Data Science: Predictive Analytics, Übung 3

Henry Haustein

Modeling & Evaluation

```python
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import train_test_split
from keras.models import Sequential
from keras.layers import Dense
import numpy as np

data = pd.read_csv("data_ex3.csv", index_col = 0)
print(data.info())
print(data.describe())

# Task 1
X = data.drop(columns = ["Rented_Bike_Count", "Date"])
y = data["Rented_Bike_Count"]
x_train, x_test, y_train, y_test = train_test_split(X, y, test_size = 0.3)
model = LinearRegression()
model.fit(x_train, y_train)
print(model.score(x_test, y_test)) # 0.55 for full model

# using R's step() function, I get the following better model
X2 = data.drop(columns = ["Rented_Bike_Count", "Date", "Functioning_Day", "Month"])
y2 = data["Rented_Bike_Count"]
x2_train, x2_test, y2_train, y2_test = train_test_split(X2, y2, test_size = 0.3)
model_reduced = LinearRegression()
model_reduced.fit(x2_train, y2_train)
print(model_reduced.score(x2_test, y2_test)) # 0.47 for better (?) model

model2 = DecisionTreeRegressor()
model2.fit(x_train, y_train)
print(model2.score(x_test, y_test)) # 0.04 for full model

model2_reduced = DecisionTreeRegressor()
model2_reduced.fit(x2_train, y2_train)
```

```
print(model2_reduced.score(x2_test, y2_test)) # 0.02 for better (?) model

newData = pd.DataFrame({"Hour": 9, "Temperature": -2, "Humidity": 42, "Wind_speed": 6, "Visibility": 2000, "Solar_Radiation": 0, "Rainfall": 0, "Snowfall": 0, "Seasons": 1, "Holiday": 0, "Functioning_Day": True, "Weekday": 1, "Day_Night": 0, "Month": 3}, index = [0])
print(model.predict(newData)) # 453.36
print(model2.predict(newData)) # 206 (but huge spread depending on training data)

# Task 2
print(model.coef_[1]) # Temperature coefficient = 28.25
print(model.coef_[6]) # Rainfall cieffiecient = -56.82
print(model_reduced.coef_[1]) # Temperature coefficient = 29.19
print(model_reduced.coef_[6]) # Rainfall cieffiecient = -63.89

# Task 3
data["Functioning_Day"] = np.asarray(data["Functioning_Day"]).astype(np.float32) # boolean values don't work
X3 = data.drop(columns = ["Date"]).copy()
y3 = X3.pop("Rented_Bike_Count")
X3 = np.array(X3)
y3 = np.array(y3)

x3_train, x3_test, y3_train, y3_test = train_test_split(X3, y3, test_size = 0.3)

model3 = Sequential()
model3.add(Dense(32, input_dim = 14))
model3.add(Dense(128))
model3.add(Dense(64))
model3.add(Dense(64))
model3.add(Dense(64))
model3.add(Dense(32))
model3.add(Dense(1))
model3.compile(
  optimizer = 'adam',
  loss = 'mean_absolute_error',
  metrics = ['mean_absolute_error']
)
model3.fit(x3_train, y3_train)
print(model3.evaluate(x3_test, y3_test)) # [404.8241882324219, 404.8241882324219]
model3.fit(x3_train, y3_train, epochs = 10)
print(model3.evaluate(x3_test, y3_test)) # [386.04815673828125, 386.04815673828125]

newData = pd.DataFrame({"Hour": 9, "Temperature": -2, "Humidity": 42, "Wind_speed": 6, "Visibility": 2000, "Solar_Radiation": 0,
 "Rainfall": 0, "Snowfall": 0, "Seasons": 1, "Holiday": 0, "
Functioning_Day": 0, "Weekday": 1, "Day_Night": 0, "Month":
3}, index = [0])
print(model3.predict(newData)) # 197.68306 (but huge spread
    depending on training data)
```