Modellbildung 1 - Übungen

I. Laden und Visualisieren von Daten

Im folgenden soll ein statisches polynomiales Modell gebildet werden, welches die Maschinenparameter auf die resultierende Bauteilqualität abbildet. Für das Modelltraining und die Modellvalidierung stehen Ihnen jeweils ein Datensatz zur Verfügung: `setpoint_data_train.pkl` und `setpoint_data_val.pkl`

1. Laden Sie die Datensätze mittels der `load()` Methode der Bibliothek pickle
2. Die Datensätze liegen im Format eines pandas Dataframe vor, machen Sie sich vertraut, wie Sie Zeilen, Spalten und Zellen in einem Dataframe adressieren können. Hilfreich hierfür ist das Pandas Cheat Sheet.
3. Nutzen Sie die stripplot() Methode der seaborn bibliothek, um die Trainings- und Validierungsdaten zu visualisieren. Nutzen Sie folgenden Code als Vorlage

```
import matplotlib.pyplot as plt
import seaborn as sns

fig, ax = plt.subplots(figsize=(20, 10))      # Create Figure and ax object

sns.stripplot(x=…,y=…, ,ax=ax)                # Plot Training Data
sns.stripplot(x=…,y=…, ,ax=ax)                # Plot Validation Data
```

II. Polynomiales Modell

Ziel ist den Bauteilinnendurchmesser aus den eingestellten Maschinenparametern zu bestimmen. Die Bezeichnungen (Label) der Eingangsgrößen und Ausgangsgrößen in den Dataframes sind

```
u_label=['Düsentemperatur','Werkzeugtemperatur','Einspritzgeschwindigkeit',
'Umschaltpunkt', 'Nachdruckhöhe', 'Nachdruckzeit', 'Staudruck', 'Kühlzeit']

y_label = ['Durchmesser_innen']
```

Als Modellansatz wird ein Polynom zweiter Ordnung mit Interaktionstermen gewählt. Der Code um das Modell als CasADi-Gleichung zu definieren lautet

```
theta0 = cs.MX.sym('theta',1,1)
theta1 = cs.MX.sym('theta',8,1)
theta2 = cs.MX.sym('theta',36,1)

theta = cs.vcat([theta0,theta1,theta2])

u = cs.MX.sym('u',8,1)
interact = cs.mtimes(u,u.T)
interact = [interact[i:,i] for i in range(0,8)]
interact = cs.vcat(interact)

y = theta0 + cs.mtimes(theta1.T,u) + cs.mtimes(theta2.T,interact)

f_model = cs.Function('f_model',[u,theta],[y],['u','theta'],['y'])
```

1. Werten Sie das Modell auf den Trainingsdaten aus und definieren Sie die Kostenfunktion in Abhängigkeit der unbekannten Parameter theta. Nutzen Sie folgenden Code als Vorlage

```
L = 0

for k in data_train.index:
    u_k = …
    y_k = …

    y_hat = …
    L = L + 0.5*(y_hat - y_k)**2
```

2. Nutzen Sie die qpsol() Methoder der casadi Bibliothek, um die optimalen Modellparameter zu ermitteln.
3. Werten Sie das trainierte Modell auf den Trainings- und Validierungsdaten aus und Visualisieren Sie das Ergebnis.

III. MLP

Im Folgenden soll ein Mulilayer-Perceptron (MLP) mit einem Neuron in der verdeckten Schicht genutzt werden, um aus der Werkzeuginnentemperatur und dem Werkzeuginnendruck extrahierte Features auf die Bauteilqualität abzubilden. Nutzen Sie folgenden Code, um die Modellgleichung als CasADi-Funktion zu definieren

```
W_h = cs.MX.sym('W_h',1,8)
b_h = cs.MX.sym('b_h',1,1)

W_o = theta1 = cs.MX.sym('W_o',1,1)
b_o = cs.MX.sym('b_o',1,1)

theta = cs.vcat([W_h.reshape((-1,1)),b_h,W_o.reshape((-1,1)),b_o])

u = cs.MX.sym('u',8,1)

h = cs.tanh(cs.mtimes(W_h,u)+b_h)
y = cs.mtimes(W_o,h)+b_o


f_model = cs.Function('f_model',[u,theta],[y],['u','theta'],['y'])
```

1. Laden Sie die Datensätze : `feature_data_train.pkl` und `feature_data_val.pkl`
2. Werten Sie das Modell auf den Trainingsdaten aus und definieren Sie die Kostenfunktion in Abhängigkeit der unbekannten Parameter `theta`. Nutzen Sie folgenden Code als Vorlage

```
L = 0

for k in data_train.index:
    u_k = …
    y_k = …

    y_hat = …
    L = L + 0.5*(y_hat - y_k)**2
```

3. Nutzen Sie die nlpsol() Methode der casadi Bibliothek, um die optimalen Modellparameter zu ermitteln.
4. Werten Sie das trainierte Modell auf den Trainings- und Validierungsdaten aus und Visualisieren Sie das Ergebnis.

5. Entspricht das Ergebnis Ihren Erwartungen? Skalieren Sie die Trainings- und Validierungsdaten auf das Intervall [-1,1]. Die Bildungsvorschrift für diese Skalierung ist

$$x_{[-1,1]} = \frac{x - x_{\min}}{x_{\max} - x_{\min}} - 1$$

Nutzen Sie die min() und max() Methode des Pandas Dataframes, um $x_{\min}$ und $x_{\max}$ zu ermitteln.

6. Wiederholen Sie die gesamte Prozedur mit den skalierten Daten und vergleichen Sie.