Vorhersage von Immobilienpreisen mittels maschinellen Lernens: Eine explorative Datenanalyse und Modellierung von Grundstücksdaten aus Milwaukee

Im Rahmen des Moduls "Applied Data Science II: Machine Learning und Reporting"
an der Digital Business University of Applied Sciences DBU Berlin

Autor:	Simon Sahli
Studiengang:	Data Science und Management (M.Sc.)
Immatrikulation:	190252
Adresse:	CH-3065 Bern
E-Mail:	simon.sahli@student.dbuas.de
Datum der Einreichung:	22.12.2024

Zusammenfassung

Diese Arbeit untersucht die Anwendung von Machine-Learning-Methoden zur Analyse und Prognose von Immobilienpreisen in einem heterogenen Datensatz. Die zugrunde liegenden Daten umfassen Verkaufsinformationen von 2002 bis 2022 und zeichnen sich durch unterschiedliche Formate, fehlende Werte und starke Ausreisser aus. Nach einer umfassenden Datenaufbereitung und explorativen Datenanalyse wurden verschiedene Machine-Learning-Modelle, darunter lineare Regression, Entscheidungsbäume, Random Forests und Gradient Boosting, getestet und verglichen. Die Ergebnisse zeigen, dass insbesondere Gradient Boosting in Kombination mit optimierten Hyperparametern eine hohe Vorhersagegenauigkeit bietet, wobei das finale Modell einen Test-Score von R2 = 0.80 bieten kann. Die Analyse identifiziert Schlüsselvariablen wie Grundstücksgrösse, Gebäudealter und geografische Lage als wesentliche Einflussfaktoren auf die Verkaufspreise. Neben den technischen Ergebnissen untersucht die Arbeit die Herausforderungen der Datenvorbereitung. Abschliessend werden Vorschläge für zukünftige Arbeiten, wie die Integration zusätzlicher sozioökonomischer Daten und der Einsatz neuronaler Netze, zur weiteren Verbesserung der Modellleistung gegeben.

Inhaltsverzeichnis

1 Einleitung

In den letzten Jahren hat die Anwendung maschineller Lernverfahren in der Immobilienbranche erheblich an Bedeutung gewonnen. So untersuchten beispielsweise Coleman et al. (2022)[5] verschiedene maschinelle Lernansätze zur Vorhersage von Immobilienpreisen.

Diese Studienarbeit beschäftigt sich mit der Analyse von Immobilienverkäufen in Milwaukee, Wisconsin, und zielt darauf ab, ein robustes Vorhersagemodell für Verkaufspreise zu entwickeln. Durch die Verwendung maschineller Lerntechniken wird die Beziehung zwischen Eigenschaften wie Baujahr, Stil und geografischer Lage der Immobilien sowie deren Verkaufspreis untersucht. Die Arbeit zeigt, dass durch eine stärkere Datenbereinigung und die Anwendung von Gradient-Boosting-Modellen eine Steigerung der Vorhersagegenauigkeit erzielt werden kann.

Die Arbeit ist wie folgt aufgebaut: Nach der Einleitung wird in Kapitel 2 die Datenquelle vorgestellt und die Schritte der Datenvorbereitung beschrieben. Dabei werden zwei verschiedene Ansätze für das Daten-Preprocessing miteinander verglichen. Anschliessend erfolgt eine explorative Datenanalyse (EDA), um erste Erkenntnisse über die Daten und deren Verteilung zu gewinnen. In Kapitel 3 werden verschiedene maschinelle Lernmodelle getestet und miteinander verglichen. Die Arbeit schliesst mit einer kurzen Diskussion der Ergebnisse und einem Ausblick in Kapitel 4, in dem mögliche Weiterentwicklungen und alternative Ansätze vorgeschlagen werden.

Eine Übersicht über dieses Projekt sowie die Studienarbeit selbst sind online unter folgendem Link einsehbar:
`https://www.simonsahli.ch/project_milwaukee.html`

2 Daten & Analyse

Dieses Kapitel der Studienarbeit referenziert auf das erste Notebook “1-eda.ipynb”.

2.1 Datenquelle

Die gemäss Aufgabenstellung zu untersuchenden Daten stammen von der Stadt Milwaukee im Bundesstaat Wisconsin[1]. Mitte November 2024 wurden zunächst die Datenreihen von 2002 bis 2018 aufgearbeitet, bevor im weiteren Verlauf separate Datensätze zu den Jahren 2019 bis 2022 entdeckt und entsprechend in die Analyse integriert wurden. Hier zeigte sich bereits eine erste Herausforderung: Der aggregierte Datensatz von 2002 bis 2018 weist in einzelnen Variablen leicht abweichende Bezeichnungen und Formatierungen im Vergleich zu den darauffolgenden jährlichen Datensätzen auf. Dieses Beispiel

verdeutlicht, wie wichtig eine im Voraus gut durchdachte Datenstruktur und konsistente Bezeichnung von Datensätzen ist, um nachträglichen Mehraufwand zu vermeiden.

Insgesamt stehen nach der Aggregation der verschiedenen Quelldateien und vor der Datenbereinigung 60'743 Beobachtungen und 19 Variablen zur Verfügung. Eine detaillierte Tabelle der einzelnen Variablen mit Bezeichnung ist im Bereich 1-2 des erwähnten Notebooks "1-eda-ipynb" ersichtlich. Grundsätzlich lässt sich festhalten, dass verschiedenste Grundstücke bzw. Liegenschaften je Region ("District") und Grundstückskategorie ("PropType", "Style") zu unterschiedlichen Zeitpunkten ("Sale_date") mit unterschiedlichen Eigenschaften ("Year_Built", "Bdrms", ...) und entsprechenden Preisen ("Sale_price") verkauft wurden. Letzteres ist die – im Kontext dieser Studienarbeit – abhängige Variable, die mithilfe eines oder mehrerer Machine-Learning-Modelle prognostiziert werden soll. Zuvor liegt die Herausforderung in der Aufbereitung der sehr ungleichmässig verteilten Daten, was Gegenstand des nächsten Unterkapitels ist.

2.2 Datenvorbereitung

Im Folgenden wurde eine EDA (Explorative Datenanalyse) und eine dafür notwendige Datenvorbereitung (Preprocessing) insgesamt zweimal durchgeführt. Zunächst ist ein Ansatz verfolgt worden, der die fehlenden Werte in den kategoriellen Variablen beibehaltet, diese jedoch als "Unbekannt" überschreibt. Dieser Ansatz, nachfolgend Ansatz 1 genannt, ist gewählt worden, um möglichst keine Daten oder Informationen zu verlieren.
Zum Vergleich der Resultate wurden die Daten im Anschluss erneut und nach einem radikaleren Ansatz aufbereitet. Im Rahmen dieses Ansatzes, nachfolgend Ansatz 2 genannt, wurden die Zeilen mit fehlenden Werten grundsätzlich komplett aus dem Datensatz entfernt. Im weiteren Verlauf der Arbeit wird auf beide Ansätze Bezug genommen und die Ergebnisse im Kapitel 3 gegenübergestellt. In den Anhängen, d.h. im Programmcode, wird aus Gründen der Übersichtlichkeit ausschliesslich Ansatz 2 im Detail aufgeführt. Der Programmcode für Ansatz 1 kann bei Interesse zusätzlich angefragt werden.

Ein erheblicher Teil der Daten enthält fehlende Werte ("NaN"). Die Variablen mit den grössten Lücken im Datensatz sind "CondoProject" (83 %) und "Extwall" (22 %). Da eine Imputation dieser Variablen wenig sinnvoll erschien, wurden beide Variablen für die weitere Modellierung entfernt.

Eine weitere Schwierigkeit besteht in Kategorien wie beispielsweise "PropType", die sehr ungleich verteilt sind. Dies führt dazu, dass einige Kategorien nur wenige Datenpunkte für das Training und Testen der Modelle enthalten. In Variante 1 wurde dies unverändert belassen. In Variante 2 wurde erneut stärker eingegriffen: Es wurde ein Vorgehen definiert, bei dem sehr seltene Ausprägungen von

“PropType”, wie “Vacant Land”, komplett aus dem Datensatz entfernt werden (232 Zeilen).

Insbesondere schwierig ist es, in den Datensätzen zu plausibilisieren, in welchen Fällen ein Wert von 0 tatsächlich eine valide Angabe oder einen fehlenden Wert darstellt. Beispielsweise zeigt die Variable “Fin_sqft” – welche die gesamte Quadratmeterzahl der fertiggestellten Fläche einer Immobilie beschreibt – bei 306 Zeilen den Wert 0, was bei einem “PropType” von “Commercial” nicht plausibel ist. In der radikalen Variante 2 wurden auch diese Datenzeilen gelöscht.

In einem weiteren Schritt der Datenvorbereitung wurden numerische Variablen kategorisiert, indem die Ausprägungen in sogenannte “Bins” eingeteilt und mit einem “Label” versehen wurden. In Variante 1 sind die Anzahl der Bins sowie deren Abgrenzungen arbiträr gewählt. So ist beispielsweise die Variable “Year_Built” in verschiedene historische Zeitspannen eingeteilt, wie “Gebäude vor der Jahrhundertwende”, “Frühes 20. Jahrhundert”, “Zwischenkriegszeit” usw. Dies führte jedoch zu einer ungleichen Verteilung der Beobachtungen in den jeweiligen Kategorien.

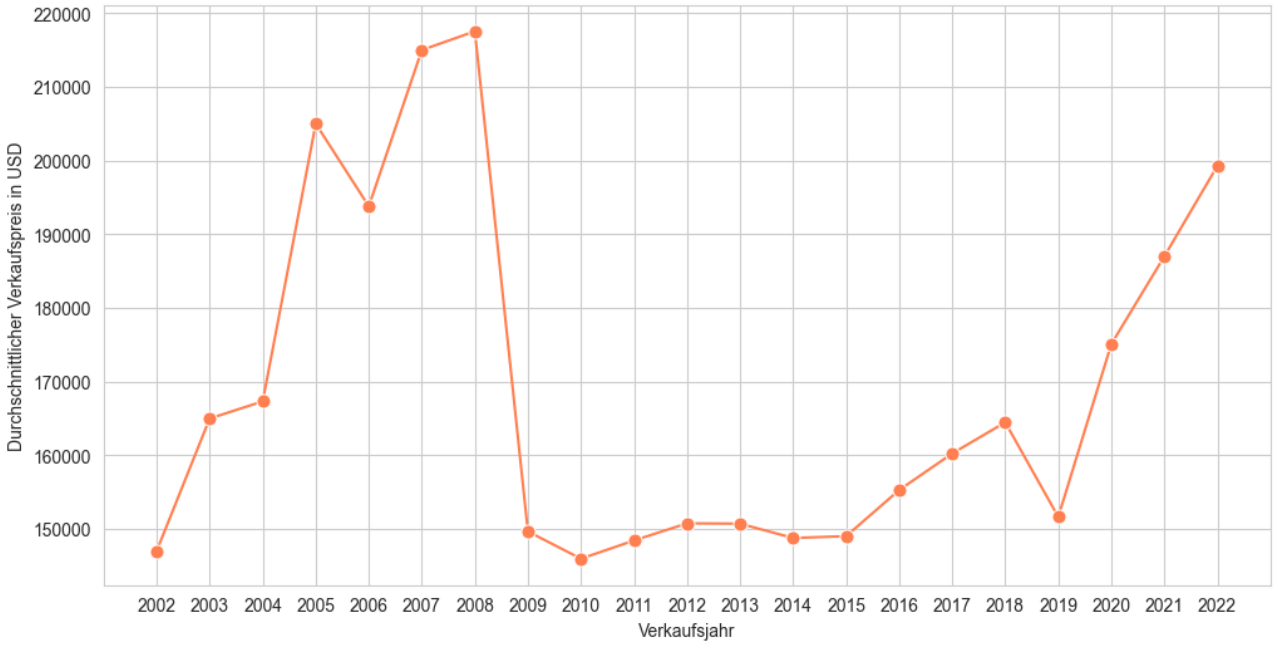
Im Rahmen der Variante 2 wird die Kategorisierung automatisiert durchgeführt, sodass die Bins so gewählt werden, dass für eine festgelegte Anzahl an Bins jeweils ungefähr gleich viele Beobachtungen vorhanden sind.

Um dem Fluch der Dimensionalität im späteren One-Hot-Encoding entgegenzuwirken, wurde in beiden Varianten die Ausprägungen der Variable “Style” reduziert. Mithilfe von Regex-Patterns wurden 393 verschiedene Ausprägungen der Variable in 11 Hauptstile zusammengefasst.
Bei Verkaufspreisen (“Sale_price”) und Grundstücksgrössen (“Fin_sqft”, “Lotsize”) wie auch weiteren numerischen Variablen sind starke Ausreisser identifiziert worden, die zwar grundsätzlich plausibel erscheinen, jedoch stark verzerrend wirken. Hier zeigte sich erneut das Problem der sehr ungleich verteilten Arten von Liegenschaften. In Variante 1 wurden lediglich die extremsten Ausreisser durch manuelle Untersuchung der Boxplots entfernt. In Variante 2 wurde ein radikalerer Ansatz gewählt: Eine Funktion identifiziert alle Ausreisser automatisiert auf Basis eines modifizierten Interquartilsabstands – berechnet aus dem 5. und 95. Perzentil. Im Rahmen von Variante 2 führte dies zur Löschung von 2’905 Zeilen.

Ein abschliessendes Feature-Engineering versucht weitere Informationen in den Datensatz zu integrieren. Dabei ist zunächst – unabhängig davon, ob Variante 1 oder Variante 2 verwendet wurde – das Verkaufsdatum “Sale_date” in “Month_sold” und “Year_sold” aufgeteilt worden. Zudem berechnet die Variable “Building_age_at_sale” das Alter des Hauses bzw. der Liegenschaft zum Verkaufszeitpunkt. Eine weitere eingeführte Boolean-Variable ist “is_post_financial_crisis”, welche die Verkäufe vor und

nach der Weltfinanzkrise 2007 klassifiziert. Abbildung A verdeutlicht die starken Auswirkungen der Krise auf die Immobilienpreise.

Abbildung A: Durchschnittlicher Verkaufspreis über die Jahre



Zu guter Letzt wird mit der Einführung eines Ratio versucht eine Flächeneffizienz ("efficiency_ratio") zu erfassen. Dabei setzt sich die effektive Wohnfläche ins Verhältnis zur Grundstücksgrösse, um eine Kennzahl bzgl. der Raumnutzung zu erhalten. Dieses Feature enthält einige NaN-Werte (bei 7'270 Zeilen) – ein Wert von 0 für "Loftsize" bei PropType "Condominium" ist plausibel.

Nach Abschluss des Preprocessings stehen bei Variante 1 60'418 Zeilen und 32 Spalten zur Verfügung, während Variante 2 56'323 Zeilen mit derselben Anzahl an Features umfasst. Die folgenden Grafiken beziehen sich ausschließlich auf Variante 2.

2.3 Explorative Datenanalyse (EDA)

In diesem Unterkapitel werden nach der Datenvorbereitung und mit Hilfe von Visualieriungen erste Erkenntnisse abgeleitet. Abbildung B ermöglicht einen Vergleich der Verkaufspreise zwischen verschiedenen Bezirken der Stadt. Auffällig ist eine ungleiche Preisverteilung und die Anzahl der Ausreisser deutet auf darauf hin, dass in allen Bezirken regelmässig sehr teure Immobilien verkauft werden.

Abbildung B: Verkaufspreise in verschiedenen Regionen

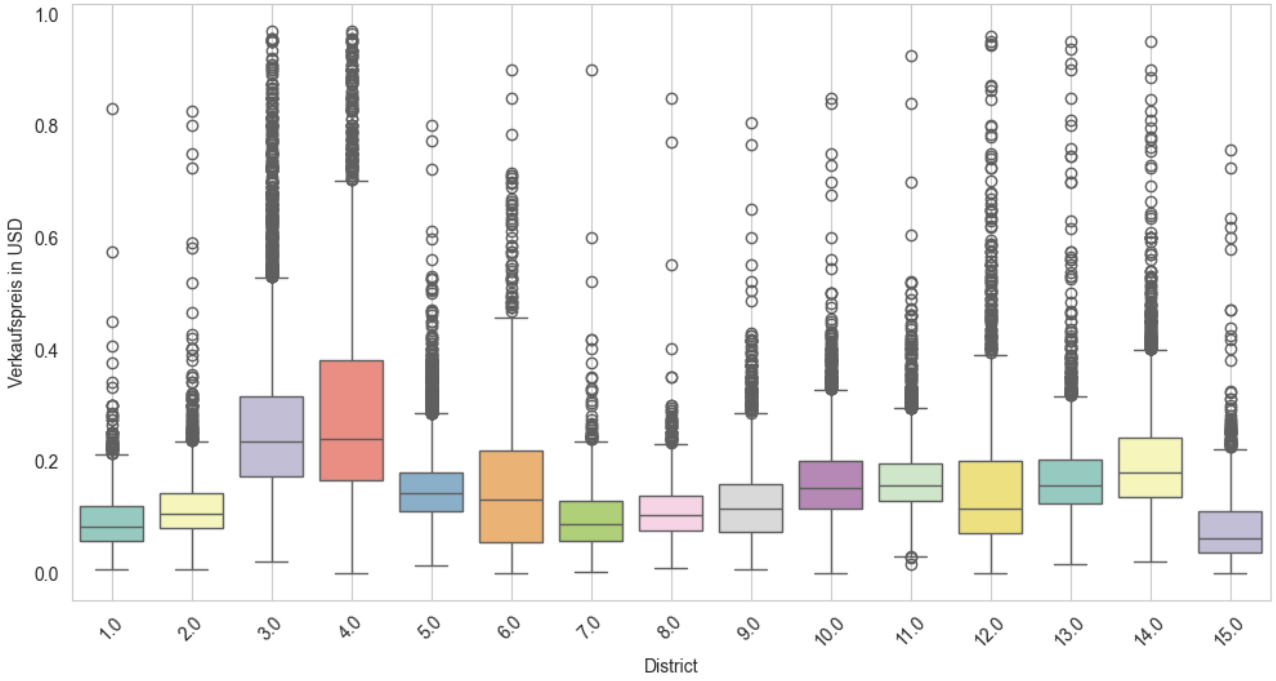


Abbildung C: Verteilung der Verkaufspreise

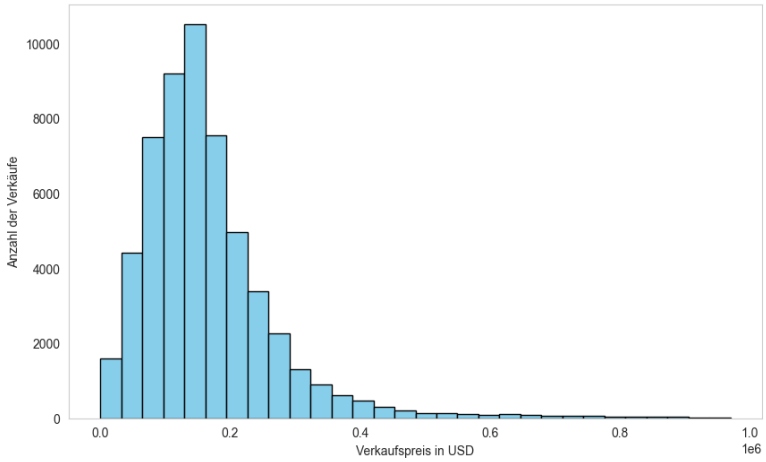
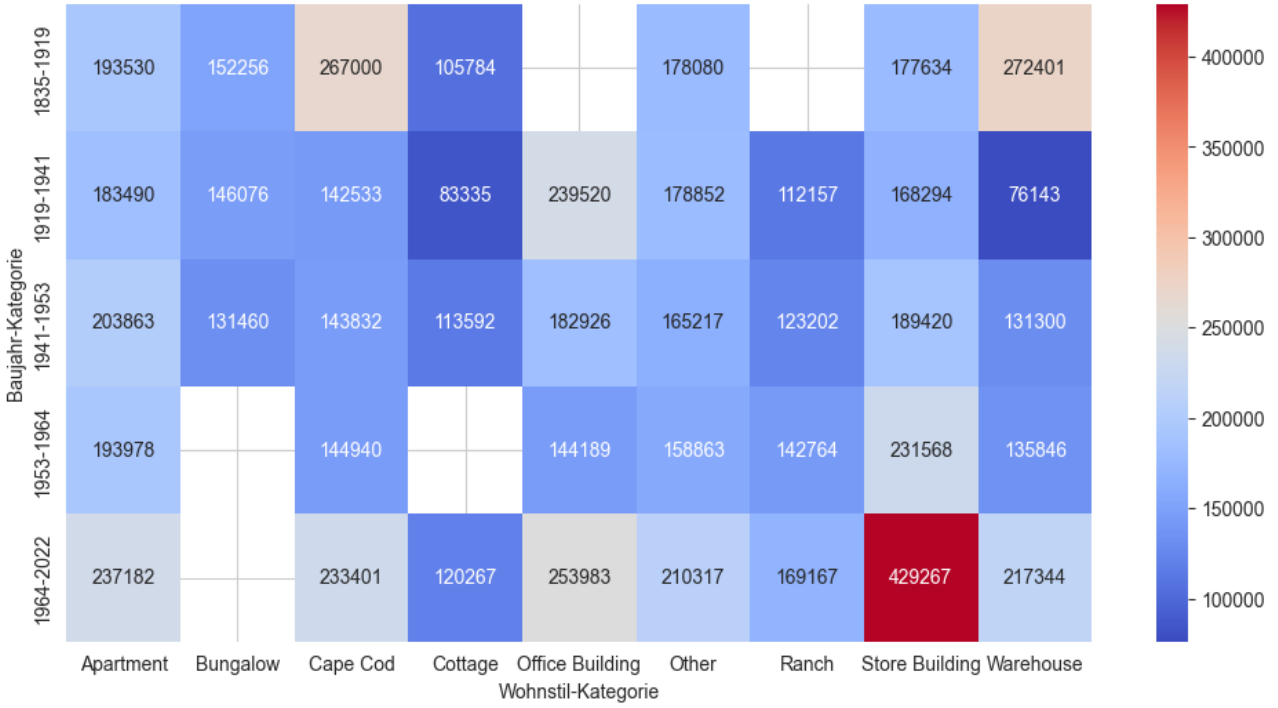


Abbildung C zeigt die Verteilung der Verkaufspreise, wobei die meisten Verkäufe zwischen 100'000 USD und 200'000 USD liegen. Der Graph weist eine rechtsschiefe Verteilung auf, was darauf hinweist, dass höhere Verkaufspreise seltener vorkommen und nur wenige Objekte sehr teuer sind.

Abbildung D zeigt eine Heatmap, welche den durchschnittlichen Verkaufspreis in Abhängigkeit von Baujahr-Kategorien (links) und Wohnstil-Kategorien (unten) darstellt. Die höchsten durchschnittlichen Verkaufspreise treten bei "Store Building" (429'267 USD) in der Baujahr-Kategorie 1964-2022 auf, während ältere Kategorien und Stile wie "Cottage" und "Warehouse" tendenziell niedrigere Preise erzielen. Moderne Wohnstile in jüngeren Baujahren erzielen insgesamt höhere Durchschnittspreise.

Abbildung D: Durchschnittlicher Verkaufspreis basierend auf Baujahr- und Wohnstil-Kategorien



3 Ergebnisse

Dieses Kapitel der Studienarbeit referenziert auf das zweite und dritte Notebook “2-testing-model.ipynb” sowie “3-final-model.ipynb”.

3.1 Modellierung

Im Folgenden orientiert sich die Modellierung und das Testen verschiedener Modelle an den in der Veranstaltung “ADSC21 Applied Data Science II” vorgestellten Ansätzen, basierend auf dem Python-Paket “scikit-learn”[6]. Dabei wurde das Konzept von Pipelines verwendet, um die Modellierungsschritte effizient zu kombinieren. Numerische Features werden, wo notwendig, mit einem IterativeImputer vervollständigt und mit dem StandardScaler standardisiert. Kategorische Features werden mithilfe des OneHotEncoder in numerische Werte umgewandelt. Untersucht wurden die Ergebnisse verschiedener Modelle, um schliesslich ein Supervised-Learning-Verfahren auszuwählen, das die Labels bestmöglich approximiert. Die Reihenfolge der ausgewählten Modelle orientiert sich an der Veranstaltung “ADSC21 Applied Data Science II”.

Eine sehr einfache lineare Regression bildet die Basis der Modellierung. Darauf aufbauend wird ein Decision-Tree-/Entscheidungsbaum-Modell erstellt, um nichtlineare Zusammenhänge erfassen zu können. Song und Lu (2015)[7] verdeutlichen in ihrer Arbeit die Vorteile eines solchen Modells. Neben der einfachen Interpretierbarkeit zeichnen sich Entscheidungsbäume durch eine vergleichsweise hohe Flexibilität aus: Sie ermöglichen die Mischung unterschiedlicher Datentypen und die Darstellung nichtlinearer Zusammenhänge.

Beim Bagging, wie von Abellán und Masegosa (2009)[2] beschrieben, wird ein Ensemble aus Modellen erstellt, die alle denselben Trainingsalgorithmus nutzen. Dabei wird jedes Modell nicht mit der gesamten Trainingsmenge, sondern mit verschiedenen Teilmengen der Daten trainiert. Wird dieser Ansatz mit Entscheidungsbäumen umgesetzt, spricht man von einem Random Forest.

Zuletzt wird das von scikit-learn implementierte Modell des GradientBoostingRegressors angewendet. Gradient Boosted Decision Trees (GBDT) stellen eine Verallgemeinerung des Boosting-Ansatzes auf beliebige differenzierbare Verlustfunktionen dar. Boosting bezieht sich auf eine Ensemble-Methode, bei der viele Modelle nacheinander trainiert werden, wobei jedes Modell aus den Fehlern seines Vorgängers lernt. Laut der scikit-learn-Dokumentation[3] gilt GBDT als ein hervorragendes Modell für sowohl Regression als auch Klassifikation, insbesondere für tabellarische Daten.

3.2 Modellbeurteilung und Interpretation

Die folgende Tabelle 1 fasst die Ergebnisse der verschiedenen Modelle zusammen und vergleicht die Resultate beider Varianten des Datenprocessings.

Tabelle 1: Vergleich der Modell-Scores (Variante 1 vs. Variante 2)

Model	**Variante**	**Train R^2**	**Test R^2**	**CrossVal R^2 (Mean)**	**CrossVal R^2 (Std)**
Pipeline 1 (Linear Regression)	Variante 1	0.309513	0.135780	0.329099	0.164644
	Variante 2	0.136456	0.130473	0.135933	0.004021
Pipeline 2 (Decision Tree)	Variante 1	0.986633	0.785585	0.180337	0.215737
	Variante 2	0.957831	0.485755	0.469317	0.003212
Pipeline 3 (Random Forest)	Variante 1	0.255182	0.112345	0.246213	0.089127
	Variante 2	0.456041	0.460253	0.450993	0.002677
Pipeline 4 (Gradient Boosting)	Variante 1	0.832039	0.760789	0.403357	0.113442
	Variante 2	0.766343	0.754305	0.741219	0.000881
Pipeline 5 (Gradient Boosting + PCA)	Variante 1	0.718994	0.351943	0.274953	0.136954
	Variante 2	0.680135	0.641985	0.631190	0.006327

Das Test-Score (R^2) der sehr simplen linearen Regression ist in beiden Varianten sehr niedrig, was darauf hindeutet, dass die Komplexität der Beziehungen mit diesem einfachen Modell nicht angemessen dargestellt werden kann.
Das Decision-Tree-Modell weist in beiden Varianten des Preprocessings einen extrem hohen Trainings-Score auf, was stark auf Overfitting hindeutet. Dies bedeutet, dass das Modell die Trainingsdaten nahezu perfekt abbildet, jedoch Schwierigkeiten hat, sich auf neue (unbekannte) Daten zu generalisieren. Nach Bramer (2002)[4] können Ursache hierfür die fehlende Begrenzungen der Baumtiefe oder der Anzahl der Blätter sowie die durch das One-Hot-Encoding geschaffene breite Baumstruktur mit vielen Kategorien sein, die die Flexibilität des Modells erhöhen und somit das Risiko von Overfitting verstärken.

Beim Random-Forest-Modell zeigt die zweite, radikalere Variante des Datenprocessings eine bessere Leistung in allen Metriken. Besonders hervorzuheben sind die Cross-Validation-Werte, die darauf hindeuten, dass Variante 2 eine bessere Generalisierung ermöglicht.

Das Gradient-Boosting-Modell zeigt, mit und ohne PCA, das beste Gesamtverhalten mit den vorhandenen Daten. Es profitiert besonders von der stärkeren Datenbereinigung, die in Variante 2 der EDA angewandt wurde.

Zusammenfassend zeigt die stärkere Datenbereinigung durch Variante 2 der EDA eine Verbesserung der Modellrobustheit. Variante 2 ermöglicht eine bessere Generalisierungsfähigkeit und reduziert die Varianz zwischen den einzelnen Folds, wodurch die Cross-Validation-Ergebnisse zuverlässiger werden. Im Gegensatz dazu weist Variante 1 eine hohe Streuung der Cross-Validation-Scores auf, was auf eine geringe Stabilität der Modelle hindeutet. Diese Instabilität könnte auf die heterogene Datenaufteilung in den Kategorien zurückzuführen sein (z. B. ungleiche Verteilung der Kategorien über die Folds oder unzureichende Datenmengen für bestimmte Features).

Tabelle 2: Zusammenfassung der besten Ergebnisse aus GridSearchCV

Rank	**Mean Test Score**	**Std Test Score**	**Learning Rate**	**Max Depth**	**N Estimators**	**Subsample**
1	0.8033	0.0055	0.1	6	300	0.9
2	0.8026	0.0076	0.1	6	300	0.8
3	0.7971	0.0074	0.1	5	300	0.9
4	0.7969	0.0053	0.1	5	300	0.8
5	0.7965	0.0061	0.1	6	200	0.9

Das Modell des Gradient Boosting Regressors, das unter der Datenaufbereitungs-Variante 2 die besten Ergebnisse erzielt hat, bildet die Grundlage für das folgende Hyperparameter-Tuning. Das finale Modell wurde mithilfe von "GridSearchCV" optimiert. Dabei wurden 48 verschiedene Parameterkombinationen getestet, basierend auf einem definierten Wertebereich für vier Parameter ("n_estimators", "earning_rate", "subsample", "max_depth").

Die beste Parameterkombination ergab, dass das finale Modell eine Lernrate von 0.1 aufweist, was bedeutet, dass das Modell bei jedem Schritt langsamere, aber stabilere Aktualisierungen der Gewichte vornimmt, um Overfitting zu vermeiden. Mit einer maximalen Baumtiefe von 6 wird die Komplexität der Entscheidungsbäume begrenzt, sodass das Modell flexibel bleibt, ohne zu stark zu überanpassen. Die Konfiguration mit 300 Entscheidungsbäumen und einem Teil-Sample von 0.9 stellt sicher, dass das Modell robuste Vorhersagen liefert, indem jede Iteration auf 90 % der Daten basiert und so die Varianz reduziert wird.

Das finale Modell mit den optimierten Hyperparametern erzielt einen Test-Score (R^2) von 0.8018. Der Mean Test Score aus der Cross-Validation bestätigt die Generalisierungsfähigkeit des Modells mit einem Wert von 0.8033. Die geringe Standardabweichung der Cross-Validation-Scores (0.0055) unterstreicht die Robustheit des Modells. Der Mean Absolute Error (MAE) von 30'549,9 zeigt, dass die durchschnittliche Abweichung zwischen den vorhergesagten und den tatsächlichen Verkaufspreisen 30'550 USD beträgt.

4 Diskussion und Ausblick

4.1 Diskkussion

Aus den Ergebnissen des vorherigen Kapitels lässt sich schlussfolgern, dass Faktoren wie Grundstücksgrösse, Alter der Gebäude, Stil der Liegenschaft und geografische Lage den Immobilien- bzw. Grundstückspreis beeinflussen. Das abgeleitete Modell konnte etwa 80 % der Varianz in den Verkaufspreisen erklären, was bedeutet, dass ein wesentlicher Teil der Preisbildung in den Daten erfasst ist.

Es ist jedoch zu betonen, dass Vereinfachungen vorgenommen wurden – insbesondere in Bezug zum Data-Preprocessing im Rahmen der Variante 2. Beispielsweise sind bestimmte Ausprägungen von Kategorien, wie vakantes (Bau-)Land, aufgrund geringer Anzahl an Beobachtungen aus der Analyse ausgeschlossen. Zudem wurden Informationen in den Daten durch Aggregation innerhalb der Kategorisierungen zusammengefasst. So sind beispielsweise die 393 Ausprägungen

der Variable “Style“ auf 11 Hauptstile in “Style_category“ reduziert worden. Diese Zusammenfassungen führen dazu, dass Detailinformationen verloren gehen, die folglich nicht in das Modell einfliessen.

4.2 Ausblick

Zukünftige Arbeiten könnten das Vorhersageproblem mithilfe von neuronalen Netzen adressieren, um möglicherweise eine noch genauere Vorhersageleistung zu erzielen. Darüber hinaus könnte zusätzliches Feature-Engineering, etwa unter Einbeziehung sozioökonomischer Daten, dazu beitragen, die Vorhersagekraft der Modelle weiter zu verbessern.

5 Quellenverzeichnis

[1] Data: milwaukee.gov. 2023. «Property Sales Data». Zugegriffen 17. November 2024 über Kaggle. https://www.kaggle.com/datasets/agungpambudi/property-sales-data-real-estate-trends?resource=download
Direkter Zugriff auf data.milwaukee.gov verweigert: «The owner of this website (data.milwaukee.gov) has banned the country or region your IP address is in (CH) from accessing this website.»

[2] Abellán, Joaquín, und Andrés R. Masegosa. 2009. «An Experimental Study about Simple Decision Trees for Bagging Ensemble on Datasets with Classification Noise». In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, herausgegeben von Claudio Sossai und Gaetano Chemello, 446–56. Berlin, Heidelberg: Springer. https://doi.org/10.1007/978-3-642-02906-6_39

[3] «scikit-learn: ensembles: Gradient Boosting, Random Forests, Bagging, Voting, Stacking». o. J. Scikit-Learn. Zugegriffen 18. Dezember 2024. https://scikit-learn.org/stable/modules/ensemble.html

[4] Bramer, Max. 2002. «Using J-Pruning to Reduce Overfitting in Classification Trees». In *Research and Development in Intelligent Systems XVIII*, herausgegeben von Max Bramer, Frans Coenen, und Alun Preece, 25–38. London: Springer. https://doi.org/10.1007/978-1-4471-0119-2_3

[5] Coleman, Walter, Ben Johann, Nicholas Pasternak, Jaya Vellayan, Natasha Foutz, und Heman Shakeri. 2022. «Using Machine Learning to Evaluate Real Estate Prices Using Location Big Data». arXiv. https://doi.org/10.48550/arXiv.2205.01180

[6] «scikit-learn: machine learning in Python — scikit-learn 1.3.2 documentation». o. J. scikit-learn - documentation. Zugegriffen 21. Dezember 2024. https://scikit-learn.org/stable/

[7] SONG, Yan-yan, und Ying LU. 2015. «Decision tree methods: applications for classification and prediction». *Shanghai Archives of Psychiatry* 27 (2): 130–35. https://doi.org/10.11919/j.issn.1002-0829.215044

6 Appendix

6.1 Code

Die Studienarbeit kann im Rahmen eines Repositories durch drei zugrunde liegende .ipynb-Notebooks rekonstruiert werden. Die Notebooks sind in der folgenden Reihenfolge auszuführen:
1-eda.ipynb
2-testing-model.ipynb
3-final-model.ipynb

Das Repository wurde als komprimierte .zip-Datei im Rahmen der Studienarbeit an die Digital Business University of Applied Sciences (DBU) in Berlin übermittelt. Auf Anfrage können die Dateien zur Verfügung gestellt werden. Falls das zugrunde liegende Repository dieser Arbeit veröffentlicht wird, ist es über die Projektseite zugänglich.