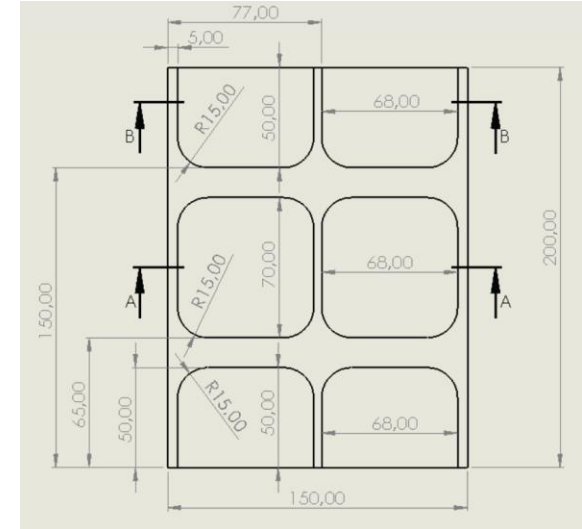
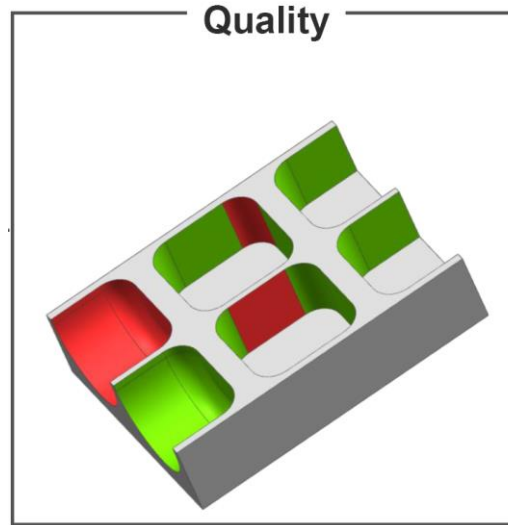


Data Challenge – Gruppe 5

Präsentation

Ilyass Afkir, Sara Ourza, Clemens Schlegel



- Ziel des Projekts
- Programmstruktur
- Datenvorverarbeitung
- Explorative Datenanalyse
- ML-Modelle
- Ergebnisse
 - Feature Selection
- Limitationen
- Zusammenfassung

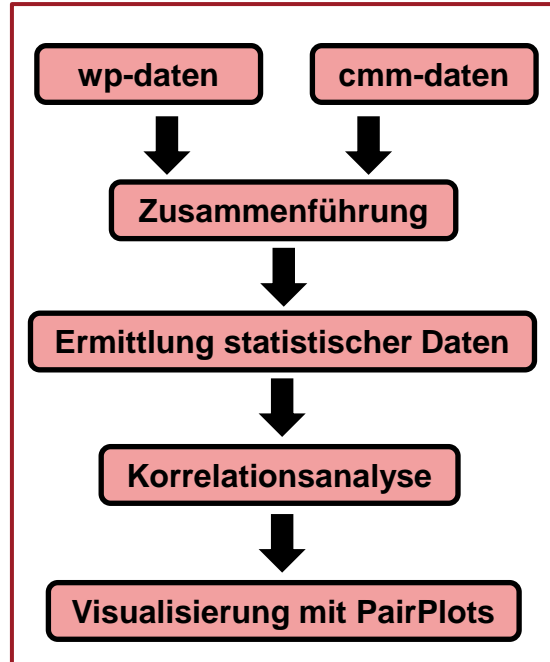
Ziel des Projekts

- Regression für die Messwerte der CMM-Daten
 - Vorhersage der Abweichung zum Nennmaß
- Klassifizierung für jedes geometrische Element und für die gesamte Pocket
 - i.O. oder n.i.O.
 - Verworfen und ursprünglich gedachte Idee: Statt Klassifizierung einen Qualitätsscore berechnen, allerdings ist eine Klassifizierung mit zusätzlicher Messabweichung vom Nennmaß aussagekräftiger im praktischen Gebrauch
- Feature Selection, um eine Vorhersage mit wenigen Maschinensignalen zu ermöglichen
 - Korrelationsanalyse
 - Feature Importance

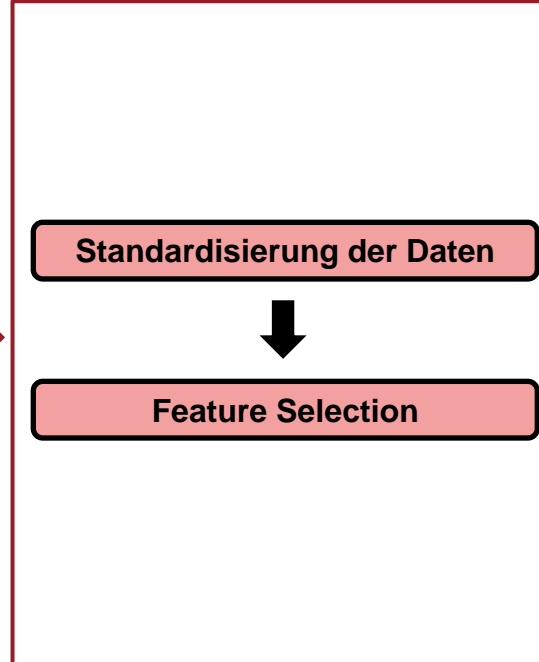
PROGRAMMSTRUKTUR

Programmstruktur

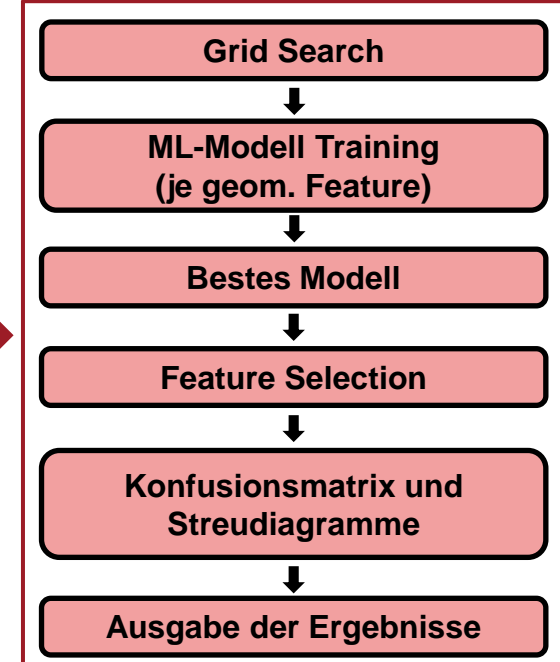
Datenexploration und -analyse



Datenvorbereitung



Auswahl und Training von ML-Modellen




DATENVOVERBARBEITUNG

CMM- und Fräsmaschinendaten werden geladen und nach geometrischen Features sortiert

Korrelationsanalyse der Maschinendaten
→ Überflüssige Signale werden in Listen gespeichert

Gruppieren nach Pockets, berechnen statistischer Feature
→ **Abweichungen vom Nennwert (CMM) und Fräsmaschinendaten zusammenführen**

Standardisierung der Daten vor dem Machine Learning
(scikit learn – StandardScaler)

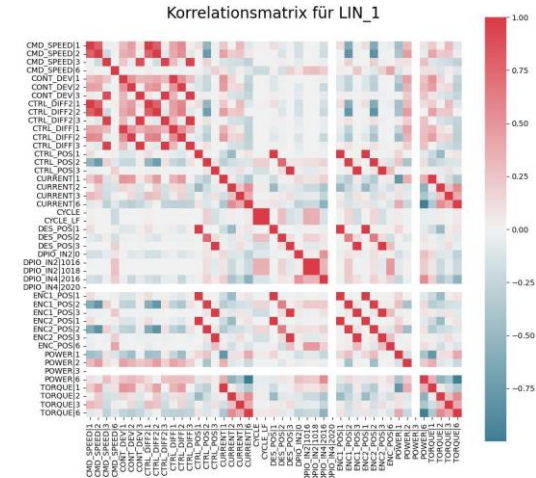
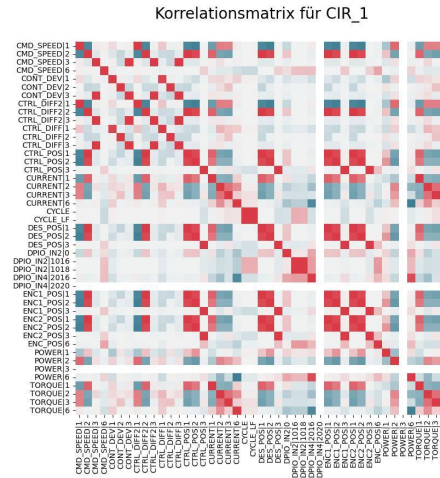


```
all_data_gF = {dict: 8} {'LIN_1': WPNR xPos ... Parallelität_LIN_1_MI
> 'LIN_1' = (DataFrame: (392, 191)) WPNR xPos ... Parallelität_LIN
> 'CIR_1' = (DataFrame: (392, 189)) WPNR xPos ... Lage_CIR_1_MI
> 'LIN_2' = (DataFrame: (392, 191)) WPNR xPos ... Parallelität_LIN
> 'CIR_2' = (DataFrame: (392, 189)) WPNR xPos ... Lage_CIR_2_MI
> 'LIN_3' = (DataFrame: (392, 191)) WPNR xPos ... Parallelität_LIN
> 'CIR_3' = (DataFrame: (392, 189)) WPNR xPos ... Lage_CIR_3_MI
> 'LIN_4' = (DataFrame: (392, 191)) WPNR xPos ... Parallelität_LIN
> 'CIR_4' = (DataFrame: (392, 189)) WPNR xPos ... Lage_CIR_4_MI
len_ = {int} 8
```

EXPLORATIVE DATENANALYSE

Explorative Datenanalyse – Korrelation

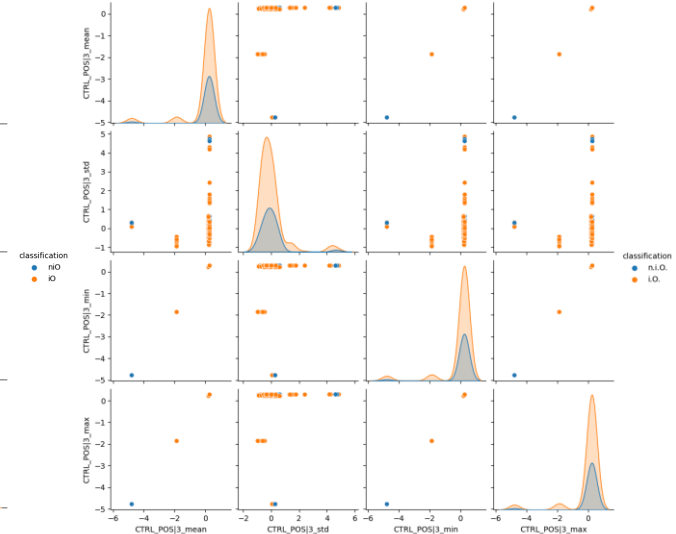
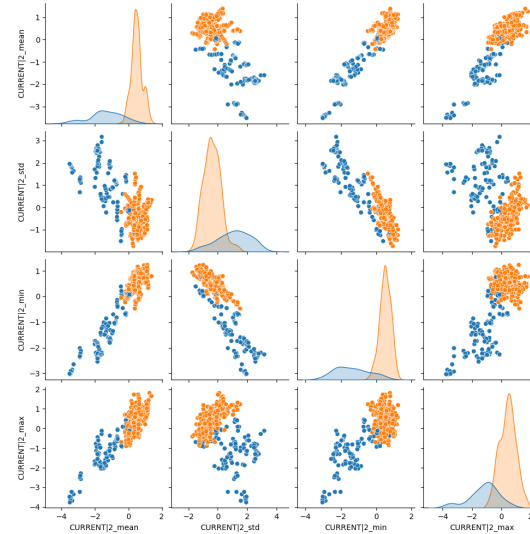
- Korrelationsanalyse zur Auswahl von relevanten Maschinensignalen
- Analyse für jedes geometrische Feature getrennt



- Hohe Korrelation z.B. zwischen Regelabweichung und vorgegebener Achsgeschwindigkeit
- Einige Signale sind konstant (weiß)

Explorative Datenanalyse – Pairplots

- Klassifikation mit allen Messwerten zu einem geometrischen Feature pro Pocket
- Darstellung für CIR 1



- Teilweise gute Trennung zwischen n.i.O. und i.O.
- Keine vollständige Trennung möglich, da Messwerte ein kontinuierliches Spektrum abbilden

MACHINE LEARNING MODELLE

KNeighborsRegressor

- Das einem Abfragepunkt (query point) zugewiesene Label wird auf der Grundlage des **Mittelwerts** der Labels seiner nächsten **k-Nachbarn** berechnet.

DecisionTreeRegressor

- Wert einer Zielvariablen wird vorhergesagt, indem einfache **Entscheidungsregeln** aus den Datenmerkmalen abgeleitet werden.

RandomForestRegressor

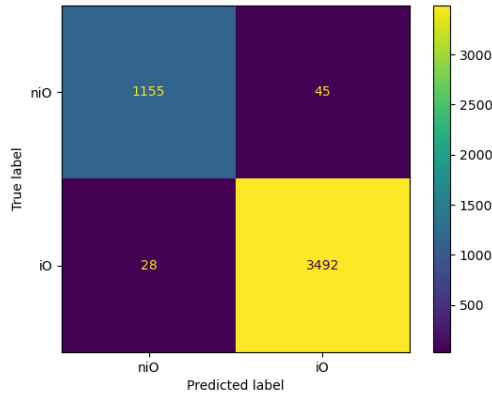
- Reihe von klassifizierenden **Entscheidungsbäumen** (Decision Trees) für verschiedene Teilstichproben des Datensatzes

- **Alle Modelle mit Rastersuche zum Finden der optimalen Modellparameter implementiert**
- **Für jedes geometrische Feature wird eine eigenes Modell trainiert**

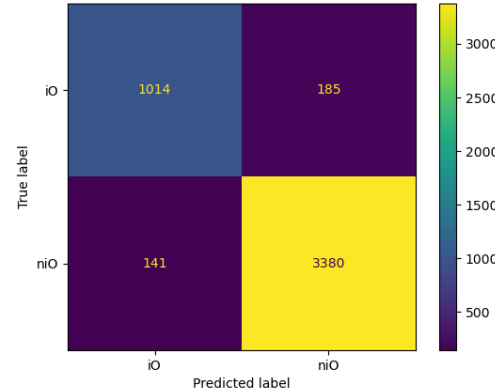
VERGLEICH DER MODELLE

Ergebnisse Modelle - Konfusionsmatrizen

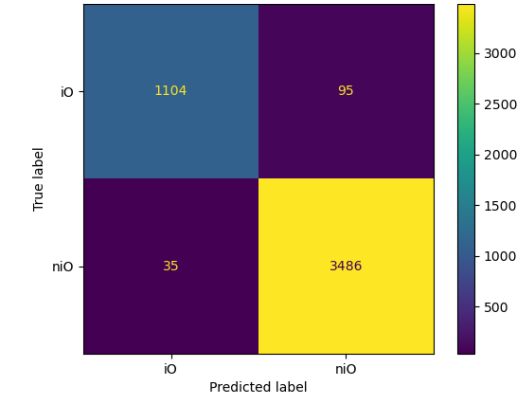
Konfusionsmatrix für alle Klassifikationen
kNeighbor



Konfusionsmatrix mit allen Klassifikationen (DecisionTree)



Konfusionsmatrix mit allen Klassifikationen (RandomForest)



- kNeighborsRegressor hat die beste Performance
 - Genaueste Vorhersage mit wenigen Fehlklassifizierungen
- kNeighborsRegressor wird für weitere Analysen verwendet

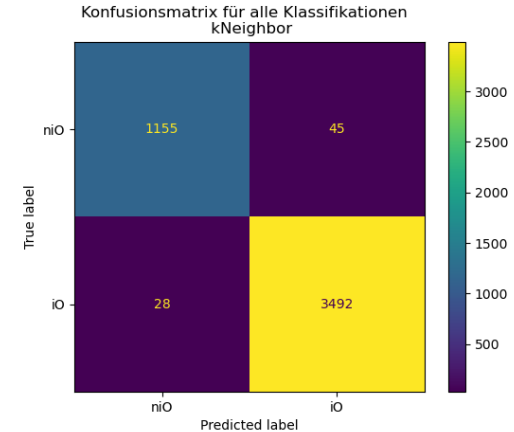
kNeighborsRegressor

ERGEBNISSE

Ergebnisse kNeighborsRegressor

- Bewertungsmetriken für alle geometrischen Feature ähnlich
- Höchste Genauigkeit für *LIN* 1 nach R^2
 - Geringe Streuung der Punkte im Streudiagramm zu erwarten
- Niedrigste Genauigkeit für *LIN* 2

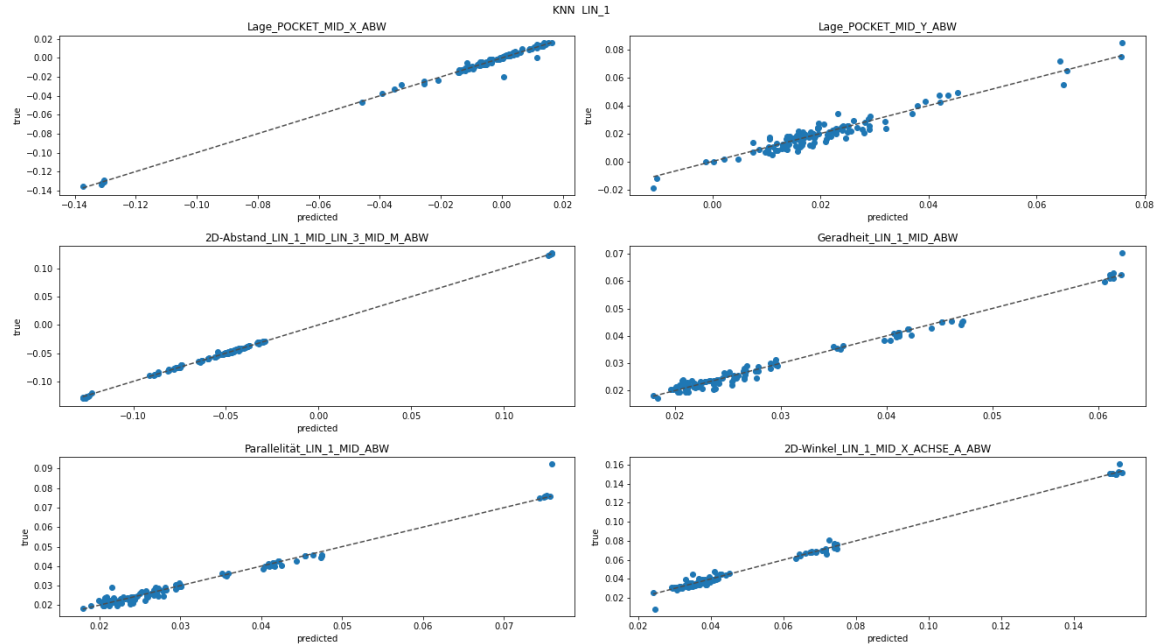
geometrisches Feature	KNeighborRegressor		
	Mean Squared Error	Mean Absolute Error	R^2
LIN_1	6.481e-06	0.001526	0.9782
CIR_1	1.013e-05	0.001942	0.9271
LIN_2	7.881e-06	0.001433	0.8357
CIR_2	1.478e-05	0.002170	0.9013
LIN_3	5.659e-06	0.001332	0.8609
CIR_3	8.605e-06	0.001873	0.9642
LIN_4	8.412e-06	0.001504	0.8734
CIR_4	1.028e-05	0.001848	0.9131



Alle Werte werden mit guter Genauigkeit vorhergesagt

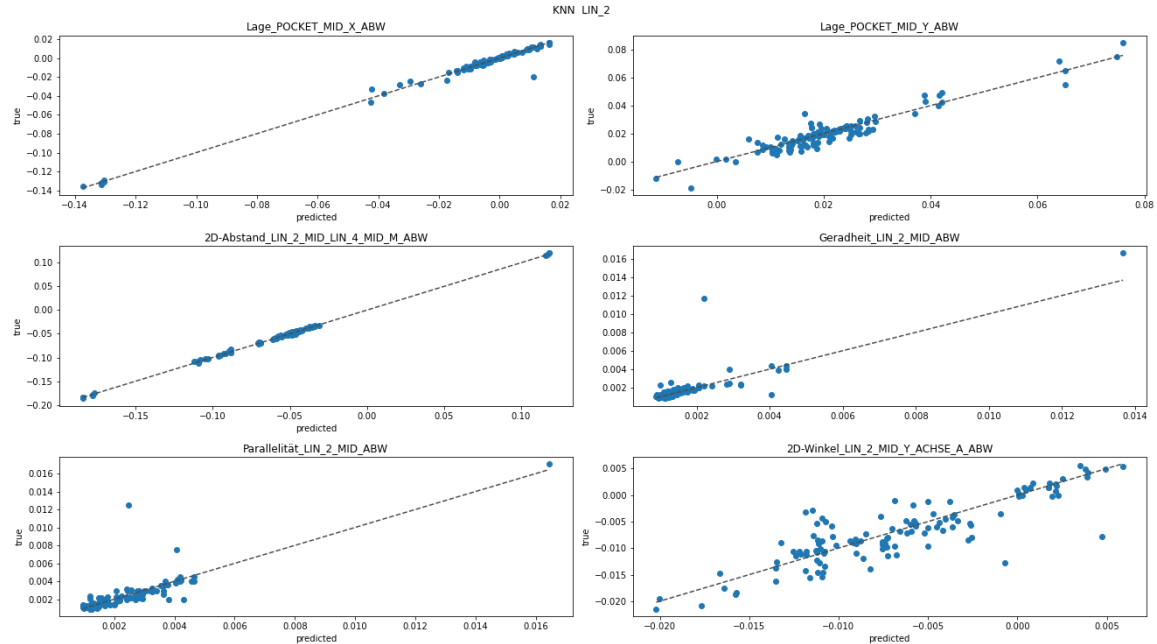
LIN 1 – Streudiagramm

- Wirklicher Wert über dem vorhergesagten Wert aufgetragen
- Vorhergesagte Werte sollten auf der eingezeichneten Geraden liegen
- Keine starken Ausreißer



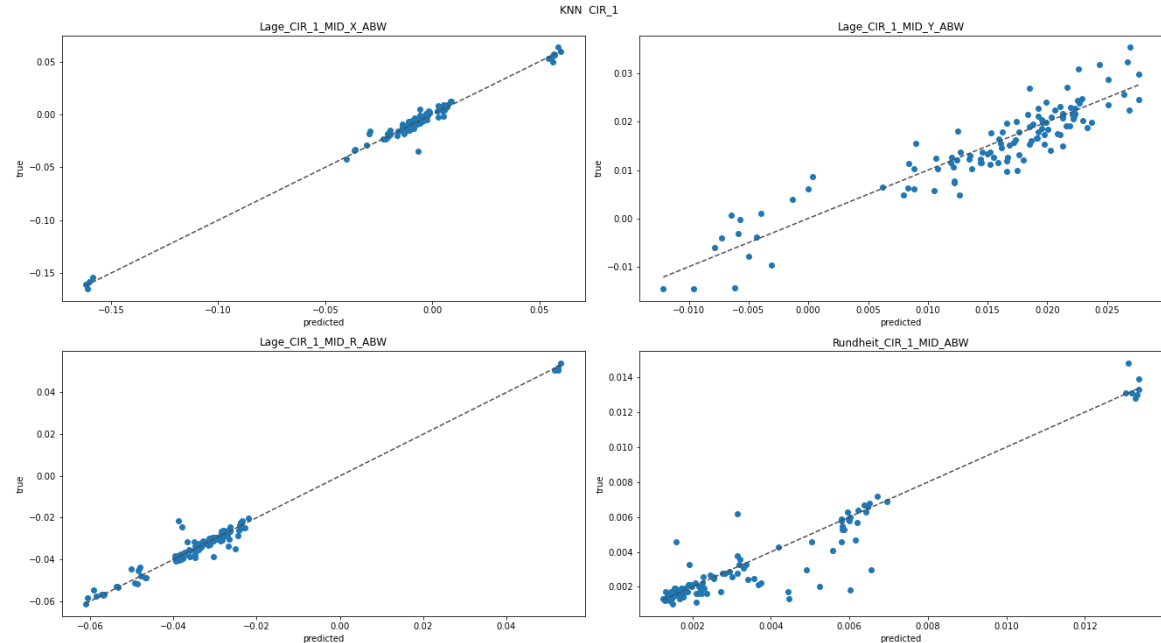
LIN 2 – Streudiagramm

- Streuung im Vergleich zu *LIN 1* geringfügig höher
- Einzelne Ausreißer bzw. falsch vorhergesagte Werte bei den Kennwerten *Parallelität* und *Geradheit*
- Möglicher Grund für die etwas geringere Genauigkeit im Vergleich zu *LIN 1*



CIR 1 – Streudiagramm

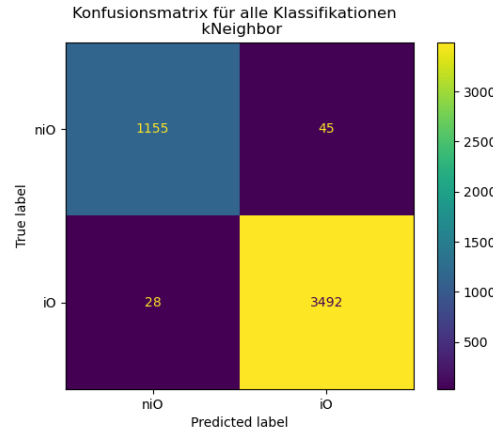
- Geringe Streuung bei der Vorhersage der Lage
 - Besonders große Abweichungen vom Nennmaß werden genau Vorhergesagt
 - Obere Toleranzgrenze: 0,04 mm
 - Untere Toleranzgrenze: -0,04 mm
- Streuung relativ zu den Toleranzgrenzen für den Kennwert *Rundheit* etwas höher
 - Obere Toleranzgrenze: 0,01 mm
 - Untere Toleranzgrenze: 0 mm



Over- / Underfitting

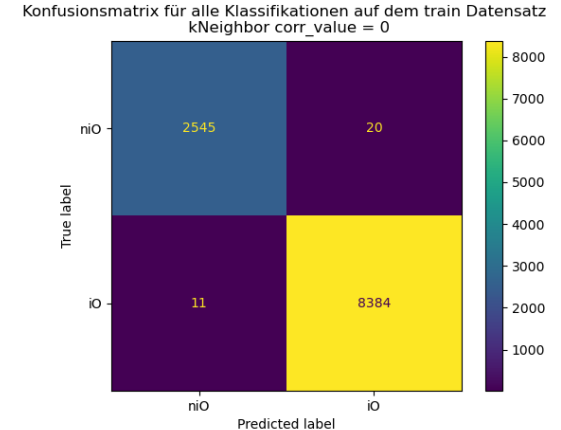
- Nur eine geringe Genauigkeitsabweichung zwischen der Vorhersage der Testdaten und der Trainingsdaten
- Alle Bewertungsmetriken und Streudiagramme wurden mit einem getrennten Testdatensatz erstellt und zeigen eine hohe Performance der Modelle

Kein signifikantes Over- oder Underfitting des Modell erkennbar



Testdatensatz:

- 98,43 % der Werte wurden korrekt klassifiziert



Trainingsdatensatz:

- 99,72 % der Werte wurden korrekt klassifiziert

FEATURE SELECTION

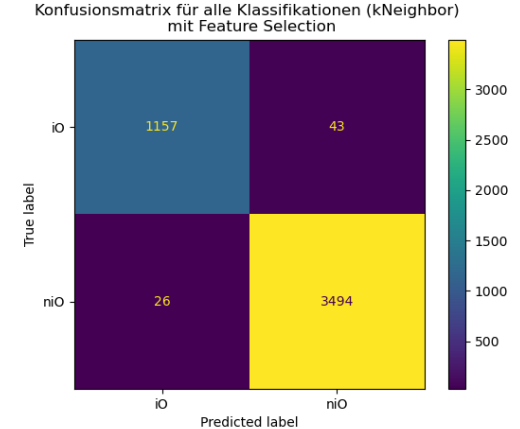
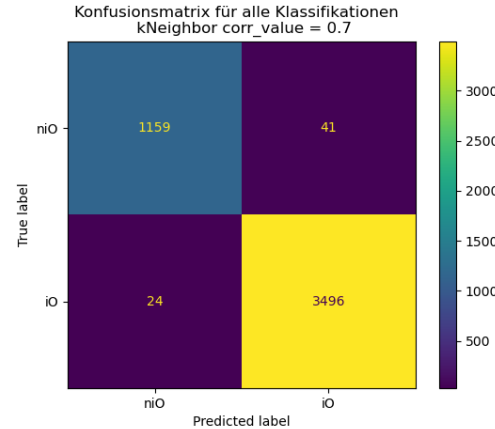
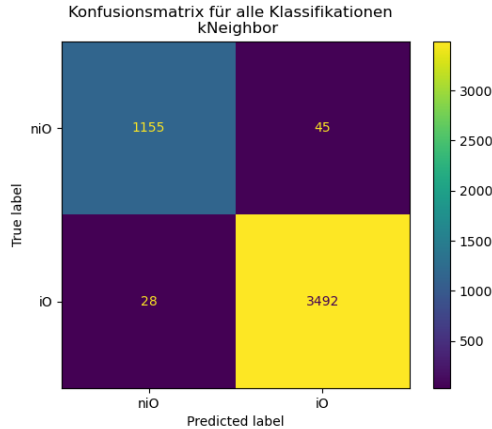
Korrelationsanalyse

- Korrelationsanalyse in der Datenvorbereitung **vor dem Training**
- Corr_value als Grenzwert
- Je nach Corr_value werden die zu löschenden Signale als Liste gespeichert
- Zugriff zu späterem Zeitpunkt möglich

Feature Importance

- **Nach dem Training** werden die wichtigste statistischen Feature für das Modell ermittelt
- Permutationsalgorithmus
- Score gesamtes Modell und für Modell ohne das zu bewertende Feature
- Wichtigkeit über Differenz der beiden Ergebnisse
- Anzahl der verbleibenden statistischen Feature gleich wie bei Korrelationsanalyse

Feature Selection – Vergleich



Für *LIN_1*:

- 180 statistische Feature
- 45 unterschiedliche Maschinensignale

Für *LIN_1*:

- 68 statistische Feature
- 17 unterschiedliche Maschinensignale

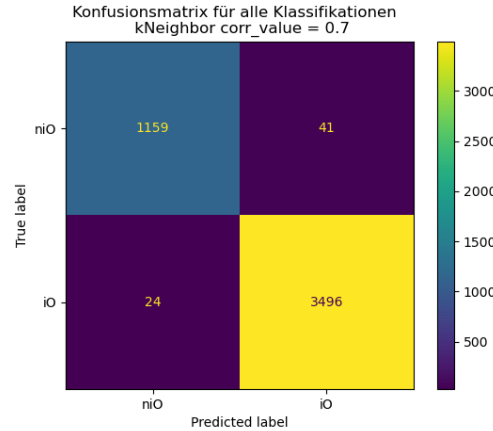
Für *LIN_1*:

- 68 statistische Feature
- 30 unterschiedliche Maschinensignale

Over- / Underfitting

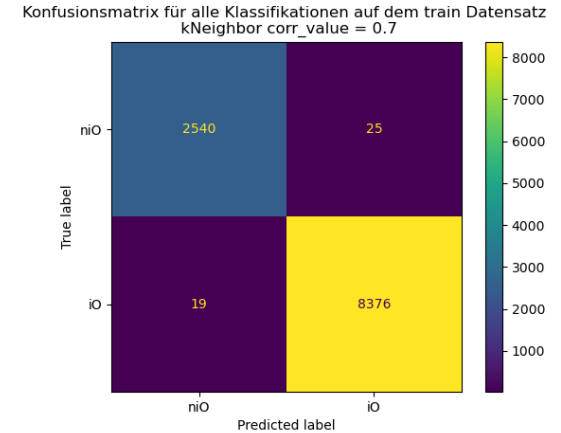
- Reduzieren der Signale führt zu höherer Robustheit
- Vorhersage der Trainingsdaten wird etwas schlechter
- Vorhersage der Testdaten wird besser

Unterschied zwischen Vorhersage auf Test- und Trainingsdaten durch Feature Selection verringert



Testdatensatz:

- 98,6 % der Werte wurden korrekt klassifiziert



Trainingsdatensatz:

- 99,6 % der Werte wurden korrekt klassifiziert

LIMITATIONEN UND ZUSAMMENFASSUNG

- Kritisch ist besonders der Fall, das ein n.i.O. Bauteil versehentlich als i.O. klassifiziert wird
 - Durch Feature Selection und eine möglichst gute Modellauswahl ist dieser Fall selten (0,51 % aller Fälle)
 - Vorhergesagte Abweichung zum Nennmaß und der Toleranz gibt weiterhin Aufschluss über die Zuverlässigkeit einer Klassifizierung
 - Geringe Abweichung zur Toleranz → möglicherweise falsche Klassifizierung → Nachuntersuchung sinnvoll

Modell mit genauer Vorhersage

- Klassifikation zu 98,6 % korrekt
- Verbesserung durch Feature Selection und Korrelationsanalyse
- Kein signifikantes Over- oder Underfitting
- Beste Performance durch kNeighborRegressor

Umrechnung der Regression in eine Klassifikation der Bauteilqualität möglich

- i.O. und n.i.O.
- Unter Verwendung der Toleranzwerte
- Praxisanwendung: Ausgabe einer Klassifikation für jeden Messwert und zusätzlich die vorhergesagte Abweichung vom Nennmaß

- Weitere Methoden der Feature Selection untersuchen
 - Principle Component Analysis / Dimension Reduction
- Nutzen von neuronalen Netzen für die Regressionsaufgabe
- Generalisieren der Modelle
 - Ein Modell für alle geometrischen Elemente *LIN*
 - Ein Modell für alle geometrischen Elemente *CIR*
- Implementieren der Modelle als Inline-Prozessüberwachung

<https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.StandardScaler.html>
<https://scikitlearn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.html>
<https://scikitlearn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html>
<https://scikitlearn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html>
https://scikit-learn.org/stable/modules/permutation_importance.html
<https://scikit-learn.org/stable/modules/multiclass.html>
<https://machinelearningmastery.com/multi-output-regression-models-with-python/>
<https://machinelearningmastery.com/feature-selection-machine-learning-python/>

**Vielen Dank für die
Aufmerksamkeit!**

ANHANG

Begriffserklärung

Geometrische Feature: LIN 1, CIR 1,..., LIN 4, CIR 4

Maschinensignale: Aufgenommene Daten aus dem Regelkreis während des Fräsvorgangs; insgesamt 45 Signale

Statistische Feature: Mittelwert, Standardabweichung, Maximum und Minimum der Maschinensignale

- Vor dem Import der Daten in das jeweilige Modell, werden die Maschinensignale standardisiert
 - Mittelwert = 0
 - Standardabweichung = 1
- Die unterschiedlichen Signale haben verschiedene Wertebereiche. Um einen gleichmäßigen Einfluss aller Signale zu gewährleisten und ein Bias des Modells zu verhindern, werden die Daten entsprechend einer Normalverteilung vorbereitet.
- *StandardScaler* von *scikit learn* verwendet

- Abweichung zum Nennmaß über angegebene Toleranzen in Klassifikation umgewandelt
 - Klassifizierung für jede Pocket und jedes geometrische Feature
 - Wenn ein Messwert innerhalb der Pocket n.i.O., dann ist die gesamte Pocket n.i.O.
- Für jedes Maschinensignal wird ein Pairplot pro geometrischem Feature jeder Pocket erstellt
 - Vier Spalten pro Signal: Mittelwert, Standardabweichung, Maximum, Minimum
- Je besser die Trennung der Punktwolken für die Klassifikation n.i.O. und i.O., desto einfacher ist das Vorhersagen der Abweichung mittels ML Methoden

- Mean absolute error (MAE)

- Dabei wird die Genauigkeit bzw. mittlere Abweichung der Vorhersage berechnet.

- $MAE = \frac{1}{n} \sum_{i=1}^n |Y_i - \hat{Y}|$

n = Anzahl der Vorhersagewerte
 Y_i = wahrer Wert
 \hat{Y} = vorhergesagter Wert
 \bar{Y} = mittlerer Wert

- Mean squared error (MSE)

- Dieser Wert zeigt wie sehr eine Vorhersage um den angegebenen Wert streut.

- $MSE = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y})^2$

- Coefficient of determination (R^2)

- Dieser Wert zeigt wie viel Streuung in den Daten durch ein vorliegendes Regressionsmodell erklärt werden kann. Der Wertebereich liegt zwischen 0 und 1.

- $R^2 = 1 - \frac{\sum_{i=1}^n (Y_i - \hat{Y})^2}{\sum_{i=1}^n (Y_i - \bar{Y})^2}$

ML Modelle – Implementierung

- Alle Modelle mit Rastersuche zum Finden der optimalen Modellparameter implementiert
- Für jedes geometrische Feature wird ein eigenes Modell trainiert

KNeighborsRegressor

Beste Parameter *LIN 1*

'algorithm': ['auto', "ball_tree", "kd_tree", "brute"]

'leaf_size': [15, 20, 25, 30, 35, 40]

'n_neighbors': [1, 2, 3, 4]

'p': [1, 2]

'weights': ['uniform', "distance"]

DecisionTreeRegressor

"splitter": ["best", "random"]

"max_depth": [1, 3, 5, 7, 9, 11, 12]

"min_samples_leaf": [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

"min_weight_fraction_leaf": [0.1, 0.2, 0.3, 0.4, 0.5]

"max_features": ["auto", "log2", "sqrt", None]

"max_leaf_nodes": [None, 10, 20, 30, 40, 50, 60, 70, 80, 90]

RandomForestRegressor

'n_estimators': [500, 600]

'max_depth': [15, 100]

'min_samples_split': [5, 100]

'min_samples_leaf': [5, 100]

Ergebnisse Modelle - Vergleich

- Der **KNeighborsRegressor** ist das beste ML-Modell

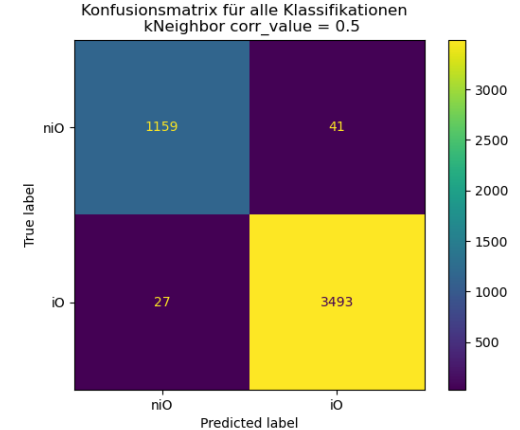
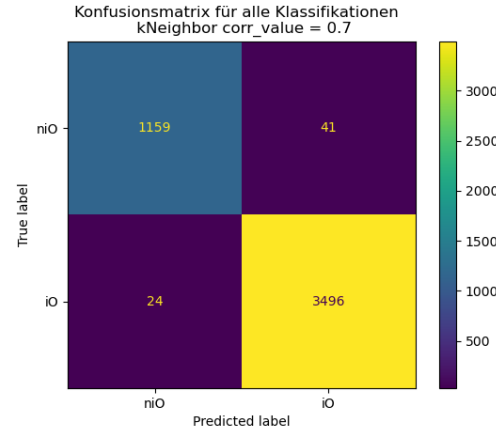
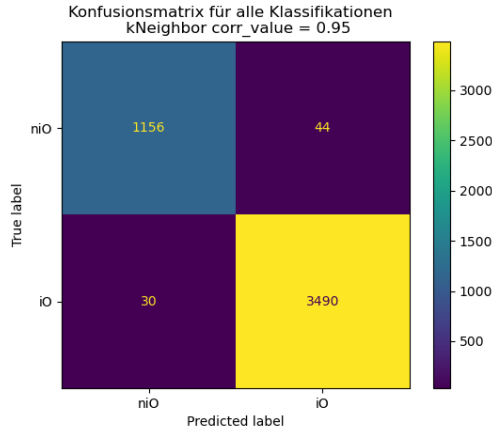
geometrisches Feature	KNeighborRegressor			RandomForestRegressor			DecisionTreeRegressor		
	Mean Squared Error	Mean Absolute Error	R ²	Mean Squared Error	Mean Absolute Error	R ²	Mean Squared Error	Mean Absolute Error	R ²
LIN_1	6.481e-06	0.001526	0.9782	5e-05	0.00290	0.90038	0.00036	0.00934	0.48502
CIR_1	1.013e-05	0.001942	0.9271	9e-05	0.00286	0.85866	0.00028	0.00797	0.43253
LIN_2	7.881e-06	0.001433	0.8357	2e-05	0.00239	0.64785	0.00034	0.00751	0.24623
CIR_2	1.478e-05	0.002170	0.9013	6e-05	0.00311	0.83309	0.00021	0.00731	0.51795
LIN_3	5.659e-06	0.001332	0.8609	2e-05	0.00206	0.78306	0.00025	0.00688	0.25818
CIR_3	8.605e-06	0.001873	0.9642	2e-05	0.00248	0.94469	0.00026	0.00787	0.40336
LIN_4	8.412e-06	0.001504	0.8734	2e-05	0.00215	0.75986	0.00029	0.00750	0.29524
CIR_4	1.028e-05	0.001848	0.9131	3e-05	0.00277	0.87398	0.00028	0.00774	0.40155

Feature Selection – Korrelationsanalyse

- Trainingsdaten variieren, je nach Korrelationsanalyse:
 - Unterschiedliche Grenzwerte *Corr_value* ab denen ein Maschinensignal nicht mehr berücksichtigt wird

geometrisches Feature	KNeighborRegressor			KNeighborRegressor Corr_value = 0,95			KNeighborRegressor Corr_value = 0,7			KNeighborRegressor Corr_value = 0,5		
	Mean Squared Error	Mean Absolute Error	R ²	Mean Squared Error	Mean Absolute Error	R ²	Mean Squared Error	Mean Absolute Error	R ²	Mean Squared Error	Mean Absolute Error	R ²
LIN_1	6.481e-06	0.001526	0.9782	6.427e-06	0.001566	0.9786	6.161e-06	0.001526	0.9790	6.799e-06	0.001555	0.9780
CIR_1	1.013e-05	0.001942	0.9271	8.907e-06	0.001829	0.9325	9.548e-06	0.001887	0.9281	1.157e-05	0.002058	0.9191
LIN_2	7.881e-06	0.001433	0.8357	7.182e-06	0.001339	0.84666	7.351e-06	0.001311	0.8361	7.351e-06	0.001312	0.8361
CIR_2	1.478e-05	0.002170	0.9013	1.342e-05	0.002135	0.9046	1.490e-05	0.002116	0.8990	1.729e-05	0.002297	0.8894
LIN_3	5.659e-06	0.001332	0.8609	7.799e-06	0.001397	0.8271	6.012e-06	0.001323	0.8076	6.591e-06	0.001389	0.8602
CIR_3	8.605e-06	0.001873	0.9642	8.209e-06	0.001809	0.9674	8.378e-06	0.001837	0.9659	1.1363e-05	0.001999	0.95812
LIN_4	8.412e-06	0.001504	0.8734	8.284e-06	0.001461	0.8685	8.360e-06	0.001479	0.8753	7.901e-06	0.001413	0.8756
CIR_4	1.028e-05	0.001848	0.9131	1.136e-05	0.001968	0.9041	1.021e-05	0.001866	0.9033	1.451e-05	0.002278	0.8801

Feature Selection – Korrelationsanalyse



Für *LIN_1*, corr_value=0,95:

- 18 Maschinensignale
- 72 statistische Feature

Für *LIN_1*, corr_value=0,7:

- 17 Maschinensignale
- 68 statistische Feature

Für *LIN_1*, corr_value=0,5:

- 13 Maschinensignale
- 52 statistische Feature

Im Vergleich beste Vorhersage

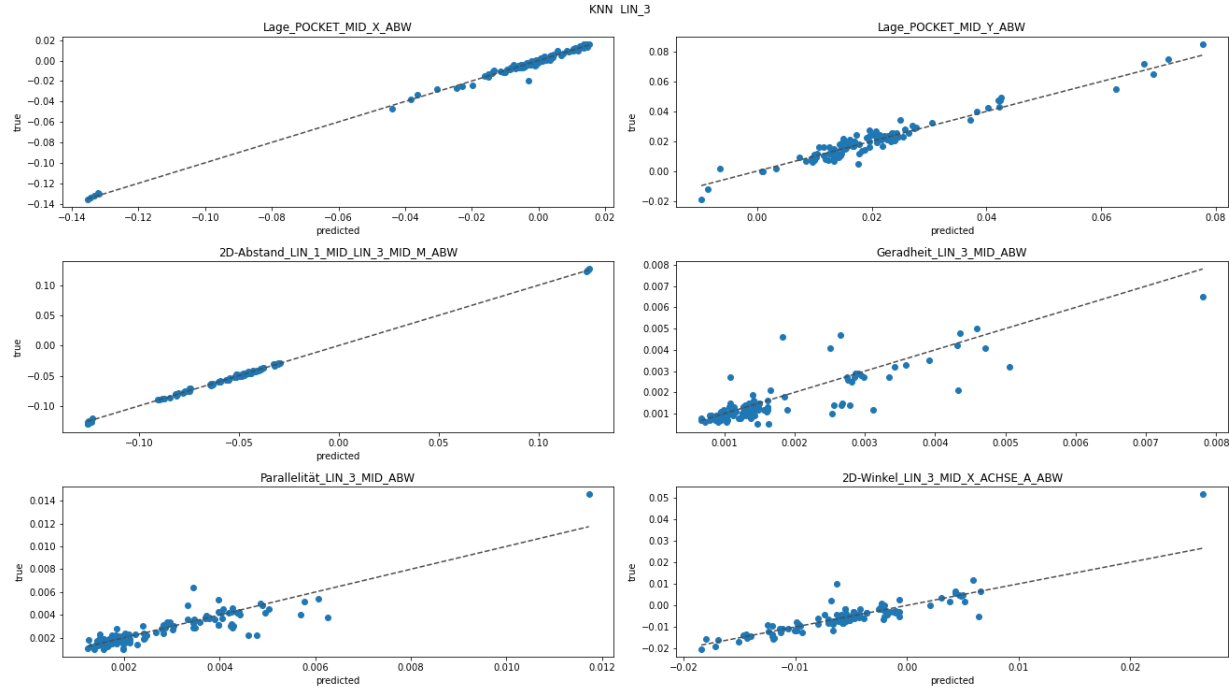
Feature Selection – ausgewählte Signale *LIN 1*

Korrelationsanalyse $\text{corr_val} = 0,7$

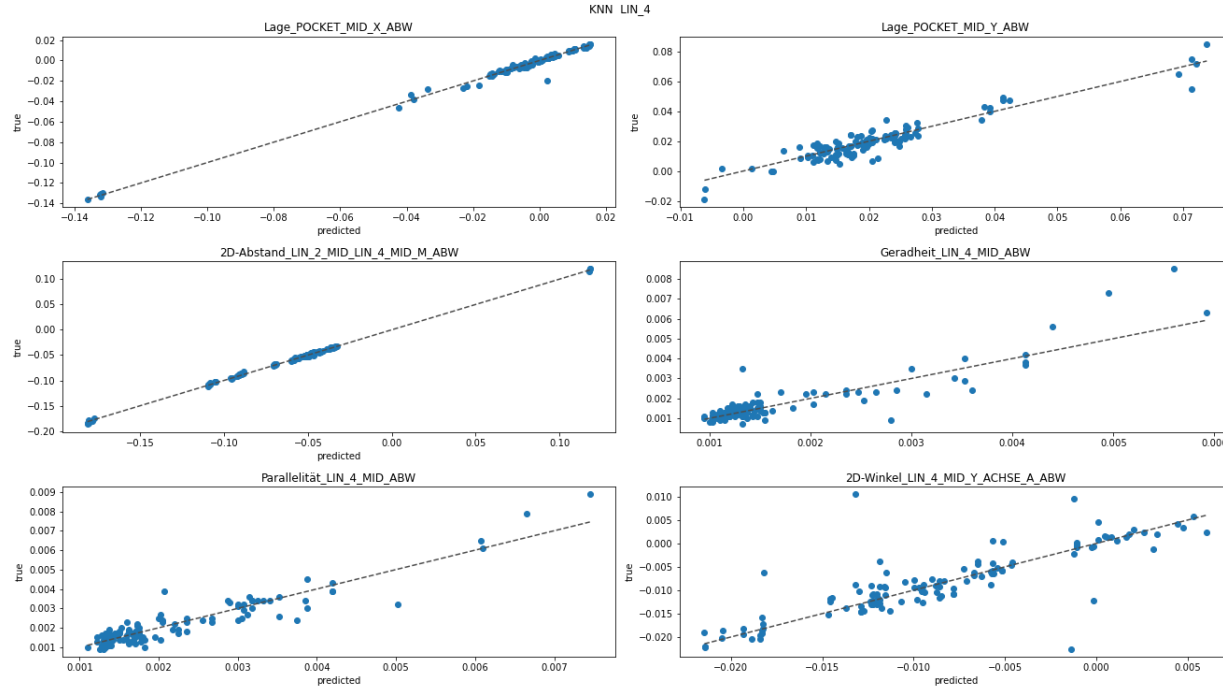
Feature Importance

CMD_SPEED 1	CONT_DEV 1	CURRENT 2	POWER 1	CTRL_POS 1	'CURRENT 1'	'CMD_SPEED 2'	'TORQUE 1'	'POWER 1'	'CTRL_DIFF 1'
CMD_SPEED 3	CONT_DEV 2	CURRENT 3	POWER 2	CTRL_POS 2	'CURRENT 2'	'CMD_SPEED 3'	'TORQUE 2'	'POWER 2'	'CTRL_DIFF 2'
CMD_SPEED 6	CURRENT 1	CURRENT 6	POWER 6	CTRL_POS 3	'CURRENT 3'	'CMD_SPEED 6'	'TORQUE 3'	'POWER 6'	'CTRL_DIFF 3'
ENC_POS 6	CYCLE				'CURRENT 6'	'CONT_DEV 1'	'TORQUE 6'	'ENC1_POS 2'	'CTRL_DIFF2 2'
					'DPIO_IN2 0'	'CONT_DEV 2'	'DPIO_IN2 1016'	'ENC1_POS 3'	'CTRL_DIFF2 3'
					'DPIO_IN2 1018'	'CONT_DEV 3'	'ENC_POS 6'	'ENC2_POS 3'	'CTRL_POS 3'

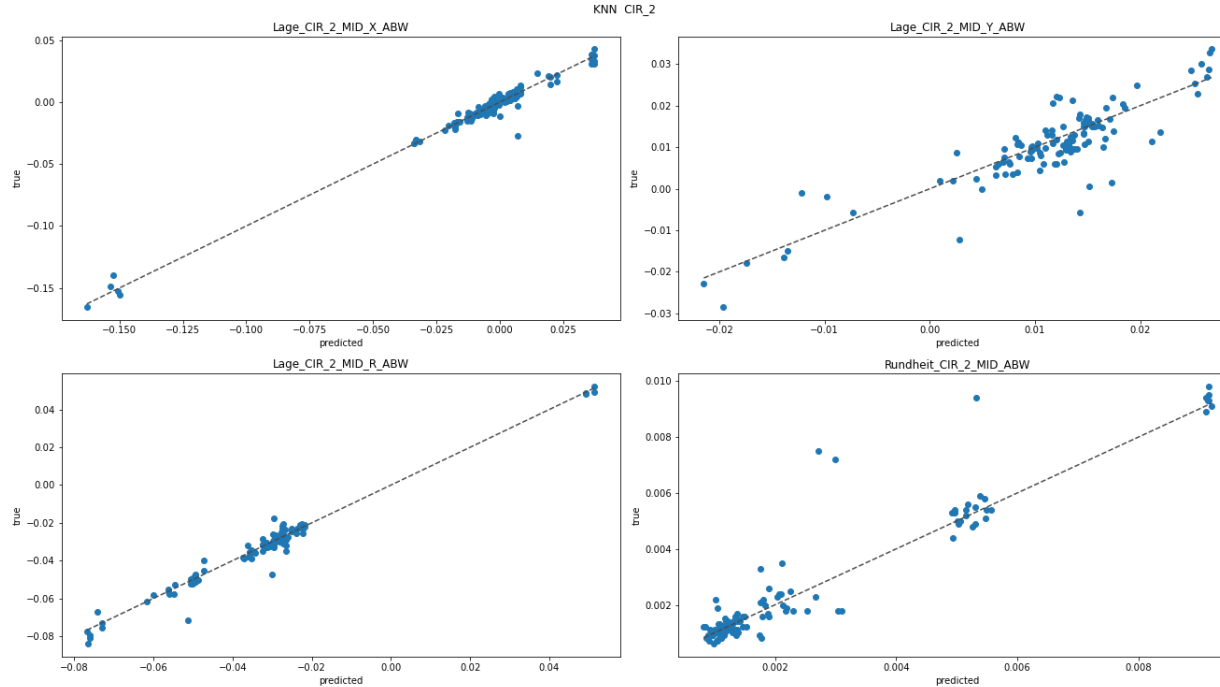
LIN 3 - Streudiagramm



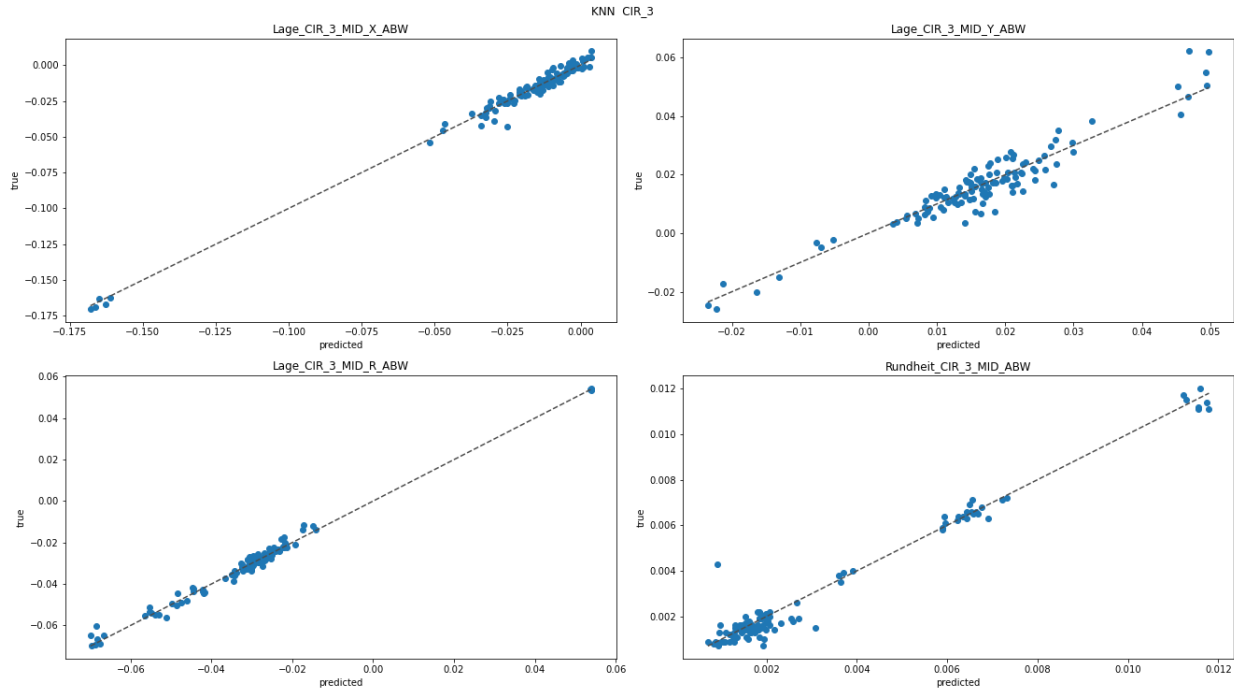
LIN 4 - Streudiagramm



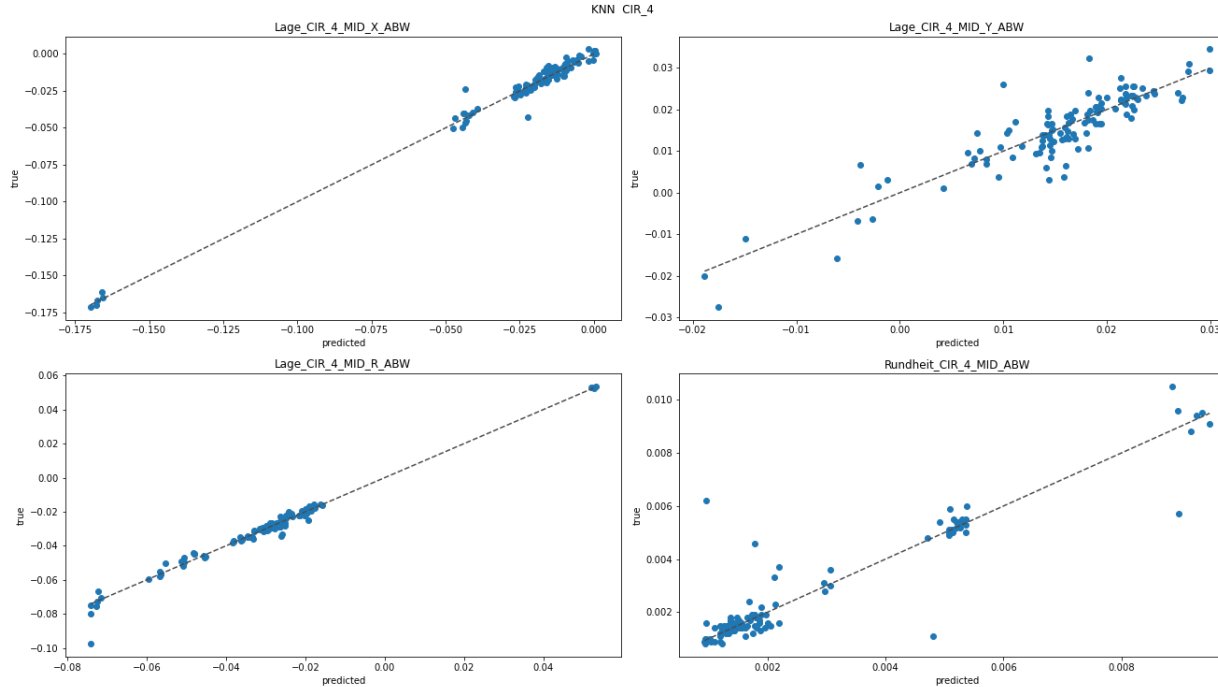
CIR 2 - Streudiagramm



CIR 3 - Streudiagramm



CIR 4 - Streudiagramm



- Programmstruktur aufgeteilt in eine erste Datenvorverarbeitung sowie explorative Datenanalyse, eine Datenvorbereitung für die ML Modelle und das Trainieren und Auswählen der Modelle
 - Dargestellt auf Folie 5
- Python Code orientiert sich an Programmstruktur:
 - Skript 01 für Datenvorverarbeitung
 - Skript 02 für Datenanalyse
 - Skript 03 für Auswahl der Modelle
 - Skript 04 für Konfusionsmatrix und Bewertung des Modells, sowie Feature Selection

1. Alle CMM und Fräsmaschinendaten werden geladen und jeweils in einem *Pandas Dataframe* gespeichert
2. In beiden Datensätzen wird die Werkstücknummer *WPNR*, sowie *xPos* und *yPos* aus */Channel/ProgramInfo/msg/u1* ausgelesen und in neue Spalten geschrieben
3. Daten werden nach geometrischen Features sortiert und in Python *Dictionaries* geschrieben
4. Korrelationsanalyse der Maschinendaten
 - Für verschiedene Korrelationswerte werden die überflüssigen Signale in Listen gespeichert → Bei der Feature Selection im späteren Trainingsprozess kann auf die Listen zugegriffen werden und die Daten für entsprechende Korrelationswerte reduziert werden

5. Maschinendaten werden für jedes geometrische Feature einzeln nach Pockets gruppiert und statistische Kennwerte berechnet
 - Mittelwert, Standardabweichung, Maximum, Minimum

```
▼ all_data_gF = {dict: 8} {'LIN_1': WPNR xPos ... Parallelität_LIN_1_M
> 'LIN_1' = {DataFrame: (392, 191)} WPNR xPos ... Parallelität_LIN
> 'CIR_1' = {DataFrame: (392, 189)} WPNR xPos ... Lage_CIR_1_MI
> 'LIN_2' = {DataFrame: (392, 191)} WPNR xPos ... Parallelität_LIN
> 'CIR_2' = {DataFrame: (392, 189)} WPNR xPos ... Lage_CIR_2_MI
> 'LIN_3' = {DataFrame: (392, 191)} WPNR xPos ... Parallelität_LIN
> 'CIR_3' = {DataFrame: (392, 189)} WPNR xPos ... Lage_CIR_3_MI
> 'LIN_4' = {DataFrame: (392, 191)} WPNR xPos ... Parallelität_LIN
> 'CIR_4' = {DataFrame: (392, 189)} WPNR xPos ... Lage_CIR_4_MI
01 _len_ = {int} 8
```


6. CMM-Daten und Maschinendaten werden zusammengeführt
 - Zuordnung eines Messwertes pro Zeile bzw. Pocket
7. Korrelationsanalyse und exemplarische Klassifizierung der Abweichung in i.O. und n.i.O.

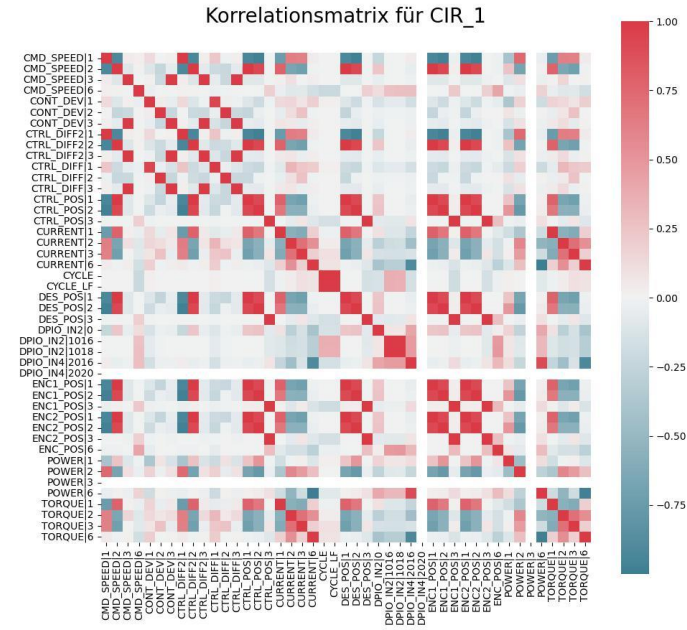
↕ WPNR	↕ xPos	↕ yPos	↕ geomFeature	↕ /Channel/ProgramInfo/msg u1	↕ CMD_SPEED 1_mean	↕ CMD_SPEED 1_std	↕ CMD_SPEED 1_min	↕ CMD_SPEED 1_max
1	0	0	LIN_1	WPNR_1_xPos_0_yPos_0_EC_24	-17.12691	3.25977	-18.13557	0.51279
1	0	1	LIN_1	WPNR_1_xPos_0_yPos_1_EC_24	-17.13558	3.26529	-18.16204	0.48979
1	0	2	LIN_1	WPNR_1_xPos_0_yPos_2_EC_24	-17.13042	3.27389	-18.16571	0.49329
1	0	3	LIN_1	WPNR_1_xPos_0_yPos_3_EC_21	-17.13797	3.28507	-18.20576	0.77680
1	0	4	LIN_1	WPNR_1_xPos_0_yPos_4_EC_21	-17.13433	3.29185	-18.21576	0.77930
1	0	5	LIN_1	WPNR_1_xPos_0_yPos_5_EC_24	-17.13498	3.26746	-18.18443	0.53079

↕ Lage_POCKET_MID_X_ABW	↕ Lage_POCKET_MID_Y_ABW	↕ 2D-Abstand_LIN_1_MID_LIN_3_MID_M_...	↕ Geradheit_LIN_1_MID_ABW	↕ Parallelität_LIN_1_MID_ABW	↕ 2D-Winkel_LIN_1_MID_X_ACHSE_A_ABW
-0.00410	0.01700	-0.07080	0.03820	0.04010	0.08100
0.00060	0.01690	-0.07410	0.03960	0.03990	0.06740
0.00340	0.01670	-0.07570	0.04140	0.04160	0.06650
0.01320	0.03440	-0.12090	0.05990	0.07490	0.15050
0.01550	0.03620	-0.12410	0.06130	0.07440	0.15220
0.01030	0.01960	-0.08160	0.04270	0.04280	0.07170

- 45 Maschinensignale
- 9 Werkstücke mit jeweils 49 Pockets
- 8 geometrische Feature (LIN, CIR)
- Messeigenschaften: 4 für CIR und 6 für LIN
- Wichtigste Features aus CMM: ABW und Toleranzwerte

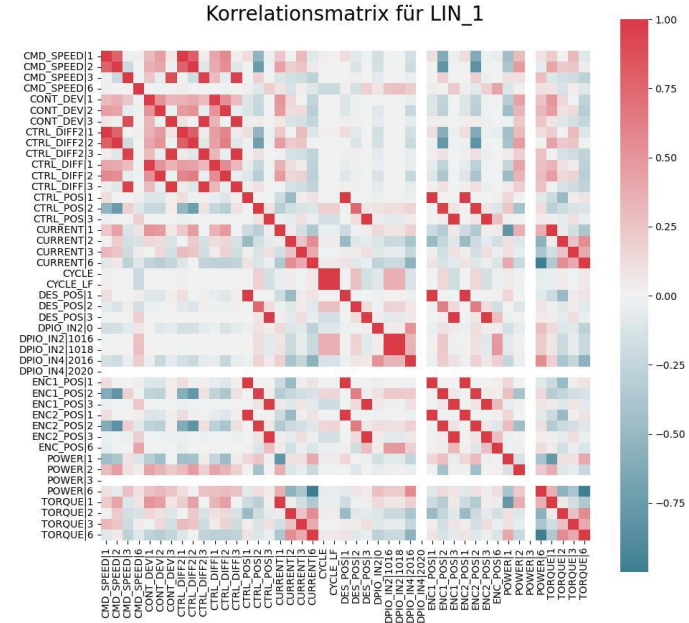
Explorative Datenanalyse – Korrelation (1/2)

- Untersuchung und Vergleich der aufgenommenen Maschinensignale
 - Korrelationsanalyse zur Auswahl von relevanten Maschinensignalen
- Analyse für jedes geometrische Feature getrennt
- Hohe Korrelation z.B. zwischen Motorenströmen und Drehmoment oder zwischen Regelabweichung und vorgegebener Achsgeschwindigkeit
- Einige Signale sind konstant und werden daher nicht berücksichtigt



Explorative Datenanalyse – Korrelation (2/2)

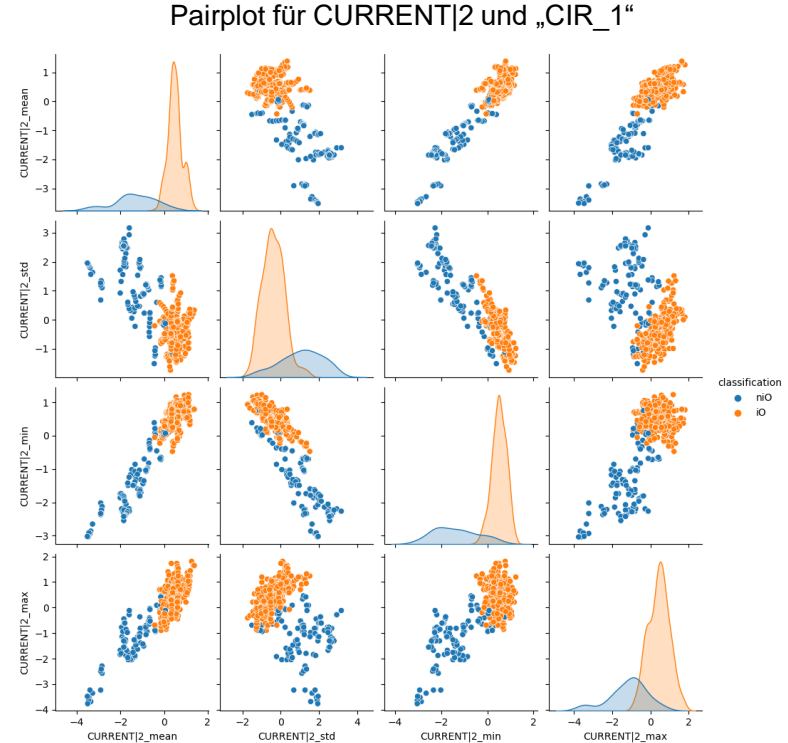
- Je nach geometrischem Feature korrelieren unterschiedliche Signale
- Weniger korrelierende Signale bei geraden Kanten *LIN*



- Abweichung zum Nennmaß über angegebene Toleranzen in Klassifikation umgewandelt
 - Klassifizierung für jede Pocket und jedes geometrische Feature
 - Wenn ein Messwert innerhalb der Pocket n.i.O., dann ist die gesamte Pocket n.i.O.
- Für jedes Maschinensignal wird ein Pairplot pro geometrischem Feature jeder Pocket erstellt
 - Vier Spalten pro Signal: Mittelwert, Standardabweichung, Maximum, Minimum
- Je besser die Trennung der Punktwolken für die Klassifikation n.i.O. und i.O., desto einfacher ist das Vorhersagen der Abweichung mittels ML Methoden

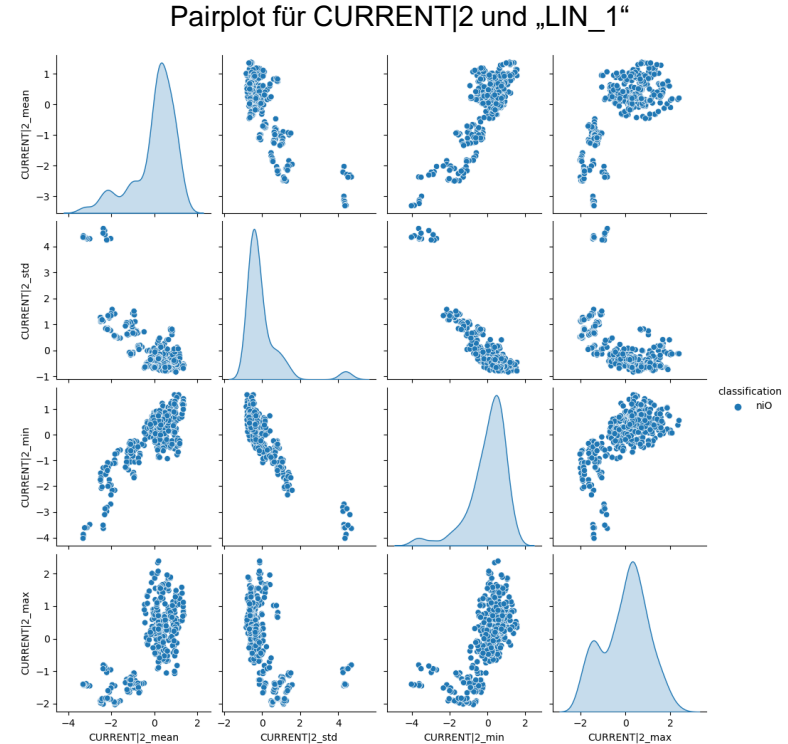
Explorative Datenanalyse – Pairplots CIR

- Teilweise gute Trennung zwischen n.i.O. und i.O. Messwerten für die runden Kanten CIR
 - Keine vollständige Trennung möglich
 - Messwerte für i.O. und n.i.O. nah an der Toleranzgrenze sehr ähnlich
- Gute Trennung bei großen Abweichungen zum Toleranzwert



Explorative Datenanalyse – Pairplots *LIN*

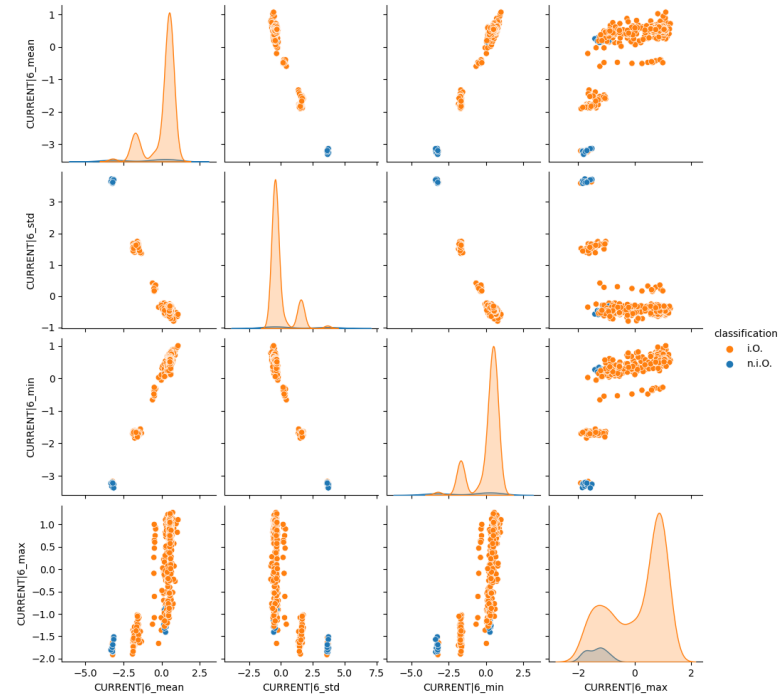
- Keine Trennung zwischen n.i.O. und i.O. bei geometrischem Element *LIN* möglich, da alle Pockets n.i.O.
- Gilt für alle geraden Kanten *LIN*
- Messwert für *2D-Abstand* ist stets außerhalb der Toleranz



Explorative Datenanalyse – Pairplots *LIN*

- Für einzelne Messwerte ist auch bei geraden Kanten „LIN“ eine teilweise Trennung der Punkte für i.O. und n.i.O. möglich
- Beispiel: Pairplot von Maschinensignal *CURRENT|6* zum Messwert *Lage_POCKET_MID_Y* des geometrischen Features *LIN_1*

Pairplot für *CURRENT|6* und *LIN_1 Lage_POCKET_MID_Y*



BEWERTUNGSMETRIKEN

- Mean absolute error (MAE)

- Dabei wird die Genauigkeit bzw. mittlere Abweichung der Vorhersage berechnet.

- $MAE = \frac{1}{n} \sum_{i=1}^n |Y_i - \hat{Y}|$

n = Anzahl der Vorhersagewerte
 Y_i = wahrer Wert
 \hat{Y} = vorhergesagter Wert
 \bar{Y} = mittlerer Wert

- Mean squared error (MSE)

- Dieser Wert zeigt wie sehr eine Vorhersage um den angegebenen Wert streut.

- $MSE = \frac{1}{n} \sum_{i=1}^n (Y_i - \hat{Y})^2$

- Coefficient of determination (R^2)

- Dieser Wert zeigt wie viel Streuung in den Daten durch ein vorliegendes Regressionsmodell erklärt werden kann.

- $R^2 = 1 - \frac{\sum_{i=1}^n (Y_i - \hat{Y})^2}{\sum_{i=1}^n (Y_i - \bar{Y})^2}$

ML Modelle - KNeighborsRegressor

- Der KNeighborsRegressor kann verwendet werden, falls es sich bei den Datenlabels um **kontinuierliche** Variablen handelt.
- Das einem Abfragepunkt (query point) zugewiesene Label wird auf der Grundlage des **Mittelwerts** der Labels seiner nächsten **k-Nachbarn** berechnet.
- **Weitere Informationen unter:**
<https://scikitlearn.org/stable/modules/generated/sklearn.neighbors.KNeighborsRegressor.html>

- Entscheidungsbäume (Decision Trees) sind eine nichtparametrische überwachte Lernmethode, die für Klassifizierung und **Regression** verwendet wird.
- Ziel ist es, ein Modell zu erstellen, das den Wert einer Zielvariablen vorhersagt, indem einfache **Entscheidungsregeln** aus den Datenmerkmalen abgeleitet werden. Ein Baum kann als eine stückweise konstante Näherung betrachtet werden.
- **Weitere Informationen unter:**
<https://scikitlearn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html>

ML Modelle - RandomForestRegressor

- Der Random Forest Regressor passt eine Reihe von klassifizierenden **Entscheidungsbäumen** (Decision Trees) auf verschiedene Teilstichproben des Datensatzes an
- Er verwendet die **Mittelwertbildung**, um die Vorhersagegenauigkeit und die Kontrolle der **Überanpassung** (Overfitting) zu verbessern.
- **Weitere Informationen unter:**
<https://scikitlearn.org/stable/modules/generated/sklearn.ensemble.RandomForestRegressor.html>

ML Modelle – Implementierung

- Alle Modelle mit Rastersuche zum Finden der optimalen Modellparameter implementiert
- Für jedes geometrische Feature wird ein eigenes Modell trainiert

KNeighborsRegressor

'algorithm': ['auto', "ball_tree", "kd_tree", "brute"]
'leaf_size': [15, 20, 25, 30, 35, 40]
'n_neighbors': [1, 2, 3, 4]
'p': [1, 2]
'weights': ['uniform', "distance"]

DecisionTreeRegressor

"splitter": ["best", "random"]
"max_depth": [1, 3, 5, 7, 9, 11, 12]
"min_samples_leaf": [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
"min_weight_fraction_leaf": [0.1, 0.2, 0.3, 0.4, 0.5]
"max_features": ["auto", "log2", "sqrt", None]
"max_leaf_nodes": [None, 10, 20, 30, 40, 50, 60, 70, 80, 90]

RandomForestRegressor

'n_estimators': [500, 600]
'max_depth': [15, 100]
'min_samples_split': [5, 100]
'min_samples_leaf': [5, 100]

Ergebnisse kNeighborRegressor

- Höchste Genauigkeit für *LIN 1* nach R^2
 - Geringe Streuung der Punkte im Streudiagramm zu erwarten
- Niedrigste Genauigkeit für *LIN 2*
- Bewertungsmetriken für alle geomterischen Feature ähnlich
 - Alle Werte werden mit guter Genauigkeit vorhergesagt

KNeighborRegressor			
geometrisches Feature	Mean Squared Error	Mean Absolute Error	R^2
LIN_1	6.481e-06	0.001526	0.9782
CIR_1	1.013e-05	0.001942	0.9271
LIN_2	7.881e-06	0.001433	0.8357
CIR_2	1.478e-05	0.002170	0.9013
LIN_3	5.659e-06	0.001332	0.8609
CIR_3	8.605e-06	0.001873	0.9642
LIN_4	8.412e-06	0.001504	0.8734
CIR_4	1.028e-05	0.001848	0.9131

- Modell mit sehr genauer Vorhersage
 - Klassifikation zu 98,6 % korrekt
 - Verbesserung durch Feature Selection und Korrelationsanalyse
 - Kein signifikantes Over- oder Underfitting
- Beste Performance durch kNeighborRegressor
- Umrechnung der Regression in eine Klassifikation der Bauteilqualität möglich
 - i.O. und n.i.O.
 - Unter Verwendung der Toleranzwerte
 - Praxisanwendung: Ausgabe einer Klassifikation für jeden Messwert und zusätzlich die vorhergesagte Abweichung vom Nennmaß

Ergebnisse – Streudiagramm

- Das in den vorherigen Folien gezeigte Verhalten, kann auch bei den übrigen geometrischen Elementen beobachtet werden
 - Streudiagramme für *LIN 3*, *LIN 4*, *CIR 2*, *CIR 3* und *CIR 4* im Anhang

- Limitationen: Genaue Vorhersage und Klassifikation an der Toleranzgrenze schwierig und teilweise fehlerbehaftet
 - Verwendung der Vorhergesagten Abweichung als Entscheidungsgrundlage für eine mögliche Nachuntersuchung des Bauteils
- Kritisch ist besonders der Fall, dass ein n.i.O. Bauteil versehentlich als i.O. klassifiziert wird
 - Durch Feature Selection und eine möglichst gute Modellauswahl ist dieser Fall selten (0,51 % aller Fälle)
 - Vorhergesagte Abweichung zum Nennmaß und der Toleranz gibt weiterhin Aufschluss über die Zuverlässigkeit einer Klassifizierung
 - Geringe Abweichung zur Toleranz → möglicherweise falsche Klassifizierung → Nachuntersuchung sinnvoll

Feature Selection – Feature Importance

- Von insgesamt 180 statistischen Feature werden die 68 wichtigsten ausgewählt
- Vergleichbarkeit zur Korrelationsanalyse gewährleistet:
Mit $\text{corr_value}=0,7$ bleiben 68 statistische Feature
- Für *LIN_1* sind darin 30 unterschiedliche Maschinensignale enthalten

geometrisches Feature	KNeighborRegressor			KNeighborRegressor mit Feature Selection		
	Mean Squared Error	Mean Absolute Error	R ²	Mean Squared Error	Mean Absolute Error	R ²
LIN_1	6.481e-06	0.001526	0.9782	1.268e-05	0.001893	0.9694
CIR_1	1.013e-05	0.001942	0.9271	1.257e-05	0.002055	0.9251
LIN_2	7.881e-06	0.001433	0.8357	6.7436e-06	0.001246	0.8393
CIR_2	1.478e-05	0.002170	0.9013	1.467e-05	0.002182	0.8979
LIN_3	5.659e-06	0.001332	0.8609	7.875e-06	0.001407	0.8231
CIR_3	8.605e-06	0.001873	0.9642	7.947e-06	0.001899	0.9669
LIN_4	8.412e-06	0.001504	0.8734	7.610e-06	0.001464	0.8468
CIR_4	1.028e-05	0.001848	0.9131	1.661e-05	0.002194	0.8718

Feature Selection – Vergleich

- Starke Überschneidung der verbleibenden Maschinensignale aus der Feature Importance und der Korrelationsanalyse
- Algorithmus lernt mit Maschinensignalen, die mit Ingenieurwissen nachvollziehbar sind
- Korrelationsanalyse ($\text{corr_value} = 0,7$): **17 Maschinensignale** und **68 statistische Feature** verbleiben
- Feature Importance: **30 Maschinensignale** und **68 statistische Feature** verbleiben