Laborversuch 2

Versuch	**Lineare Regression und Regularisierung**
Fach	Intelligente Lernende Systeme
Semester	WS 2021/22
Fachsemester	CPS 5 und ITS5/WIN5
Labortermine	25.11.2021
	02.12.2021
Abgabe bis spätestens	10.12.2021

Versuchsteilnehmer

Name: Vorname:

Semester: Matrikelnummer:

Bewertung des Versuches

Aufgabe:	1	2	3	
Punkte maximal:	45	30	35 (ZP)	
Punkte erreicht:				
Gesamtpunktezahl:	/75	Note:		Zeichen:

Anmerkungen:

Aufgabe 1: (10+9+14+12 = 45 Punkte)

Thema: Lineare "Least-Squares" Regression mit Regularisierung in Python

Gegeben seien Daten $\{(x_n, t_n)|n = 1, ..., N\}$ welche ursprünglich von der Parabel $f(x) = w_0 + w_1 x + w_2 x^2$ mit $w_0 = 2$, $w_1 = -1$, $w_2 = 3$ gesampelt wurden, aber nun mit Rauschen behaftet sind. Zu diesen Daten soll ein lineares Regressionsmodell $y = \mathbf{w}^{\mathrm{T}} \boldsymbol{\phi}(x)$ mit polynomiellen Basisfunktionen $\boldsymbol{\phi}$ bestimmt werden.

a) Betrachten Sie das Programmgerüst `V2A1_LinearRegression.py` aus dem Praktikumsverzeichnis:

- Erklären Sie kurz in eigenen Worten (jeweils 1-2 Sätze) wozu die Funktionen `fun_true(.)`, `generateDataSet(.)`, `getDataError(.)` und `phi_polynomial(.)` dienen. Versuchen Sie den Python-Code zu verstehen (muss nicht dokumentiert werden).
- Von welcher Funktion sind die Original-Daten (x_n, t_n) gesampelt?
- Wie lauten die Basisfunktionen $\phi_j(x)$ für $j = 1, ...,$`deg` des linearen Modells?
- Welche Rolle hat die Variable lmbda?
- Worin unterscheiden sich die Variablen `X`,`T` von `X_test`,`T_test`?
- Was stellen im Plot die grünen Kreuze/Punkte, grüne Kurve, rote Kurve dar?

b) Vervollständigen Sie das Programm:

- Implementieren Sie die Berechnung der regularisierten Least-Squares-Gewichte `W_LSR` als $M \times 1$-Matrix
- Implementieren Sie die Berechnung der Prognosewerte `Y` als $N \times 1$-Matrix.

c) Testen Sie das Programm zunächst ohne Regularisierung ($\lambda = 0$) für $N = 10$:

- Welche optimalen Gewichte $\mathbf{W}_{\mathrm{LSR}}$ erhalten Sie für Polynomgrad 5? Wie groß ist der Lern-Datenfehler $E_D(\mathbf{W}_{\mathrm{LSR}})$? Wie groß ist der Fehler auf den Testdaten? Warum ist der Test-Daten-Fehler größer als der Lern-Daten-Fehler?
- Vergleichen Sie Lern- und Test-Datenfehler für verschiedene Polynomgrade 1, 2, 3, 4, 5, 7, 9? Welche Phänomene treten bei zu niedrigem bzw. zu hohem Polynomgrad auf? Warum?
- Bestimmen Sie das mittlere Gewicht $\frac{1}{M} \sum_{j=0}^{M-1} |W_{\mathrm{LSR}j}|$ für verschiedene Polynomgrade 1,2,4,6,9? Warum werden die Gewichte im Mittel größer?
- Bestimmen Sie den mittleren Lern- bzw. Test-Datenfehler (pro Datenpunkt) für Polynomgrad 9 und verschiedene Größen des Datensets $N = 10, 100, 1000, 10000$. Warum wird der eine Fehler kleiner und der andere größer?
- Wieviele Daten N braucht man, damit für Polynomgrad 2 die tatsächlichen Koeffizienten der Original-Funktion `fun_true` bis auf 10% Genauigkeit geschätzt werden können? (ungefährer Wert reicht)

d) Testen Sie das Programm für Polynomgrad 9, $N = 10$ Daten und Regularisierung $\lambda \geq 0$:

- Beschreiben Sie kurz (1-2 Sätze) den Nutzen einer Regularisierung?
- Bestimmen Sie den Lern- und Test-Datenfehler sowie das mittlere Gewicht für verschiedene Werte λ =0, 0.01, 0.1, 1, 10, 100, 1000, 10000. Welche Probleme treten auf wenn λ zu klein oder zu groß gewählt wird?
- Für welches λ wird der Generalisierungsfehler (auf den Test-Daten) minimal?

Aufgabe 2: (6+8+9+3+4 = 30 Punkte)

Thema: Python-Modul für Lineare und k-Nearest-Neighbor (KNN) Regression

Da wir im folgenden verschiedene Regressions-Verfahren implementieren wollen lohnt es sich ein eigenes Python-Modul dafür zu erstellen (siehe Programmgerüst `V2A2_Regression.py`).

a) Versuchen Sie zunächst den Aufbau des Moduls `V2A2_Regression.py` zu verstehen:

- Betrachten Sie den Aufbau des Moduls durch Eingabe von `pydoc V2A2_Regressifier`. Welche Klassen gehören zu dem Modul und welchen Zweck haben sie jeweils?
- Betrachten Sie nun die Basis-Klasse `Regressifier` im Quelltext: Wozu dienen jeweils die Methoden `fit(self,X,T)`, `predict(self,x)` und `crossvalidate(self,S,X,T)` ?
- Worin unterscheidet sich `crossvalidate(.)` von der entsprechenden Methode für Klassifikation (siehe vorigen Versuch)?

b) Betrachten Sie nun die Funktion `phi_polynomial(x,deg)`:

- Was berechnet die Funktion? Welches Ergebnis liefert `phi_polynomial([3],5)`? Welches Ergebnis liefert `phi_polynomial([3,5],2)`?
- Geben Sie eine allgemeine Formel an für das Ergebnis von `phi_polynomial([x1,x2],2)`?
- Wozu braucht man diese Funktion im Zusammenhang mit Regression?
- Bis zu welchem Polynomgrad kann die Funktion Basisfunktionen berechnen? Erweitern Sie die Funktion mindestens bis Grad 5.

c) Betrachten Sie die Klasse `LSRRegressifier`:

- Welche Art von Regressions-Modell berechnet diese Klasse?
- Wozu dienen jeweils die Parameter `lmbda`, `phi`, `flagSTD` und `eps` ?
- Welche Rolle spielt hier die Klasse `DataScaler`?
  In welchen Methoden und zu welchem Zweck werden die Daten ggf. umskaliert?
  Welches Problem kann auftreten wenn man dies nicht tut?
  Wozu braucht man die Variablen `Z` und `maxZ` in der Methode `fit(.)`?
- Vervollständigen Sie die Methoden `fit(self,X,T,...)` und `predict(self,x,...)` (vgl. vorige Aufgabe).

d) Betrachten Sie die Klasse `KNNRegressifier`:

- Welche Art von Regressions-Modell berechnet diese Klasse?
- Wozu dienen jeweils die Parameter `K` und `flagKLinReg`?
- Beschreiben Sie kurz in eigenen Worten (2-3 Sätze) auf welche Weise die Prädiktion $\mathbf{y}(\mathbf{x})$ berechnet wird.

e) Betrachten Sie abschließend den Modultest:

- Beschreiben Sie kurz was im Modultest passiert.
- Welche Gewichte W werden gelernt? Wie lautet also die gelernte Prädiktionsfunktion? Welche Funktion sollte sich idealerweise (für $N \to \infty$) ergeben?
- Welche Ergebnisse liefert die Kreuzvalidierung? Was bedeuten die Werte?
- Vergleichen und Bewerten Sie die Ergebnisse von Least Squares Regression gegenüber der KNN-Regression (nach Optimierung der Hyper-Parameter λ, K, ...).

Zusatzaufgabe 3 14+9+2+10 = 35 Punkte

Thema: Lineare Regression auf Airfoil-Noise-Daten

Betrachten Sie nun die Excel-Datensammlung `airfoil_self_noise.xls` (siehe Vorlesungsverzeichnis): Sie besteht aus $N = 1503$ Datensätzen der Dimension $D = 6$, welche den Schalldruck (Spalte 6) einer Flugzeugtragfläche in Abhängigkeit verschiedener Parameter (wie z. B. Anströmwinkel und -geschwindigkeit; Spalten 1-5) darstellen. Sie sollen auf diesen Daten ein Regressionsmodell lernen, um für neue Parametersätze den resultierenden Schalldruck vorhersagen zu können.

a) Vervollständigen Sie das Programmgerüst `V2A3_regression_airfoilnoise.py` um eine **Least-Squares-Regression** auf den Daten zu berechnen. Optimieren Sie die Hyper-Parameter um bei einer $S = 3$-fachen Kreuzvalidierung möglichst kleine Fehlerwerte zu erhalten.

   - Welche Bedeutung haben jeweils die Hyper-Parameter `lmbda`, `deg`, `flagSTD`?
   - Was passiert ohne Skalierung der Daten (`flagSTD`=0) bei höheren Polynomgraden (achten Sie auf die Werte von `maxZ`)?
   - Geben Sie Ihre optimalen Hyper-Parameter sowie die resultierenden Fehler-Werte an.
   - Welche Prognosen ergibt Ihr Modell für die neuen Datenvektoren `x_test_1=[1250,11,0.2,69.2,0.0051]` bzw. `x_test_2=[1305,8,0.1,57.7,0.0048]` ?
   - Welchen Polynomgrad und wieviele Basisfunktionen verwendet Ihr Modell?

b) Vervollständigen Sie das Programmgerüst `V2A3_regression_airfoilnoise.py` um eine **KNN-Regression** auf den Daten zu berechnen. Optimieren Sie die Hyper-Parameter um bei einer $S = 3$-fachen Kreuzvalidierung möglichst kleine Fehlerwerte zu erhalten.

   - Welche Bedeutung haben jeweils die Hyper-Parameter `K` und `flagKLinReg`?
   - Geben Sie Ihre optimalen Hyper-Parameter sowie die resultierenden Fehler-Werte an.
   - Welche Prognosen ergibt Ihr Modell für die neuen Datenvektoren `x_test_1=[1250,11,0.2,69.2,0.0051]` bzw. `x_test_2=[1305,8,0.1,57.7,0.0048]` ?

c) Vergleichen Sie die beiden Modelle. Welches liefert die besseren Ergebnisse?

   AIRFOIL-CHALLENGE: Vergleichen Sie Ihre Ergebnisse mit den anderen Praktikums-Gruppen: Wer erreicht die kleinsten Fehlerwerte? (Muss nicht dokumentiert werden.)

d) Testen Sie Least-Squares und KNN auch mit Hilfe der Machine-Learning-Bibliothek **Scikit-Learn** und vergleichen Sie mit Ihrer Implementierung! Testen Sie auch andere Regressions-Algorithmen Ihrer Wahl (z.B. Neuronale Netze, Random Forests, SVM, ...). Hinweise siehe auch Vorlesung zu Machine-Learning-Bibliotheken.