Deep Learning mit Python

Logistische Regression

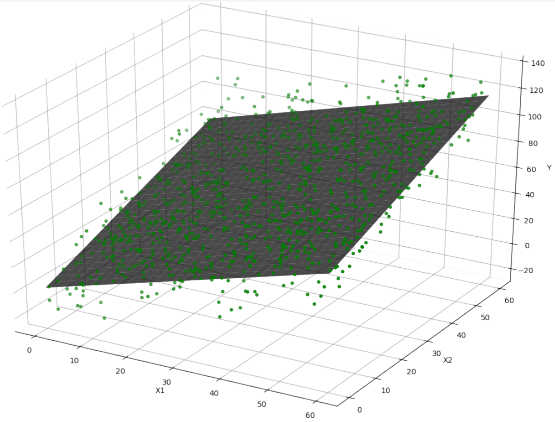
Fabian Gieseke & Moritz Seiler

Department of Information Systems
University of Münster

Übersicht

Übersicht

Wdh.: Multiple Lineare Regression



Allgemeine (mehrdimensionale) Form

- Gegeben: Trainingsdatensatz $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\} \subset \mathbb{R}^d \times \mathbb{R}$
- Ziel: Lineares Modell der Form $f(\mathbf{z}; \mathbf{w}) = w_0 + w_1 z_1 + w_2 z_2 + \ldots + w_d z_d$

Logistische Regression

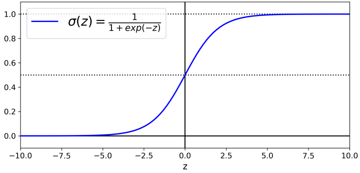
- Sei $\mathcal{Y} = \{0, 1\}$, d.h. wir betrachten binäre Klassifikations-Szenarien.

Logistische Regression

- Sei $\mathcal{Y} = \{0, 1\}$, d.h. wir betrachten binäre Klassifikations-Szenarien.
- Für Regression haben wir Modelle der Form $f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^\mathsf{T}\mathbf{x}$ betrachtet, welche beliebige Werte in $\mathbb{R}$ annehmen können $\rightarrow$ unpassend für binäre Klassifikation.

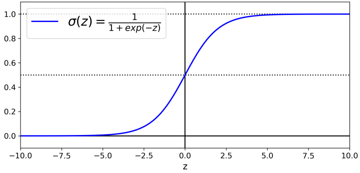
Logistische Regression

- Sei $\mathcal{Y} = \{0, 1\}$, d.h. wir betrachten binäre Klassifikations-Szenarien.
- Für Regression haben wir Modelle der Form $f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^\mathsf{T}\mathbf{x}$ betrachtet, welche beliebige Werte in $\mathbb{R}$ annehmen können $\rightarrow$ unpassend für binäre Klassifikation.
- Die Sigmoid-Funktion $\sigma(z) = \frac{1}{1+\exp(-z)}$ bildet den Definitionsbereich $\mathbb{R}$ auf $[0, 1]$ ab:



Logistische Regression

- Sei $\mathcal{Y} = \{0, 1\}$, d.h. wir betrachten binäre Klassifikations-Szenarien.
- Für Regression haben wir Modelle der Form $f(\mathbf{x}; \mathbf{w}) = \mathbf{w}^\mathsf{T}\mathbf{x}$ betrachtet, welche beliebige Werte in $\mathbb{R}$ annehmen können $\rightarrow$ unpassend für binäre Klassifikation.
- Die Sigmoid-Funktion $\sigma(z) = \frac{1}{1+\exp(-z)}$ bildet den Definitionsbereich $\mathbb{R}$ auf $[0, 1]$ ab:

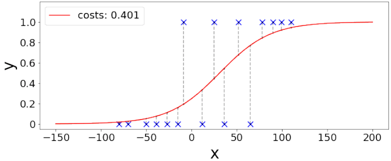


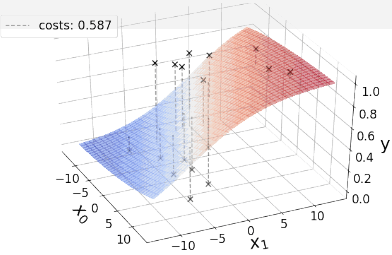
- Wir erhalten dadurch Modelle der Form

$$f(\mathbf{x}; \mathbf{w}) = \frac{1}{1 + \exp(-\mathbf{w}^\mathsf{T}\mathbf{x})},$$

welche für jedes beliebige $\mathbf{x}$ einen Wert in $[0, 1]$ liefern.

Logistische Regression





Modell & Vorhersagen

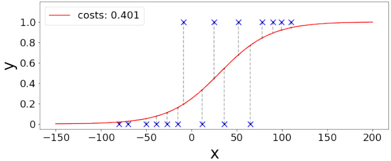
- Wir erhalten dadurch Modelle der Form

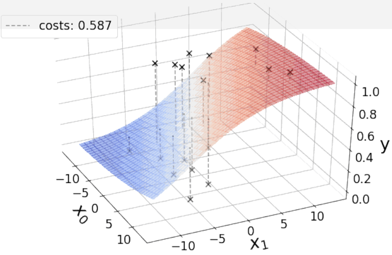
$$f(\mathbf{x};\mathbf{w}) = \frac{1}{1+\exp(-\mathbf{w}^\mathsf{T}\mathbf{x})},$$

welche für jedes beliebige $\mathbf{x}$ einen Wert in $[0,1]$ liefern.

Quelle: https://towardsdatascience.com/animations-of-logistic-regression-with-python-31f8c9cb420

Logistische Regression





Modell & Vorhersagen

- Wir erhalten dadurch Modelle der Form

$$f(\mathbf{x};\mathbf{w}) = \frac{1}{1+\exp(-\mathbf{w}^\mathsf{T}\mathbf{x})},$$

  welche für jedes beliebige $\mathbf{x}$ einen Wert in $[0, 1]$ liefern.

- Die Ausgabe $f(\mathbf{x};\mathbf{w})$ can als Wahrscheinlichkeit für Klasse 1 interpretiert werden. Entscheidung für eine der beiden Klassen mittels Schwellwert (z.B. 0.5):

$$\hat{y} = \begin{cases} 0 & \text{if } f(\mathbf{x};\mathbf{w}) < 0.5 \\ 1 & \text{if } f(\mathbf{x};\mathbf{w}) \geq 0.5 \end{cases}$$

Quelle: https://towardsdatascience.com/animations-of-logistic-regression-with-python-31f8c9cb420

Bestimmung des Fehlers

Verlustfunktion

- Sei $\pi_i = f(\mathbf{x}_i; \mathbf{w}) = \frac{1}{1+\exp\left(-\mathbf{w}^\mathsf{T}\mathbf{x}_i\right)}$ die geschätzte Wahrscheinlichkeit für die i-te Instanz $(\mathbf{x}_i, y_i)$ unseres Datensatzes $T = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\} \subset \mathbb{R}^d \times \{0, 1\}$.

Bestimmung des Fehlers

Verlustfunktion

- Sei $\pi_i = f(\mathbf{x}_i; \mathbf{w}) = \frac{1}{1+\exp(-\mathbf{w}^\intercal \mathbf{x}_i)}$ die geschätzte Wahrscheinlichkeit für die i-te Instanz $(\mathbf{x}_i, y_i)$ unseres Datensatzes $T = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\} \subset \mathbb{R}^d \times \{0, 1\}$.
- Für eine einzelne Instanze betrachten wir die folgende Verlustfunktion:

$$c(\mathbf{w}) = \begin{cases} -\log(\pi_i) & \text{if } y_i = 1 \\ -\log(1-\pi_i) & \text{if } y_i = 0 \end{cases}$$

Kommentar: Der Verlust ist groß falls $y_i = 1$ und π_i nah bei 0 ist bzw. falls $y_i = 0$ und π_i nah bei 1 ist. Der Verlust ist klein falls $y_i = 1$ und π_i nah bei 1 ist bzw. $y_i = 0$ und π_i nah bei 0 ist.

Bestimmung des Fehlers

Verlustfunktion

- Sei $\pi_i = f(\mathbf{x}_i; \mathbf{w}) = \frac{1}{1+\exp\left(-\mathbf{w}^\intercal \mathbf{x}_i\right)}$ die geschätzte Wahrscheinlichkeit für die i-te Instanz $(\mathbf{x}_i, y_i)$ unseres Datensatzes $T = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\} \subset \mathbb{R}^d \times \{0, 1\}$.
- Für eine einzelne Instanze betrachten wir die folgende Verlustfunktion:

$$c(\mathbf{w}) = \begin{cases} -\log(\pi_i) & \text{if } y_i = 1 \\ -\log(1-\pi_i) & \text{if } y_i = 0 \end{cases}$$

Kommentar: Der Verlust ist groß falls $y_i = 1$ und π_i nah bei 0 ist bzw. falls $y_i = 0$ und π_i nah bei 1 ist. Der Verlust ist klein falls $y_i = 1$ und π_i nah bei 1 ist bzw. $y_i = 0$ und π_i nah bei 0 ist.

- Der (durchschnittliche) Verlust für alle Instanzen kann wie folgt bestimmt werden:

$$G(\mathbf{w}) = -\frac{1}{n} \sum_{i=1}^{n} y_i \log \pi_i + (1 - y_i) \log(1 - \pi_i)$$

Bestimmung des Fehlers

Verlustfunktion

- Sei $\pi_i = f(\mathbf{x}_i; \mathbf{w}) = \frac{1}{1+\exp(-\mathbf{w}^\intercal \mathbf{x}_i)}$ die geschätzte Wahrscheinlichkeit für die i-te Instanz $(\mathbf{x}_i, y_i)$ unseres Datensatzes $T = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\} \subset \mathbb{R}^d \times \{0, 1\}$.
- Für eine einzelne Instanze betrachten wir die folgende Verlustfunktion:

$$c(\mathbf{w}) = \begin{cases} -\log(\pi_i) & \text{if } y_i = 1 \\ -\log(1-\pi_i) & \text{if } y_i = 0 \end{cases}$$

  Kommentar: Der Verlust ist groß falls $y_i = 1$ und π_i nah bei 0 ist bzw. falls $y_i = 0$ und π_i nah bei 1 ist. Der Verlust ist klein falls $y_i = 1$ und π_i nah bei 1 ist bzw. $y_i = 0$ und π_i nah bei 0 ist.
- Der (durchschnittliche) Verlust für alle Instanzen kann wie folgt bestimmt werden:

$$G(\mathbf{w}) = -\frac{1}{n} \sum_{i=1}^{n} y_i \log \pi_i + (1 - y_i) \log(1 - \pi_i)$$

- Wir wollen $G(\mathbf{w})$ minimieren. Im Gegensatz zur linearen Regression kann man allerdings keine Lösung in „geschlossener Form" bestimmen. Deshalb greift man i.A. auf das Gradientenabstiegsverfahren zurück ...

Logistische Regression: Demo

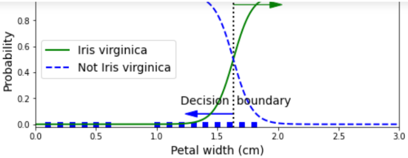
jupyter Logistic Regression (Iris Dataset) Last Checkpoint: a few seconds ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Trusted Python 3

Run Code



Logistic regression with two input features ($d = 2$).

In [4]:
```
# Consider two features
X = iris["data"][:, (2, 3)]  # petal length, petal width
y = (iris["target"] == 2).astype(np.int32)

# fit logistic regression model
log_reg = LogisticRegression(solver="lbfgs", C=10**10, random_state=42)
log_reg.fit(X, y)

# generate mesh grid of points to visualize the decision surface
x0, x1 = np.meshgrid(
        np.linspace(2.9, 7, 500).reshape(-1, 1),
        np.linspace(0.8, 2.7, 200).reshape(-1, 1),
    )
X_new = np.c_[x0.ravel(), x1.ravel()]
y_proba = log_reg.predict_proba(X_new)

# visualize results
plt.figure(figsize=(10, 10))
plt.plot(X[y==0, 0], X[y==0, 1], "bs")
plt.plot(X[y==1, 0], X[y==1, 1], "g^")
```

Übersicht

Kettenregel

Kettenregel

Seien $I \subset \mathbb{R}$ und $g : I \to \mathbb{R}$ und $f : g(I) \to \mathbb{R}$ differenzierbare Funktionen. Dann ist die Verkettung $f \circ g$ der Funkionen differenzierbar und es gilt für $x \in I$:

$$(f \circ g)'(x) = f'(g(x))g'(x)$$

Kettenregel

Kettenregel

Seien $I \subset \mathbb{R}$ und $g : I \to \mathbb{R}$ und $f : g(I) \to \mathbb{R}$ differenzierbare Funktionen. Dann ist die Verkettung $f \circ g$ der Funkionen differenzierbar und es gilt für $x \in I$:

$$(f \circ g)'(x) = f'(g(x))g'(x)$$

- Sei $g : \mathbb{R}^m \to \mathbb{R}$ partiell differenzierbar in $\mathbf{w}$ und $f : \mathbb{R} \to \mathbb{R}$ partiell differenzierbar in $g(\mathbf{w})$. Dann ist $f \circ g$ partiell differenzierbar in $\mathbf{w}$ mit:

$$\frac{\partial (f \circ g)}{\partial w_i}(\mathbf{w}) = f'(g(\mathbf{w}))\frac{\partial g}{\partial w_i}(\mathbf{w})$$

Somit gilt für den Gradienten: $\nabla(f \circ g)(\mathbf{w}) = f'(g(\mathbf{w}))\nabla g(\mathbf{w})$

Kettenregel

Kettenregel

Seien $I \subset \mathbb{R}$ und $g : I \to \mathbb{R}$ und $f : g(I) \to \mathbb{R}$ differenzierbare Funktionen. Dann ist die Verkettung $f \circ g$ der Funkionen differenzierbar und es gilt für $x \in I$:

$$(f \circ g)'(x) = f'(g(x))g'(x)$$

- Sei $g : \mathbb{R}^m \to \mathbb{R}$ partiell differenzierbar in $\mathbf{w}$ und $f : \mathbb{R} \to \mathbb{R}$ partiell differenzierbar in $g(\mathbf{w})$. Dann ist $f \circ g$ partiell differenzierbar in $\mathbf{w}$ mit:

$$\frac{\partial (f \circ g)}{\partial w_i}(\mathbf{w}) = f'(g(\mathbf{w}))\frac{\partial g}{\partial w_i}(\mathbf{w})$$

  Somit gilt für den Gradienten: $\nabla(f \circ g)(\mathbf{w}) = f'(g(\mathbf{w}))\nabla g(\mathbf{w})$
- Beispiel: Sei $g : \mathbb{R}^2 \to \mathbb{R}$ mit $g(w_1, w_2) = w_1^2 + w_2^2$ und $f : \mathbb{R} \to \mathbb{R}$ mit $f(x) = x^2$. Dann gilt $f \circ g(w_1, w_2) = f(g(w_1, w_2)) = (w_1^2 + w_2^2)^2$

Kettenregel

Kettenregel

Seien $I \subset \mathbb{R}$ und $g : I \to \mathbb{R}$ und $f : g(I) \to \mathbb{R}$ differenzierbare Funktionen. Dann ist die Verkettung $f \circ g$ der Funkionen differenzierbar und es gilt für $x \in I$:

$$(f \circ g)'(x) = f'(g(x))g'(x)$$

- Sei $g : \mathbb{R}^m \to \mathbb{R}$ partiell differenzierbar in $\mathbf{w}$ und $f : \mathbb{R} \to \mathbb{R}$ partiell differenzierbar in $g(\mathbf{w})$. Dann ist $f \circ g$ partiell differenzierbar in $\mathbf{w}$ mit:

$$\frac{\partial (f \circ g)}{\partial w_i}(\mathbf{w}) = f'(g(\mathbf{w}))\frac{\partial g}{\partial w_i}(\mathbf{w})$$

  Somit gilt für den Gradienten: $\nabla(f \circ g)(\mathbf{w}) = f'(g(\mathbf{w}))\nabla g(\mathbf{w})$

- Beispiel: Sei $g : \mathbb{R}^2 \to \mathbb{R}$ mit $g(w_1, w_2) = w_1^2 + w_2^2$ und $f : \mathbb{R} \to \mathbb{R}$ mit $f(x) = x^2$. Dann gilt $f \circ g(w_1, w_2) = f(g(w_1, w_2)) = (w_1^2 + w_2^2)^2$ und

$$\frac{\partial (f \circ g)}{\partial w_i}(w_1, w_2) = f'(g(w_1, w_2))\frac{\partial g}{\partial w_i}(w_1, w_2) = 2(w_1^2 + w_2^2)2w_i$$

Spaß (?) mit der Kettenregel ...

1. Sei $\sigma : \mathbb{R} \to [0,1]$ die Sigmoid-Funktion mit $\sigma(z) = \frac{1}{1+\exp(-z)}$.
   Zeige: $\sigma'(z) = \sigma(z)(1 - \sigma(z))$.
2. Sei $h : \mathbb{R}^{d+1} \to \mathbb{R}$ mit $h(\mathbf{w}) = y_i \log \pi_i = y_i \log \left(\frac{1}{1+\exp\left(-\mathbf{w}^\intercal \mathbf{x}_i\right)} \right)$.
   Wie lautet $\nabla h(\mathbf{w})$?

Anwendung der Kettenregel

Beispiel 1

- Sei $\sigma : \mathbb{R} \to [0, 1]$ die Sigmoid-Funktion $\sigma(z) = \frac{1}{1+\exp(-z)}$.

Anwendung der Kettenregel

Beispiel 1

- Sei $\sigma : \mathbb{R} \to [0,1]$ die Sigmoid-Funktion $\sigma(z) = \frac{1}{1+\exp(-z)}$.
- Durch die mehrfache Anwendung der Kettenregel erhalten wir:

$$\sigma'(z) = -\frac{1}{(1+\exp(-z))^2}\exp(-z)(-1)$$

Anwendung der Kettenregel

Beispiel 1

- Sei $\sigma : \mathbb{R} \to [0,1]$ die Sigmoid-Funktion $\sigma(z) = \frac{1}{1+\exp(-z)}$.
- Durch die mehrfache Anwendung der Kettenregel erhalten wir:

$$\begin{aligned}
\sigma'(z) &= -\frac{1}{(1+\exp(-z))^2}\exp(-z)(-1) \\
&= \frac{\exp(-z)}{(1+\exp(-z))^2}
\end{aligned}$$

Anwendung der Kettenregel

Beispiel 1

- Sei $\sigma : \mathbb{R} \to [0,1]$ die Sigmoid-Funktion $\sigma(z) = \frac{1}{1+\exp(-z)}$.
- Durch die mehrfache Anwendung der Kettenregel erhalten wir:

$$\begin{aligned}
\sigma'(z) &= -\frac{1}{(1+\exp(-z))^2}\exp(-z)(-1) \\
&= \frac{\exp(-z)}{(1+\exp(-z))^2} \\
&= \frac{1}{1+\exp(-z)}\left(\frac{1+\exp(-z)}{1+\exp(-z)} - \frac{1}{1+\exp(-z)}\right)
\end{aligned}$$

Anwendung der Kettenregel

Beispiel 1

- Sei $\sigma : \mathbb{R} \to [0,1]$ die Sigmoid-Funktion $\sigma(z) = \frac{1}{1+\exp(-z)}$.
- Durch die mehrfache Anwendung der Kettenregel erhalten wir:

$$
\begin{aligned}
\sigma'(z) &= -\frac{1}{(1+\exp(-z))^2}\exp(-z)(-1) \\
&= \frac{\exp(-z)}{(1+\exp(-z))^2} \\
&= \frac{1}{1+\exp(-z)}\left(\frac{1+\exp(-z)}{1+\exp(-z)} - \frac{1}{1+\exp(-z)}\right) \\
&= \sigma(z)(1-\sigma(z))
\end{aligned}
$$

Anwendung der Kettenregel

Beispiel 2

- Sei $h : \mathbb{R}^{d+1} \to \mathbb{R}$ mit $h(\mathbf{w}) = y_i \log \pi_i = y_i \log \left(\frac{1}{1+\exp\left(-\mathbf{w}^\top \mathbf{x}_i\right)} \right)$

Anwendung der Kettenregel

Beispiel 2

- Sei $h : \mathbb{R}^{d+1} \to \mathbb{R}$ mit $h(\mathbf{w}) = y_i \log \pi_i = y_i \log \left(\frac{1}{1+\exp\left(-\mathbf{w}^\mathsf{T}\mathbf{x}_i\right)} \right)$
- Dann gilt $h = f \circ g$ mit $f(z) = y_i \log \sigma(z)$ und $g(\mathbf{w}) = \mathbf{w}^\mathsf{T}\mathbf{x}_i$.

Anwendung der Kettenregel

Beispiel 2

- Sei $h : \mathbb{R}^{d+1} \to \mathbb{R}$ mit $h(\mathbf{w}) = y_i \log \pi_i = y_i \log \left(\frac{1}{1+\exp(-\mathbf{w}^\mathsf{T}\mathbf{x}_i)} \right)$
- Dann gilt $h = f \circ g$ mit $f(z) = y_i \log \sigma(z)$ und $g(\mathbf{w}) = \mathbf{w}^\mathsf{T}\mathbf{x}_i$. Daher:

$$\nabla h(\mathbf{w}) = f'(g(\mathbf{w}))\nabla g(\mathbf{w})$$

Anwendung der Kettenregel

Beispiel 2

- Sei $h : \mathbb{R}^{d+1} \to \mathbb{R}$ mit $h(\mathbf{w}) = y_i \log \pi_i = y_i \log \left(\frac{1}{1+\exp(-\mathbf{w}^\mathsf{T} \mathbf{x}_i)} \right)$
- Dann gilt $h = f \circ g$ mit $f(z) = y_i \log \sigma(z)$ und $g(\mathbf{w}) = \mathbf{w}^\mathsf{T} \mathbf{x}_i$. Daher:

$$\begin{aligned} \nabla h(\mathbf{w}) &= f'(g(\mathbf{w})) \nabla g(\mathbf{w}) \\ &= \frac{y_i}{\sigma(g(\mathbf{w}))} \sigma'(g(\mathbf{w})) \mathbf{x}_i \end{aligned}$$

Anwendung der Kettenregel

Beispiel 2

- Sei $h : \mathbb{R}^{d+1} \to \mathbb{R}$ mit $h(\mathbf{w}) = y_i \log \pi_i = y_i \log \left(\frac{1}{1+\exp(-\mathbf{w}^\mathsf{T}\mathbf{x}_i)} \right)$
- Dann gilt $h = f \circ g$ mit $f(z) = y_i \log \sigma(z)$ und $g(\mathbf{w}) = \mathbf{w}^\mathsf{T}\mathbf{x}_i$. Daher:

$$
\begin{aligned}
\nabla h(\mathbf{w}) &= f'(g(\mathbf{w}))\nabla g(\mathbf{w}) \\
&= \frac{y_i}{\sigma(g(\mathbf{w}))}\sigma'(g(\mathbf{w}))\mathbf{x}_i \\
&= \frac{y_i}{\sigma(g(\mathbf{w}))}\sigma(g(\mathbf{w}))(1-\sigma(g(\mathbf{w})))\mathbf{x}_i
\end{aligned}
$$

Anwendung der Kettenregel

Beispiel 2

- Sei $h : \mathbb{R}^{d+1} \to \mathbb{R}$ mit $h(\mathbf{w}) = y_i \log \pi_i = y_i \log \left(\frac{1}{1+\exp(-\mathbf{w}^\mathsf{T}\mathbf{x}_i)} \right)$
- Dann gilt $h = f \circ g$ mit $f(z) = y_i \log \sigma(z)$ und $g(\mathbf{w}) = \mathbf{w}^\mathsf{T}\mathbf{x}_i$. Daher:

$$\begin{aligned}
\nabla h(\mathbf{w}) &= f'(g(\mathbf{w}))\nabla g(\mathbf{w}) \\
&= \frac{y_i}{\sigma(g(\mathbf{w}))}\sigma'(g(\mathbf{w}))\mathbf{x}_i \\
&= \frac{y_i}{\sigma(g(\mathbf{w}))}\sigma(g(\mathbf{w}))(1-\sigma(g(\mathbf{w})))\mathbf{x}_i \\
&= y_i(1-\sigma(g(\mathbf{w})))\mathbf{x}_i
\end{aligned}$$

Anwendung der Kettenregel

Beispiel 2

- Sei $h : \mathbb{R}^{d+1} \to \mathbb{R}$ mit $h(\mathbf{w}) = y_i \log \pi_i = y_i \log \left(\frac{1}{1+\exp(-\mathbf{w}^\mathsf{T} \mathbf{x}_i)} \right)$
- Dann gilt $h = f \circ g$ mit $f(z) = y_i \log \sigma(z)$ und $g(\mathbf{w}) = \mathbf{w}^\mathsf{T} \mathbf{x}_i$. Daher:

$$
\begin{aligned}
\nabla h(\mathbf{w}) &= f'(g(\mathbf{w})) \nabla g(\mathbf{w}) \\
&= \frac{y_i}{\sigma(g(\mathbf{w}))} \sigma'(g(\mathbf{w})) \mathbf{x}_i \\
&= \frac{y_i}{\sigma(g(\mathbf{w}))} \sigma(g(\mathbf{w}))(1 - \sigma(g(\mathbf{w}))) \mathbf{x}_i \\
&= y_i (1 - \sigma(g(\mathbf{w}))) \mathbf{x}_i \\
&= y_i (1 - \pi_i) \mathbf{x}_i
\end{aligned}
$$

Anwendung der Kettenregel

Beispiel 2

- Sei $h : \mathbb{R}^{d+1} \to \mathbb{R}$ mit $h(\mathbf{w}) = y_i \log \pi_i = y_i \log \left(\frac{1}{1+\exp(-\mathbf{w}^\mathsf{T}\mathbf{x}_i)} \right)$
- Dann gilt $h = f \circ g$ mit $f(z) = y_i \log \sigma(z)$ und $g(\mathbf{w}) = \mathbf{w}^\mathsf{T}\mathbf{x}_i$. Daher:

$$\begin{aligned}
\nabla h(\mathbf{w}) &= f'(g(\mathbf{w}))\nabla g(\mathbf{w}) \\
&= \frac{y_i}{\sigma(g(\mathbf{w}))}\sigma'(g(\mathbf{w}))\mathbf{x}_i \\
&= \frac{y_i}{\sigma(g(\mathbf{w}))}\sigma(g(\mathbf{w}))(1-\sigma(g(\mathbf{w})))\mathbf{x}_i \\
&= y_i(1-\sigma(g(\mathbf{w})))\mathbf{x}_i \\
&= y_i(1-\pi_i)\mathbf{x}_i
\end{aligned}$$

- Für $\bar{h} : \mathbb{R}^{d+1} \to \mathbb{R}$ mit $\bar{h}(\mathbf{w}) = (1-y_i)\log(1-\pi_i)$ erhält man analog:

$$\nabla \bar{h}(\mathbf{w}) = -(1-y_i)\pi_i\mathbf{x}_i$$

Bestimmung des Gradienten (binäre Klassifikation)

$$\nabla G(\mathbf{w}) \quad = \quad -\frac{1}{n}\sum_{i=1}^{n}(\nabla y_i \log \pi_i) + (\nabla(1-y_i)\log(1-\pi_i))$$

Bestimmung des Gradienten (binäre Klassifikation)

$$
\begin{aligned}
\nabla G(\mathbf{w}) &= -\frac{1}{n}\sum_{i=1}^{n}(\nabla y_i \log \pi_i) + (\nabla(1-y_i)\log(1-\pi_i)) \\
&= -\frac{1}{n}\sum_{i=1}^{n} y_i(1-\pi_i)\mathbf{x}_i - (1-y_i)\pi_i\mathbf{x}_i
\end{aligned}
$$

Bestimmung des Gradienten (binäre Klassifikation)

$$
\begin{aligned}
\nabla G(\mathbf{w}) &= -\frac{1}{n}\sum_{i=1}^{n}(\nabla y_i \log \pi_i) + (\nabla(1-y_i)\log(1-\pi_i)) \\
&= -\frac{1}{n}\sum_{i=1}^{n} y_i(1-\pi_i)\mathbf{x}_i - (1-y_i)\pi_i\mathbf{x}_i \\
&= -\frac{1}{n}\sum_{i=1}^{n}(y_i - y_i\pi_i - \pi_i + y_i\pi_i)\mathbf{x}_i
\end{aligned}
$$

Bestimmung des Gradienten (binäre Klassifikation)

$$
\begin{aligned}
\nabla G(\mathbf{w}) &= -\frac{1}{n}\sum_{i=1}^{n}\left(\nabla y_i \log \pi_i\right) + \left(\nabla(1-y_i)\log(1-\pi_i)\right) \\
&= -\frac{1}{n}\sum_{i=1}^{n} y_i(1-\pi_i)\mathbf{x}_i - (1-y_i)\pi_i\mathbf{x}_i \\
&= -\frac{1}{n}\sum_{i=1}^{n}(y_i - y_i\pi_i - \pi_i + y_i\pi_i)\mathbf{x}_i \\
&= \frac{1}{n}\sum_{i=1}^{n}(\pi_i - y_i)\mathbf{x}_i
\end{aligned}
$$

Logistische Regression: Gradientenverfahren

Logistic Regression (batch gradient descent)

Require: Training set $T = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\} \subset \mathbb{R}^d \times \{0, 1\}$ and learning rate $\eta > 0$.
Ensure: Weights $\mathbf{w}$ for the model $f(\mathbf{x}; \mathbf{w}) = \sigma(-\mathbf{w}^\mathsf{T}\mathbf{x})$

```
// generate augmented data matrix (prepend column of ones)
// ...
// small random values (e.g., normally distributed)
Initialize w ∈ ℝ^{d+1}
repeat
    // gradient of the objective (based on all training instances)
    Compute ∇G(w) = 1/n Σ_{i=1}^{n} (π_i − y_i) x_i = 1/n Σ_{i=1}^{n} ( 1/(1+exp(−wᵀx_i)) − y_i ) x_i
    // model parameter update
    w ← w − η∇G(w)
until stopping criterion is met
```

$$\nabla G(\mathbf{w}) = \frac{1}{n}\sum_{i=1}^{n}(\pi_i - y_i)\mathbf{x}_i = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{1+\exp\left(-\mathbf{w}^\mathsf{T}\mathbf{x}_i\right)} - y_i\right)\mathbf{x}_i$$

$$\mathbf{w} \leftarrow \mathbf{w} - \eta\nabla G(\mathbf{w})$$

Logistische Regression: Implementation in Python

jupyter Logistic Regression Implementation Last Checkpoint: a few seconds ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help Trusted Python 3

Run Code

In [1]:
```
import numpy
from sklearn import datasets
import matplotlib.pyplot as plt
```

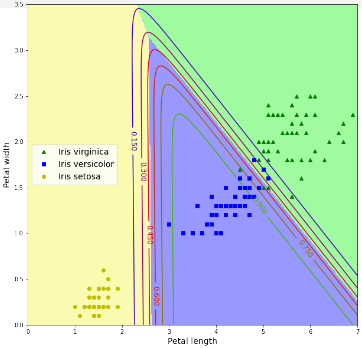
In [2]:
```
iris = datasets.load_iris()
# Consider two features
X = iris["data"][:, (2, 3)]  # petal length, petal width
y = (iris["target"] == 2).astype(numpy.int)
```

A gradient descent implementation for Logistic Regression

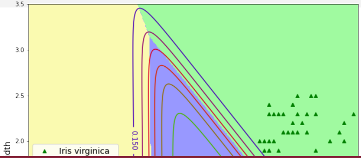
In [3]:
```
from sklearn.base import BaseEstimator, ClassifierMixin

class GradientDescentLogisticRegression(BaseEstimator, ClassifierMixin):
    """
    A Logistic Regression classifier implementation using
    gradient descent.

    Parameters
    ----------
    lr : float, default 0.0001
        The learning rate used for gradient descent.
    seed : int, defeault 0
        The seed used to initialize the random number
        generator
    verbose : int, defeault 0
        If verbose > 0, then log messages are generated.
    """

    def __init__(self, lr=0.0001, seed=0, verbose=0):

        self.lr = lr
        self.seed = seed
        self.verbose = verbose

    def fit(self, X, y, batch_size=None, n_epochs=1000):
        """
        Fits the logistic regression model

        Parameters
        ----------
        X : array-like of shape (n_samples, n_features)
            Matrix containing the training samples
```

Übersicht

Multinomiale Logistische Regression



Multinomiale Logistische Regression



Softmax-Regression: Modell

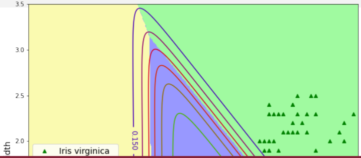
- Trainingsdatensatz $T = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\} \subset \mathbb{R}^d \times \{1, \ldots, K\}$
- Erstelle für jede Klasse k ein Modell $s_k(\mathbf{x}) = \left(\mathbf{w}^{(k)}\right)^\mathsf{T} \mathbf{x}$ mit $\mathbf{w}^{(k)} \in \mathbb{R}^{d+1}$
- Die Werte (*scores*) der einzelnen Modelle werden durch die sogenannte Softmax-Funktion zu Wahrscheinlichkeiten $\hat{p}_1, \ldots, \hat{p}_K$ transformiert:

$$\hat{p}_k(\mathbf{x}) = \frac{\exp(s_k(\mathbf{x}))}{\sum_{j=1}^{K} \exp(s_j(\mathbf{x}))} \in [0, 1]$$

- Klassifikation: Diejenige Klasse mit der höchsten (geschätzten) Wahrscheinlichkeit:

$$\hat{y}(\mathbf{x}) = \underset{k}{\operatorname{argmax}}\, \hat{p}_k(\mathbf{x}) = \underset{k}{\operatorname{argmax}}\, s_k(\mathbf{x}) = \underset{k}{\operatorname{argmax}} \left(\left(\mathbf{w}^{(k)}\right)^\mathsf{T} \mathbf{x} \right)$$

Multinomiale Logistische Regression



Softmax-Regression: Training

- Bestimmung der Qualität eines Modells $(\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(K)})$ auf Basis der Kreuzentropie (*cross entropy*):

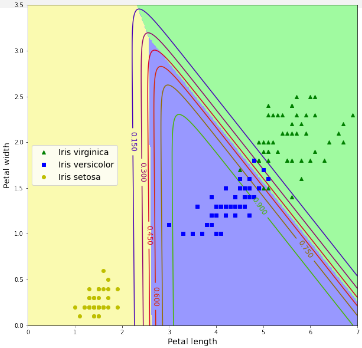
$$J(\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(K)}) = -\frac{1}{n} \sum_{i=1}^{n} \sum_{k=1}^{K} y_i^{(k)} \log(\hat{p}_k(\mathbf{x}_i))$$

  Hierbei gibt $y_i^{(k)}$ die Wahrscheinlichkeit an, dass $\mathbf{x}_i$ zur Klasse k gehört.
  (z.B. $y_i^{(1)} = 0$, $y_i^{(2)} = 0$, $y_i^{(3)} = 1$, $y_i^{(4)} = 0$)

- Bestimmung der Modellparameter mittels des Gradientenverfahrens …

Übersicht

Heute



- (Multinomiale) Logistische Regression, Sigmoid, Softmax, ...
- Gradientverfahren, Kettenregel, Python-Implementation, ...

Literatur I