

Versuch 1: Modelltraining und Hyperparameter-Optimierung

Datengrundlage: complete_data.csv

Features: Mean, Min, Max von P,D und F

Für die Vorhersage der Zugfestigkeit wurden folgende maschinelle Lernmodelle evaluiert:

- Lineare Modelle:
 - Linear Regression
 - Ridge
 - Lasso
 - ElasticNet
- Baumbasierte Modelle:
 - DecisionTreeRegressor
 - RandomForest
 - GradientBoosting
- Nichtlineare Modelle:
 - Support Vector Regression (SVR)
 - KNeighborsRegressor
 - MLPRegressor

Um optimale Modellparameter zu finden, wurde RandomizedSearchCV mit Kreuzvalidierung eingesetzt. Dabei wurden verschiedene Hyperparameter getestet, um Überanpassung zu vermeiden und die Generalisierungsfähigkeit der Modelle zu verbessern.

Evaluation der Modelle

Die Modelle wurden anhand folgender Metriken evaluiert:

- Mean Absolute Error (MAE)
- Visualisierung der Vorhersagen durch Soll-Ist-Vergleiche

Die Ergebnisse wurden gespeichert und zur weiteren Analyse in einer CSV-Datei festgehalten.

Ergebnisse

Model	MAE
LinearRegression	270,746211
Ridge	270,747179
Lasso	270,746431
ElasticNet	270,843469
RandomForest	95,7732876
GradientBoosting	96,6659558
SVR (linear)	288,879041
KNeighborsRegressor	127,794689
MLPRegressor	170,164034
DecisionTreeRegressor	110,06712

Die Modellbewertung zeigte deutliche Unterschiede in der Vorhersagequalität:

- Baumbasierte Modelle wie RandomForest und GradientBoosting lieferten robuste, aber teilweise unzuverlässige Vorhersagen, insbesondere bei komplexen Zusammenhängen.
- Lineare Modelle wie Ridge und Lasso schnitten ähnlich ab, was auf redundante Merkmale in den Daten hindeuten könnte.
- Neuronale Netzwerke (MLPRegressor) hatten vergleichsweise höhere MAE-Werte, was auf die Notwendigkeit einer tieferen Hyperparameter-Optimierung oder größerer Datenmengen hindeutet.
- SVR mit nichtlinearem Kernel zeigte eine gute Anpassung, erforderte jedoch eine hohe Rechenleistung und lange Trainingszeiten.

Die durchschnittlichen MAE-Werte deuten darauf hin, dass die Generalisierungsfähigkeit der Modelle weiter optimiert werden kann.

Nächste Schritte

- Weitere nichtlineare Modelle testen
- Anpassung & Erweiterung der Features
- $Z = 0$ aus Rohdaten entfernen & Versuch wiederholen
- Rohdaten normalisieren

Model	parameters	optimal MAE
LinearRegression		-270,7462108
Ridge	'alpha': 0,0001	-270,7471786
Lasso	'alpha': 0,0001	-270,7464309
ElasticNet	'l1_ratio': 0,7368421052631579; 'alpha': 0,0001	-270,8434686
RandomForest	'n_estimators': 75; 'min_samples_split': 2; 'min_samples_leaf': 2; 'max_features': None; 'max_depth': 13;	-95,77328756
GradientBoosting	'n_estimators': 175; 'min_samples_split': 8; 'min_samples_leaf': 7; 'max_depth': 4; 'learning_rate': 0,05455594781168517	-96,66595582
SVR (linear)	'gamma': 'auto'; 'epsilon': 6,158482110660261; 'degree': 3; 'C': 24,420530945486497	-288,8790408
KNeighborsRegressor	'weights': 'uniform'; 'p': 1; 'n_neighbors': 8	-127,7946889
MLPRegressor	'learning_rate_init': 0,00042813323987193956; 'hidden_layer_sizes': (50; 50); 'alpha': 0,009102981779915217; 'activation': 'relu'	-170,1640337
DecisionTreeRegressor	'min_samples_split': 9; 'min_samples_leaf': 13; 'max_depth': 14;	-110,06712

