Mustererkennung Übung 5

Benjamin Kiesling, Peewee Manzer

1. Aufgabe (5 Punkte): Lineare Regression

Importieren Sie den Datensatz fish.txt in matlab. Das Format ist: index ; the age of the fish ; the water temperature in degrees Celsius ; the length of the fish

a. (3 Punkte)

Schätzen Sie den Wert für “length” anhand der Parameter “age” und “temperature” mit linearer Regression. Visualisieren Sie dreidimensional die tatsächlichen Datenpunkte, die geschätzten Datenpunkte, sowie die Abstände der tatsächlichen zu den geschätzten Datenpunkten.

In [1]:
```
%matplotlib inline

import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from mpl_toolkits.mplot3d import Axes3D
# make plots bigger

plt.rcParams['figure.figsize'] = 16, 16
```

In [2]:
```
#loadData
fishData = pd.read_csv("./fish.txt",delim_whitespace=True, dtype={'names': ('ind
ex', 'fish age', 'water temperature','fish length'),
                     'formats': ('i4','i4', 'i4','i4' )},index_col=0, names = [N
one, 'fish age', 'water temperature','fish length'])
fishData.head()
```

Out[2]:

	fish age	water temperature	fish length
1	14	25	620
2	28	25	1315
3	41	25	2120
4	55	25	2600
5	69	25	3110

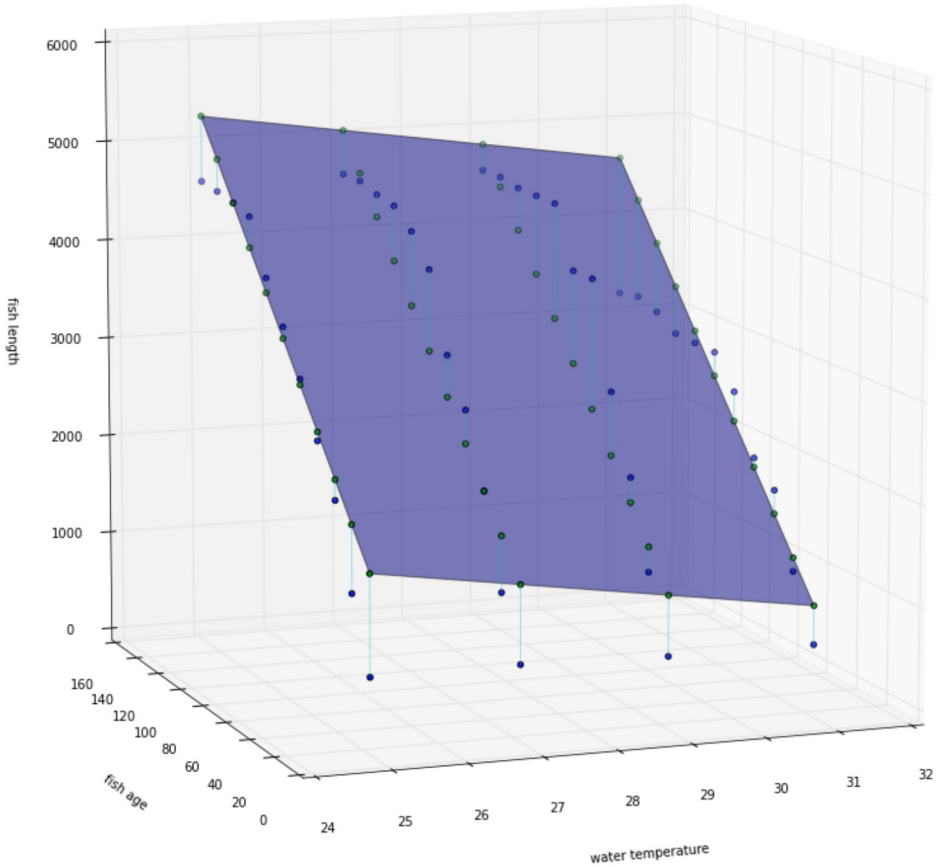
In [3]:
```
from sklearn.linear_model import LinearRegression

y = fishData['fish length']
features = ['water temperature','fish age']
x = fishData[features]
lm = LinearRegression()
lm.fit(x,y)

print lm.intercept_
print lm.coef_
zip(features, lm.coef_)
```

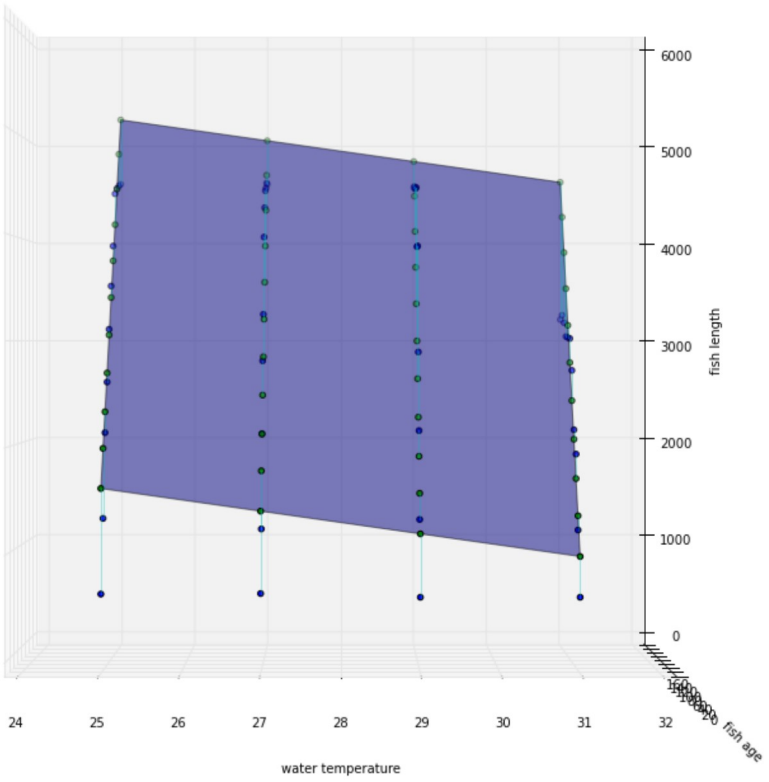
```
3904.26601675
[-106.41363636   26.24068177]
```

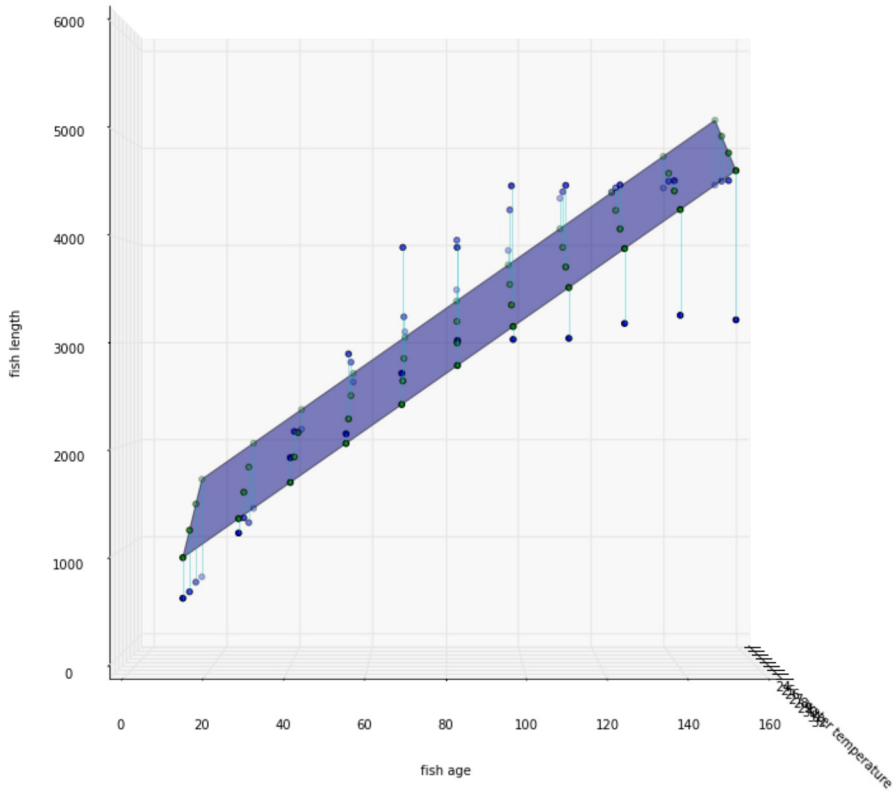
Out[3]:
```
[('water temperature', -106.41363636363634), ('fish age', 26.240681765825098)]
```

b. (2 Punkte) Visualisieren Sie die durch die in Aufgabe a) berechneten Koeffizienten definierte Ebene.

In [4]:
```
x = list(fishData['water temperature'])
y = list(fishData['fish age'])
z = list(fishData['fish length'])

x_min = fishData['water temperature'].min()
x_max = fishData['water temperature'].max()

y_min = fishData['fish age'].min()
y_max = fishData['fish age'].max()


x_range = np.array([x_min,x_max])
y_range = np.array([y_min,y_max])
x_surf, y_surf = np.meshgrid(x_range, y_range)
lr_plane = lm.intercept_+lm.coef_[0]*x_surf+lm.coef_[1]*y_surf

def plot_fig(fignb,azim,elev):
    fig = plt.figure(fignb)
    ax = fig.add_subplot(111, projection='3d',azim = azim,elev = elev)


    ax.scatter(x,y,z)

    ax.plot_surface(x_surf,
                    y_surf,
                    lr_plane.reshape((2, 2)),
                    alpha=.5)


    ax.set_xlabel('water temperature')
    ax.set_ylabel('fish age')
    ax.set_zlabel('fish length')
    z_i_hat = lm.intercept_+np.multiply(lm.coef_[0],x)+np.multiply(lm.coef_[1],y
)

    ax.scatter(x,y,z_i_hat,c='g')
    for i,z_i in enumerate(z):
        ax.plot([x[i],x[i]],[y[i],y[i]],[z[i],z_i_hat[i]], c='c',alpha =.35)

plot_fig(0,-110,10)
plot_fig(1,-90,0)
plot_fig(2,0,0)



plt.show()
```


fish length
6000
5000
4000
3000
2000
1000
0
fish age
160
140
120
100
80
60
40
20
0
water temperature
24
25
26
27
28
29
30
31
32


6000
5000
4000
3000
2000
1000
0
fish length
24
25
26
27
28
29
30
31
32
water temperature
fish age



2. Aufgabe (5 Punkte): Subset Selection

Importieren Sie den Datensatz winequalityred. txt in matlab. Das Format ist: fixed acidity; volatile acidity; citric acid; residual sugar; chlorides; free sulfur dioxide; total sulfur dioxide; density; pH; sulphates; alcohol; quality (score between 0 and 10) Schätzen Sie den Wert für “quality” mit linearer Regression anhand aller möglichen Kombinationen der anderen Parameter (also jeweils für alle Einer, Zweier, Dreierkombinationen, usw.) und berechnen jeweils die Summe der quadratischen Abweichungen zwischen den geschätzten und tatsächlichen Werten für “quality”. Visualisieren Sie dies als zweidimensionalen Plot. Auf der xAchse steht dabei die Anzahl der verwendeten Parameter, auf der yAchse die Summe der quadratischen Abweichungen (für alle Kombinationen der jeweiligen Anzahl der Parameter) .

In [5]:
```
wineDataPath = "./winequality-red.txt"
wineData = pd.read_csv(wineDataPath, delimiter=';',dtype={'names': ('fixed_acidi
ty', 'volatile acidity', 'citric acid', 'residual sugar', 'chlorides', 'free sul
fur dioxide', 'total sulfur dioxide', 'density', 'pH', 'sulphates', ' alcohol',
'quality'),
                    }, names = ['fixed_acidity','volatile acidity', 'citric acid
','residual sugar', 'chlorides', 'free sulfur dioxide', 'total sulfur dioxide',
'density', 'pH', 'sulphates', ' alcohol', 'quality'])
print wineData.shape
wineData.head()
```

```
(1599, 12)
```

Out[5]:

	fixed_acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcol
0	7.4	0.70	0.00	1.9	0.076	11	34	0.9978	3.51	0.56	9.4
1	7.8	0.88	0.00	2.6	0.098	25	67	0.9968	3.20	0.68	9.8
2	7.8	0.76	0.04	2.3	0.092	15	54	0.9970	3.26	0.65	9.8
3	11.2	0.28	0.56	1.9	0.075	17	60	0.9980	3.16	0.58	9.8
4	7.4	0.70	0.00	1.9	0.076	11	34	0.9978	3.51	0.56	9.4

In [6]:
```
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error as mse

def linregCombi(features):

    x = wineData[features]
    y = wineData['quality']

    lm = LinearRegression()
    lm.fit(x,y)

    featData = {}
    for feat in features:
        featData[feat] = wineData[feat]
    X_new = pd.DataFrame(featData)

    #pred = lm.predict(X_new)
    pred = lm.intercept_

    feat = zip(features, lm.coef_)
    for f in feat:
        pred +=f[1]*wineData[f[0]]


    return np.sum(np.square(y-pred)) #mse(y,pred) or np.sum(np.square(y-preds))
for SSD
```

In [7]:
```
import itertools

features = ['fixed_acidity', 'volatile acidity', 'citric acid', 'residual sugar'
, 'chlorides', 'free sulfur dioxide', 'total sulfur dioxide', 'density', 'pH', '
sulphates', ' alcohol',]

fig = plt.figure()
ax = fig.add_subplot(111)
for L in range(1,len(features)+1):
    for fsubset in itertools.combinations(features,L):
        tmp = linregCombi(list(fsubset))
        ax.scatter(L,tmp,alpha = .4)
plt.axis('auto')
plt.xticks(np.arange(12))
plt.show()
```