Künstliche Intelligenz WS 19/20

ÜBUNGSBLATT 2

Abgabe bis: 9. Dezember 2019

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN

Vorlesung: Prof. Dr. Paul Lukowicz
Übungen: M.Sc Peter Hevesi, Kunal Oberoi, M.Sc Vitor Fortes, B.Sc Matthias Tschöpe

Hinweis: Alle Programmieraufgaben sind in Python abzugeben.

Im Tutorium wurde eine Klasse Optimierer geschrieben, die zwei Lernalgorithmen zur Verfügung stellt (Ridge Regression und SGD). In der Datei *exercise_2_template.py* finden Sie die Grundstruktur einer ähnlichen Klasse, die in den folgenden Aufgaben fertig implementiert werden soll. Die implementierte Klasse soll in der Lage sein, die Parameter eines Polynoms p-ten Grades von Trainingspunkten zu bestimmen und anschließend soll das gefittete Polynom-Modell neue Testdaten vorhersagen können.

Aufgabe 2.1

Laden Sie die Daten aus dem *Train_Dataset.csv* (siehe Materialordner/Übungsblatt 2) und speichern Sie die zwei Spalten separat in die Variablen `x_train` und `y_train` ab. Plotten Sie die eingelesenen Werte als x-y Punktpaare. Wie viele Samples sind in den Trainingsdaten ($n =$?)?

Aufgabe 2.2

In dieser Aufgabe soll die ”Feature”-Matrix X generiert werden. Wenn p der Grad des Polynoms ist (Modell) und

$$x = [x_1, x_2, ...x_i, ...x_n] \tag{2.1}$$

dann soll X wie folgt aussehen:

$$X = \begin{bmatrix} 1 & x_1 & \dots & x_1^p \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & \dots & x_n^p \end{bmatrix} \tag{2.2}$$

Ergänzen Sie dafür die Methode `_generate_features(...)` der Klasse. *Hinweis*: p wird über den Konstuktor der Klasse übergeben (*degree*).

Aufgabe 2.3

(a) Für die `fit`-Methode der Klasse implementieren Sie eine der Optimierungsverfahren, die im Tutorium gezeigt wurden (z.B. Gradientenabstieg oder Normalgleichung). Die Methode soll die Werte für die `model`-Klassenvariable setzen, so dass die Fehlerfunktion (Mean Squared Error - MSE) für die ermittelten Parameter minimal wird.

(b) Mit den Daten aus dem Train_Dataset.csv fitten Sie die Datenpunkte mit einem Modell vom Grad $2, 3$ und 4 (d.h. quadratische, kubische und quartische Approximationen). Plotten Sie Ihre Ergebnisse (Input-Datenpunkte und Funktionswerte für alle drei Polynome).

Hinweis: Das Modell für Polynomgrad p hat $p + 1$ Parameter.
Hinweis: Abhängig von der gewählten Optimierungsmethode, dürfen Sie weitere Argumente (z.B. learning Rate, Toleranz, usw.) für den Konstruktor oder fit-Methode einführen.
Hinweis: Für die Lösung des Optimierungsproblems finden Sie weitere Hinweise in den Übungsfolien von 3. und 4. Woche.

Aufgabe 2.4

In dieser Aufgabe werden Sie Vorhersagen für neue Daten berechnen. Beachten Sie, dass diese Aufgabe daher auf der vorherigen Aufgaben aufbaut. Im Materialordner finden Sie die Datei `Test_Dataset.csv`. Lösen Sie damit die folgenden Aufgaben:

(a) Ergänzen Sie die Methode `predict`, damit sie für neue x-Werte mit Verwendung der optimierten Modellparameter die dazugehörige y-Werte (Funktionswerte des Polynoms an den Stellen x_test) berechnet.

(b) Verwenden Sie nun die Methode `predict` und die vorher trainierten weights ($model$) um die Funktionswerte aus der Datei Test_Dataset.csv vorherzusagen. Plotten Sie auch hier Ihre Ergebnisse für y_test (aus der Datei) und $y\_predict$ (Ergebnis der Vorhersage).

(c) Wieso ist die Approximation mit einem Modell vom Grad 3 oder 4 schlechter als die Approximation vom Grad 2?

Hinweis: Sie sollten die Werte aus der Datei diesmal in die Variablen x_test und y_test abspeichern.
Hinweis: Nachdem die Feature-Matrix X für die x-Werte generiert ist, kann man mit einer einfachen Matrix-Multiplikation die vorhersageten Werte berechnen (siehe Tutorium).

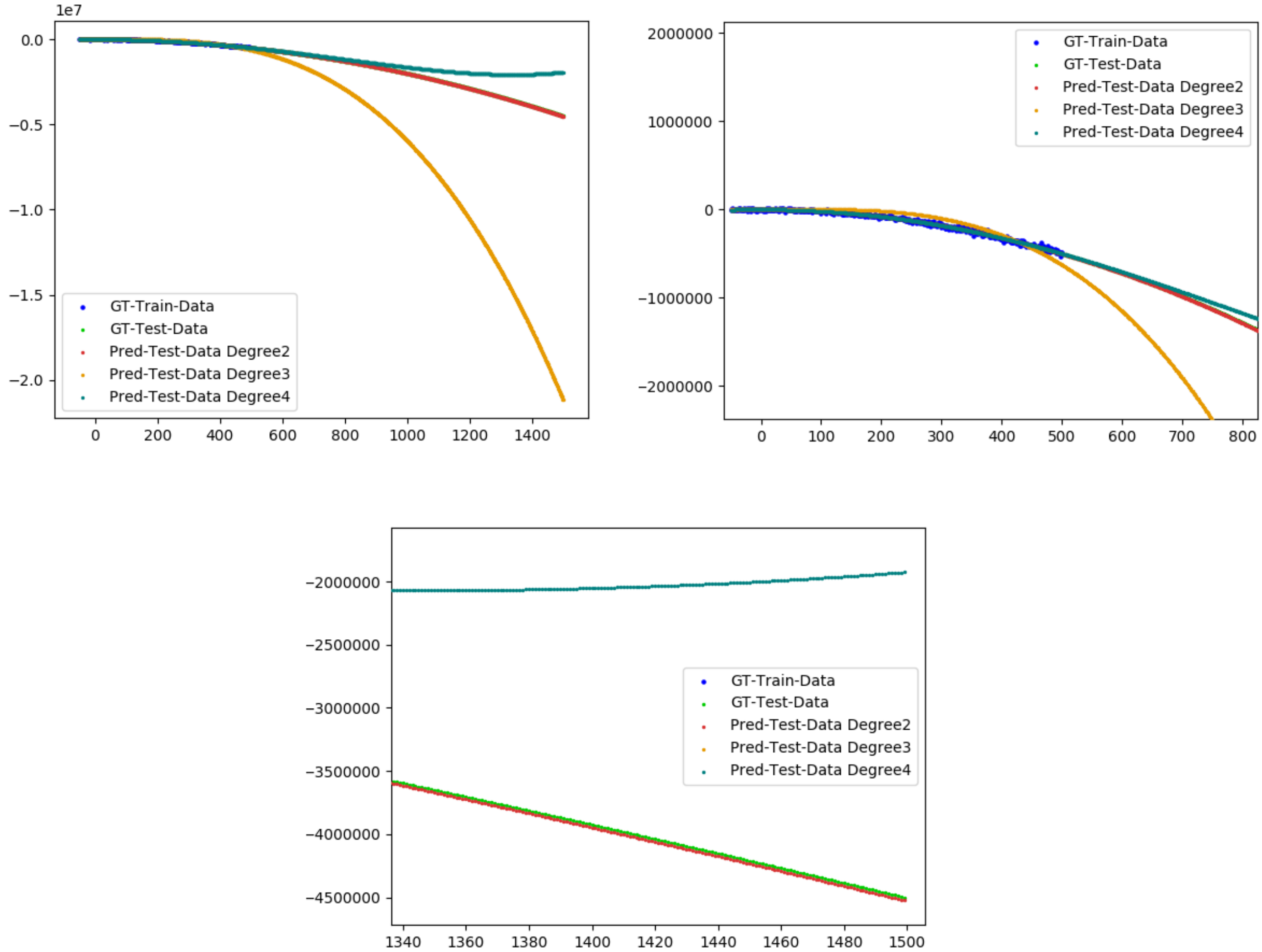

Abbildung 2.1: Ihre Plots zu Aufgabe 2.3 und 2.4 sollten in etwa so aussehen

Aufgabe 2.5

Im Tutorium haben wir die Mean Squared Error (MSE) verwendet um den Fehler zwischen den Modell-Daten und den Label-Daten zu berechnen. Darüber hinaus gibt es noch viele weitere Fehler-Funktionen (engl. Loss-Function), die in unterschiedlichen Aufgaben ihre Vor- und Nachteile haben können. Eine weitere Fehler-Funktion ist die euklidische Norm. Diese ist wie folgt definiert:

$$ED(y, \widehat{y}) := \left\| y - \widehat{y} \right\|_2 = \sqrt{\sum_{i=1}^{n} (y_i - \widehat{y}_i)^2}$$

Implementieren Sie eine Methode $ed(y, \widehat{y})$, die die ED zwischen den Label-Daten `y` und den Modell-Daten $\widehat{y}$ berechnet. Für die Berechnung nehmen wir die folgenden Werte:

`y` $:=$ `[0.8, 0.43, 1.74, 0.26, 4.06, 0.73, 2.8, 3.37]`

$\widehat{y}$ $:=$ `[3.49, 1.3, 1.49, 4.12, 2.19, 4.24, 4.67, 0.22]`