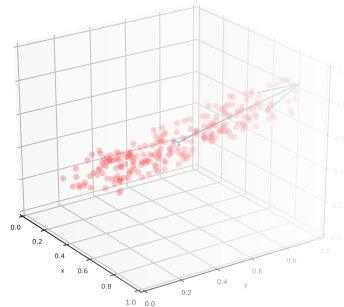
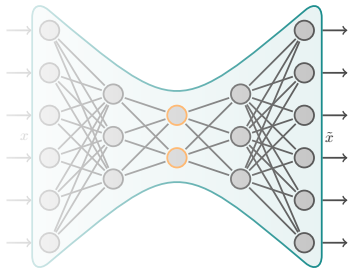
Principal Component Analysis und Autoencoder

Prof. Dr. Jörg Frochte

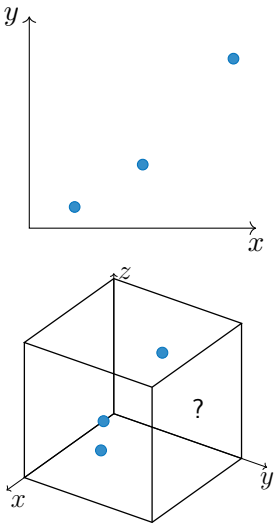
Maschinelles Lernen





Wdh. Fluch der Dimensionalität

- Für viele maschinelle Lernverfahren ist ein großer Merkmalsraum problematisch.
- Als **Fluch der Dimensionalität** bzw. **Curse of Dimensionality** bezeichnet man das Problem, wenn die Dimension des Merkmalsraums nicht zu der Datenmenge passt.
- Die Anzahl Parameter einer Hyperebene, um z. B. eine Regression durchzuführen, beträgt in 2D zwei, in 3D drei usw. Es würden ebensoviele Daten benötigt um die Parameter eindeutig zu bestimmen.
- Wenn man von einem nicht-linearen Modell ausgeht, benötigt man wesentlich mehr Daten um ein gutes Modell zu erhalten. Mit vielen Dimensionen müsste man für alle möglichen Kombinationen von Werten Daten haben.

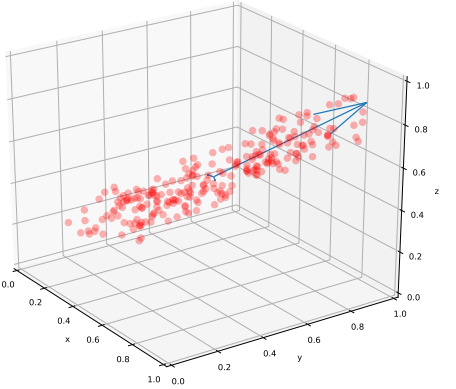


Motivation

- Bis jetzt haben wir unverändert Merkmale benutzt oder aus einer Menge von Merkmalen ausgewählt.
- Unser nächster Schritt geht dahin Merkmale zusammenzuführen und dabei möglichst viel Information zu erhalten.
- Das Verfahren, welches wir heute betrachten ist die **Hauptkomponentenanalyse** bzw. **Principal Component Analysis** (PCA)
  - Es handelt sich um einen Ansatz über eine Projektion von einem großen Raum in einen kleineren Raum die Dimension des Merkmalsraums zu reduzieren.
  - Der größere Raum wird durch die Merkmale aufgespannt, der kleinere durch eine neue Basis.
- Eine andere Kategorie, die heute nicht dran kommt sind **Autoencoder**.
  - Dies sind Neuronale Netze mit einer bestimmten Architektur, die als Zielwerte die Eingangswerte lernen.

Principal Component Analysis

- Die Idee bei der PCA liegt darin, anhand von Korrelationen zwischen Merkmalen Muster in den Daten zu entdecken und zu nutzen.
- **Ziel**: Richtungen mit maximaler Varianz in einem hochdimensionalen Raum zu finden und diese besonderen Richtungen als Grundlage für eine Projektion zu verwenden.
- In der Abb. haben wir eine Menge, die am stärksten eine Varianz in Richtung w_1 aufweist.



- Nun gilt es, hierzu weitere Basisvektoren w_2 und w_3 zu finden.
- Diese drei Vektoren sollen alle orthogonal sein.
- Die neuen Vektoren zeigen jeweils in die Richtung der verbliebenen max. Varianz.

Principal Component Analysis

- Die Reihenfolge gibt dabei an, wie stark die Varianz in dieser Richtung im Vergleich zu den anderen ist.
- Wenn wir also nur einen zweidimensionalen Raum wünschen, werden wir w_1 und w_2 als Basis nutzen und w_3 entfallen lassen.
- Um auf diesen von w_1 und w_2 aufgespannten Untervektorraum zu kommen, werden wir eine Projektionsmatrix W_p konstruieren.
- Die Annahme für diesen Ansatz lautet dabei, dass die Richtungen mit der größten Varianz in X auch die meiste Information bzgl. y beinhaltet.
- Das ist oft richtig, aber trotzdem eine Annahme, die auch falsch sein kann.
- In den nächsten Seiten greifen wir stark auf die lineare Algebra und dabei das Wissen über Projektionen zurück.

Merkmale zusammenlegen, ist nichts ungewöhnliches . . .

- Nehmen wir an, es geht um z. B. Kreuzfahrtschiffe.
- Auf `https://de.wikipedia.org/wiki/Liste_von_Kreuzfahrtschiffen` sind u. a. diverse Daten zu Kreuzfahrtschiffen aufgeführt, die wir vermutlich nutzen würden, wenn es um eine entsprechende Fragestellung ginge.

- Zu den Daten dieser Schiffe gehören u. a. Länge, Breite, Bruttoraumzahl und Kabinenanzahl.
- Sicherlich lässt sich z. B. die Information über die *Schiffgröße* auch komprimierter vermitteln.
- In gewisser Weise generiert die PCA eine solches komprimiertes Merkmal für uns.

Kreuzfahrtschiff vor Funchal 2015 (Quelle: eigene Aufnahme)

Mathematische Herleitung und Motivation der PCA

- Die Hauptkomponente w_1 soll diejenige sein, entlang der unsere Datenmenge am meisten ausgedehnt ist.
- Es geht nur um die Richtung. Wir legen fest, dass w_1 normiert sein soll, d. h. $\|w_1\| = 1$.

Zur Erinnerung:

Wenn wir einen Vektor x auf ein normiertes w projizieren wollen erhalten wir die Projektion z durch die Gleichung

$$z = w \cdot x = w^\top x \ .$$

z war dabei nur die Länge der Projektion in Richtung w. Ist ein Vektor inklusive Richtung gewünscht, muss man $z\,w$ nutzen.

- Nun benötigen wir noch ein weiteres Hilfsmittel, die **Kovarianzmatrix**.
- Die Kovarianzmatrix ist wie folgt definiert:

$$\Sigma = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1n} \\ \sigma_{21} & \sigma_2^2 & \cdots & \sigma_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{n1} & \sigma_{n2} & \cdots & \sigma_n^2 \end{pmatrix}$$

- Hierbei ist $\sigma_{ij} = \mathrm{Cov}(x_i, x_j)$ wieder die Kovarianz der Merkmale x_i und x_j.
- **Achtung:** Der griechischen Buchstaben Σ ist nicht im Sinne von Summe zu verstehen, sondern ein Symbol für die Matrix.
- Damit enthält die Kovarianzmatrix alle paarweisen Kovarianzen der Merkmale und somit die Informationen über Streuungen und Korrelationen.

- Σ ist eng verwandt mit der bekannten Korrelationsmatrix

$$P = \begin{pmatrix} \rho_{11} & \rho_{12} & \cdots & \rho_{1n} \\ \rho_{21} & \rho_{22} & \cdots & \rho_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{n1} & \rho_{n2} & \cdots & \rho_{nn} \end{pmatrix} = \begin{pmatrix} 1 & \rho_{12} & \cdots & \rho_{1n} \\ \rho_{21} & 1 & \cdots & \rho_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{n1} & \rho_{n2} & \cdots & 1 \end{pmatrix} \text{ mit } \rho_{ij} = \frac{\mathrm{Cov}(x_i, x_j)}{\sqrt{\sigma_i^2 \cdot \sigma_j^2}} = \frac{\sigma_{ij}}{\sqrt{\sigma_i^2 \cdot \sigma_j^2}}$$

- Beide unterscheiden i. W. durch eine Skalierung. Definiert man die Diagonalmatrix $D = \mathrm{diag}(\sigma_{11}, \sigma_{22}, \ldots, \sigma_{nn})$, erhält man p durch Σ und umgekehrt:

$$P = D^{-1} \cdot \Sigma \cdot D^{-1} \qquad \Sigma = D \cdot P \cdot D$$

- Das macht die beiden zu kongruenten Matrizen. In der linearen Algebra nennt man zwei quadratische Matrizen P und Σ so, wenn es eine invertierbare Matrix R gibt, sodass gilt $\Sigma = R^T \cdot P \cdot R$

- Da für eine Diagonalmatrix $D = D^T$ gilt, ist dies erfüllt, wenn alle $\sigma_{ii} \neq 0$ sind.
- Darüber hinaus sind die beiden Matrizen symmetrische Matrizen.

Wichtig:

Im Falls von Daten, bei denen alle Merkmale standardisiert wurden, also $\sigma_i = 1$ gilt:

$$P = \Sigma$$

- Die Varianz $\mathrm{Var}(w)$ kann durch

$$\mathrm{Var}(w) = w^\top \Sigma w$$

  berechnet werden.
- Wir verzichten hier auf einen formalen Beweis, gehen jedoch Beispiele durch.

Beispiel 1

Nehmen wir zunächst an, wir hätten $w = (1, 0, 0)^\top$:

$$\begin{pmatrix} 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_2^2 & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_3^2 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix} \begin{pmatrix} \sigma_1^2 \\ \sigma_{21} \\ \sigma_{31} \end{pmatrix} = \sigma_1^2$$

Beispiel 1

Nun verwenden wir eine Linearkombination aus den ersten beiden Merkmalen $w = (1/\sqrt{2}, 1/\sqrt{2}, 0)^\top$:

$$\begin{aligned} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} & 0 \end{pmatrix} \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \sigma_{13} \\ \sigma_{21} & \sigma_2^2 & \sigma_{23} \\ \sigma_{31} & \sigma_{32} & \sigma_3^2 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \\ 0 \end{pmatrix} &= \frac{1}{2} \begin{pmatrix} 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} \sigma_1^2 + \sigma_{12} \\ \sigma_{21} + \sigma_2^2 \\ \sigma_{31} + \sigma_{32} \end{pmatrix} \\ &= \frac{\sigma_1^2 + \sigma_{12} + \sigma_{21} + \sigma_2^2}{2} \end{aligned}$$

Wie erwartet, erhalten wir dann eine Mittelung der Varianzen der beteiligten Merkmale.

- Wir suchen nach der Richtung, in der die Varianz maximal ist.
- Das ist ein Optimierungsproblem und lässt sich wie folgt notieren:

$$\arg\max_{w_1} \underbrace{w_1^\top \Sigma w_1}_{(*)} - \underbrace{\lambda(w_1 \cdot w_1 - 1)}_{(**)} \tag{1}$$

- Der Term $(*)$ ist die Varianz, die wir maximieren wollen.
- $(**)$ ist eine Lagrange-Zwangsbedingung, die bewirkt, dass w_1 die Norm 1 hat.
- λ ist dabei der aus Analysis 2 bekannte Lagrange-Parameter.
- Wir gehen über die notwendige Bedingung und differenzieren (1) nach w_1.
- Indem wir die Ableitung gleich dem Nullvektor setzen, erhalten wir als Bedingung:

$$2\Sigma w_1 - 2\lambda w_1 = 0$$

- Die 2 kann gekürzt werden, und dann bringen wir λw_1 auf die rechte Seite.

$$\Sigma w_1 = \lambda w_1$$

$$\Sigma w_1 = \lambda w_1$$

- Wenn w_1 auf Σ angewendet wird, wird es lediglich in seiner Länge verändert.
- Das ist die Definition eines Eigenvektors, weshalb also w_1 ein Eigenvektor von Σ ist und λ ein Eigenwert.
- Σ ist eine symmetrische Matrix ist, weshalb es nur reelle Eigenwerte besitzt.
- Diese Bedingung, dass jede Lösung ein Eigenvektor sein muss, können wir nun in (1) nutzen:

$$w_1^\top \underbrace{\Sigma w_1}_{=\lambda w_1} = w_1^\top \lambda w_1 = \lambda \underbrace{w_1^\top w_1}_{=1} = \lambda \tag{2}$$

- Das bedeutet, dass der Term, den wir maximieren wollen, eben dann maximal wird, wenn wir den Eigenvektor mit dem größten Eigenwert wählen.

- Seien $\lambda_1 > \lambda_2 > \ldots$ die nach der Größe sortierten Eigenwerte von Σ, so wählen wir also $\lambda = \lambda_1$.
- Dies ist nun die erste Hauptkomponente, also mit der größten Varianz.
- w_2 soll nun die verbleibende Varianz maximieren und auf Eins normiert sein.
- Darüber hinaus fordern wir, dass w_2 senkrecht auf w_1 steht.
- Diese Bedingung bewirkt dann, dass unsere neuen Merkmalsachsen nicht mehr miteinander korrelieren.
- Daher erhalten wir nun für die zweite Achse eine etwas komplexere Bedingung:

$$\arg\max_{w_2} w_2^\top \Sigma w_2 - \lambda(w_2^\top w_2 - 1) - \beta(w_2^\top w_1 - 0) \tag{3}$$

- Der dritte Term fordert also nun als Nebenbedingung $w_2^\top w_1 = 0$.

$$\arg\max_{w_2} w_2^\top \Sigma w_2 - \lambda(w_2^\top w_2 - 1) - \beta(w_2^\top w_1 - 0)$$

- Wieder differenzieren wir diese Gleichung – nun eben nach w_2 – um die notwendige Bedingung für einen Extremwert zu bekommen:

$$2\Sigma w_2 - 2\lambda w_2 - \beta w_1 = 0 \tag{4}$$

- Wir verwenden das w_1 und w_2 orthogonal sein sollen.
- Hierzu multiplizieren wir die Gleichung (4) von links mit $w_1^\top$ und erhalten:

$$2w_1^\top \Sigma w_2 - 2\lambda w_1^\top w_2 - \beta w_1^\top w_1 = 0$$

- Hier fällt nun viel weg, da zum einen $w_1^\top w_1 = 1$ wegen der Normierung gilt und $w_1^\top w_2 = 0$, da beide Vektoren senkrecht aufeinander stehen sollen.

$$2w_1^\top \Sigma w_2 - \beta = 0 \tag{5}$$

- Der Term $w_1^\top \Sigma w_2$ als Ganzes ist nichts Anderes als ein Skalar.
- Skalare dürfen einfach transponiert werden, ohne dass sich etwas ändert.
- Es gilt also:
$$w_1^\top \Sigma w_2 = (w_1^\top \Sigma w_2)^\top = w_2^\top \Sigma w_1$$
- Da w_1 aber ein Eigenvektor ist, gilt $\Sigma w_1 = \lambda_1 w_1$ und somit wegen der geforderten Orthogonalität wieder
$$w_2^\top \Sigma w_1 = w_2^\top \lambda_1 w_1 = \lambda_1 w_2^\top w_1 = 0$$
- Setzen wir dies in (5) ein, vereinfacht sich das zu
$$\beta = 0$$
und wir können (4) vereinfachen zu
$$2\Sigma w_2 - 2\lambda w_2 = 0 \quad \Longleftrightarrow \quad \Sigma w_2 = \lambda w_2 \tag{6}$$
- Das bedeutet, dass w_2 der Eigenvektor von Σ mit dem zweitgrößten Eigenwert sein sollte, und wir setzen also $\lambda = \lambda_2$.

- Dies läuft für die restlichen Hauptkomponenten analog.
- Die anderen Achsen sind dann durch die Eigenvektoren entsprechend der absteigenden Eigenwerte gegeben.
- Da Σ eine symmetrische Matrix ist, gilt, dass die Eigenvektoren w_i, w_j zu zwei verschiedenen Eigenwerten einer reellen symmetrischen Matrix immer orthogonal sind.
- Es ist jedoch nicht automatisch klar, dass unsere Matrix regulär ist.
- Ist Σ nicht regulär, sondern singulär, so erhalten wir nur $l < n$ unterschiedliche Eigenwerte.
- Diese l Achsen bieten dann einen verkleinerten Raum, den wir als neue Basis nutzen können und das sogar in der Theorie ohne Informationsverlust.
- In der Praxis nimmt man oft die ersten k Hauptkomponenten, bis zu denen große Eigenwerte auftreten, und lässt die kleinen freiwillig entfallen.

Kochrezept für die PCA

1. Im Zweifel die Daten bzgl. aller Merkmale standardisieren bzw. so aufbereiten wie für Einsatz sonst sinnvoll.
2. Kovarianzmatrix (gleich Korrelationsmatrix nach Standardisierung) aufstellen.
3. Eigenwerte und Eigenvektoren der Kovarianzmatrix berechnen.
4. Abhängig von den Vorgaben bzw. der Entwicklung der Eigenwerte die k Eigenvektoren mit den größten Eigenwerten auswählen.
5. Aus diesen k Eigenvektoren die Projektionsmatrix W_P konstruieren.
6. Die Daten X mit der Projektionsmatrix W_P projizieren.

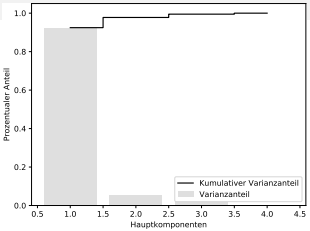
Umsetzung am Iris-Beispiel

```
1  import numpy as np
2  import matplotlib.pyplot as plt
3  from CARTDecisionTree import bDecisionTree
4
5  dataset = np.loadtxt("iris.csv", delimiter=",")
6  X = dataset[:,0:4]
7  Y = dataset[:,4]
8  skale = False
9  if skale: X[:,0] = 100*X[:,0]
10
11 Sigma = np.cov(X.T)
12 (lamb, W) = np.linalg.eig(Sigma)
13 eigenVar = np.sort(lamb)[::-1]
```

Varianzaufklärung

```
14 sumEig = np.sum(lamb)
15 eigenVar = eigenVar/sumEig
16 cumVar= np.cumsum(eigenVar)
17 plt.figure()
18 plt.bar(range(1,len(eigenVar)+1),eigenVar, alpha=0.25, align='center',
19         label='Varianzanteil', color='gray')
20 plt.step(range(1,len(eigenVar)+1),cumVar, where='mid',
21         label='Kumulativer Varianzanteil', c='k')
22 plt.xlabel('Hauptkomponenten'); plt.ylabel('Prozentualer Anteil')
23 plt.legend()
```

- Schon mit den beiden ersten Hauptachsen reicht man über 97% Varianzaufklärung.
- Dabei leistet allein der erste Vektor ca. 90%.
- Man kann folglich mit 2 statt 4 Merkmalen arbeiten.

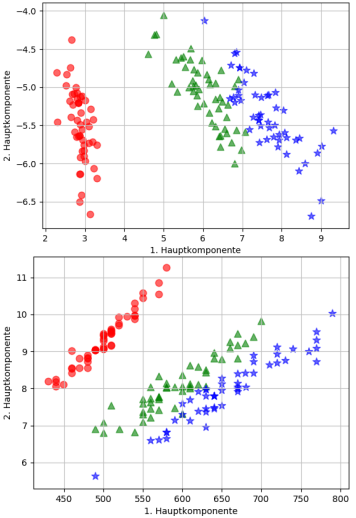


Eigene Implementierung der PCA für Autopreis-Regression

- Nun muss nur noch auf den verkleinerten Merkmalsraum projiziert werden:

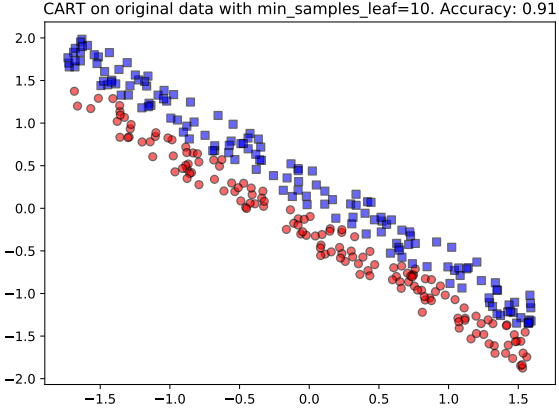
```
24
25 eigenVarIndex = np.argsort(lamb)[::-1]
26 WP = W[:,eigenVarIndex[0:2]]
27 XProj = ( WP.T@X.T ).T
```

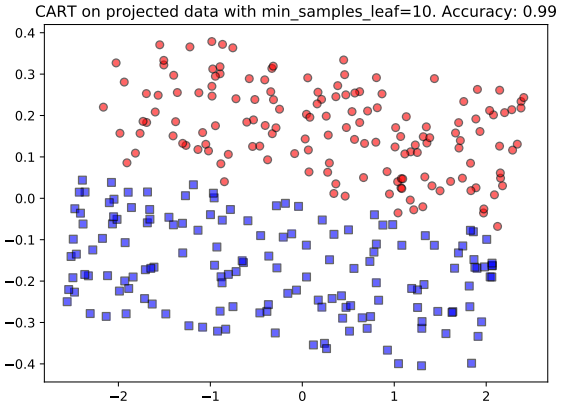
- Anschließend könnten Sie jedes Verfahren eben auf diesen Merkmalsraum anwenden.
- Genauigkeit vergleichbar zu vollen Merkmalsraum
- Skalierung/Standardisierung von Daten hat eine Auswirkung auf die PCA
- **Achtung:** Auch wenn die Daten anfangs standardisiert wurden, ist es die Projektion i.d.R. nicht mehr



PCA als Vorverarbeitung ohne Kompression

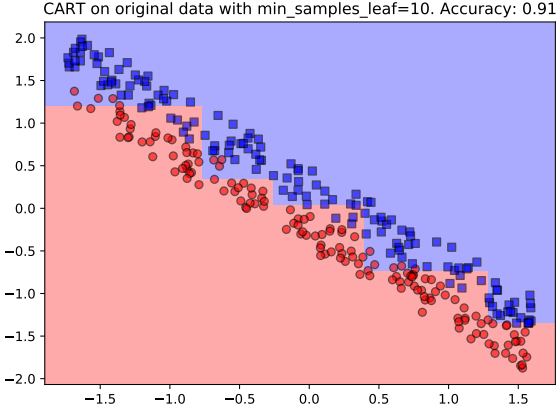
- PCA muss nicht zur Kompression eingesetzt werden.
- Insbesondere Verfahren mit achsparallelen Schnitten (CART, Random Forest etc.) können von der PCA als Vorverarbeitung profitieren, müssen aber nicht:

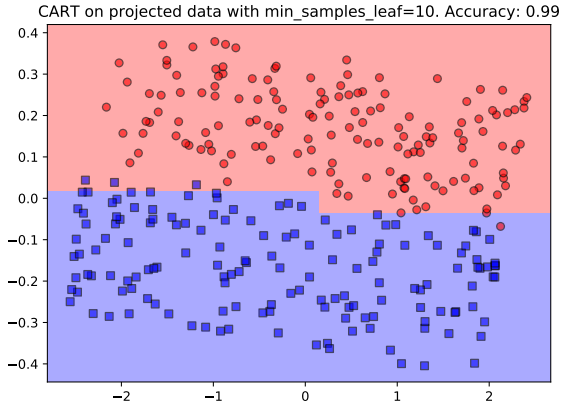




PCA als Vorverarbeitung ohne Kompression

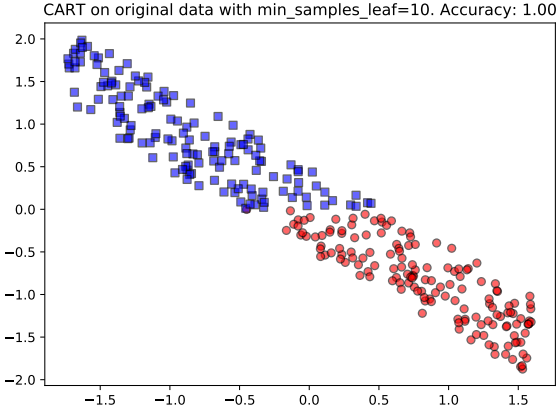
- PCA muss nicht zur Kompression eingesetzt werden.
- Insbesondere Verfahren mit achsparallelen Schnitten (CART, Random Forest etc.) können von der PCA als Vorverarbeitung profitieren, müssen aber nicht:

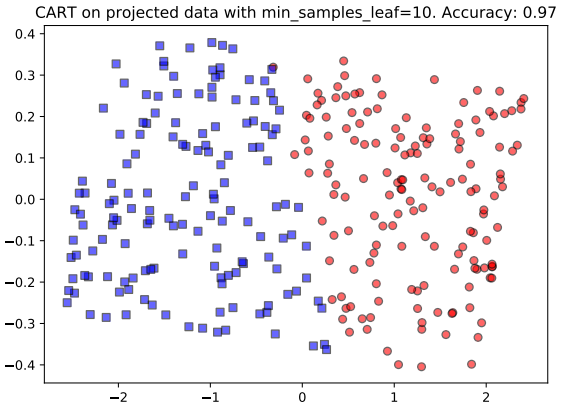




PCA als Vorverarbeitung ohne Kompression

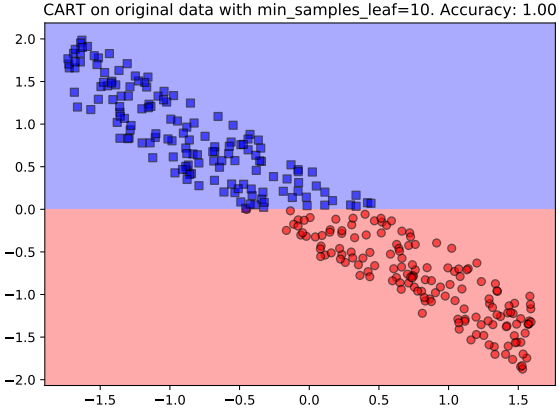
- PCA muss nicht zur Kompression eingesetzt werden.
- Insbesondere Verfahren mit achsparallelen Schnitten (CART, Random Forest etc.) können von der PCA als Vorverarbeitung profitieren, müssen aber nicht:

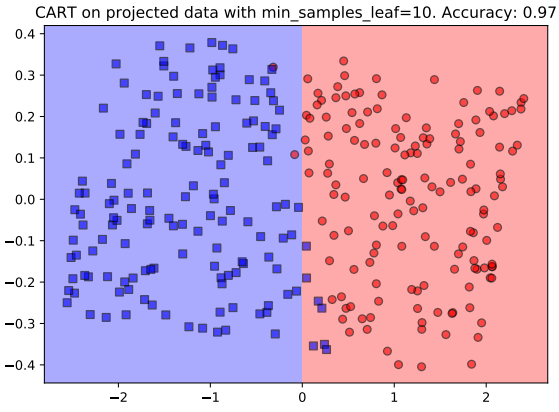




PCA als Vorverarbeitung ohne Kompression

- PCA muss nicht zur Kompression eingesetzt werden.
- Insbesondere Verfahren mit achsparallelen Schnitten (CART, Random Forest etc.) können von der PCA als Vorverarbeitung profitieren, müssen aber nicht:





Ausblick

Es gibt noch weitere Dimensionsreduktionstechniken, hier eine Auswahl:

- sklearn unterstützt diese klassische PCA auch direkt
- sklearn bietet sogenannte Pipelines für Vorverarbeitungen (ggf. einen Blick wert).
- Für die PCA funktioniert der Kernel-Trick, der im Buch für SVM erklärt wird und mehr Möglichkeiten eröffnet.
- Isomap Embedding ist ein graphbasierter Algorithmus, der versucht die Distanz (im Graphen) zu seinen k nächsten Nachbarn im reduzierten Raum zu bewahren. In scikit-learn: `sklearn.manifold.Isomap`
- t-distributed Stochastic Neighbor Embedding eignet sich gut um hochdimensionalen Daten zu visualisieren. In scikit-learn: `sklearn.manifold.TSNE`
- Lineare Diskriminantenanalyse (LDA) berücksichtigt die Klassenlabels Y. Die Anzahl der projizierten Dimensionen muss kleiner als die Anzahl Klassen sein! In scikit-learn: `sklearn.discriminant_analysis.LinearDiscriminantAnalysis`