, USA, 785–794,2016

謝辞

修士の生活で、終始適切な助言を賜り、また丁寧に指導して下さった由井薗隆也先生に心から感謝いたします。

本研究を遂行するに当たり、様々なご指導、ご鞭撻を賜りました、副指導の吉高 淳夫准教授に厚く御礼申し上げます。

修士の課程でおよび最終審査では、吉高 淳夫准教授、池田 心教授、小谷 一孔教授より、貴重なご指導とご助言を賜りました。感謝申し上げます。

所属する由井薗ゼミのみなさまには多くのご支援をいただきました。お礼申し上げます。

そして何よりも、経済面、精神面から支えてくださり、常に寛大な心で接してくださいました両親に深い敬意と感謝を示し、心より御礼申し上げます。

付録 A

OPENDOTA で取れる一つ試合の全特徴

ID	特徴名	値	説明
1	match_id	integer	試合の ID、それぞれの試合は独特な ID がある
2	barracks_status_dire	integer	試合が終わった時にバラックがまだ存在しているかどうかを表示するビットマスク（dire 方）
3	barracks_status_radiant	integer	試合が終わった時にバラックがまだ存在しているかどうかを表示するビットマスク（radiant 方）
4	chat	object	試合内のチャット内容を含む配列
5	time	integer	メッセージが送信した時間（秒まで）
6	unit	string	メッセージを送信したプレイヤーの名前
7	key	string	メッセージ内容
8	slot	integer	スロット
9	player_slot	integer	プレイヤーがどのスロットにいる。0-127 は dire 方、128-255 は radiant 方
10	cluster	integer	クラスタ
11	cosmetics	object	プレイヤーが使ったスキンなど
12	dire_score	integer	試合が終わった時の dire 方が倒したヒーローの数
13	draft_timings	object	ヒーロー選択時のインフォメーション
14	order	integer	ヒーロー選択時の順
15	pick	boolean	特定のヒーローを選択したか
16	active_team	integer	選択しているチーム
17	hero_id	integer	ヒーローの ID、それぞれのヒーローは独特な ID がある
18	player_slot	integer	プレイヤーがどのスロットにいる。0-127 は dire、128-255 は radiant

ID	特徴名	値	説明
19	extra_time	integer	ヒーローを選択する時ボーナス時間
20	total_time_taken	integer	ヒーローを選択でかかったトータル時間
21	duration	integer	試合がかかった時間（秒単位）
22	engine	integer	エンジン
23	first_blood_time	integer	ファーストボラードが発生した時間
24	game_mode	integer	ゲームのモードを表示する
25	human_players	integer	試合でのプレイヤーの数
26	leagueid	integer	リーグの ID
27	lobby_type	integer	ロビーのタイプを表示する整数
28	match_seq_num	integer	試合の順の番号
29	negative_votes	integer	ゲームのクライアントが受け取った試合のリプレイに対する反対投票の数
30	objectives	object	対象
31	picks_bans	pick_bans	ヒーロー選択時のインフォメーション
32	positive_votes	integer	ゲームのクライアントが受け取った試合のリプレイに対する賛成投票の数
33	radiant_gold_adv	object	ゲーム中の各時点での radiant 方のゴールドアドバンテージを表示する配列。負数なら dire 方
34	radiant_score	integer	試合が終わった時の radiant 方が倒したヒーローの数
35	radiant_win	boolean	1 の場合は radiant 方が勝った
36	radiant_xp_adv	object	ゲーム中の各時点での radiant 方の経験値ドアドバンテージを表示する配列。負数なら dire 方
37	start_time	integer	ゲームが始める時の UNIX 時間
38	teamfights	object	チームの戦い
39	tower_status_dire	integer	試合が終わった時にタワーがまだ存在しているかどうかを表示するビットマスク（dire 方）
40	tower_status_radiant	integer	試合が終わった時にタワーがまだ存在しているかどうかを表示するビットマスク（radiant 方）
41	version	integer	ゲームのバージョン
42	replay_salt	integer	試合をダウンロードする時に必要な暗証番号
43	series_id	integer	試合シリーズの ID
44	series_type	integer	シリーズのタイプ

ID	特徴名	値	説明
45	radiant_team	object	radiant チーム
46	dire_team	object	dire チーム
47	league	object	リーグのインフォメーション
48	skill	integer	プレイヤーのスキルレベル（Normal, High, Very High）
49	player	object	それぞれのプレイヤーのインフォメーションを表示する配列
50	match_id	integer	試合の ID
51	player_slot	integer	プレイヤーがどのスロットにいる。0-127 は dire 方、128-255 は radiant 方
52	ability_upgrades_arr	array of integer	ヒーローの能力上がりを表示する配列
53	ability_uses	object	プレイヤーがヒーローのスキルを使用した回数
54	ability_targets	object	プレイヤーがヒーローのスキルを使用した相手
55	damage_targets	object	プレイヤーが敵ヒーローに与えたダメージ値と与えた形を表示するオブジェクト
56	account_id	integer	アカウントの ID
57	actions	object	プレイヤーがヒーローに与えた操作の数と形を表示するオブジェクト
58	additional_units	object	プレイヤーが操作した他の単位に関するインフォメーション
59	assists	object	プレイヤーのアシストの数
60	backpack_0	integer	バックパック 0 のアイテム
61	backpack_1	integer	バックパック 1 のアイテム
62	backpack_2	integer	バックパック 2 のアイテム
63	buyback_log	object	プレイヤーのバイバックに関するインフォメーションを表示する配列
64	time	integer	プレイヤーがバイバックした時間（秒単位）
65	slot	integer	スロット
66	player_slot	integer	プレイヤーがどのスロットにいる。0-127 は dire 方、128-255 は radiant 方
67	camps_stacked	integer	中立のクリープをスタックの数
68	connection_log	object	プレイヤーの連接に関するインフォメーションを表示する配列
69	time	integer	連接事件が起きたゲーム時間
70	event	string	事件

ID	特徴名	値	説明
71	player_slot	integer	プレイヤーがどのスロットにいる。0-127 は dire 方、128-255 は radiant 方
72	creep_stacked	integer	クリープをスタックの数
73	damage	object	プレイヤーが敵に与えたダメージ
74	damage_inflictor	object	プレイヤーが敵に与えたダメージの根源インフォメーション
75	damage_inflictor_received	object	プレイヤーが受けた敵ヒーローのダメージの根源インフォメーション
76	damage_taken	object	プレイヤーがダメージを受けた対象に関するインフォメーション
77	deaths	integer	プレイヤーが倒された数
78	denies	integer	Deny（味方ユニットのとどめを刺すこと）の数
79	dn_t	array of integer	各時間の deny の数を表示する配列
80	gold	integer	試合が終わった時のゴールドの数
81	gold_per_min	integer	プレイヤーが得たゴールドの数（分単位）
82	gold_reasons	object	得たゴールドのソース
83	gold_spent	integer	プレイヤーが使ったゴールド
84	gold_t	array of integer	各時間のゴールドの数を表示する配列
85	hero_damage	integer	敵ヒーローに与えたダメージ総量
86	hero_healing	integer	味方へのヒーリング総量
87	hero_hits	object	プレイヤーが敵に与えた攻撃の数
88	hero_id	integer	プレイヤーが使ったヒーローの ID、それぞれのヒーローは独特な ID がある
89	item_0	integer	スロット 1 にあるアイテム
90	item_1	integer	スロット 2 にあるアイテム
91	item_2	integer	スロット 3 にあるアイテム
92	item_3	integer	スロット 4 にあるアイテム
93	item_4	integer	スロット 5 にあるアイテム
94	item_5	integer	スロット 6 にあるアイテム
95	item_uses	object	プレイヤーがアイテムを使用した回数
96	kill_streaks	object	プレイヤーのキルストリーク（連続キル）に関するインフォメーション
97	killed	object	プレイヤーを倒したユニットに関するインフォメーション
98	killed_by	object	プレイヤーを倒した相手

ID	特徴名	値	説明
99	kills	object	プレイヤーがヒーローを倒した数
100	kills_log	object	プレイヤーがどの敵ヒーローがとどの時間で倒したに関するインフォメーション
101	time	integer	プレイヤーが敵ヒーローを倒した時間（秒単位）
102	key	string	倒されたヒーロー
103	lane_pos	object	レーンの位置に関するインフォメーション
104	last_hits	integer	クリープをとどめを刺した数
105	leaver_status	integer	プレイヤーがゲームを退出したかを表示する整数 0:退出しない 1:安全に退出した 2+:放棄した
106	level	integer	試合が終わった時のレベル
107	lh_t	array of integer	各時間でとどめを刺したの数を表示する配列
108	life_state	object	HP のステータス
109	max_hero_hit	object	プレイヤーが敵に与えたマックスダメージ（1 回）
110	multi_kills	object	プレイヤーがあったマルチキルの数
111	obs	object	プレイヤーが Observer Ward(味方ヒーローはその周囲の視野を得ることができる)を放置した場所
112	obs_left_log	object	Observer Ward が消したインフォメーション
113	obs_log	object	プレイヤーが Observer Ward を放置した場所と時間
114	obs_placed	integer	プレイヤーが Observer Ward を放置した数
115	party_id	integer	パーティーの ID
116	permanent_buffs	object	ゲームが終わった時のプレイヤーがある永久の buff
117	pings	integer	ping の総量
118	purchase	object	プレイヤーが買ったアイテムに関するインフォメーション
119	purchase_log	object	プレイヤーが買ったアイテムの時間に関するインフォメーション
120	time	integer	アイテムが買った時間（秒単位）
121	key	string	アイテムの ID
122	charges	integer	同じアイテムの数
123	rune_pickups	integer	ルーンを拾った数
124	runes	runes	拾ったルーンの種類

ID	特徴名	値	説明
125	runes_log	object	ルーンを拾った時間を表示する配列
126	sen	object	Sentries Ward（周囲のインビジブルなことを見えるようになる）を放置した場所
127	sen_left_log	object	Sentries Ward が消したインフォメーション
128	sen_log	object	プレイヤーが Sentries Ward を放置した場所と時間
129	sen_placed	integer	プレイヤーが Sentries Ward を放置した数
130	stuns	number	受けたスターンの総時間
131	times	array of integer	他の時間配列を対応した時間（秒単位）
132	tower_damage	integer	プレイヤーがタワーに与えた総ダメージ
133	xp_per_min	integer	プレイヤーが得た経験値の数（分単位）
134	xp_reasons	object	得た経験値のソース
135	xp_t	array of integer	各時間でプレイヤーの経験値
136	personaname	string	プレイヤーの ID
137	name	string	プレイヤーの ID
138	last_login	string<date-time>	最後にログインした時間
139	radiant_win	boolean	1 は radiant が勝った、0 は dire が勝った
140	start_time	integer	1970 から試合が開始する時間（秒単位）
141	duration	integer	試合が続いた時間（秒単位）
142	cluster	integer	クラスター
143	lobby_type	integer	ロビーのタイプを表示する整数
144	game_mode	integer	ゲームのモードを表示する整数
145	patch	integer	ゲームのバージョンを表示する整数
146	region	integer	ゲームの地域を表示する整数
147	isRadiant	boolean	プレイヤーが Radiant なら 1、Dire なら 0
148	win	integer	プレイヤーが勝ったか
149	lose	integer	プレイヤーが負けたか
150	total_gold	integer	試合が終わった時のトータルゴールド
151	total_xp	integer	試合が終わった時のトータル経験値
152	kills_per_min	number	プレイヤーが敵を倒すの数（分）

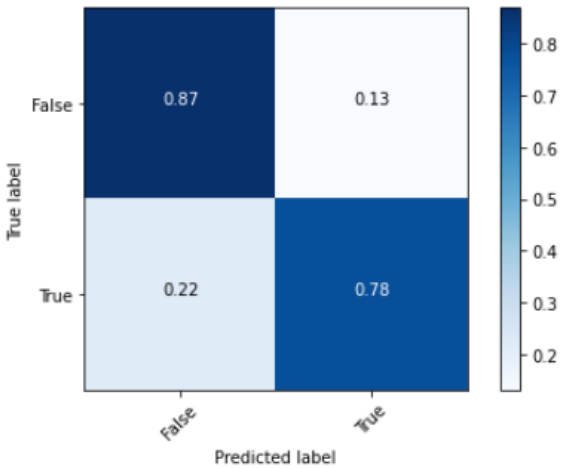
ID	特徴名	値	説明
153	kda	number	kill/dead/assistant
154	abandons	integer	試合を放棄したか
155	neutral_kills	integer	中立クリープを倒した数
156	tower_kills	integer	タワーを破壊した数
157	courier_kills	integer	Courier（アイテム運んでくれるユニット）を倒した数
158	lane_kills	integer	プレイヤーがレーンのクリープを倒した数
159	hero_kills	integer	プレイヤーが敵ヒーローを倒した数
160	observer_kills	integer	プレイヤーが Observer Ward を倒した数
161	sentry_kills	integer	プレイヤーが sentry Ward 倒した数
162	roshan_kills	integer	プレイヤーが roshan（ゲーム内のボス）を倒した数
163	necronomicon_kills	integer	プレイヤーが Necronomicon が召喚したクリープを倒した数
164	ancient_kills	integer	味方クリープ、タワーが敵クリープを倒した数
165	buyback_count	integer	プレイヤーがバイバックを使った数
166	observer_uses	integer	Observer Ward が使った数
167	sentry_uses	integer	sentry Ward が使った数
168	lane_efficiency	number	レーンの効率
169	lane_efficiency_pct	number	レーンの効率
170	lane	integer	ゲーム開始する時のヒーローがどのレーンに行った
171	lane_role	integer	プレイヤーがレーンでの役割
172	is_roaming	boolean	プレイヤーがローミングしていたか
173	purchase_time	object	プレイヤーが最後に買ったアイテムに関するインフォメーション
174	first_purchase_time	object	プレイヤーが最初に買ったアイテムに関するインフォメーション
175	item_win	object	アイテムが勝ったか
176	item_usage	object	このアイテムはプレイヤーが買ったか（いつも 1）
177	purchase_tpscroll	object	プレイヤーTP（転送のスクロール）を買った数
178	actions_per_min	integer	プレイヤーの APM（毎分操作する回数）
179	life_state_dead	integer	倒された時のステータス
180	rank_tier	integer	プレイヤーがプレイする時のランク
181	cosmetics	array of integer	プレイヤーのスキンなど

ID	特徴名	値	説明
182	benchmarks	object	GPM、XPM などインフォメーションを含めているオブジェクト
183	patch	integer	試合のバージョンを表示する整数
184	region	integer	試合が行った地域
185	all_word_counts	object	試合でのすべてのチャットのワード個数
186	my_word_counts	object	試合でプレイヤーのチャットのワード個数
187	throw	integer	試合に負けた場合のプレーヤーチームの最大のゴールドアドバンテージ
188	comeback	integer	試合に勝った場合のプレーヤーチームの最大のゴールドディスアドバンテージ
189	loss	integer	試合に負けた場合のプレーヤーチームの最大のゴールドディスアドバンテージ
190	win	integer	試合に勝った場合のプレーヤーチームの最大のゴールドアドバンテージ

付録 B

4.5.5 選手パフォーマンスに着目したチーム特徴とゴールドに着目したチーム特徴両方もベースラインに追加した三つのモデルそれぞれの訓練結果

```
検証データ
              precision    recall  f1-score   support

       False       0.77      0.77      0.77      7256
        True       0.78      0.78      0.78      7534

    accuracy                           0.77     14790
   macro avg       0.77      0.77      0.77     14790
weighted avg       0.77      0.77      0.77     14790

*****************************************************
テストデータ
              precision    recall  f1-score   support

       False       0.78      0.87      0.82      7001
        True       0.87      0.78      0.82      7789

    accuracy                           0.82     14790
   macro avg       0.82      0.82      0.82     14790
weighted avg       0.83      0.82      0.82     14790
```



チームに関する特徴全体を追加したナイーブベイズの訓練結果

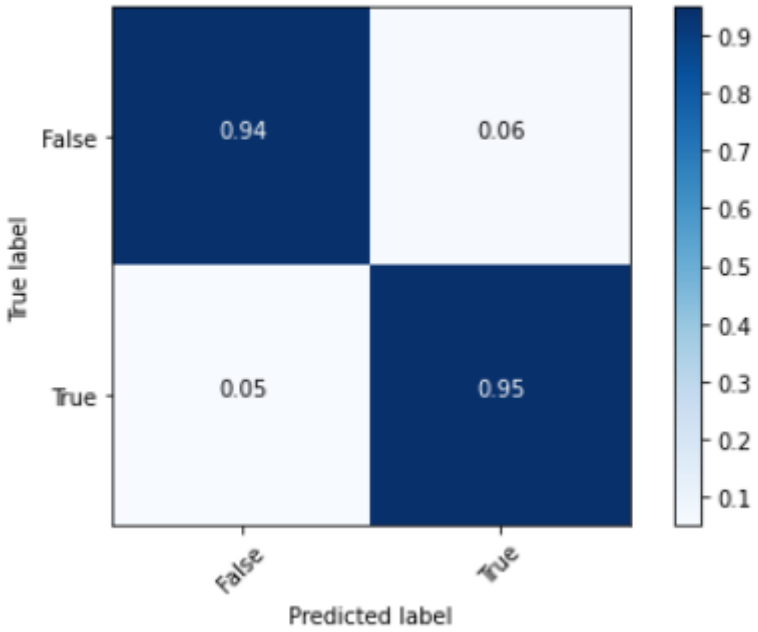
検証データ

	precision	recall	f1-score	support
False	0.94	0.92	0.93	7256
True	0.93	0.94	0.93	7534
accuracy			0.93	14790
macro avg	0.93	0.93	0.93	14790
weighted avg	0.93	0.93	0.93	14790

****

テストデータ

	precision	recall	f1-score	support
False	0.94	0.94	0.94	7001
True	0.95	0.95	0.95	7789
accuracy			0.95	14790
macro avg	0.95	0.95	0.95	14790
weighted avg	0.95	0.95	0.95	14790



チームに関する特徴全体を追加したランダムフォレストの訓練結果

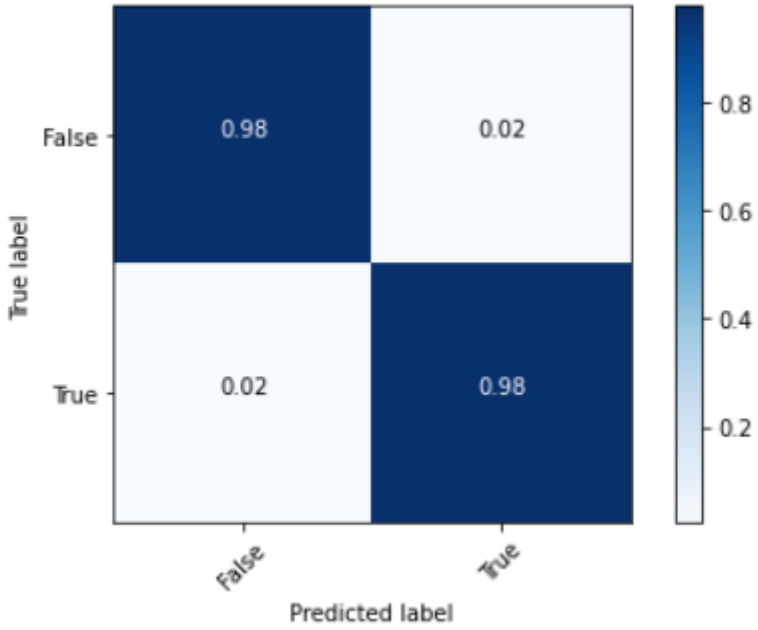
検証データ

	precision	recall	f1-score	support
False	0.97	0.96	0.96	7256
True	0.96	0.97	0.96	7534
accuracy			0.96	14790
macro avg	0.96	0.96	0.96	14790
weighted avg	0.96	0.96	0.96	14790

****

テストデータ

	precision	recall	f1-score	support
False	0.97	0.98	0.97	7001
True	0.98	0.98	0.98	7789
accuracy			0.98	14790
macro avg	0.98	0.98	0.98	14790
weighted avg	0.98	0.98	0.98	14790



チームに関する特徴全体を追加した XGBoost の訓練結果