Künstliche Intelligenz WS 20/21

3. ÜBUNGSBLATT

TECHNISCHE UNIVERSITÄT KAISERSLAUTERN

Aufgabe 3.3 (2,5 Punkte)

Fügen Sie Ihrer Klasse Optimizer (siehe Aufgaben der 5. Woche) folgende Funktionen hinzu:

`gradient_descent(self, X, w, y, tol=0.00001, learning_rate=0.1)` berechnet den *Gradient Descent* auf f mit Startwerten w. Den Algorithmus finden Sie auch im Zusatzmaterial zur 6. Woche (siehe Algorithmus 1). Sie können entweder die euklidische Norm verwenden (wie in den Folien) oder den *Mean Squared Error* (MSE). Beachten Sie, dass sich der Gradient ändert, wenn Sie MSE statt der euklidische Norm verwenden.

`compute_gradient(self, X, w, y)` berechnet den Gradienten $\nabla f(w)$ (siehe Aufgaben der 6. Woche und Zusatzmaterial).

Testen Sie Ihre Methoden mit Hilfe der Datenpunkte aus den Dateien `fkt_1.txt`, `fkt_2.txt`, `fkt_3.txt`, `fkt_4.txt`, `fkt_5.txt` (siehe: `/home/share/` auf dem Jupyterhub). Visualisieren Sie Ihre Ergebnisse für jede Datei. Wie schätzen Sie Ihre Ergebnisse ein? Welche Approximationen empfinden Sie für gut und welche für weniger gut?

Aufgabe 3.4 (1 Punkte)

Beantworten Sie folgende Fragen:

1. Wieso verwendet man einen Trainings- Validierungs- und Testdatensatz und welche Eigenschaften sollten Sie erfüllen? Gehen Sie auch auf die prozentuale Größenverteilung ein.
2. Angenommen Sie trainieren ein Modell und überwachen sowohl das Trainingsloss als auch das Validationloss. Wann würden Sie aufhören zu trainieren?
3. Erklären Sie den Unterschied zwischen Underfitting und Overfitting?
4. Welche Möglichkeiten gibt es für das Validationloss sich zu ändern? In welchem Fall würden Sie weiter trainieren und in welchem würden Sie ggf aufhören zu trainieren? Nutzen Sie eine Zeichnung um Ihre Gedanken zu visualisieren.
5. Während des Trainings stellen Sie fest, dass sich sowohl das Trainingsloss als auch das Validationloss kaum noch ändert. Was können Sie ändern um dieses Problem zu beheben?

Aufgabe 3.5 (2,5 Punkte)

In dieser Aufgabe sollen Sie Ihren WG-Datensatz (siehe erstes Übungsblatt) verwenden. Nutzen Sie das Pytorch-Grundgerüst aus dem Jupyter-Notebook von Woche 6, um ein Neuronales Netz mit mindestens 2 Hidden-Layern zu erstellen. Trainieren Sie anschließend das Netz so, dass folgende Parameter als Eingabe verwendet werden:

(1) Größe der Wohnung

(2) Art der Wohnung

(3) Ort der Wohnung

und damit die WG-Preise und Wohnungspreise vorhergesagt werden. Achten Sie beim Erstellen des Modells darauf, dass sie nun drei Eingabewerte verarbeiten müssen. Erstellen Sie für jeden Ort (Kaiserslautern-Uni-Wohngebiet, Kaiserslautern-Innenstadt, Kaiserslautern Betzenberg, etc.) und für alle Orte zusammen jeweils einen $3D$—Plot. Dabei soll die x-Achse die Größe, die y-Achse die Art und die z-Achse die Preise der Wohnungen repräsentieren. Plotten Sie in die gleiche Abbildung die vorhergesagten Preise, die durch ihr Modell erzeugt werden. Wie gut schätzen Sie Ihr trainiertes Modell für zukünftige neue Wohnungen ein? Nennen Sie einige Verbesserungsvorschläge für ein besseres Modell.

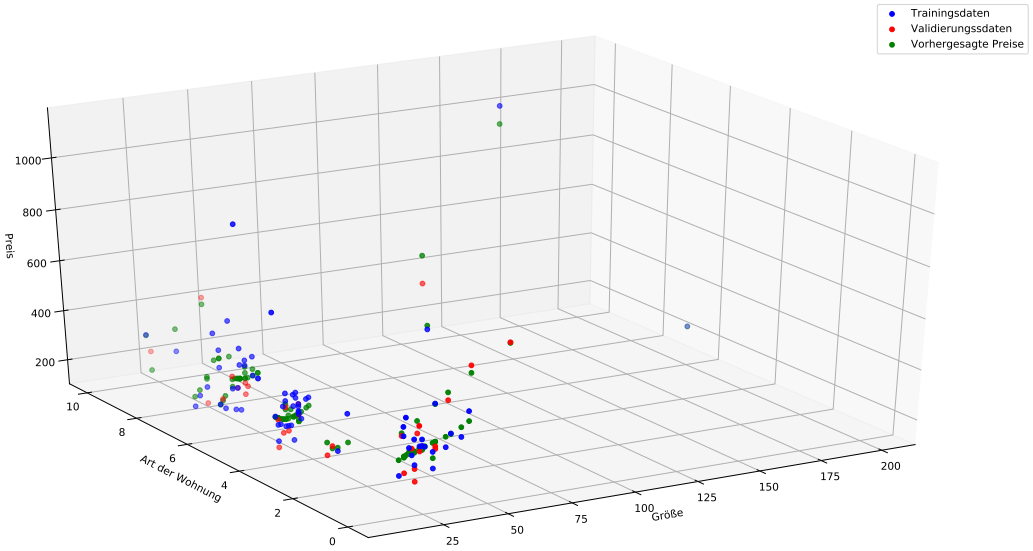

Abbildung 3.2: Ein $3D$—Plot für die Aufgabe zur Vorhersage der WG-Preise, könnte so aussehen.