Data-preprocessing_activpal20_all_respondents_walking

January 12, 2021

[1]:
```
from helpers import math_helper as mth
from helpers import pandas_helper as pdh
from sensors.activpal import *
from scipy import stats
from sklearn.preprocessing import StandardScaler

import pandas as pd
import seaborn as sns

activpal = Activpal()

activity_focus = 'lopen'
training_validation_respondents = ['BMR002', 'BMR004', 'BMR011', 'BMR012',␣
 ↪'BMR014', 'BMR018', 'BMR031', 'BMR032', 'BMR033', 'BMR034', 'BMR036',␣
 ↪'BMR041', 'BMR042', 'BMR043', 'BMR044', 'BMR052', 'BMR053', 'BMR055',␣
 ↪'BMR058', 'BMR064', 'BMR097', 'BMR098']
```

0.1 Defining helper functions

[2]:
```
def get_timestamps(respondent):
    activities_df = read_functions.read_activities(respondent)
    start = activities_df.loc[activity_focus].start
    stop = activities_df.loc[activity_focus].stop

    return (start, stop)
```

[3]:
```
def get_speed(x_acc, y_acc, z_acc):
    activpal_time = 0.05

    x_vel = x_acc.diff()**2
    y_vel = y_acc.diff()**2
    z_vel = z_acc.diff()**2

    return (x_vel + y_vel + z_vel) / activpal_time
```

[4]:
```
def get_respondent_number(respondent):
    if ('BMR0' in respondent):
        return int(respondent.replace('BMR0', ''))

    if ('BMR' in respondent):
        return int(respondent.replace('BMR', ''))
```

[5]:
```
def remove_upper_outliers(series):
    z_score = 2.5
    mean = series.mean()
    return series < mean*z_score
```

[6]:
```
def get_activpal20_df(respondent, start, stop):
    df = activpal.read_data(respondent, start, stop)

    mask = (df.index >= start) & (df.index < stop)

    df = df.loc[mask]
    df = df[['pal_accX', 'pal_accY', 'pal_accZ']].apply(mth.convert_value_to_g)
    df['sum_mag_acc'] = mth.to_mag_acceleration(df['pal_accX'], df['pal_accY'],␣
 ↪df['pal_accZ'])
    df['mean_speed'] = get_speed(df['pal_accX'], df['pal_accY'], df['pal_accZ'])

    return df
```

[7]:
```
def get_vyntus_df(respondent, weight, start, stop):
    vyntus_df = pdh.read_csv_vyntus(respondent)
    mask = (vyntus_df.index >= start) & (vyntus_df.index < stop)
    vyntus_df = vyntus_df.loc[mask]

    min_index = vyntus_df.index.min()
    max_index = vyntus_df.index.max()

    vyntus_df['vyn_VO2'] = [float(vo2.replace(',', '.')) if type(vo2) == str␣
 ↪else vo2 for vo2 in vyntus_df['vyn_VO2']]
    vyntus_df['met'] = mth.calculate_met(vyntus_df['vyn_VO2'], weight)

    #vyntus_df = vyntus_df.resample('15s').mean()[:-1]

    return (vyntus_df[['met', 'vyn_VO2']], min_index, max_index)
```

[8]:
```
def get_regression_df(respondent):
    start, stop = get_timestamps(respondent)

    # read in all dataframes necessary
    respondents_df = pdh.read_csv_respondents_cleaned()
```

```
    res_number = get_respondent_number(respondent)

    vyntus_df, min_index, max_index = get_vyntus_df(respondent,␣
↪respondents_df['weight_kg'][res_number], start, stop)

    activpal20_df = get_raw_df(respondent, min_index, max_index)

    # add met and mag acc to new dataframe
    new_df = pd.DataFrame(index=raw_df.index)
    new_df['mean_met'] = vyntus_df['met']
    new_df['sum_mag_acc'] = raw_df['sum_mag_acc']

    # add features to new dataframe
    new_df['length_cm'] = respondents_df['lengte'][res_number]
    new_df['weight_kg'] = respondents_df['gewicht'][res_number]
    new_df['mean_speed'] = raw_df['mean_speed']
    new_df['bmi'] = mth.calculate_bmi(new_df['weight_kg'], new_df['length_cm'])
    new_df['gender'] = int(respondents_df['geslacht'][res_number].
↪replace('vrouw',str(0)).replace('man', str(1)))
    new_df['age_category'] = respondents_df['leeftijdscategorie'][res_number]
    convert_age_to_number(new_df, "age_category")
    new_df['is_sporter'] = respondents_df['sporter'][res_number]
    new_df['meets_activity_guidelines'] = int(respondents_df['voldoet aan␣
↪beweegrichtlijn 2nee17'][res_number].replace('ja', '1').replace('nee', '0'))
    new_df['does_muscle_bone_exercises'] = int(respondents_df['voldoet aan␣
↪richtlijn bot en spierversterkende activiteiten'][res_number].replace('ja',␣
↪'1').replace('nee', '0'))
    new_df['meets_balance_guidelines'] = int(respondents_df['voldoet aan␣
↪richtlijn balansoefeningen'][res_number].replace('ja', '1').replace('nee',␣
↪'0'))

    return new_df
```

[9]:
```
def get_all_activpal20_vyntus_dfs():
    all_activpal20_dfs = pd.DataFrame(index=pd.to_datetime([]))
    all_vyntus_dfs = pd.DataFrame(index=pd.to_datetime([]))
    respondents_df = pdh.read_csv_respondents_cleaned()

    for respondent in training_validation_respondents:
        start, stop = get_timestamps(respondent)

        vyntus_df, min_index, max_index = get_vyntus_df(respondent,␣
↪respondents_df['weight_kg'][get_respondent_number(respondent)], start, stop)
        activpal20_df = get_activpal20_df(respondent, min_index, max_index)

        vyntus_df['respondent'] = respondent
```

```
        activpal20_df['respondent'] = respondent

        all_activpal20_dfs = pd.concat([activpal20_df, all_activpal20_dfs])
        all_vyntus_dfs = pd.concat([vyntus_df, all_vyntus_dfs])

    all_activpal20_dfs.sort_index(inplace=True)
    all_vyntus_dfs.sort_index(inplace=True)

    return (all_activpal20_dfs, all_vyntus_dfs)
```

[10]:
```
all_activpal20_dfs, all_vyntus_dfs = get_all_activpal20_vyntus_dfs()
```

[11]:
```
all_activpal20_dfs.columns
```

[11]:
```
Index(['pal_accX', 'pal_accY', 'pal_accZ', 'sum_mag_acc', 'mean_speed',
       'respondent'],
      dtype='object')
```

[12]:
```
all_vyntus_dfs.columns
```

[12]:
```
Index(['met', 'vyn_VO2', 'respondent'], dtype='object')
```

1 Data Preprocessing

1.1 Missing Values

[13]:
```
all_activpal20_dfs.isnull().sum()
```

[13]:
```
pal_accX        0
pal_accY        0
pal_accZ        0
sum_mag_acc     0
mean_speed     22
respondent      0
dtype: int64
```

In the case of mean_speed, the null values is when they start walking, so put it to 0

[14]:
```
all_activpal20_dfs.mean_speed.fillna(0, inplace=True)
all_activpal20_dfs.isnull().sum()
```

[14]:
```
pal_accX       0
pal_accY       0
pal_accZ       0
sum_mag_acc    0
```

```
mean_speed     0
respondent     0
dtype: int64
```

[15]:
```
all_vyntus_dfs.isnull().sum()
```

[15]:
```
met           0
vyn_VO2       0
respondent    0
dtype: int64
```

1.2 Convert values to numerical

[16]:
```
all_activpal20_dfs.dtypes
```

[16]:
```
pal_accX       float64
pal_accY       float64
pal_accZ       float64
sum_mag_acc    float64
mean_speed     float64
respondent      object
dtype: object
```

[17]:
```
all_vyntus_dfs.dtypes
```

[17]:
```
met           float64
vyn_VO2       float64
respondent     object
dtype: object
```

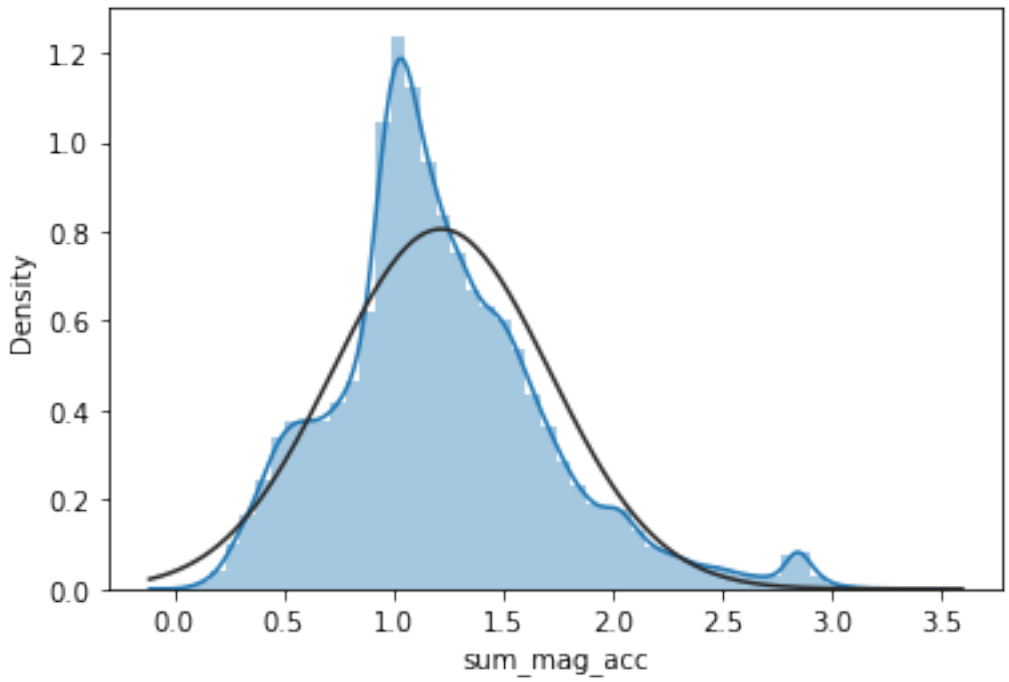
1.3 Remove Outliers

1.3.1 Outliers sum_mag_acc

[18]:
```
sns.distplot(all_activpal20_dfs.sum_mag_acc, fit=stats.norm)
```

```
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/seaborn/distributions.py:2551: FutureWarning: `distplot` is a
deprecated function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar flexibility)
or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

[18]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f01464ab048>
```

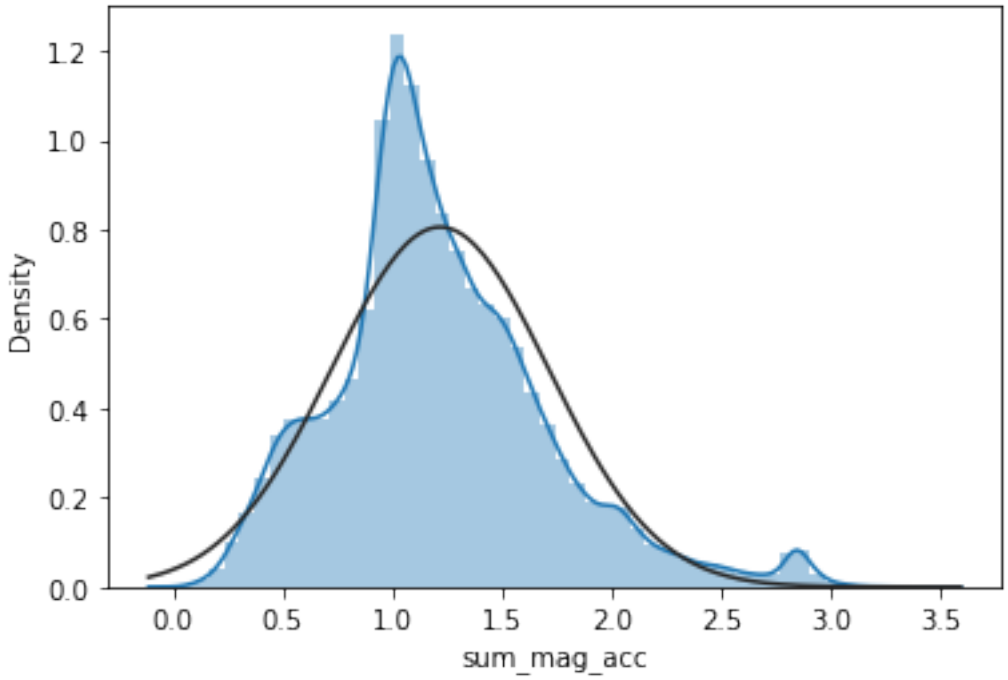


Remove only outliers 3.5 times the mean from only the high end, since the low end is when the person starts walking so it makes sense for the values to start from 0

[19]:
```
high_range = all_activpal20_dfs.sum_mag_acc.mean() * 2.5
all_activpal20_dfs_cleaned = all_activpal20_dfs#[all_activpal20_dfs.sum_mag_acc␣
↪< high_range]
sns.distplot(all_activpal20_dfs_cleaned.sum_mag_acc, fit=stats.norm)
```

```
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/seaborn/distributions.py:2551: FutureWarning: `distplot` is a
deprecated function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar flexibility)
or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

[19]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f014602de80>
```

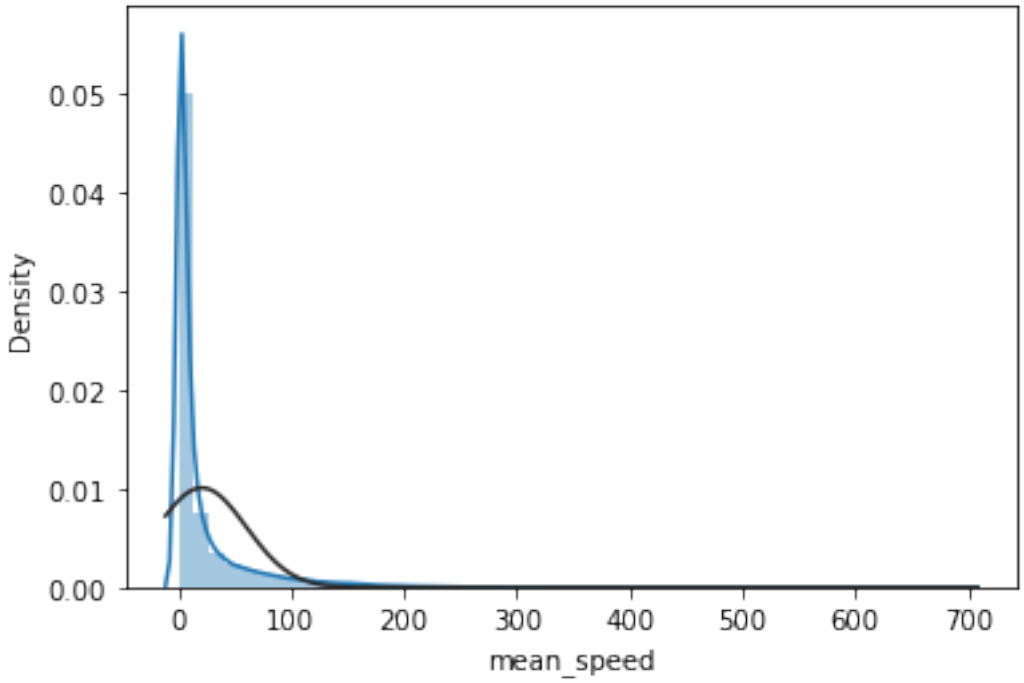


1.3.2 Outliers mean_speed

[20]:
```
sns.distplot(all_activpal20_dfs_cleaned.mean_speed, fit=stats.norm)
```

```
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/seaborn/distributions.py:2551: FutureWarning: `distplot` is a
deprecated function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar flexibility)
or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

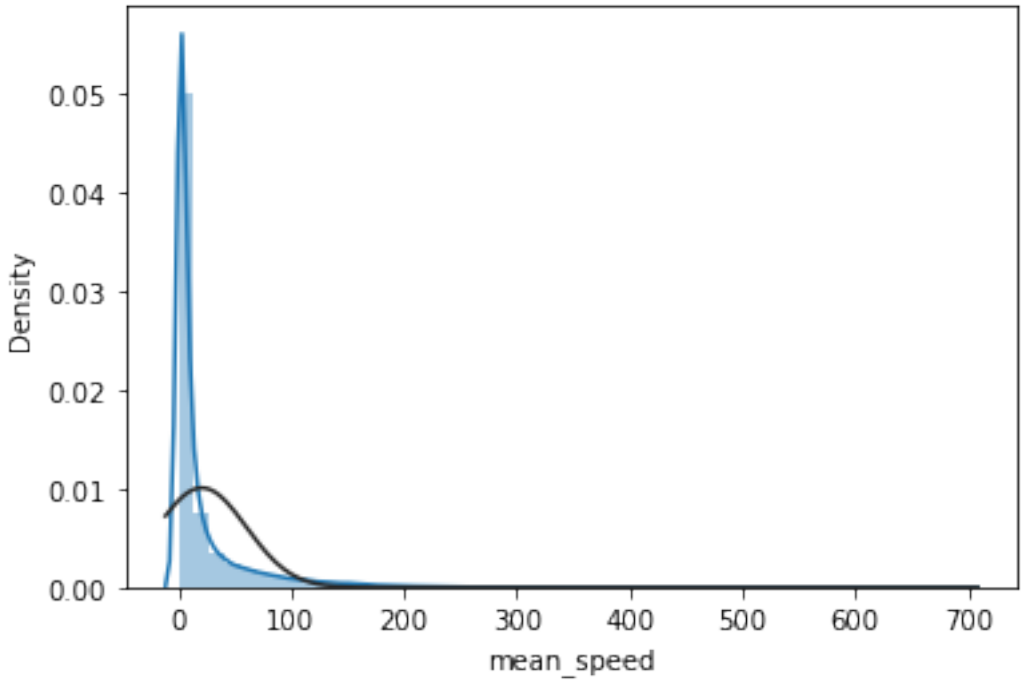
[20]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f01456b01d0>
```



[21]:
```
high_range = all_activpal20_dfs_cleaned.mean_speed.mean() * 2.5
#all_activpal20_dfs_cleaned =␣
 ↪all_activpal20_dfs_cleaned[all_activpal20_dfs_cleaned.mean_speed <␣
 ↪high_range]
sns.distplot(all_activpal20_dfs_cleaned.mean_speed, fit=stats.norm)
```

```
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/seaborn/distributions.py:2551: FutureWarning: `distplot` is a
deprecated function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar flexibility)
or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

[21]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f0144093748>
```

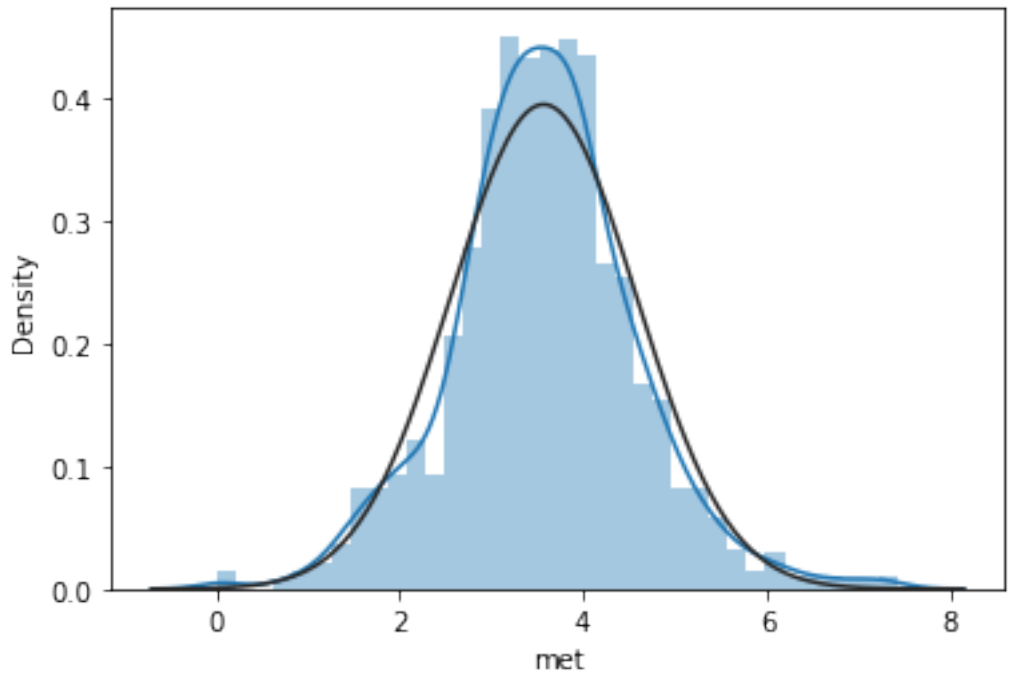


1.4 Outliers MET

[22]:
```
sns.distplot(all_vyntus_dfs.met, fit=stats.norm)
```

```
/opt/jupyterhub/anaconda/lib/python3.6/site-
packages/seaborn/distributions.py:2551: FutureWarning: `distplot` is a
deprecated function and will be removed in a future version. Please adapt your
code to use either `displot` (a figure-level function with similar flexibility)
or `histplot` (an axes-level function for histograms).
  warnings.warn(msg, FutureWarning)
```

[22]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f013478c908>
```



MET is normally distributed with no extreme outliers, no need for removing outliers

1.5 Resample, add respondent features and save cleaned df

[23]:
```
all_regression_dfs = pd.DataFrame(index=pd.to_datetime([]))
respondents_df = pdh.read_csv_respondents_cleaned()
print(respondents_df.columns)

for respondent, vyntus_df in all_vyntus_dfs.groupby('respondent'):
    resp_number = get_respondent_number(respondent)
    activpal20_df = all_activpal20_dfs_cleaned.loc[all_activpal20_dfs_cleaned.
 ↪respondent == respondent].resample('15s').agg({'sum_mag_acc':'sum',␣
 ↪'mean_speed':'mean'})
    vyntus_df = vyntus_df.resample('15s').mean()[:-1]

    regression_df = pd.concat([activpal20_df, vyntus_df])

    regression_df['respondent'] = respondent
    regression_df['estimated_level'] = respondents_df.loc[resp_number,␣
 ↪'estimated_level']
    regression_df['length_cm'] = respondents_df.loc[resp_number, 'length_cm']
    regression_df['weight_kg'] = respondents_df.loc[resp_number, 'weight_kg']
    regression_df['waist_circumference'] = respondents_df.loc[resp_number,␣
 ↪'waist_circumference']
```

```
    regression_df['gender'] = respondents_df.loc[resp_number, 'gender']
    regression_df['age_category'] = respondents_df.loc[resp_number,␣
 ↪'age_category']

    all_regression_dfs = pd.concat([regression_df, all_regression_dfs])

all_regression_dfs.sort_index(inplace=True)
print(all_regression_dfs.isnull().sum())
print(all_regression_dfs.shape)
```

```
Index(['estimated_level', 'length_cm', 'weight_kg', 'waist_circumference',
       'gender', 'age_category', 'running_speed_km'],
      dtype='object')
sum_mag_acc            430
mean_speed             430
met                    449
vyn_VO2                449
respondent               0
estimated_level          0
length_cm                0
weight_kg                0
waist_circumference      0
gender                   0
age_category             0
dtype: int64
(879, 11)
```

[ ]: