Einfaches Beispiel für Support Vector Regression (SVR)

Vorhersage einer Multiplen Linearen Regression (mit Daten aus Kapitel 5).

Wir nutzen die Datenpunkte einer Multiplen Linearen Regression als Beispiel (3-dimensional). Dies erfogt in mehreren Schritten:

1. Datenerstellung: wir nehem die Datenpunkte aus Aufgabe (5.7): Wir suchen eine ”least square fit” Ebene z = a + b*x* + *c*y für z:=erreichte Punktzahl (Score) des Examens abhängig von den zwei Parametern: x:=”Aufwand der Examens Vorbereitung in Std.” and y:=Aufwand für Hausaufgaben in Std.“. Daten der Trainings-Sets TS: = {(x, y; z)|(Examvorb., Homework; Score)} = {(7, 5; 41), (3, 4; 27), (5, 5; 35), (3, 3; 26), (8, 9; 48), (7, 8; 45), (10, 10; 46), (3, 5; 27), (5, 3; 29), (3, 3; 19)}
2. SVR-Modell: Ein linearer-Kernel wird verwendet, da es sich um eine lineare Regression handelt. Die Parameter C und gamma steuern die Anpassung des Modells. Baue das SVR Modell SVR.
3. Vergleich der Regressionsebne aus Aufgabe (5.7) mit der Ebene die durch SVR approximiert wurde. Prüfe das Ergebnis mit einem Ergebnis von Aufgabe (5.7): z = 13.264 + 2.488*x* + *1.382*y

Autor: Dr. H. Völlinger, Januar 2025

In [29]:
```
# Import and check needed libraries

import numpy as np
import matplotlib.pyplot as plt
from sklearn.svm import SVR
from mpl_toolkits.mplot3d import Axes3D

# to check the time of execution, import function time
import time

# check versions of libraries
print('numpy version is: {}'.format(np.__version__))
```

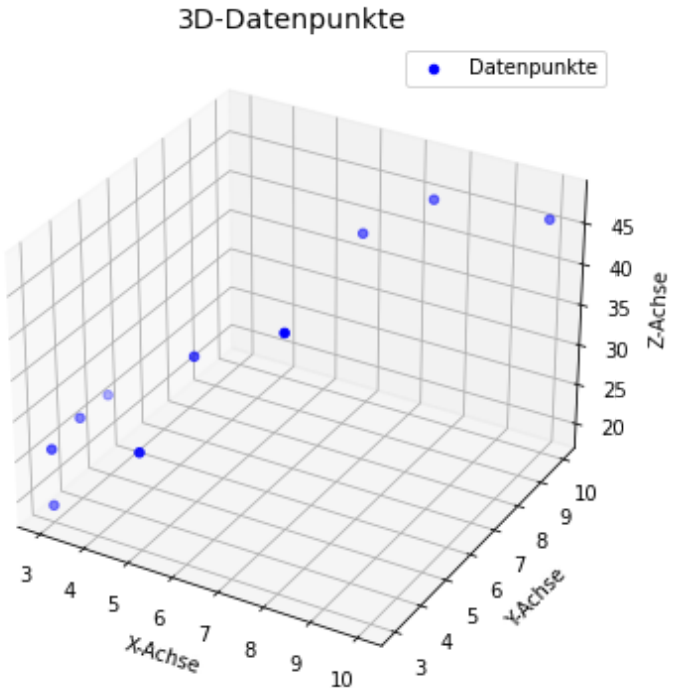
```
numpy version is: 1.18.1
```

Step1: Daten bereistellen und ausplotten

In diesem zweiten Schritt nehmen wir die Daten aus Übungsaufgabe (5.7). Diese Daten sind gegeben als 10 Datenpunkten in einem 3-dimensionalen Raum. Wir plotten diese Daten aus.

In [30]:

```python
# Definieren der Datenpunkte
data_points = [
    (7, 5, 41),
    (3, 4, 27),
    (5, 5, 35),
    (3, 3, 26),
    (8, 9, 48),
    (7, 8, 45),
    (10, 10, 46),
    (3, 5, 27),
    (5, 3, 29),
    (3, 3, 19)
]

# Extrahieren von x, y, z Koordinaten
x_coords = [point[0] for point in data_points]
y_coords = [point[1] for point in data_points]
z_coords = [point[2] for point in data_points]

# Plotten der Punkte im 3D-Raum
fig = plt.figure(figsize=(8, 6))
ax = fig.add_subplot(111, projection='3d')

# Scatter-Plot der Punkte
ax.scatter(x_coords, y_coords, z_coords, color='blue', label='Datenpunkte')

# Achsentitel und Label
ax.set_title("3D-Datenpunkte", fontsize=14)
ax.set_xlabel("X-Achse")
ax.set_ylabel("Y-Achse")
ax.set_zlabel("Z-Achse")
ax.legend()

# Plot anzeigen
plt.show()
```



Step2: 3 SVR Modelle definieren (Typ =linear) und anwenden

Setzen der 2 linearen SVR-Modell-Parameter: $C = 100, \gamma = auto$

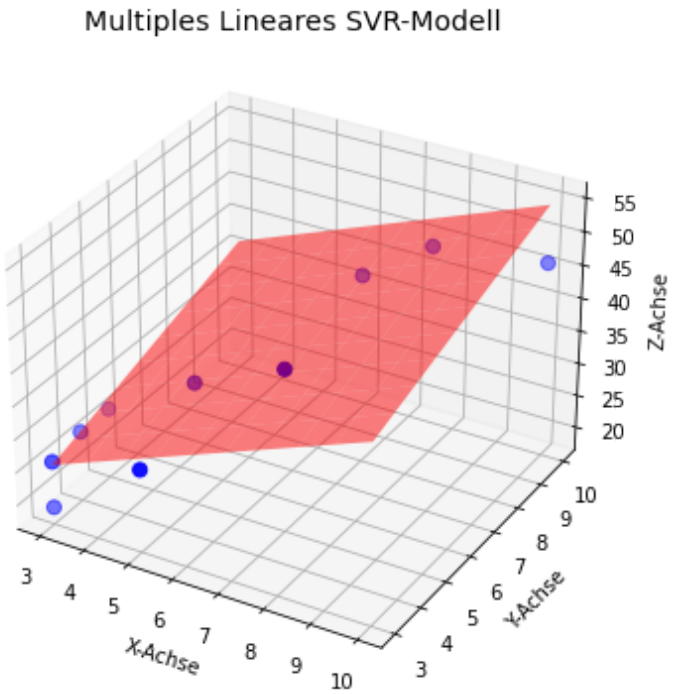
In [31]:
```
# Gegebene Datenpunkte
data_points = np.array([
    [7, 5, 41],
    [3, 4, 27],
    [5, 5, 35],
    [3, 3, 26],
    [8, 9, 48],
    [7, 8, 45],
    [10, 10, 46],
    [3, 5, 27],
    [5, 3, 29],
    [3, 3, 19]
])

# Eingabeparameter (x, y) und Zielwerte (z)
X = data_points[:, :2]  # x und y
y = data_points[:, 2]   # z

# Lineares SVR-Modell definieren
svr_linear = SVR(kernel='linear', C=1.0, epsilon=0.1)

# Trainieren des Modells auf den gesamten Daten
svr_linear.fit(X, y)

# Modellkoeffizienten und Intercept
coef = svr_linear.coef_
intercept = svr_linear.intercept_

print("Modellkoeffizienten (x und y):", coef)
print("Modellintercept:", intercept)
```

```
Modellkoeffizienten (x und y): [[2.58 1.58]]
Modellintercept: [13.04]
```

Ergebnisse visualisieren

Anzeige der SVR-Vorhersage und der Datenpunkte

In [32]:
```
# 3D-Visualisierung der Datenpunkte und der SVR-Ebene
fig = plt.figure(figsize=(10, 6))
ax = fig.add_subplot(111, projection='3d')

# Datenpunkte plotten
ax.scatter(X[:, 0], X[:, 1], y, color='blue', label='Datenpunkte', s=50)

# Bereich für die SVR-Ebene
x_range = np.linspace(X[:, 0].min(), X[:, 0].max(), 10)
y_range = np.linspace(X[:, 1].min(), X[:, 1].max(), 10)
x_mesh, y_mesh = np.meshgrid(x_range, y_range)
z_mesh = (coef[0][0] * x_mesh + coef[0][1] * y_mesh + intercept).reshape(x_m
esh.shape)

# SVR-Ebene plotten
ax.plot_surface(x_mesh, y_mesh, z_mesh, color='red', alpha=0.5, label='SVR-M
odell')

# Achsentitel und Label
ax.set_title("Multiples Lineares SVR-Modell", fontsize=14)
ax.set_xlabel("X-Achse")
ax.set_ylabel("Y-Achse")
ax.set_zlabel("Z-Achse")
# ax.legend()

plt.show()
```



Step3: Vergleich mit Ergebnissen der Übungsaufagbe (5.7)

In [33]:
```
print('Lösungen bei Aufgabe (5.7) waren Steigungen  ~ 2,48 und 1,38 und Achs
enabschnitt ~ 13.26')
print('Lösungen hier Modellkoeffizienten (x und y): [[2.58 1.58]] und Modell
intercept: [13.04]')
print('Lösungen sind also bei beiden Verfahren durchaus vergleichbar!')
```

```
Lösungen bei Aufgabe (5.7) waren Steigungen  ~ 2,48 und 1,38 und Achsenabsc
hnitt ~ 13.26
Lösungen hier Modellkoeffizienten (x und y): [[2.58 1.58]] und Modellinterc
ept: [13.04]
Lösungen sind also bei beiden Verfahren durchaus vergleichbar!
```

In [34]:
```
# Print current date and time
print('******* Aktuelles Datum und Zeit*********')
print("Date & Time:",time.strftime("%d.%m.%Y  %H:%M:%S"))
# end of import test
print ("Ende Python-Programm ***SVR_LinRegression ***")
```

```
******* Aktuelles Datum und Zeit*********
Date & Time: 02.01.2025  15:56:26
Ende Python-Programm ***SVR_LinRegression ***
```