Supervised Learning -Falling Prediction using KNN

Submitted by Manish Bafna
Student Id: 19655
Instructor: Dr. Henry Chang
GIT: MachineLearning-Overfitting

Table of Content

Introduction

Design

Implementation

Test

Conclusion

Enhancement Ideas

References

Introduction

Injuries due to unintentional falls cause high social cost in which several systems have been developed to reduce them.

- The market is dominated by fall detection systems, which activate an alarm after a fall occurrence
- Personal devices, such as smartphones, are being exploited for implementing fall systems, because they are commonly carried by the user most of the day

there are systems with the goal of predicting and preventing a fall, called fall prediction and prevention systems (FPPSs)

Such systems track and report data from wearable sensors without engaging the users in the monitoring process

Introduction

FPPSs include sensors to collect data and software applications to process them

a machine learning algorithm is applied on the obtained data

Since smartphones are used as personal digital assistance and are equipped with precise sensors and communication component, they are commonly used in FPPSs to monitor and collect data.

Accelerometer and gyroscope are sensors embedded in a smartphone that can be used in fall prediction and prevention systems

Kinematic features obtained from the data collected from accelerometer and gyroscope can be evaluated in combination with different machine learning algorithms to predict and prevent fall

Introduction

Accelerometer: An accelerometer is a device that measures acceleration, i.e, the rate of change of the velocity of an object.

Accelerometer let iOS

know how the device is being held and if it is being moved.

handles autorotation, and many games use it as a control mechanism.

detects shakes and other sudden movement.

Introduction

Gyroscope: A gyroscope gives the angular rate around one or more axes of the space. Angular measurement around lateral, longitudinal and vertical plane are referred to as pitch, roll and yaw, respectively. Typically, in FPPSs, the gyroscope is used in combination with an accelerometer.

Ios includes a gyroscope sensor that lets you read values describing the devices rotation around its axes.

Rather than registering an absolute rotation value, the gyroscopes tell you about changes to the devices rotation as they happen.

Accelerometer and gyroscope are sensors embedded in a smartphone that can be used in fall prediction and prevention systems

Introduction

Features of the acceleration signal of human upper trunk in a short time interval before the fall are denoted as λ. After obtaining the ATS of the user, P(ATS|λ) states the probability of a fall occurrence during the user motion. Two thresholds P1 and P2 are specified to predict and detect a fall

The output of P(TS|λ) is an input to the algorithm. Then, based on predefined thresholds, if P is higher than P1, the fall risk is notified, if P is higher than P2, a possible fall is noticed.
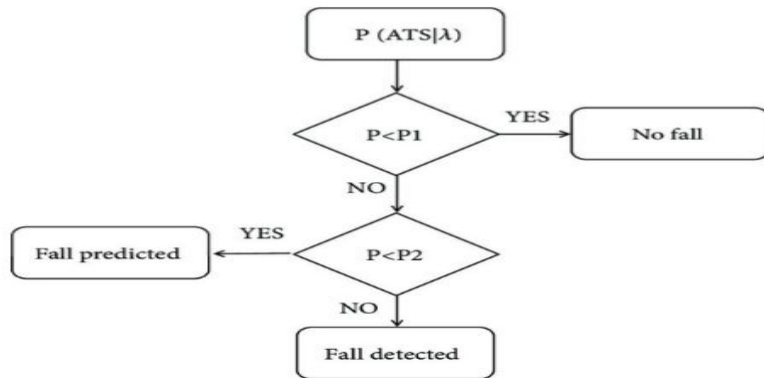
Introduction

The k-nearest neighbors (KNN) algorithm is a simple, easy-to-implement supervised machine learning algorithm that can be used to solve both classification and regression problems

- K nearest neighbor algorithm is based on minimum distance from the query instance to the training samples to determine the K-nearest neighbors.
- After we gather K nearest neighbors, we take simple majority of these K-nearest neighbors to be the prediction of the query instance.
- The data for KNN algorithm consist of several multivariate attributes name $_i$ that will be used to classify Y.
    - The data of KNN can be any measurement scale from ordinal, nominal, to quantitative scale but for the moment let us deal with only quantitative $_i$ and binary (nominal) Y

Design

Using KNN to manually calculate the distance and predict the result.

● This is the tranining data and the test data:

Accelerometer Data			Gyroscope Data			Fall (+), Not (-)
x	y	z	x	y	z	+/-
1	2	3	2	1	3	-
2	1	3	3	1	2	-
1	1	2	3	2	2	-
2	2	3	3	2	1	-
6	5	7	5	6	7	+
5	6	6	6	5	7	+
5	6	7	5	7	6	+
7	6	7	6	5	6	+
7	6	5	5	6	7	??

Design

- Use Python to implement the application of using kNN to predict fall.
- Create the code by modifying KNN from scratch
  - Explain the code
- Run the code on Colab

- Compare the result from the Python program and the result of manual calculation.

Implementation

Manually calculate the distance and predict the result for the data set using KNN

Calculate the K

K = sqrt(number of data samples)

= sqrt (7)

= 2.64

=3

Implementation

Accelerometer Data			Gyroscope Data			Fall (+), Not (-)	Distance to each neighbor for Accelerometer and Gyroscope	K =Number of nearest neighbors =sqrt(number of neighbors) =sqrt(number of data samples) =sqrt(7) =3
x	y	z	x	y	z	+/-		
1	2	3	2	1	3	-	$((7-1)^2+(6-2)^2+(5-3)^2+(5-2)^2+(6-1)^2+(7-3)^2)=106$	
2	1	3	3	1	2	-	$((7-2)^2+(6-1)^2+(5-3)^2+(5-3)^2+(6-1)^2+(7-2)^2)=108$	
1	1	2	3	2	2	-	$((7-1)^2+(6-1)^2+(5-2)^2+(5-3)^2+(6-2)^2+(7-2)^2)=115$	
2	2	3	3	2	1	-	$((7-2)^2+(6-2)^2+(5-3)^2+(5-3)^2+(6-2)^2+(7-1)^2)=101$	
6	5	7	5	6	7	+	$((7-6)^2+(6-5)^2+(5-7)^2+(5-5)^2+(6-6)^2+(7-7)^2)=6$	+
5	6	6	6	5	7	+	$((7-5)^2+(6-6)^2+(5-6)^2+(5-6)^2+(6-5)^2+(7-7)^2)=7$	+
5	6	7	5	7	6	+	$((7-5)^2+(6-6)^2+(5-7)^2+(5-5)^2+(6-7)^2+(7-6)^2)=10$	
7	6	7	6	5	6	+	$((7-7)^2+(6-6)^2+(5-7)^2+(5-6)^2+(6-5)^2+(7-6)^2)=7$	+
7	6	5	5	6	7	??(+)		

Implementation

Calculate the Euclidean distance between two vwectors

Euclidean Distance = sqrt(sum i to N (X1_i - X2_i)^2)

#Result:

10.29563

10.3923

10.72381

10.04988

2.44949

2.645751

3.162278

2.645751

Implementation

Locate the most similar neighbors

[6,5,7,5,6,7] => **+**

[5,6,6,6,5,7] => **+**

[7,6,7,6,5,6] => **+**

Hence [7,6,5,5,6,7] => +

Implementation

KNN Algorithm

Dataset =

[[7,6,5,5,6,7,1],[1,2,3,2,1,3,0],[2,1,3,3,1,2,0],[1,1,2,3,2,2,0],[2,2,3,3,2,1,0],[6,5,7,5,6,7,1],[5,6,6,6,5,7,1],[5,6,7,5,7,6,1],[7,6,6,7,6,5,6,1]]
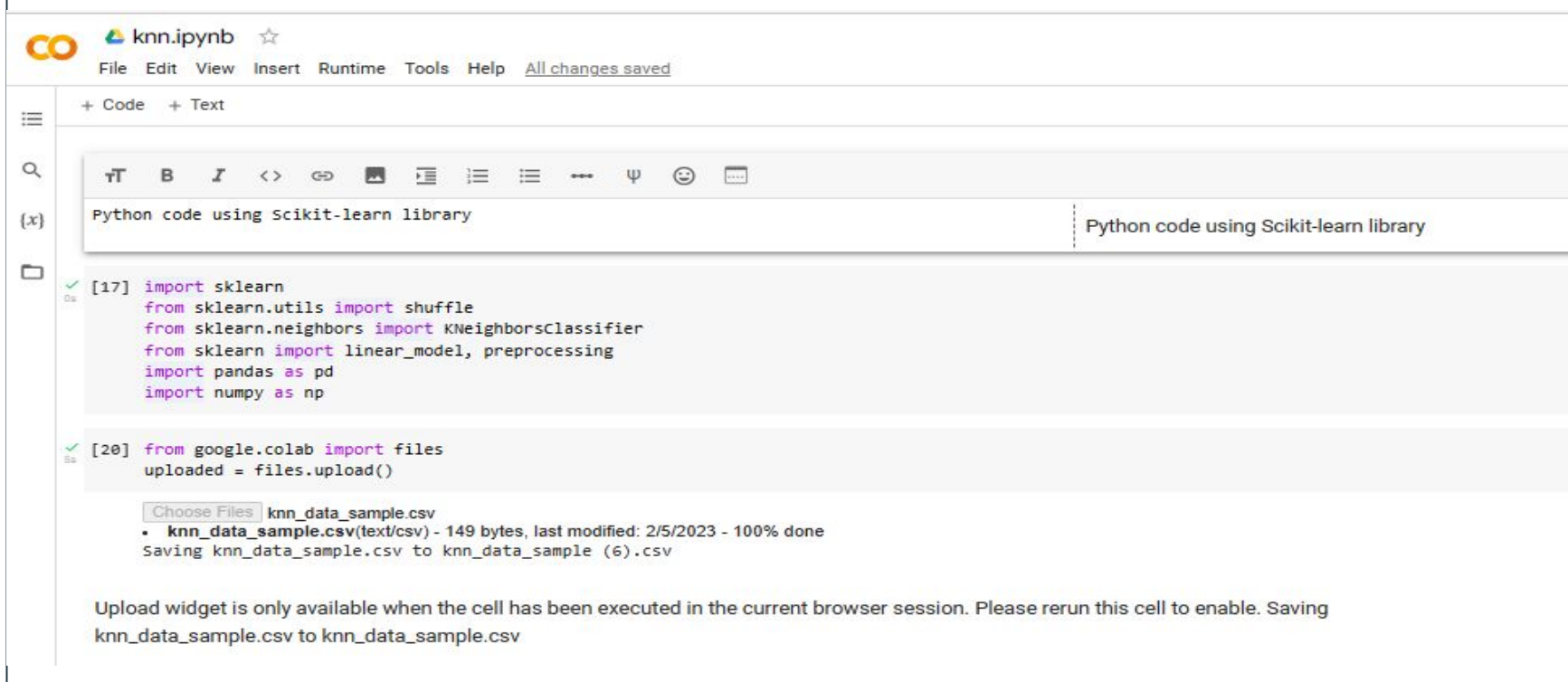
```python
# Caluclate euclidean_distance
print("Euclidean distance between two vectors")
for i in range(1,9):
        print(euclidean_distance(dataset[0],dataset[i]))


# row 0 (i.e., dataset[0]) is the one to be predicted
prediction = predict_classification(dataset, dataset[0], 3)

# - dataset[0][-1] is the last element of row 0 of dataset
# - Display
#    Expected 1, Got 1.
print('Expected %d, Got %d.' % (dataset[0][-1], prediction))
```

Implementation

Code

Test (Scikit-Learn library)

```
[1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0]
Text(0, 0.5, 'Testing Accuracy')
```
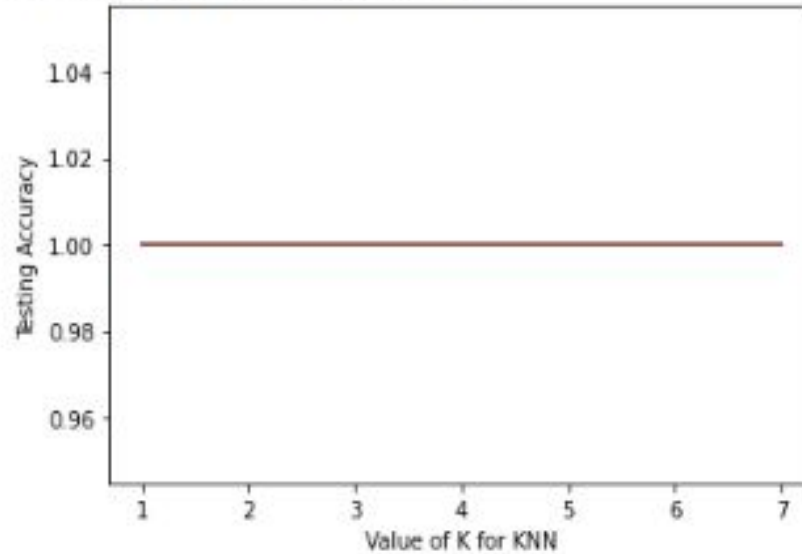
Test - Predict kNN Value

```
[1] prediction = predict_classification(dataset, dataset[0], 3)

    # - dataset[0][-1] is the last element of row 0 of dataset
    # - Display
    #    Expected 1, Got 1.
    print('Expected %d, Got %d.' % (dataset[0][-1], prediction))
```

```
Euclidean distance between two vectors
10.295630140987
10.392304845413264
10.723805294763608
10.04987562112089
2.449489742783178
2.6457513110645907
3.1622776601683795
2.6457513110645907
Expected 1, Got 1.
```

Enhancement Ideas

We can use weighted kNN. In traditional k-NN algorithm, each of the K nearest neighbors has equal weight in determining the target variable. In weighted kNN, the nearest k points are given a weight using a function called as the kernel function. The intuition behind weighted kNN, is to give more weight to the points which are nearby and less weight to the points which are farther away.

Conclusion

We analysed different aspects of fall prediction and prevention systems

We understood the working of Gyroscope and Accelerometer in how fall predictions can be made

We understood K nearest neighbor and manually calculated the K for the collected data set from Accelerometer and Gyroscope

We manually predicted the Euclidean Distance

We used colab to programmatically calculate the K and Euclidean Distance

Manual and programmatic calculations match

References

[K-NN Example (sfbu.edu)](#)

[A Review on Fall Prediction and Prevention System for Personal Devices: Evaluation and Experimental Results (hindawi.com)](#)

[Machine Learning Basics with the K-Nearest Neighbors Algorithm | by Onel Harrison | Towards Data Science](#)

[Sensors (sfbu.edu)](#)

[Whee, Gyro and Accelerometer (sfbu.edu)](#)