import pandas as pd import numpy as np import matplotlib.pyplot as plt import logging as log from dataclasses import dataclass import sys

@dataclass class DataConfig: window\_duration:float=30

def plot(df, title): ’’’ Plotting the dataframe with pupil\_timestamps on x-axis and diameter values on y-axis in a line chart fro checkdata notebook. Creating histograms showing the distribution of the variables confidence, diameter and diameter\_3d.

parameter  
 ---------  
 df: Dataframe which is plotted.   
 title: Title for the plot.  
 '''  
 fig, ax = plt.subplots(2,2)  
 ax[0,0].set\_title(title)  
 sub=df.plot(ax=ax[0,0],x='pupil\_timestamp', y='diameter', kind='line')  
 df.plot(ax=sub,x='pupil\_timestamp', y='diameter\_100', kind='line')  
 ax[0,1].set\_title("confidence")  
 ax[0,1].hist(df['confidence'], bins=100, density=True)  
 ax[1,0].set\_title("diameter")  
 ax[1,0].hist(df['diameter'], bins=100, density=True)  
 ax[1,1].set\_title("diameter\_3d")  
 ax[1,1].hist(df['diameter\_3d'], bins=100, density=True)  
 plt.show()

def load\_df(path, usecols): ’’’ Controlling csv dataframe if pickle file already exists. If already created, pickle file is loaded. If no pickle file is found, it is created and the csv dataframe included.

parameter  
---------  
 path: Path to files which are loaded.   
 usecols: Columns which are selected.  
'''  
import os  
import pandas as pd  
if os.path.exists(f"{path}.pickle"):  
 log.info("read pickle")  
 return pd.read\_pickle(f"{path}.pickle")   
df=pd.read\_csv(path,index\_col=False, usecols=usecols)  
df.to\_pickle(f"{path}.pickle")  
return df

def prepare(data\_dir,subject\_id,eye\_id, config:DataConfig): ’’’ Load dataframes used for the code. Select the important columns. Load the annotation\_timestamps and slice the dataframes into replicates. Add a timeslot column.

###Add Moving Average???  
###only used in checkdata notebook  
  
parameter  
---------  
 data\_dir: Path to files which are loaded.   
 subject\_id: Subject\_id, important for selecting the files.  
 eye\_id: Eye side.  
 config:DataConfig: Selected dataclass.  
'''  
csv\_cols = ['pupil\_timestamp', 'diameter\_3d', 'diameter','eye\_id','confidence']  
df=load\_df(f"{data\_dir}/{subject\_id[:4]}/{subject\_id}/exports/000/pupil\_positions.csv", csv\_cols)  
# df = pd.read\_csv(f"{data\_dir}/{subject\_id[:4]}/{subject\_id}/exports/000/pupil\_positions.csv", index\_col=False, usecols=csv\_cols)  
# add moving average for the whole dataset  
df['diameter\_100']=df['diameter'].rolling(window=100).mean()  
annotation\_timestamps = np.load(f"{data\_dir}/{subject\_id[:4]}/{subject\_id}/annotation\_timestamps.npy")  
res=[]  
for annotation\_timestamp in annotation\_timestamps:  
 # Calculate the start and end timestamps for the window after the annotation  
 window\_start = annotation\_timestamp - 5.0  
 window\_end = window\_start + config.window\_duration  
   
 # Select the rows that fall within the window  
 df\_sliced = df[(df['pupil\_timestamp'].between(window\_start,window\_end)) & (df['eye\_id'] == eye\_id)]  
 df\_sliced=df\_sliced.copy()  
 df\_sliced['pupil\_timestamp\_based'] = df\_sliced['pupil\_timestamp'] - window\_start  
 # Do more cleanup  
  
 # Add a timeslot column  
 df\_sliced['rowid'] = range(len(df\_sliced))   
 df\_sliced['timeslot'] = df\_sliced['rowid'] // 1000  
 res.append(df\_sliced)   
return res

# blinkreconstruct for a pandas series. Returns a numpy array.

# see https://pydatamatrix.eu/0.15/series/#function-blinkreconstructseries-vt5-vt\_start10-vt\_end5-maxdur500-margin10-smooth\_winlen21-std\_thr3-gap\_margin20-gap\_vt10-modeuoriginal

def blinkreconstruct(df, vt=5, vt\_start=10, vt\_end=5, maxdur=500, margin=10, smooth\_winlen=21, std\_thr=3, gap\_margin=20, gap\_vt=10, mode=u’advanced’): ’’’ Blinkreconstruct for a pandas series. Returns a numpy array. see https://pydatamatrix.eu/0.15/series/#function-blinkreconstructseries-vt5-vt\_start10-vt\_end5-maxdur500-margin10-smooth\_winlen21-std\_thr3-gap\_margin20-gap\_vt10-modeuoriginal ’’’

display(type(df))  
import datamatrix  
import datamatrix.series  
import datamatrix.operations  
dm=datamatrix.convert.from\_pandas(df).series  
return datamatrix.series.blinkreconstruct(dm, vt,vt\_start,vt\_end,maxdur,margin,smooth\_winlen,std\_thr,gap\_margin,gap\_vt,mode)

if **name**==“**main**”: import pandas as pd subject\_id=“PJ02\_1\_Ruhe” window\_duration=30 data\_dir=“./Users/Katharina/Desktop/Beispieldaten/” config=DataConfig(window\_duration=30) prepare(data\_dir,subject\_id,0,config)