EL elleboogafstand

December 26, 2020

[1]:

```
import sys
sys.path.append("../")
import pandas as pd
from ortho_lib import *
import os
import matplotlib.pyplot as plt
import numpy as np
```

[2]:

```
path_cats = ['..//transformed_data/Category_1/', '..//transformed_data/
↪Category_2/', '..//transformed_data/Category_3/', '..//transformed_data/
↪Category_4/']
exercise = '/EL1'
df = pd.DataFrame()

def elleboogafstand(path_cat, df = pd.DataFrame()): #bij het aanroepen van de␣
↪functie het indexnummer voor de categorie uit path_cats
    patientID = os.listdir(path_cats[path_cat])
    if path_cat == 3:
        patientID.remove('23')
        patientID.remove('21')

    for patient in patientID:
        path = path_cats[path_cat] + patient + exercise + '.txt'
        df_patient = exercise_to_df(path)
        df_patient['patientID'] = patient
        df = df.append([df_patient])
        del df['z']
        del df['y']


    elbow_df = df[df['sensor'] != '2'] #anker verwijderen uit de dataframe, dit␣
↪datapunt is nooit nodig
    elbow_df = elbow_df.set_index( ['patientID', 'frame'], drop=True,␣
↪inplace=False, verify_integrity=False)
    elbow_df = elbow_df[elbow_df['sensor'] != '3'] #sensoren verwijderen die␣
↪niet van belang zijn. Alleen de sensoren bewaren die vergeleken moeten␣
↪worden.
```

```
    elbow_df = elbow_df[elbow_df['sensor'] != '4']
    elbow_df = elbow_df[elbow_df['sensor'] != '6']
    elbow_df = elbow_df[elbow_df['sensor'] != '7']
    elbow_df = elbow_df[elbow_df['sensor'] != '9']

    minelleboogafstand_list = []
    for patient in patientID:
        dfpatient = df[df['patientID']==str(patient)]
        per_patient_5 = dfpatient[dfpatient['sensor'] == '5']
        per_patient_8 = dfpatient[dfpatient['sensor'] == '8']
        max_5 = max(per_patient_5['x'])
        min_5 = min(per_patient_5['x'])
        verschil_5 = max_5 - min_5
        max_8 = max(per_patient_8['x'])
        min_8 = min(per_patient_8['x'])
        verschil_8 = max_8 - min_8
        minelleboogafstand = min(verschil_5, verschil_8)
        minelleboogafstand_list.append(minelleboogafstand)

    elbow_distance_df = pd.DataFrame()
    elbow_distance_df['patientID'] = patientID
    elbow_distance_df.set_index(['patientID'], drop = True, inplace = True)
    elbow_distance_df['elbow distance'] = minelleboogafstand_list
    elbow_distance_df['category'] = path_cat + 1

    return elbow_distance_df
```

[3]:
```
df_ellebogen = pd.concat([elleboogafstand(0), elleboogafstand(1),␣
 ↪elleboogafstand(2), elleboogafstand(3)])
```

[4]:
```
df_ellebogen
```

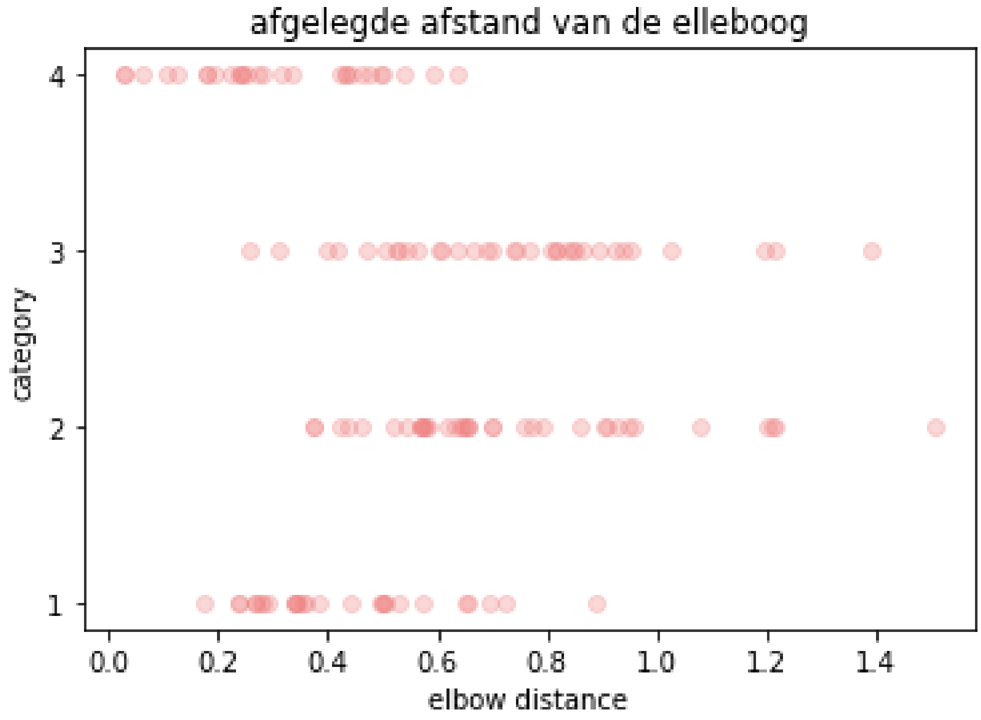
[4]:
```
           elbow distance  category
patientID
8                0.267481         1
3                0.891016         1
1                0.695687         1
22               0.234785         1
17               0.238309         1
…                     …        …
27               0.592029         4
5                0.496536         4
2                0.239724         4
4                0.432559         4
24               0.237744         4

[125 rows x 2 columns]
```

[5]:

```
plt.scatter(df_ellebogen['elbow distance'], df_ellebogen['category'], alpha = 0.
 ↪3, color ='lightcoral')
plt.title('afgelegde afstand van de elleboog')
plt.yticks([1,2,3,4])
plt.xlabel('elbow distance')
plt.ylabel('category')
```

[5]:

```
Text(0, 0.5, 'category')
```



[6]:

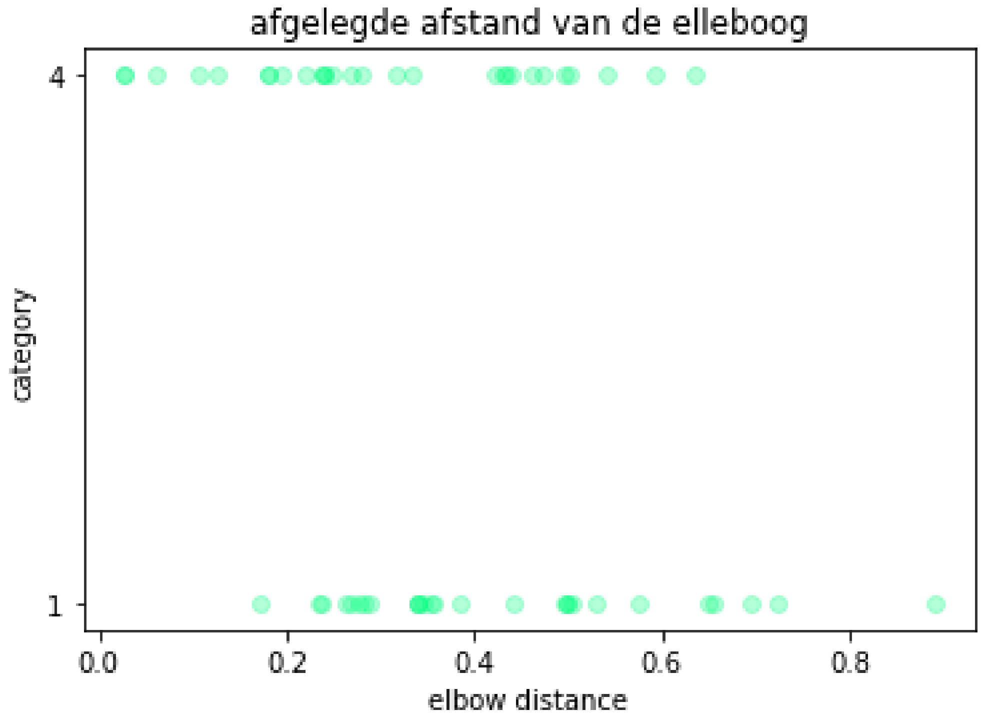
```
from sklearn.model_selection import train_test_split
from sklearn.model_selection import StratifiedKFold
import numpy as np
from sklearn.linear_model import LogisticRegression

#splitten test en train set
X = np.asarray(df_ellebogen[['elbow distance']])
y = np.asarray(df_ellebogen[['category']])

scores=[]

skf = StratifiedKFold(n_splits=6)
for train, test in skf.split(X, y):
    X_train, X_test = X[train], X[test]
    y_train, y_test = y[train], y[test]
    logistic_reg = LogisticRegression(multi_class='ovr', solver='saga')
    logistic_reg.fit(X_train,y_train)
    y_predict = logistic_reg.predict(X_test)
    score = logistic_reg.score(X_test, y_test)
    print(y_predict, score)
    scores.append(score)

print(np.mean(scores))
```

```
[4 2 2 4 4 4 2 2 4 4 2 4 2 4 2 2 2 4 4 4 2] 0.2857142857142857
[4 4 2 2 2 2 3 3 2 4 2 4 3 2 3 4 4 4 4 4 4] 0.47619047619047616
[2 4 4 4 4 2 2 2 2 2 2 2 2 2 2 2 4 4 4 4 4] 0.5238095238095238
[2 4 4 2 2 2 2 2 2 2 2 2 2 2 2 2 4 4 2 2 4] 0.42857142857142855
[2 2 4 4 2 4 2 2 2 2 2 2 2 2 2 2 4 4 4 2 2] 0.38095238095238093
[4 4 2 4 3 3 3 2 2 2 2 2 2 2 2 2 2 4 4 4] 0.3
0.3992063492063491
```

```
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
```

```
[10]: df_ellebogen_subset = df_ellebogen[(df_ellebogen['category']==␣
       ↪1)|(df_ellebogen['category']== 4)]
      plt.scatter(df_ellebogen_subset['elbow distance'],␣
       ↪df_ellebogen_subset['category'], alpha = 0.3, color = 'springgreen')
      plt.title('afgelegde afstand van de elleboog')
      plt.yticks([1,4])
      plt.xlabel('elbow distance')
      plt.ylabel('category')
```

```
[10]: Text(0, 0.5, 'category')
```



```
[11]: from sklearn.model_selection import train_test_split
      from sklearn.model_selection import StratifiedKFold
      import numpy as np
      from sklearn.linear_model import LogisticRegression

      #splitten test en train set
      X = np.asarray(df_ellebogen_subset[['elbow distance']])
      y = np.asarray(df_ellebogen_subset[['category']])

      scores=[]

      skf = StratifiedKFold(n_splits=6)
```

```
for train, test in skf.split(X, y):
    X_train, X_test = X[train], X[test]
    y_train, y_test = y[train], y[test]
    logistic_reg = LogisticRegression()
    logistic_reg.fit(X_train,y_train)
    y_predict = logistic_reg.predict(X_test)
    score = logistic_reg.score(X_test, y_test)
    print(y_predict, score)
    scores.append(score)

print(np.mean(scores))
```

```
[4 1 1 4 4 4 4 1 1 1] 0.4
[4 4 1 4 4 4 4 4 4] 0.5555555555555556
[1 4 4 4 4 4 4 4 4] 0.5555555555555556
[1 4 4 1 4 4 1 1 1] 0.4444444444444444
[1 1 4 4 4 4 4 4 1] 0.6666666666666666
[4 4 1 4 1 1 4 1 4] 0.3333333333333333
0.49259259259259264
```

```
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
```

```
(n_samples, ), for example using ravel().
  return f(**kwargs)
```

[12]:

```
from sklearn import svm
#splitten test en train set
X = np.asarray(df_ellebogen[['elbow distance']])
y = np.asarray(df_ellebogen[['category']])

scores=[]

skf = StratifiedKFold(n_splits=6)
for train, test in skf.split(X, y):
    X_train, X_test = X[train], X[test]
    y_train, y_test = y[train], y[test]
    clf = svm.SVC(kernel = 'linear')
    clf.fit(X_train,y_train)
    y_predict = logistic_reg.predict(X_test)
    score = logistic_reg.score(X_test, y_test)
    print(y_predict, score)
    scores.append(score)

print(np.mean(scores))
```

```
[4 1 1 4 4 1 1 1 1 1 1 4 1 1 1 1 1 4 4 1 1] 0.19047619047619047
[4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 4 1 4 4 4 4] 0.38095238095238093
[1 4 4 1 4 1 1 1 1 1 1 1 1 1 1 1 4 1 4 4 4] 0.2857142857142857
[1 4 4 1 1 1 1 1 1 1 1 1 1 1 1 1 4 1 1 1 4] 0.19047619047619047
[1 1 4 4 1 1 1 1 1 1 1 1 1 1 1 1 4 4 4 1 1] 0.23809523809523808
[4 4 1 4 1 1 1 1 1 1 1 1 1 1 1 1 1 4 1 4] 0.15
0.23928571428571424
```

```
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
```

```
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/sklearn/utils/validation.py:72: DataConversionWarning: A column-vector
y was passed when a 1d array was expected. Please change the shape of y to
(n_samples, ), for example using ravel().
  return f(**kwargs)
```

[ ]: