CSE572 – Data Mining

Submission data: February 25, 2020

Roey Ben Hayun

Assignment #2: User Dependent and Independent Analysis Project

[User Dependent and Independent Analysis 1](#_Toc33257698)

[Background 1](#_Toc33257699)

[Phase One: User Dependent Analysis 2](#_Toc33257700)

[Dataset split 2](#_Toc33257701)

[Selected ML methods 2](#_Toc33257702)

[Classification Report 2](#_Toc33257703)

[Phase Two: User Independent Analysis 2](#_Toc33257704)

[Dataset split 2](#_Toc33257705)

[Selected ML methods 2](#_Toc33257706)

[Classification Report 2](#_Toc33257707)

[Code and environment setup 3](#_Toc33257708)

[How to run 3](#_Toc33257709)

[Reference: 4](#_Toc33257710)

# User Dependent and Independent Analysis

## Background

The reduced feature matrix (after PCA) is the input for the user dependent and independent analysis.

I have stacked (vertically) the four matrices:

eating with fork (30x50)

eating with spoon (30x50)

non-eating with fork(30x50)

non-eating with spoon (30x50)

Into one matrix:

Feature matrix (120x50)

To reduced dimensionality, I have used PCA on the feature matrix and received new feature matrix:

reduced feature matrix (120x5)

For this report, this matrix is called X

I have also created a list(vector) of labels (120,1) with eating and non-eating labels (the first 60 are labeled ‘eating’ and the rest are labeled ‘non-eating’

For this report, this vector is called Y

## Phase One: User Dependent Analysis

For this part, I have been asked to split the new feature set (X) 60% for training and 40% for testing.

### Dataset split

For the split I used python

train\_test\_split

The inputs for this method are:

X : the feature matrix

Y : the labels vector

test\_size – the split ratio (0.4 in our case)

random\_state – set to const value (42) to make sure to get the same split every run.

This is the split result

X\_train shape: (72,5)

X\_test shape :(48,5)

Y\_train shape :(72,)

Y\_test shape : (48,)

### 

### Selected ML methods

Three ML methods were selected: decision trees, SVM and Neural Networks.

Python sklearn supports those classifiers. The pattern for using sklearn classifier is as follows:

1. Create a classifier object
2. fit(X\_train, Y\_train)
3. predict(X\_test)

As we can see, steps 2 and 3 are the same for all the classifiers.

### Classification Report

## 

## Phase Two: User Independent Analysis

### Dataset split

### Selected ML methods

### Classification Report

## Code and environment setup

Python3

Linux/OSx

Dependencies:

import scipy.io

import matplotlib.pyplot as plt

import numpy as np

import os

import math

from tempfile import TemporaryFile

import re

from sklearn.decomposition import PCA

import pandas as pd

import seaborn as sns

from sklearn.pipeline import make\_pipeline

from sklearn import preprocessing

from sklearn.preprocessing import StandardScaler

import errno

import os

import sys

### How to run

python3 main.py

This will create two directories:

‘out’ and ‘plot’

In the ‘out’ directory you will find all the extracted IMU data for all users and activities (eating, non-eating, spoon and fork).

In the ‘plot’ directory you will find the plots (which I have used for this report) organized in sub directories.

Important – make sure to delete of the ‘out’ and ‘plots’ directories before running again.

## Reference: