def drop\_columns(df, thresh\_perc, ncols = None, nrows = None):

""" Drop columns which have more than 50% missing

:param data: dataframe

:return: dataframe

"""

if ncols == None: ncols = len(df.columns)

if nrows == None: nrows = len(df)

nmissing = list(df.isna().sum().compute())

colstodrop = [a for a in range(ncols) if nmissing[a]/nrows\*100 >= thresh\_perc]

colstodrop = [df.columns[a] for a in colstodrop]

return df.drop(colstodrop, axis='columns')

#delete columns (features) with more than N % missing observations from training data set:

missing\_perc = 50

x\_train\_df = drop\_columns(x\_train\_df, missing\_perc, x\_train\_df\_ncols, x\_train\_df\_nrows)

diff = x\_train\_df\_ncols - len(x\_train\_df.columns)

if diff > 0:

print('Deleted', diff, 'features because they had more than', missing\_perc, 'percent missing observations')

x\_train\_df\_ncols = len(x\_train\_df.columns)

else:

print('No features had more than', missing\_perc, 'percent missing observations')

#impute mean for all features:

#calculate mean (note that axis needs to be 0 to get columns, which is weird)

miin = x\_train\_df.mean(axis = 0).compute()

x\_train\_df\_imean = x\_train\_df.fillna(dict(miin))

print(percent\_missing(x\_train\_df\_imean))