13 Neuronale Netze

13.1 Mathematische Grundlagen

In einem *k-schichtigen neuronalen Netz* werden durch eine Funktion f Inputs der Form $x \in \mathbb{R}^{n_0}$ in Outputs der Form $y \in \mathbb{R}^{n_k}$ transformiert. Die Funktion f ist dabei eine Verkettung von *Potentialfunktionen*, die von Aktivierungen der Schicht $l-1$ auf Potentiale der Schicht l abbilden

$$z_i^l = \sum_{j=1}^{n_{l-1}} w_{ij}^l a_j^{l-1} + w_{i,n_{l-1}+1} \, , \tag{1}$$

und *Aktivierungsfunktionen*, die von Potentialen der Schicht l auf Aktivierungen der Schicht l abbilden

$$a_i^l = \sigma \left(\sum_{j=1}^{n_{l-1}} w_{ij}^l a_j^{l-1} + w_{i,n_{l-1}+1} \right) \, , \tag{2}$$

wobei $\sigma(z)$ die gewählte *univariate Aktivierungsfunktion* ist und w_{ij}^l die Einträge der *Wichtungsmatrix* der l-ten Schicht darstellen:

$$W^l = \begin{pmatrix} w_{1,1}^l & \dots & w_{1,n_{l-1}}^l & w_{1,n_{l-1}+1}^l \\ \vdots & \ddots & \vdots & \vdots \\ w_{n_l,1}^l & \dots & w_{n_l,n_{l-1}}^l & w_{n_l,n_{l-1}+1}^l \end{pmatrix} \, . \tag{3}$$

Für einen gegebenen *binären Klassifikationsdatensatz*

$$\mathcal{D} = \{(x_1, y_1), \dots, (x_n, y_n)\} \tag{4}$$

mit $x_i \in \mathbb{R}^m$ und $y_i \in \{0, 1\}$ für $i = 1, \dots, n$ wird ein k-schichtiges neuronales Netz trainiert, indem $x = a^0$ und $a^k = y$ gesetzt wird.

Wie in der Vorlesung gezeigt wurde, können im Anschluss die optimalen Gewichte $\hat{W}^l$ für $l = 1, \dots, k$ der optimalen Abbildung $\hat{f}$ mithilfe des *Backpropagation-Algorithmus* geschätzt werden. Für einen potentiell neuen Featurevektor $\tilde{x} \in \mathbb{R}^m$ kann dann das unbekannte Klassenlabel als $\hat{y} = \hat{f}(\tilde{x})$ prädiziert werden.

In der vorliegenden Übung wollen wir die Schätzung eines neuronalen Netzes via *leave-one-out cross-validation* sowie die Prädiktion von Labels auf Grundlage der geschätzten Gewichte für einen realen Datensatz nachvollziehen.

13.2 Analyse in R

Die Datei `FADE_SAME.csv` enthält den in der ersten Seminarsitzung vorgestellten Datensatz. Erklären Sie die Funktion des folgenden R-Codes:

```
# Daten einlesen
fname = 'FADE_SAME.csv'                          # Dateiname
D     = read.csv(fname)                          # Dataframe

# Datenmatrix extrahieren
rows  = startsWith(D$subject, 'subA')            # Personen aus Studie A
cols  = c('novelty.FADE','novelty.SAME',         # Definition Variablen
          'memory.FADE', 'memory.SAME')
X     = t(as.matrix(D[rows,cols]))               # Datenmatrix
y     = t(as.matrix(D$memory[rows]))             # gruppendefinierende Variable
y_med = median(y)                                # Median-Gedächtnisleistung
y     = 0*(y <= y_med) + 1*(y > y_med)           # Labelvariable
D     = as.data.frame(cbind(t(y), t(X)))         # Dataframe
print(dim(X))                                    # Überprüfung Datenmatrix
print(dim(y))                                    # Überprüfung Labelvariable
print(sum(y))                                    # Anzahl positiver Fälle
```

13.3 Erste Programmieraufgabe

Führen Sie eine Klassifikationsanalyse auf Grundlage der Datenmatrix X und der Labelvariable y durch, indem Sie die Gewichte eines neuronalen Netzes mit dem Prinzip der *leave-one-out cross-validation* schätzen und zur Vorhersage des jeweils ausgelassenen Datenpunkts verwenden. Gehen Sie dazu wie folgt vor:

- Kopieren Sie den R-Code aus Abschnitt 8.4 des Arbeitsblatts (8) *Prädiktive Modellierung*.
- Entfernen Sie im Abschnitt "Vorbereitung Datenanalyse" die Definition der Variable `L`.
- Laden Sie stattdessen mit `library()` das R-Paket `neuralnet`. (Falls es nicht vorhanden ist, installieren Sie es mittels `install.packages("neuralnet")`.)
- Erzeugen Sie im Abschnitt "Datensatz partitionieren" innerhalb der Schleife den Datenframe `D_train` durch Auslassen der i-ten Zeile aus dem Datenframe `D` und den Datenframe `D_test` als i-te Zeile des Datenframes `D`.
- Trainieren Sie im Abschnitt "Training" innerhalb der Schleife mit `neuralnet()` ein neuronales Netz mit der Output-Schicht-Variable "V1" (y), den Input-Schicht-Variablen "novelty.FADE", "novelty.SAME", "memory.FADE", "memory.SAME" (X) aus dem Trainingsdatensatz `D_train`, ohne verdeckte Schicht, mit der Cross-Entropy-Kostenfunktion und einer sigmoiden Aktivierungsfunktion. Orientieren Sie sich bei der Konstruktion des entsprechenden Befehls am in der Vorlesung präsentierten Code. Speichern Sie den Ausgabewert in die Variable `nn_train`.

- Prädizieren Sie im Abschnitt "Test" innerhalb der Schleife den Wert der Output-Schicht-Variable für den i-ten Datenpunkt, indem Sie mit der Funktion `neuralnet()` das trainierte neuronale Netz `nn_train` auf den Testdatensatz `D_test` anwenden. Speichern Sie das Ergebnis in die Variable `pred`.
- Modifizieren Sie die Klassifikationsregel derart, dass das prädizierte Label des i-ten Datenpunkts 1 ist, wenn `pred` größer 0.5 ist, und 0 anderenfalls.
- Geben Sie nach der Schleife zur Überprüfung die Anzahl positiver Klassifikationen (`y_pred[,2]==1`) aus. Sie sollten folgende Ergebnisse erhalten:

```
[1] 116
```

13.4 Abbildung in R

Die in der Datenmatrix enthaltenen Variablen sollen nun unter Berücksichtigung ihrer tatsächlichen und prädizierten Klassenzugehörigkeit visualisiert werden. Erklären Sie dazu den folgenden R-Code und die Abbildung, die er erzeugt:

```
# Abbildungsparameter
library(latex2exp)
par(
    family    = "sans",
    mfcol     = c(1,2),
    pty       = "m",
    bty       = "o",
    lwd       = 1,
    las       = 1,
    mgp       = c(2,1,0),
    xaxs      = "i",
    yaxs      = "i",
    cex       = 1,
    cex.lab   = 1.5,
    cex.axis  = 1.5)

# Novelty-Scores: Datenpunkte
plot(X[1, y_pred[,1]==0 & y_pred[,2]==0], X[2, y_pred[,1]==0 & y_pred[,2]==0],
    pch       = 15,
    col       = "red",
    xlab      = "Novelty-FADE-Score",
    ylab      = "Novelty-SAME-Score",
    xlim      = c(-3,  0),
    ylim      = c(-3, +3))
points(X[1, y_pred[,1]==0 & y_pred[,2]==1], X[2, y_pred[,1]==0 & y_pred[,2]==1],
    pch       = 15,
    col       = "blue")
points(X[1, y_pred[,1]==1 & y_pred[,2]==0], X[2, y_pred[,1]==1 & y_pred[,2]==0],
    pch       = 16,
```

```
    col       = "red")
points(X[1, y_pred[,1]==1 & y_pred[,2]==1], X[2, y_pred[,1]==1 & y_pred[,2]==1],
    pch       = 16,
    col       = "blue")

# Novelty-Scores: Legende
legend("topleft", c("y = 0, y_hat = 0", "y = 0, y_hat = 1",
                    "y = 1, y_hat = 0", "y = 1, y_hat = 1"),
    lty        = 0,
    pch        = c(15, 15, 16, 16),
    col        = c("red", "blue", "red", "blue"),
    bty        = "n",
    y.intersp  = 1.5)

# Memory-Scores: Datenpunkte
plot(X[3, y_pred[,1]==0 & y_pred[,2]==0], X[4, y_pred[,1]==0 & y_pred[,2]==0],
    pch       = 15,
    col       = "red",
    xlab      = "Memory-FADE-Score",
    ylab      = "Memory-SAME-Score",
    xlim      = c(-3,  0),
    ylim      = c(-3, +3))
points(X[3, y_pred[,1]==0 & y_pred[,2]==1], X[4, y_pred[,1]==0 & y_pred[,2]==1],
    pch       = 15,
    col       = "blue")
points(X[3, y_pred[,1]==1 & y_pred[,2]==0], X[4, y_pred[,1]==1 & y_pred[,2]==0],
    pch       = 16,
    col       = "red")
points(X[3, y_pred[,1]==1 & y_pred[,2]==1], X[4, y_pred[,1]==1 & y_pred[,2]==1],
    pch       = 16,
    col       = "blue")

# Memory-Scores: Legende
legend("topleft", c("y = 0, y_hat = 0", "y = 0, y_hat = 1",
                    "y = 1, y_hat = 0", "y = 1, y_hat = 1"),
    lty        = 0,
    pch        = c(15, 15, 16, 16),
    col        = c("red", "blue", "red", "blue"),
    bty        = "n",
    y.intersp  = 1.5)

# Speichern
dev.copy2pdf(
    file      = "Abbildungen/Neuronale_Netze_1.pdf",
    width     = 16,
    height    = 9)
```

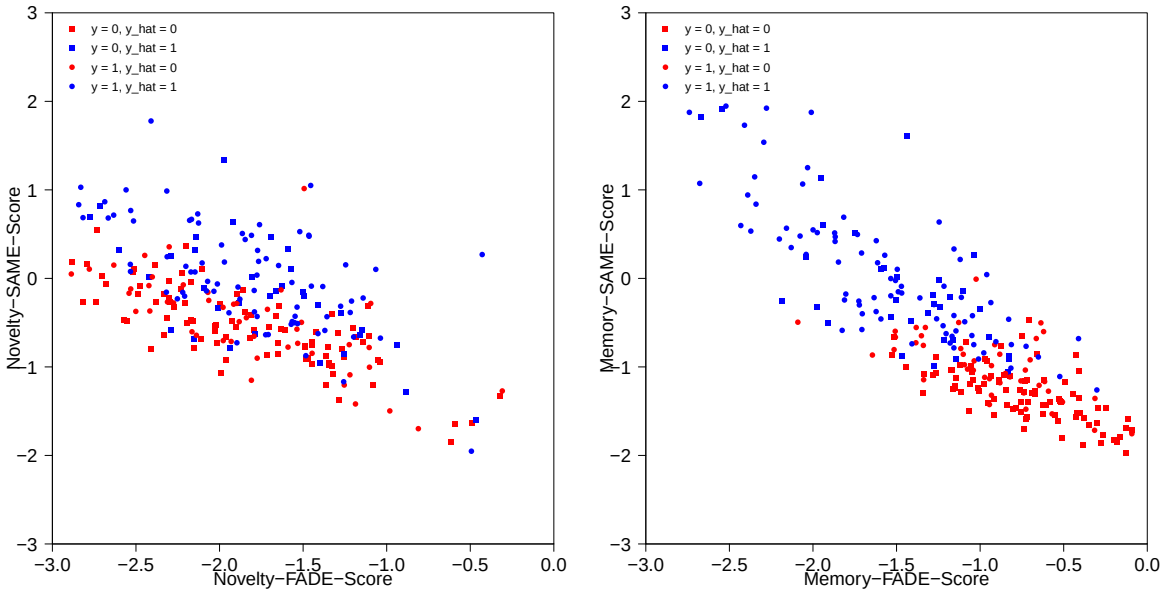

Abbildung 1. Novelty-Scores und Memory-Scores, getrennt nach Gedächtnisleistung (Vierecke: $y = 0$; Kreise: $y = 1$) und eingefärbt nach prädizierter Klassenzugehörigkeit (rot: $\hat{y} = 0$; blau: $\hat{y} = 1$).

13.5 Zweite Programmieraufgabe

Evaluieren Sie die Klassifikationsperformanz, indem Sie die Sensitivität, Spezifizität und Genauigkeit der Klassifikation berechnen. Gehen Sie dazu wie folgt vor:

- Orientieren Sie sich für diese Aufgabe an Abschnitt 8.5 des Arbeitsblatts (8) *Prädiktive Modellierung*.
- Berechnen Sie die Einträge der 2×2 Konfusionsmatrix und speichern Sie die Ergebnisse als `TN`, `FP`, `FN` und `TP`.
- Berechnen Sie mithilfe der in Vorlesung (8) *Prädiktive Modellierung* angegebenen Formeln die true positive rate (TPR) und true negative rate (TNR) sowie accuracy (ACC).
- Geben Sie die Resultate ihrer Analyse aus. Sie sollten folgende Ergebnisse erhalten:

```
true positive rate (sensitivity) :  0.628
true negative rate (specificity) :  0.731
Genauigkeit (accuracy)           :  0.68
```

13.6 Lückentext

Füllen Sie mit den in der Übung gewonnenen Erkenntnissen den folgenden Lückentext aus und präsentieren Sie die Ergebnisse im Seminar:

Lückentext: Die funktionale Architektur neuronaler Netze besteht aus Potentialfunktionen $\Phi^l : \mathbb{R}^{n_{l-1}} \to \mathbb{R}^{n_l}$ mit _____ $W^l \in \mathbb{R}^{n_l \times (n_{l-1}+1)}$ sowie aus Aktivierungsfunktionen $\Sigma^l : \mathbb{R}^{n_l} \to \mathbb{R}^{n_l}$ für $l = 1, \ldots, k$, wobei k die _____ ist. Klassifikation mittels neuronaler Netze besteht darin, dass die Gewichte anhand eines Trainingsdatensatzes angepasst werden, um im Anschluss die unbekannten Labels in einem Testdatensatz vorherzusagen. Im vorliegenden Datensatz wurden die Labels aus der Variable _____ gewonnen, wobei "negative Fälle" _____ und "positive Fälle" _____. Die Klassifikation der so erzeugten Labelvariable aus _____ Features erfolgt mittels *leave-one-out cross-validation*. Die true positive rate (TPR, "Sensitivität") ist _____, d.h. _____, und die true negative rate (TNR, "Spezifizität") ist _____, d.h. _____. Insgesamt ergibt sich eine Klassifikationsgenauigkeit in Höhe von _____.

13.7 Mögliche Klausurfrage

Präsentieren Sie im Seminar folgende Klausurfrage und erklären Sie die richtige Antwort:

Frage: Die Formel für die Aktivierung von Neuron i in Schicht l lautet: $a_i^l = \sigma\left(\sum_{j=1}^{n_{l-1}} w_{ij}^l a_j^{l-1} + w_{i,n_{l-1}+1}^l\right)$. Wofür steht die Variable a_j^{l-1} in dieser Formel?

a) für die Aktivierung von Neuron j in Schicht $l-1$

b) für die Aktivierung von Neuron i in Schicht l

c) für das Gewicht der Verbindung von Neuron j in Schicht $l-1$ und Neuron i in Schicht l

d) für die Aktivierungsfunktion des neuronalen Netzes

13.8 Kinderwitz

Wie heißt der Hausmeister einer Nudelfabrik?

Antwort: Fussility Manager.