

# Data Science 1

## Übungsblatt 7

### Aufgabe 1 (Klassifikation von Schoko-Kugeln)

Das Schoko-Kugeln-Problem besteht aus einem Datensatz mit RGB-Farbwerten, bei dem jede Zeile einem Farbwert zugeordnet ist. Die Aufgabe ist es nun, für eine Kombination von RGB-Werten die entsprechende Kugelfarbe vorherzusagen.

Der Datensatz ist über die URL

<https://data.hsbo.de/kugeln.csv>

erreichbar. In dieser Aufgabe sollen Sie den Datensatz laden und darauf ein lineares SVM-Modell trainieren. Das Modell sollen Sie dann benutzen, um eigene Kombinationen von RGB-Werten damit vorherzusagen.

1. Laden Sie die Daten in einen DataFrame. Wieviel Zeilen sind enthalten? Welche Spalten gibt es? Welche Farben sind in der Spalte *label* enthalten?
2. Erstellen Sie aus dem DataFrame einen neuen DataFrame in der Variablen **X**, der nur die Spalten *R*, *G* und *B* enthält. Definieren Sie eine Variable **y**, die die Spalte *label* aus dem Datensatz enthält.
3. Definieren Sie eine Variable **model**, die einen SVM-Classifer enthält und trainieren Sie das Modell auf den Daten **X**, **y** (vgl. letzte Folien von DataScience 1, Foliensatz 7).
4. Schreiben Sie eine Funktion **neue\_daten(r,g,b)**, die aus den übergebenen Parameter **r**, **g** und **b** eine DataFrame mit einer Zeile und den Spalten *R*, *G*, *B* erzeugt.
5. Benutzen Sie ihr gelerntes Modell und die Funktion **neue\_daten(...)** um die Kugelfarbe für die Werte

(24, 52, 198) und (87, 132, 29)

vorherzusagen. Welche Farben kommen dabei heraus?

## Aufgabe 2 \* (Support Vector Machine)

Schauen Sie sich das Notebook **V7-Lineare-SVM.ipynb** von der Vorlesungsseite auf dem DataScience Server an.

1. Erzeugen Sie aus dem Iris-Datensatz mit den zwei Klassen *setosa* und *versicolor* einen Trainings- und einen Test-Datensatz. Nutzen Sie dafür 80% der Daten für die Trainingsmenge.
2. Trainieren Sie das SVM-Modell auf dem Trainingsdatensatz und berechnen Sie mit Hilfe der SkLearn Funktion **accuracy\_score** den Trainingsfehler:

```
from sklearn.metrics import accuracy_score

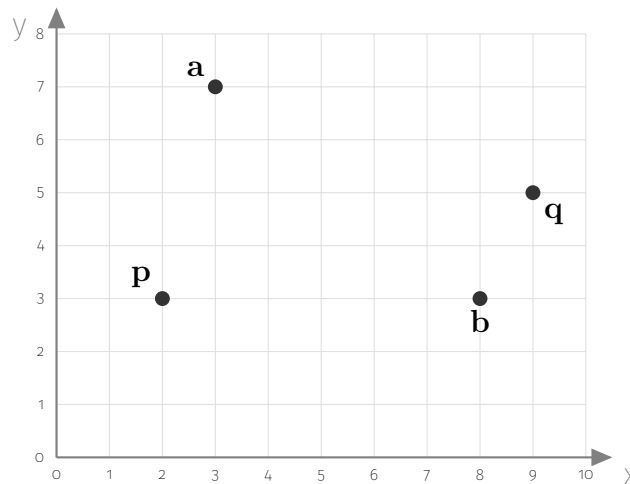
y_pred = m.predict(X_train)
train_error = 1 - accuracy_score(y_train, y_pred)
```

3. Berechnen Sie auch den Testfehler.
4. Erzeugen Sie auch ein  $k$ -NN Modell mit  $k = 5$  und trainieren Sie es auf den gleichen Trainingsdaten. Wie gut ist die Vorhersagegüte *accuracy* von  $k$ -NN im Vergleich zum SVM-Modell?
5. Schauen Sie sich für beide Modelle auch den **classification\_report** an und vergleichen Sie die beiden Modelle auch anhand der anderen Metriken.

### Aufgabe 3 \* (Vektoren mit Pandas)

Diese Aufgabe befasst sich mit linearen Modellen und der damit verbundenen Thematik der Vektorräume. Vektoren lassen sich in Pandas sehr gut als **Series** Objekte darstellen. Damit kann man mit Python/Pandas sehr komfortabel Vektorraum Dinge umsetzen.

Schauen Sie sich den folgenden Plot an.



1. Definieren Sie die Punkte **p**, **q**, **a** und **b** als **Series** Objekte.
2. Implementieren Sie eine Funktion **skalarprodukt(v,w)**, die das Skalarprodukt von zwei Vektoren **v** und **w** (gegeben jeweils als **Series** Objekt) berechnet.
3. Schreiben Sie eine Funktion **dist(v,w)**, die für zwei Vektoren (als **Series** Objekte) die Distanz zwischen den Vektoren berechnet. Welchen Abstand haben die Vektoren **p** und **q** aus der obigen Skizze?
4. Definieren Sie eine Python Funktion **f(x)**, die die Gerade beschreibt, die durch die Punkte **p** und **q** läuft.
5. Berechnen Sie den Vektor  $\mathbf{v} = \mathbf{q} - \mathbf{p}$  als weiteres **Series** Objekt und berechnen Sie die Skalarprodukte  $\langle \mathbf{b}, \mathbf{v} \rangle$  und  $\langle \mathbf{a}, \mathbf{v} \rangle$ .