DHBW
Duale Hochschule
Baden-Württemberg
Mannheim

Portfolioprüfung im Fach

Advanced Machine Learning

Studiengang Wirtschaftsinformatik

Studienrichtung Data Science

Ausarbeitung zum Projekt

Prediction of Betting Odds for Soccer Games

Studenten:	Simon Wrigg	5874903
	Pascal Schmidt	8133405
	Philipp Becht	9443009

Kurs: WWI19DSB

Dozent: Prof. Maximilian Scherer

Ehrenwörtliche Erklärung

Wir versichern hiermit, dass wir die vorliegende Ausarbeitung selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt haben.

Inhaltsverzeichnis

Abbildungsverzeichnis

1. Einleitung

Die Vorhersage von Spielergebnissen im Fußball stellt eine besonders schwierige Aufgabe dar. Wissenschaftliche Untersuchungen haben gezeigt, dass in 45% der untersuchten Spiele das vermeintlich schlechtere Team gewonnen hat. Diese Beobachtung ist auf kleine Fehler im Spielverlauf zurückzuführen, welche sich in Kombination mit einem entsprechenden Konter der Gegnermannschaft spielentscheidend auswirken können. Der Fußball ist demnach ein Sport mit einem hohen Glücksfaktor und charakterisiert sich im Vergleich zu anderen Sportarten über relativ wenige Tore. Gleichzeitig sind Wettbüros eine willkommene Anlaufstelle für begeisterte Fußballfans, die mit ihren Wetten Geld gewinnen möchten.

Im Rahmen des Machine Learning Projekts ist es das erklärte Ziel der Gruppe, Wettquoten für Fußballspiele zu errechnen, welche auf der Vorhersage des Spielergebnisses basieren. Die abgeleiteten Quoten sollen hierbei allerdings für die Nutzer ähnlich attraktiv sein, wie bei der etablierten Konkurrenz.

In der vorliegenden Ausarbeitung soll im besonderen Maße auf die Herangehensweise und die entsprechende Durchführung im Projekt eingegangen werden.

2. Grundlagen

2.1 Ensemble Learning

In diesem Kapitel soll auf die Grundlagen der im Projekt verwendeten Technologien bzw. Algorithmen eingegangen werden.

Beim Ensemble Learning werden mehrere schwache Learnalgorithmen (auch Base Models genannt) als Bausteine zur Erstellung von komplexen Modellen verwendet, indem sie kombiniert werden.[1] Base Models tendieren dazu entweder einen hohen Bias oder eine hohe Varianz aufzuweisen. Durch den Einsatz von Methoden des Ensemble Learnings wird versucht, den Fehler durch die Kombination verschiedener Base Models zu verringern, indem sie zu einem sog. „Strong Learner“ kombiniert werden.[2]

Es gibt verschiedene Arten des Ensemble Learnings. Zu den bekanntesten Vertretern gehören Bagging, Boosting und Stacking.

Beim Bagging werden homogene Weak Learner unabhängig voneinander, aber parallel erlernt. Dabei findet das Training der Prädiktore allerdings mit einer jeweils anderen, zufällig ausgewählten Teilmenge der Trainingsdaten statt. Diesen Vorgang nennt man auch Bootstrapping. Um die letztendliche Prediction zu erhalten, werden die Vorhersagen sämtlicher Prädiktoren zusammengefasst. Man spricht auch von Aggregation, was die Senkung des Bias bzw. der Varianz zur Folge hat.[3]

Dem gegenüber stehen Methoden des Stackings. Das Stacking verwendet heterogene, also andersartige, Weak Learner, welche parallel erlernt werden. Anzumerken ist hierbei, dass die Predictions der Base Models zunächst den Input, also die Trainingsdaten, für ein sog. Meta-Modell darstellen. Erst durch das Training des Meta-Modells (auch Final Learner) erhält man die finale Vorhersage.

[1] Vgl. Géron, 2018, Kap.7, S.181.
[2] Vgl. Brownlee, 2021.
[3] Vgl. Rocca, 2019.

2.2 Gradient Boosting Klassifikation

Das Gradient Boosting ist ein populärer Boosting-Algorithmus. Boosting-Algorithmen zeichnen sich dadurch aus, dass homogene Weak Learner sequentiell und adaptiv erlernt werden.[4] Damit hängt die Performance des neuen Modells von der Performance des vorherigen Modells ab.[5] Bei jedem Durchlauf werden die neuen Prädiktoren an die vom vorigen Prädiktor begangenen Restfehler angepasst. Das Ziel des Algorithmus ist die Reduzierung des Bias.

3. Herangehensweise im Projekt

Zu Beginn des Projekts sind verschiedene Recherchen zu verwandten Themen im Kontext der Vorhersage von Sportergebnissen und der damit verbundenen Quotenberechnung angestellt worden. Schließlich war zunächst zu überprüfen, wie derartige Berechnungen umgesetzt werden können und wie die Mechanik der Quotenberechnung im Allgemeinen funktioniert.

Um ein Konzept zur Durchführung des Projekts zu haben, hat die Gruppe einen Solution Sketch erarbeitet, welcher die vollständig ausgearbeitete Idee darstellt. Dieser Solution Sketch galt als Startpunkt eines adaptiven Entwicklungsansatzes, da im Sinne eines agilen Vorgehens Anpassungen an der Umsetzungsidee, falls erforderlich, erfolgen sollten.

Im Anschluss daran galt es, passende Daten zu finden, welche die Grundlage für das Machine Learning Projekt darstellen. Nach umfassender Analyse verfügbarer Datensätze hat sich die Gruppe für den „Football Match Probability Predictions“ Datensatz entschieden, welcher auf Kaggle abrufbar ist[6]. Der Datensatz ist gewählt worden, weil er deskriptive Daten zu den Fußballspielen, aber auch historische Daten zu den jeweiligen Mannschaften enthält. Diesen Datensatz galt es zunächst in die projektinterne Entwicklungsumgebung auf Google Colab zu importieren. Danach erfolgten die Schritte der Projektdurchführung, welche im nächsten Kapitel dargelegt werden.

[4] Vgl. Géron, 2018, Kap.7, S.195f.
[5] Vgl. Aliyev, 2022.
[6] Datensatz verfügbar unter https://www.kaggle.com/competitions/football-match-probability-prediction/data (Stand: 14.07.2022).

4. Projektdurchführung

4.1 Datenexploration

Der erste Schritt im Rahmen der Projektdurchführung ist eine ausführliche Exploration der ausgewählten Daten. Erst wenn die Daten, deren Schema, wichtige Verteilungen und Besonderheiten verstanden werden, kann effektiv die Entwicklung von Feautures und darauffolgend das Training eines Modells erfolgen.

Die vorher separierten Trainingsdaten enthalten Informationen zu 85.000 Fussballspielen aus 860 verschiedenen Ligen der Welt im Zeitraum von 2019 – 2021. Es handelt sich also um recht aktuelle Daten. Dabei sind Spiele von 9500 Mannschaften enthalten, für die in 190 Spalten wichtige Informationen bereitgestellt sind. Für jedes Spiel ist in der Spalte „Target“ angegeben, wie das Spiel ausgegangen ist. Die Target-Spalte kann den Wert „Win Home-team“, „Draw“ oder „Win Away-team“ annehmen. Diese drei Klassen stellen letztendlich die Labels für die Vorhersage der Spielergebnisse dar, die Spalte „Target“ ist also die Zielvariable.

Bei Berechnung der Verteilung der drei Klassen fällt auf, dass ein Sieg des Heimteams am häufigsten vorkommt. Die Klasse „Unentschieden“ (engl.: „Draw“) wird am seltensten realisiert. Zur Verdeutlichung dient das folgende Balkendiagramm mit Angabe der prozentualen Klassenverteilung.

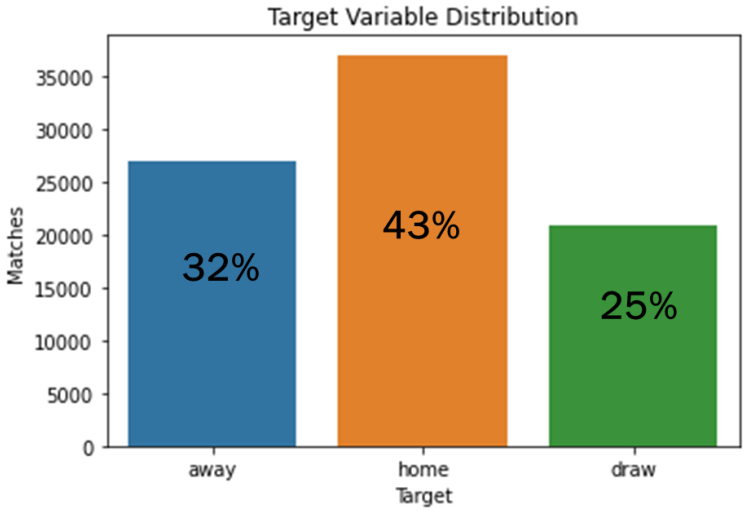

Abbildung 1: Klassenverteilung der Spielergebnisse

Wie im Diagramm ersichtlich, gehen die Spiele zu einer Wahrscheinlichkeit von rund 43% mit einem Heimsieg aus. Daraus lässt sich ein Basismodell für die Vorhersage der Spielergebnisse ableiten. Wenn immer das Ergebnis mit der höchsten Wahr-

scheinlichkeit, also in diesem Fall der Heimsieg, vorhergesagt wird, liegt die Vorhersage zu 43% richtig. Diese Genauigkeit von 43% soll im Rahmen des Projektes durch eine Machine Learning basierte Vorhersage übertroffen werden.

Die drei Klassen werden folglich in numerische Werte kodiert, um später besser Korrelationen feststellen und Modelle trainieren zu können. Die Kodierung wird wie folgt umgesetzt:

Heimsieg = 2 | Unentschieden = 1 | Auswärtssieg = 0

Die weiteren Spalten des Datensatzes teilen sich in deskriptive und historische Spalten auf. In den deskriptiven Spalten sind Meta-Informationen zum jeweiligen Fußballspiel gegeben, wie etwa die Teamnamen, das Spieldatum, die Liga, ob es sich um ein Turnier handelt und wer die Trainer der Teams sind. Die historischen Spalten dagegen enthalten jeweils für das Heim- und Auswärtsteam wertvolle Informationen über deren jeweils letzten zehn Spiele. So wird beispielsweise bekannt, wie viele Tore das Heimteam jeweils in den letzten ein bis zehn Spielen geschossen und kassiert hat, welches Rating (basierend auf dem Anbieter Octosport) das Team und dessen Gegner hatte, und in welcher Liga das Spiel jeweils stattgefunden hat. Die historischen Spalten sind hierzu mit einem {i} versehen, {1} bedeutet, dass es sich um das zuletzt gespielte Spiel handelt, {10} bedeutet, dass es sich um das Spiel handelt, das die Mannschaft vor 10 Spielen bestritt. Zur Verdeutlichung wird ein Beispiel angeführt. Die folgende Spalte trägt den Wert 2:

away_team_history_opponent_goal_{5} = 2

Hiermit ist also gemeint, dass das Auswärtsteam in seinem fünftletzten Spiel zwei Tore kassiert hat.

Diese historischen Daten bieten das Potenzial, aussagekräftige Features zur jeweils aktuellen Performance des Heim- und Auswärtsteams abzuleiten.

4.2 Feature Engineering

Das Feature Engineering wird häufig als der wichtigste Teil im gesamten Machine Learning Prozess betrachtet. Folglich nimmt es auch in diesem Projekt eine wichtige Rolle ein.

Basierend auf den historischen Daten für die jeweils letzten ein bis zehn Spiele der Mannschaften können Features gebildet werden, welche die Durchschnittswerte angeben und somit die aktuelle Team-Performance abbilden. Deshalb wird für die Attribute Rating, Rating des Gegners, Erzielte Tore und kassierte Tore der jeweilige Durchschnitt der letzten zehn Spiele berechnet und als Feature verwendet. Somit wird beispielsweise ein neues Feature kreiert, das angibt, wie viele Tore das Heimteam im Schnitt in den letzten zehn Spielen geschossen hat. Dies hat eine positive Korrelation mit dem Heimsieg.

Zudem wird aus der Information, wie viele Tore in den letzten Spielen erzielt und kassiert wurden, berechnet, wie viele Spiele gewonnen wurden. Somit lassen sich neue Features für die Team-Perfomance erstellen, wie die Anzahl an Siegen des Heimteams in den letzten fünf Spielen.

Darüber hinaus wird die Erstellung weiterer Features getestet, die nicht die Team-Perfomance berücksichtigen, sondern weitere Faktoren. Denkbar ist z.B. ein Feature, das angibt, ob es in den letzten Spielen bei der Mannschaft einen Trainerwechsel gab, oder ob eines der letzten Spiele ein Turnierspiel war. Diese Features werden für das Modell allerdings nicht verwendet, da sie entweder eine sehr geringe Wichtigkeit oder ein geringes Vorkommen aufweisen.

4.3 Modellauswahl und -training

Der separierte Trainingsdatensatz mit den neu erstellten Features wird anschließend mithilfe einer Pipeline für die Modelltrainings vorverarbeitet. Hierzu werden für das Training irrelevante Spalten aus dem Dataframe eliminiert, null-Werte mithilfe eines Simple-Imputers mit Mittelwerten ersetzt und die Daten mithilfe des StandardScaler-Moduls skaliert. Es kommt die Bibliothek scikit-learn zum Einsatz.

Nach der Vorverarbeitung werden verschiedene überwachte Klassifikationsmodelle zur Vorhersage der Spielergebnisse (2/1/0) trainiert. Ziel soll es hierbei sein, den

Bias möglichst klein zu halten, um ein möglichst genaues Modell zu erhalten, da Vorhersagen von Fußballspiel-Ergebissen schnell zu ungenau sein können.

Zunächst wird eine logistische Regression trainiert, die ausdrücklich kein Regressions-, sondern ein Klassifikationsmodell darstellt. Für den vorhandenen Datensatz erscheint dieses Modell allerdings zu simpel, wie auch das Trainieren eines einfachen Entscheidungsbaumes. Deshalb fällt die Wahl im Laufe des Projektes auf das Trainieren von Ensemble Modellen. Hierzu wird ein Random Forest Classifier (Bagging) wie auch ein Gradient Boosting Modell (Boosting) trainiert. Das Gradient Boosting Modell schneidet auf den Trainings- wie auch auf den Testdaten mit einer Genauigkeit von rund 50% deutlich besser ab. Da das Modell außerdem grundsätzlich dafür sorgt, den Bias der Vorhersage zu verringern, wird es als Modell für das Projekt gewählt, erneut trainiert und gespeichert.

4.4 Umwandlung der Predictions in Wettquoten

Mit dem trainierten Modell können über die Funktion „model.predict_proba“ die Ausgangswahrscheinlichkeiten der Spiele vorhergesagt werden. Somit kann eine Vorhersage beispielsweise wie folgt aussehen (Win Home | Draw | Win Away):

(0.55 | 0.25 | 0.20) → 2 (Heimsieg)

Diese vorhergesagten Wahrscheinlichkeiten sind essentiell für die Vorhersage der Wettquoten. Grundsätzlich werden die Wahrscheinlichkeiten zu Wettquoten weiterverrechnet. Eine faire Wettquote, bei der im Schnitt kein Gewinn für den Buchmacher entstehen würde, basiert auf der Berechnung (1 / Wahrscheinlichkeit). Um jedoch eine reale Quote mit Gewinnsicherung für den Buchmacher zu errechnen, wird ein Gewinnfaktor mit einberechnet. Zur Erklärung dieses Konzepts dient die folgende Abbildung:

Ergebnis	Wahrscheinlichkeit (ML-Model)	Faire Quote	Buchmacher-Faktor	Reale Quote
Win Home-Team	62%	100/62 = **1,61**	0,9	1,61*0,9 = **1,45**
Draw	27%	100/27 = **3,7**	0,9	3,7*0,9 = **3,33**
Win Away-Team	11%	100/11 = **9,09**	0,9	9,09*0,9 = **8,18**

→ Reale Quote = 100 / Wahrscheinlichkeit * Buchmacher-Faktor

Abbildung 2: Berechnung von fairen und realen Wettquoten

Ausgehend von den Wahrscheinlichkeiten werden für das Projekt durch eine Python-Funktion reale Wettquoten berechnet. Dabei wird der Buchmacher-Faktor als Funktionsparameter mitgegeben, um diesen dynamisch anpassen zu können. Beispielsweise könnte die Quote für ein Unentschieden so erhöht werden, indem der Faktor für Unentschieden erhöht wird. Für dieses Projekt wurde ein Faktor von 0.9 gewählt, um zwar attraktive Quoten anzubieten, aber dennoch einen sehr hohen Buchmacher-Gewinn erwirtschaften zu können.

4.5 Evaluation

Das Gradient Boosting Modell erreicht auf den rund 26000 Testdaten eine Genauigkeit von rund 50%. Dies ist besser als das erwähnte Basismodell (Vorhersage Heimsieg) mit 43% und außerdem besser als einfaches Raten, da es drei Klassen gibt, sodass die gleichverteilte Wahrscheinlichkeit bei 33% liegen würde. Wenn nur Spiele mit einer vorhergesagten Ausgangswahrscheinlichkeit von über 50% zur Evaluation herangezogen werden, steigt die Genauigkeit auf 61%.

Die Vorhersage einzig und allein nach der Genauigkeit der Ergebnisvorhersagen zu evaluieren, trifft jedoch nicht den Charakter dieses Projektes. Schließlich ist es das Ziel, Wettquoten vorherzusagen, die sich aus den Ergebniswahrscheinlichkeiten ableiten. Wenn die Vorhersage eine Genauigkeit von 100% hätte, so würde immer die niedrigste Quote gewinnen, was sehr durchschaubar wäre. Daher ist das primäre Ziel nicht, die höchstmögliche Genauigkeit zu erreichen, sondern möglichst passende Quoten vorherzusagen.

Folglich werden die vorhergesagten Quoten qualitativ mit realen Quoten, die tatsächlich von Wettanbietern angeboten wurden, verglichen. Da keine extrahierbaren Tabellen mit Quotendaten gefunden werden konnten, wird der Vergleich beispielhaft an manuell kopierten Quotendaten von zehn Bundesligaspielen durchgeführt. Die Daten wurden vom Anbieter „Oddsportal“ kopiert. Dies dient lediglich dazu, ein Gefühl dafür zu bekommen, ob die durch das Machine Learning Modell vorhergesagten Quoten den echten Buchmacher-Quoten ähneln.

Der qualitative Vergleich kann in der folgenden Abbildung eingesehen werden:

Vorhergesagte Quote			Oddsportal (echte) Quote			Delta der Quoten		
home_odd	draw_odd	away_odd	oddsportal_home	oddsportal_draw	oddsportal_away	home_odd_difference	draw_odd_difference	away_odd_difference
2.79	3.98	2.05	4.04	4.21	1.81	1.25	0.23	-0.24
3.22	3.34	2.06	3.27	3.29	2.34	0.05	-0.05	0.28
1.55	4.09	4.84	2.91	3.30	2.55	1.36	-0.79	-2.29
1.76	3.63	3.97	2.05	3.38	3.94	0.29	-0.25	-0.03
2.05	3.40	3.19	3.13	3.73	2.22	1.08	0.33	-0.97
2.40	3.70	2.44	3.29	4.00	2.08	0.89	0.30	-0.36
4.08	3.36	1.81	10.68	6.71	1.26	6.60	3.35	-0.55
1.67	4.18	3.83	2.19	3.87	3.10	0.52	-0.31	-0.73
1.34	5.13	6.34	1.38	5.25	8.13	0.04	0.12	1.79
1.46	4.53	5.13	1.64	3.93	5.73	0.18	-0.60	0.60

Abbildung 3: Beispielhafter Vergleich der vorhergesagten und echten Wettquoten

Der Vergleich zeigt, dass die vorhergesagten Quoten nur geringfügig von den echten Quoten abweichen. Tendenziell wurden die Heimquoten niedriger und die Auswärtsquoten etwas höher vorhergesagt, auf Basis von zehn Spielen ist damit jedoch keine verallgemeinernde Aussage zu treffen.

4.6 Deployment

Die Funktionalität, für ein Fußballspiel den Spielausgang, die Ausgangswahrscheinlichkeiten sowie die Wettquoten vorherzusagen, wird folglich im Rahmen einer FastAPI deployt. Das FastAPI Webframework bietet die Möglichkeit, hoch performante RESTful-APIs in Python zu bauen.

Die für dieses Projekt entwickelte API verfolgt den Zweck, Buchmachern die initialen Quoten vorherzusagen, bzw. sie bei der Errechnung der initialen Quoten zu beraten. Somit können Kunden mit der Post-Funktion der API ein Spiel als Input angeben (Name des Heimteams, Name des Auswärtsteams und Spieldatum) und bekommen die empfohlenen Wettquoten als Output.

Hierzu wird das zuvor trainierte und gespeicherte Gradient Boosting Modell im Google Colab Notebook für die FastAPI importiert. Außerdem werden die vorverarbeiteten Fußballspiel-Daten in das Notebook importiert, damit dem Modell die jeweils passenden Daten für die Vorhersage zur Verfügung stehen. Auf Basis des API-Inputs werden die zugehörigen Spieldaten gefiltert und dem Modell weitergegeben, das dann die Vorhersage für das angefragte Spiel errechnet.

Um die API über eine öffentliche URL bereitstellen zu können, wurde die FastAPI über das Google Colab Notebook mit dem Service „ngrok“ deployt. Solange die Notebook-Zelle mit dem ngrok-Befehl ausgeführt wird, ist die API über einen öffentlichen Link zugänglich. Bei Hinzufügen eines „/docs“ an die ngrok.io URL öffnet sich eine Seite, in welcher die API dokumentiert ist und ausprobiert werden kann.

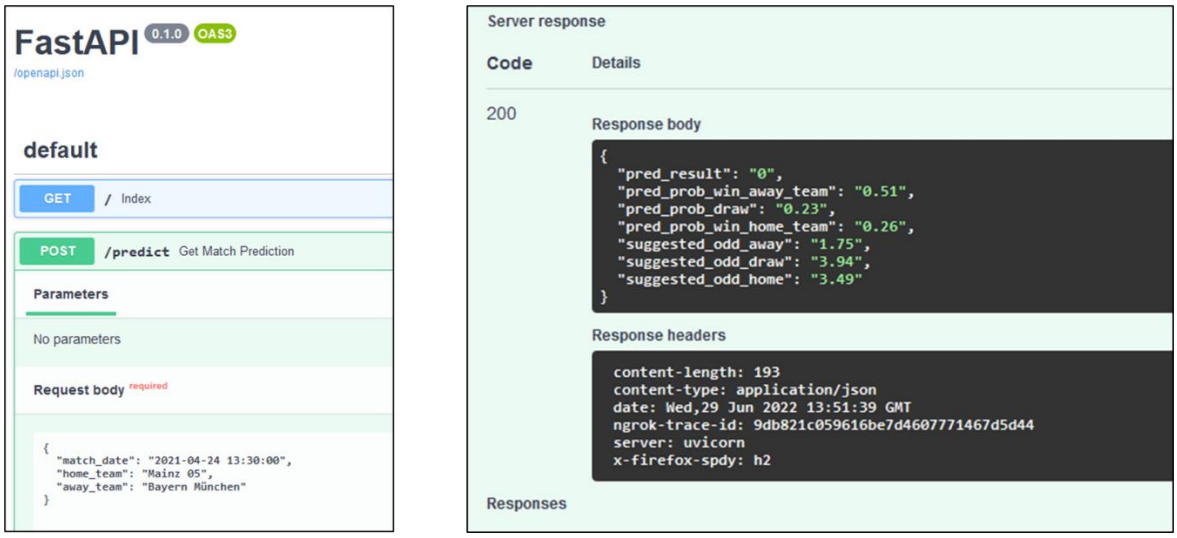

Abbildung 4: Screenshot von der entwickelten FastAPI Docs Seite

In diesem Screenshot wird der Post-Befehl beispielhaft für das Spiel Mainz 05 gegen Bayern München vom 24.04.2022 ausgeführt. Im rechten oberen Teil kann der Response Body eingesehen werden, der die passenden Vorhersagen enthält.

5. Fazit

5.1 Zusammenfassung der Ergebnisse

Zusammenfassend lässt sich sagen, dass im Rahmen des Machine Learning Projekts ein Modell zur Vorhersage von Spielergebnissen trainiert und evaluiert wurde, um darauf aufbauend attraktive Wettquoten für Fußballspiele zu berechnen. Das Ziel des Projekts ist somit erreicht worden.

Die Vorhersage der Quoten wird mithilfe einer FastAPI realisiert, welche auf den jeweiligen Wahrscheinlichkeiten der Spielausgänge basiert. Mithilfe der produzierten Lösung ist es demnach möglich, die Wettquoten für historische Fußballspiele zu ermitteln, mit denen ein potenzielles Wettbüro bestenfalls sogar Geld verdienen könnte. Nichtsdestotrotz ist an dieser Stelle anzumerken, dass die erreichten Predictions noch nicht auf dem gleichen Niveau wie die der etablierten Anbieter für Sportwetten sind.

Besonders hervorzuheben sind die Learnings der Teammitglieder durch das Projekt, da neue Herangehensweisen und Technologien erprobt worden sind. Um den Nutzen der Daten zu maximieren sind bspw. zusätzliche Features erstellt worden, welche im Verlauf des Projekts einen wichtigen Beitrag zur Vorhersage der Spielausgänge geleistet haben. Zudem sind neue Technologien verwendet worden, wobei hier besonders die Verwendung einer FastAPI hervorzuheben ist. Die ganzheitliche Umsetzung eines „Full-Stack“ Machine Learning Use Cases stellte die Gruppe zwar vor eine Herausforderung, welcher aber mit Freude entgegnet wurde.

5.2 Kritische Reflexion

An dieser Stelle soll eine kritische Reflexion der Ergebnisse vorgenommen werden.

Wie bereits erwähnt, sind die erreichten Predictions bereits zufriedenstellend, allerdings noch nicht auf dem gleichen Niveau wie die der etablierten Wettbüros. Ein Grund hierfür ist die Tatsache, dass in den hier erzeugten Quoten noch keine nachfrage-gesteuerte Quotenanpassung erfolgen kann. Besonders wenn viele Menschen den gleichen Tipp abgeben, ist eine situative Anpassung der Quoten notwendig, wenn gleichzeitig auch ein wirtschaftliches Geschäftsmodell betrieben werden soll.

Zudem sind derzeit nur Vorhersagen auf historischen Daten möglich. Eine Live-Datenanbindung ist durch den „Proof of Concept“-Charakter bis jetzt noch nicht realisiert worden. Auch weitere, denkbar relevante Daten, wie bspw. die Anzahl von roten Karten einer Mannschaft, könnten im Sinne einer Weiterentwicklung des Projekts neue Impulse zur Verbesserung der Vorhersagen liefern. Diese Daten müssten allerdings aus anderen Datenquellen bezogen werden.

5.3 Ausblick

Die bereits angedeutete Weiterentwicklung des Projekts könnte in Zukunft einen echten Mehrwert für Buchmacher darstellen.

Hierzu wäre es allerdings notwendig, dass die Daten aus Kaggle im möglichst gleichen Schema auch für zukünftige Spiele vorhanden wären. Dann wäre es möglich, die Vorhersagen für Spielausgänge für echte Spielpartien zu nutzen. Sollten die Ergebnisse zufriedenstellend sein, so könnte die entwickelte API von Buchmachern direkt genutzt werden, um zukünftige Gewinnwahrscheinlichkeiten zu berechnen.

Quellenverzeichnis

Aliyev, V.: Gradient Boosting Classification explained through Python, https://towardsdatascience.com/gradient-boosting-classification-explained-through-python-60cc980eeb3d, 2020.

Brownlee, J.: Strong Learners vs. Weak Learners in Ensemble Learning, 2021.

Géron, A.: Praxiseinstieg Machine Learning mit Scikit-Learn & Tensorflow, O´Reilly Verlag, 2018.

Rocca, J.: Ensemble methods: bagging, boosting and stacking, 2019.