all_steps_activity_recognition_logistic_regression

January 12, 2021

[2]:
```python
from helpers import math_helper
from sensors.activpal import *
from utils import read_functions
from scipy import signal
from sklearn.model_selection import train_test_split
from sklearn import tree
from sklearn.metrics import f1_score, plot_confusion_matrix, confusion_matrix,␣
↪accuracy_score, precision_score, recall_score, confusion_matrix,␣
↪classification_report
from sklearn.linear_model import LogisticRegression

import pandas as pd
import numpy as np
import statistics
import os

import matplotlib.pyplot as plt
```

```
/opt/jupyterhub/anaconda/lib/python3.6/importlib/_bootstrap.py:219:
RuntimeWarning: numpy.ufunc size changed, may indicate binary incompatibility.
Expected 192 from C header, got 216 from PyObject
  return f(*args, **kwds)
/opt/jupyterhub/anaconda/lib/python3.6/importlib/_bootstrap.py:219:
RuntimeWarning: numpy.ufunc size changed, may indicate binary incompatibility.
Expected 192 from C header, got 216 from PyObject
  return f(*args, **kwds)
/opt/jupyterhub/anaconda/lib/python3.6/importlib/_bootstrap.py:219:
RuntimeWarning: numpy.ufunc size changed, may indicate binary incompatibility.
Expected 192 from C header, got 216 from PyObject
  return f(*args, **kwds)
/opt/jupyterhub/anaconda/lib/python3.6/importlib/_bootstrap.py:219:
RuntimeWarning: numpy.ufunc size changed, may indicate binary incompatibility.
Expected 192 from C header, got 216 from PyObject
  return f(*args, **kwds)
/opt/jupyterhub/anaconda/lib/python3.6/importlib/_bootstrap.py:219:
RuntimeWarning: numpy.ufunc size changed, may indicate binary incompatibility.
Expected 192 from C header, got 216 from PyObject
  return f(*args, **kwds)
```

Adnan Akbas # Feature Extraction

[29]:
```
activpal = Activpal()

features_columns = ['standard_deviation_x', 'mean_x', 'standard_deviation_y',␣
↪'mean_y','standard_deviation_z', 'mean_z', 'activiteit']

#activity_columns = ['activity_walking', 'activity_running',␣
↪'activity_jumping', 'activity_standing', 'activity_traplopen',␣
↪'activity_sitten']
#activity_columns = ['cycling_light','cycling_hard', 'activity_walking',␣
↪'activity_running', 'activity_jumping', 'activity_standing',␣
↪'activity_traplopen', 'activity_sitten']
#activity_columns = ['cycling_light','cycling_hard', 'activity_walking',␣
↪'activity_running', 'activity_standing', 'activity_sitten']
#activity_columns = ['activity_cycling', 'activity_walking',␣
↪'activity_running', 'activity_standing', 'activity_sitten']
activity_columns = ['activity']

#activities = ['lopen', 'rennen', 'springen', 'staan', 'traplopen', 'zitten']
#activities = ['fietsen licht', 'fietsen zwaar', 'lopen', 'rennen', 'springen',␣
↪'staan', 'traplopen', 'zitten']
#Since jumping and walking on stairs don't have any vyntus
activities = ['fietsen licht', 'fietsen zwaar', 'lopen', 'rennen', 'staan',␣
↪'zitten']

test_users = ['BMR002', 'BMR004', 'BMR008']
segment_size = 9.4
```

Notes:

1. First of all we don't have vyntus data for jumping and walking on stairs. This means that it's not neccasary to recognize these activities anymore.
2. Second is that the MET models now can predict the speed of cycling, running and walking. This means that we don't have split cycling in light and hard anymore.
3. In conclusion we will work with cycling, walking, running, standing and sitten

[4]:
```
def extract_features_from_correspondent(correspondent):
    features_df = pd.DataFrame(columns=features_columns, index=pd.
↪to_datetime([]))

    # Getting dataset for a correspodent
    activities_df = read_functions.read_activities(correspondent)

    for activity_name in activities:
        activity = activities_df.loc[activity_name]
        if not activity.empty:
```

```
            start_time = activity.start
            stop_time = activity.stop
            activpal_df = activpal.read_data(correspondent, start_time,␣
↪stop_time)

            # denormalizing dataset
            activpal_df['x'] = math_helper.
↪convert_value_to_g(activpal_df['pal_accX'])
            activpal_df['y'] = math_helper.
↪convert_value_to_g(activpal_df['pal_accY'])
            activpal_df['z'] = math_helper.
↪convert_value_to_g(activpal_df['pal_accZ'])

            date_range = pd.date_range(start_time, stop_time,␣
↪freq=str(segment_size) + 'S')

            for time in date_range:
                segment_time = time + pd.DateOffset(seconds=segment_size)
                activpal_segment = activpal_df[(activpal_df.index >= time) &␣
↪(activpal_df.index <= segment_time)]

                stdev_x =  statistics.stdev(activpal_segment['x']) if␣
↪len(activpal_segment['x']) >= 2 else 0
                mean_x = activpal_segment['x'].mean()

                stdev_y =  statistics.stdev(activpal_segment['y']) if␣
↪len(activpal_segment['y']) >= 2 else 0
                mean_y = activpal_segment['y'].mean()

                stdev_z =  statistics.stdev(activpal_segment['z']) if␣
↪len(activpal_segment['z']) >= 2 else 0
                mean_z = activpal_segment['z'].mean()


                features_df.loc[segment_time] = [stdev_x, mean_x, stdev_y,␣
↪mean_y, stdev_z, mean_z, activity_name]

    return features_df
```

[5]:
```
def extract_features_from_all_correspondents():
    all_features_df = pd.DataFrame(index=pd.to_datetime([]))

    for directory in os.walk('../../data'):
        if directory[0] == '../../data':
            for respDirect in directory[1]:
```

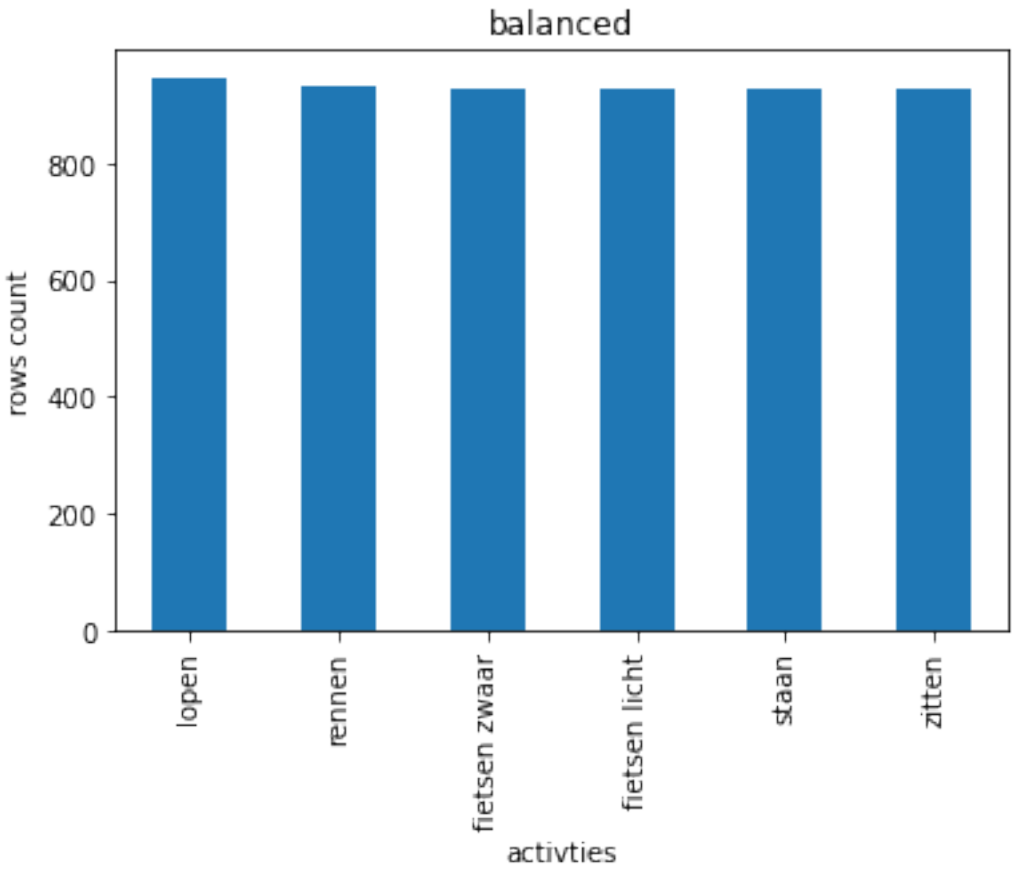
```
                if respDirect not in ['output', 'throughput', 'Test data','.
↪ipynb_checkpoints', 'BMR035', 'BMR100', 'BMR051', 'BMR027']:
                  #  if respDirect not in test_users:
                    print("Extracting " + respDirect)
                    features_df =␣
↪extract_features_from_correspondent(respDirect)
                    all_features_df = pd.concat([all_features_df, features_df])

    print("Done extracting features")

    return all_features_df
```

[31]:
```
features_dataset = extract_features_from_all_correspondents()
```

```
Extracting BMR099
Extracting BMR025
Extracting BMR060
Extracting BMR012
Extracting BMR030
Extracting BMR044
Extracting BMR043
Extracting BMR004
Extracting BMR011
Extracting BMR098
Extracting BMR034
Extracting BMR014
Extracting BMR036
Extracting BMR052
Extracting BMR002
Extracting BMR031
Extracting BMR097
Extracting BMR008
Extracting BMR015
Extracting BMR033
Extracting BMR064
Extracting BMR055
Extracting BMR041
Extracting BMR053
Extracting BMR042
Extracting BMR018
Extracting BMR058
Extracting BMR040
Extracting BMR032
Done extracting features
```

[32]:
```
features_dataset['activiteit'].value_counts().plot.bar(ylabel='rows␣
↪count',xlabel='activties',title='balanced')
```

[32]: <matplotlib.axes._subplots.AxesSubplot at 0x7ff65c7d7550>



1 model preperation

[33]:
```
features_dataset[activity_columns] = 0

#features_dataset.loc[(features_dataset['activiteit'] == 'springen'),␣
↪'activity_jumping'] = 1
#features_dataset.loc[(features_dataset['activiteit'] == 'traplopen'),␣
↪'activity_traplopen'] = 1
features_dataset.loc[(features_dataset['activiteit'] == 'lopen'), 'activity'] =␣
↪1
features_dataset.loc[(features_dataset['activiteit'] == 'rennen'), 'activity']␣
↪= 2
features_dataset.loc[(features_dataset['activiteit'] == 'staan'), 'activity'] =␣
↪3
```

```
features_dataset.loc[(features_dataset['activiteit'] == 'zitten'), 'activity']␣
↪= 4
features_dataset.loc[(features_dataset['activiteit'] == 'fietsen licht'),␣
↪'activity'] = 5
features_dataset.loc[(features_dataset['activiteit'] == 'fietsen zwaar'),␣
↪'activity'] = 5

features_dataset.drop('activiteit', axis=1, inplace=True)
features_dataset.dropna(how='any', inplace=True)

features_dataset.head()
```

[33]:

```
                         standard_deviation_x    mean_x  standard_deviation_y  \
2019-09-12 10:25:05.400              0.433126 -0.671986              0.133191
2019-09-12 10:25:14.800              0.421084 -0.679500              0.122672
2019-09-12 10:25:24.200              0.447636 -0.665231              0.126320
2019-09-12 10:25:33.600              0.441681 -0.689547              0.128438
2019-09-12 10:25:43.000              0.449540 -0.659068              0.142767

                           mean_y  standard_deviation_z    mean_z  activity
2019-09-12 10:25:05.400  0.136694              0.157682  0.843887         5
2019-09-12 10:25:14.800  0.129010              0.152290  0.861871         5
2019-09-12 10:25:24.200  0.130024              0.170394  0.852837         5
2019-09-12 10:25:33.600  0.126309              0.166221  0.861364         5
2019-09-12 10:25:43.000  0.132303              0.162122  0.850473         5
```

1.1 Preparing feature dataset for learning

1.1.1 Splitting in x and y

[34]:

```
x = features_dataset[features_columns[:-1]]
y = features_dataset[activity_columns]

x_train, x_valid, y_train, y_valid = train_test_split(x,y, test_size=0.3,␣
↪random_state=23, stratify=y)

x_train.head()
```

[34]:

```
                         standard_deviation_x    mean_x  standard_deviation_y  \
2019-09-30 14:14:15.800              0.376362 -1.025076              0.247497
2019-10-09 12:27:41.400              0.000000 -1.063492              0.000000
2019-09-30 15:50:17.600              0.007513 -0.312141              0.008811
2019-09-30 12:52:14.200              0.423447 -0.618288              0.201820
2019-10-10 15:43:00.600              0.943503 -1.054289              0.528084
```

```
                           mean_y  standard_deviation_z    mean_z
2019-09-30 14:14:15.800  0.138467              0.352398  0.210486
2019-10-09 12:27:41.400 -0.015873              0.000000 -0.047619
2019-09-30 15:50:17.600  0.099629              0.003279  1.174772
2019-09-30 12:52:14.200  0.262411              0.118608  0.903749
2019-10-10 15:43:00.600  0.158561              0.827365  0.128926
```

[35]:
```
y_train.head()
```

[35]:
```
                         activity
2019-09-30 14:14:15.800         1
2019-10-09 12:27:41.400         3
2019-09-30 15:50:17.600         4
2019-09-30 12:52:14.200         5
2019-10-10 15:43:00.600         2
```

1.2 Logistic Regression

[47]:
```
lr = LogisticRegression(multi_class='multinomial')

lr.fit(x_train, y_train)

predictions = lr.predict(x_valid)
```

```
/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)
```

1.2.1 Result

Accuracy

[48]:
```
accuracy_score(y_valid, predictions, normalize=True)
```

[48]:
```
0.9648390941597139
```

Classification report

[49]:
```
# 1 : lopen
# 2 : rennen
# 3 : staan
# 4 : zitten
# 5 : fietsen
```

```
print(classification_report(y_valid, predictions, zero_division=0))
```

```
              precision    recall  f1-score   support

           1       0.94      0.98      0.96       284
           2       0.98      0.93      0.95       280
           3       0.96      0.94      0.95       278
           4       0.92      0.99      0.95       279
           5       1.00      0.98      0.99       557

    accuracy                           0.96      1678
   macro avg       0.96      0.96      0.96      1678
weighted avg       0.97      0.96      0.96      1678
```

1.2.2 Confusion matrix

[44]:
```
import seaborn as sn

#confusion_matrix(valid_y, prediction_y)
cm = confusion_matrix(y_valid.values.argmax(axis=1), predictions.
 ↪argmax(axis=1), normalize='true')

df_cm = pd.DataFrame(cm, index=activity_columns, columns=activity_columns)
df_cm.head()
plt.figure(figsize = (10,7))
sn.heatmap(df_cm, annot=True, cmap='Blues')

plt.title("Validation dataset")
plt.xlabel("predicted label")
plt.ylabel("true label")
```

```
---------------------------------------------------------------------------
AxisError                                 Traceback (most recent call last)
<ipython-input-44-f0412aed1f8d> in <module>
      2
      3 #confusion_matrix(valid_y, prediction_y)
----> 4 cm = confusion_matrix(y_valid.values.argmax(axis=1), predictions.
 ↪argmax(axis=1), normalize='true')
      5
      6 df_cm = pd.DataFrame(cm, index=activity_columns,␣
 ↪columns=activity_columns)

AxisError: axis 1 is out of bounds for array of dimension 1
```

1.3 Diagnostics

1.3.1 Cross validation analysis

[46]:
```
from sklearn.model_selection import cross_val_score
```

[50]:
```
accuracy_scores =␣
↪cross_val_score(LogisticRegression(multi_class='multinomial'), x, y, cv=5,␣
↪scoring='accuracy')
recall_scores = cross_val_score(LogisticRegression(multi_class='multinomial'),␣
↪x, y , cv=5, scoring='recall_micro')
precision_scores =␣
↪cross_val_score(LogisticRegression(multi_class='multinomial'), x, y , cv=5,␣
↪scoring='precision_micro')

print("Accuracy: %0.2f (+/- %0.2f)" % (accuracy_scores.mean(), accuracy_scores.
↪std() ))
print("Recall: %0.2f (+/- %0.2f)" % (recall_scores.mean(), recall_scores.std()␣
↪))
print("Precision: %0.2f (+/- %0.2f)" % (precision_scores.mean(),␣
↪precision_scores.std() ))
```

```
/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/linear_model/_logistic.py:764: ConvergenceWarning: lbfgs failed
to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
/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/linear_model/_logistic.py:764: ConvergenceWarning: lbfgs failed
to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
/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/linear_model/_logistic.py:764: ConvergenceWarning: lbfgs failed
to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
/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/linear_model/_logistic.py:764: ConvergenceWarning: lbfgs failed
to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
/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/linear_model/_logistic.py:764: ConvergenceWarning: lbfgs failed
to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)
/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/linear_model/_logistic.py:764: ConvergenceWarning: lbfgs failed
to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  extra_warning_msg=_LOGISTIC_SOLVER_CONVERGENCE_MSG)

Accuracy: 0.95 (+/- 0.02)
Recall: 0.95 (+/- 0.02)
Precision: 0.95 (+/- 0.02)
```

[ ]: