Fall Detection – From Phone to Cloud

-Archiving Phone Data on Cloud Server

Student: Manvick Paliwal
Advisor: Dr. Anne Hee Hiong Ngu

What Fall Detection is all about? 🤔

Introduction

- This Independent Study is about a Fall Detection Sensor Application which will help detect if someone has fallen down. This application is specially designed for elderly people, but anyone can use it. The way this works is we collect streaming sensor data from a smart watch. The sensor data collected is them processed to extract key features related to falling on a smart phone. This processed data is then fed into a Naive Bayes machine learning algorithm which analyzes and output fall or not fall as the outcome.

Objective

- Archiving of sensor data to the cloud robustly.
- Improving the accuracy of the fall detection model by retraining using archived false positive data.

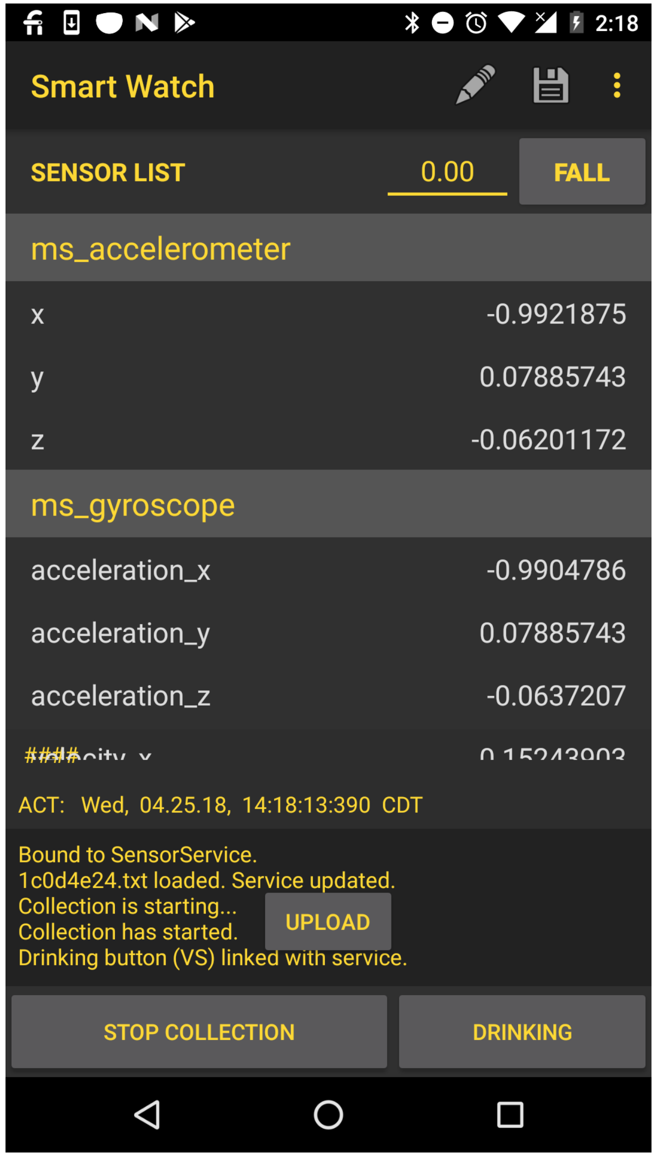
Raw Data

- Raw data is streamed from the smart watch at a sampling frequency of 32milliseconds.
- We store this data inside a folder named “RawData”.
- We write raw data to a file name cur_*timestamp under “RawData” folder*
- After every 5 minutes we rename this file to upload_*timestamp and create a new cur_timestamp file and start writing to that file.*
- Upload_*timestamp* are the files which are ready to be uploaded.

False Positive data

Two ways in which false positive data can be stored.

- Storing all the processed data (Not efficient)
- Storing Time Stamps (More Efficient)


2:18

Smart Watch

SENSOR LIST 0.00 FALL

ms_accelerometer

x -0.9921875

y 0.07885743

z -0.06201172

ms_gyroscope

acceleration_x -0.9904786

acceleration_y 0.07885743

acceleration_z -0.0637207

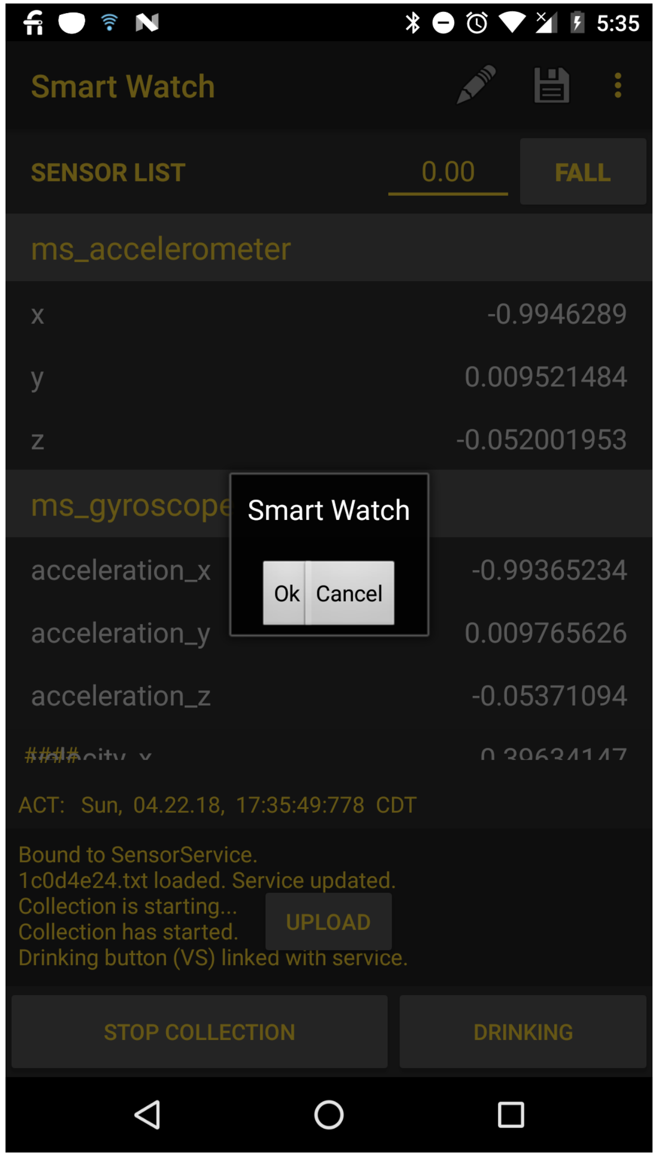
####city_x 0.15243903

ACT: Wed, 04.25.18, 14:18:13:390 CDT

Bound to SensorService.
1c0d4e24.txt loaded. Service updated.
Collection is starting...
Collection has started.
Drinking button (VS) linked with service.

UPLOAD

STOP COLLECTION DRINKING


5:35

Smart Watch

SENSOR LIST 0.00 FALL

ms_accelerometer

x -0.9946289

y 0.009521484

z -0.052001953

ms_gyroscope

acceleration_x -0.99365234

acceleration_y 0.009765626

acceleration_z -0.05371094

####city_x 0.39634147

Smart Watch

Ok Cancel

ACT: Sun, 04.22.18, 17:35:49:778 CDT

Bound to SensorService.
1c0d4e24.txt loaded. Service updated.
Collection is starting...
Collection has started.
Drinking button (VS) linked with service.

UPLOAD

STOP COLLECTION DRINKING

False Positive data Continued…

- We store false positive data inside a separate folder name “ProcessedData”.
- File name: TimeStamps.csv
- Time Stamps at the time fall happened is logged

Uploading

- Reading from all the upload_timestamp files and sending post request to server.
- If all the data of a specific file is uploaded successfully a success call back is sent from the server.
- Only after success call back is received from the server we delete the specific file from the phone.

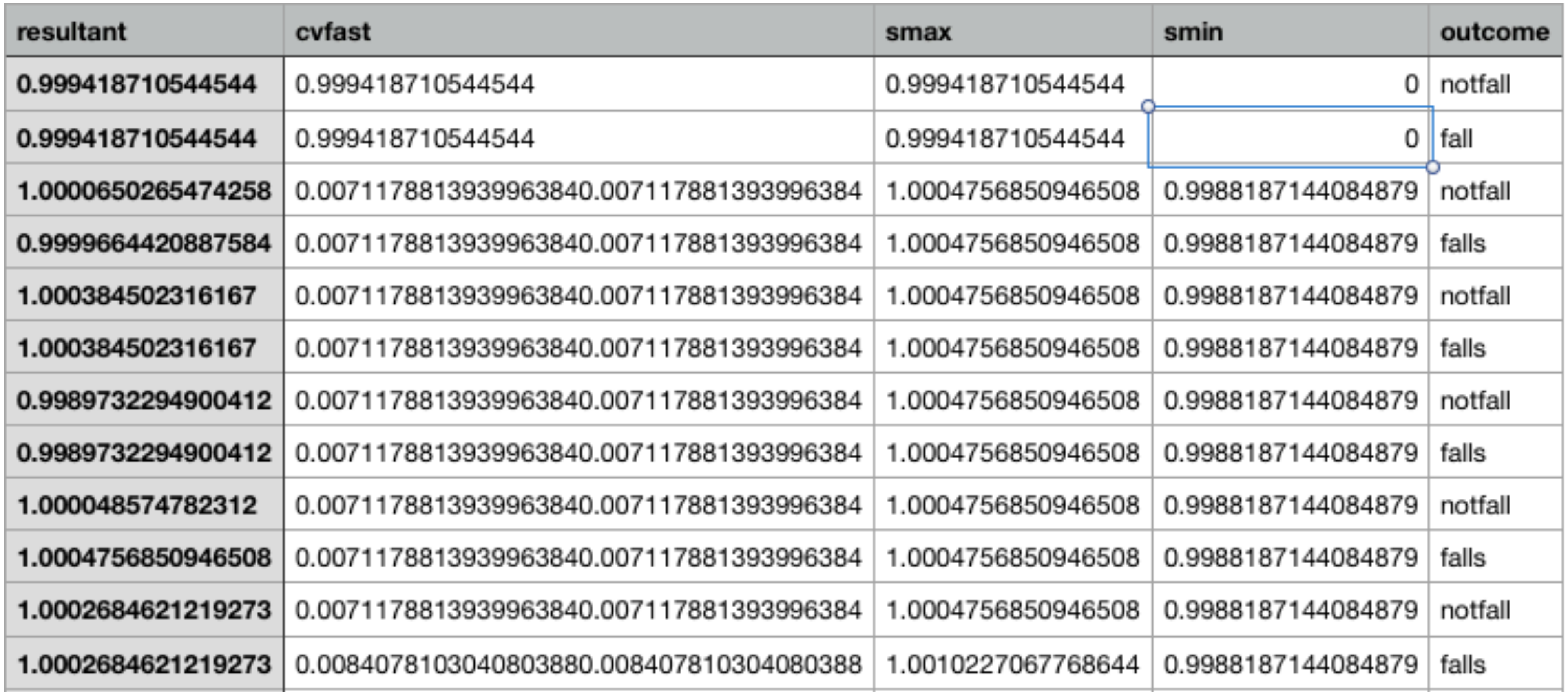
Chunking

- Challenges: If all sensor data is written to same file and upload periodically, The file gets bigger and bigger and it is not a good practice to upload a huge amount of data over http post request.
- A better solution is based on uploading Chunks.
- The size of each chunk is approximately 512kb and each file chunk is created every 5 minutes (300 seconds).

Robustness with 2 phones

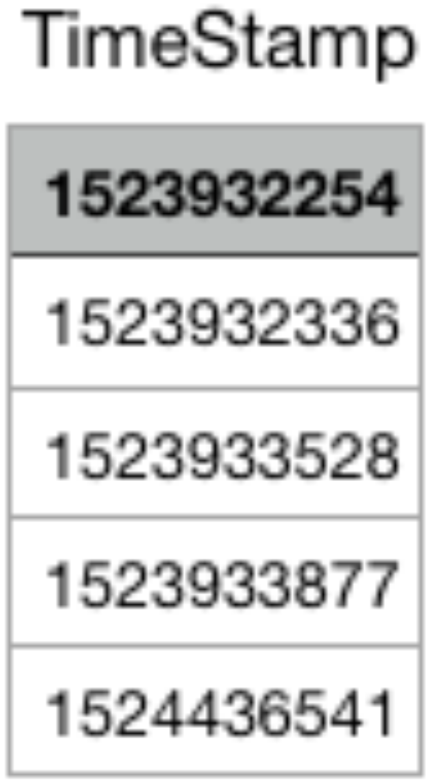
- All the data is uploaded correctly and in a robust way with Two mobiles uploading data simultaneously.
- If Program is closed in-between of the uploading process none of the data is lost.
- If the Internet connection is lost in the process of uploading of data the data which is archived is deleted and the data which is not archived is saved securely on client device.
- Each file is approximately of 512kb.

Processed data



resultant	cvfast	smax	smin	outcome
0.999418710544544	0.999418710544544	0.999418710544544	0	notfall
0.999418710544544	0.999418710544544	0.999418710544544	0	fall
1.0000650265474258	0.0071178813939963840.007117881393996384	1.0004756850946508	0.9988187144084879	notfall
0.9999664420887584	0.0071178813939963840.007117881393996384	1.0004756850946508	0.9988187144084879	falls
1.000384502316167	0.0071178813939963840.007117881393996384	1.0004756850946508	0.9988187144084879	notfall
1.000384502316167	0.0071178813939963840.007117881393996384	1.0004756850946508	0.9988187144084879	falls
0.9989732294900412	0.0071178813939963840.007117881393996384	1.0004756850946508	0.9988187144084879	notfall
0.9989732294900412	0.0071178813939963840.007117881393996384	1.0004756850946508	0.9988187144084879	falls
1.000048574782312	0.0071178813939963840.007117881393996384	1.0004756850946508	0.9988187144084879	notfall
1.0004756850946508	0.0071178813939963840.007117881393996384	1.0004756850946508	0.9988187144084879	falls
1.0002684621219273	0.0071178813939963840.007117881393996384	1.0004756850946508	0.9988187144084879	notfall
1.0002684621219273	0.0084078103040803880.008407810304080388	1.0010227067768644	0.9988187144084879	falls

Time Stamp



TimeStamp
1523932254
1523932336
1523933528
1523933877
1524436541

Processed data script

- This script creates false positive raw data on server using time stamps and raw data
- This raw data is then pushed to R script to create processed data.
- The processed data is then finally merged with existing training data to retrain the model.

Algorithm

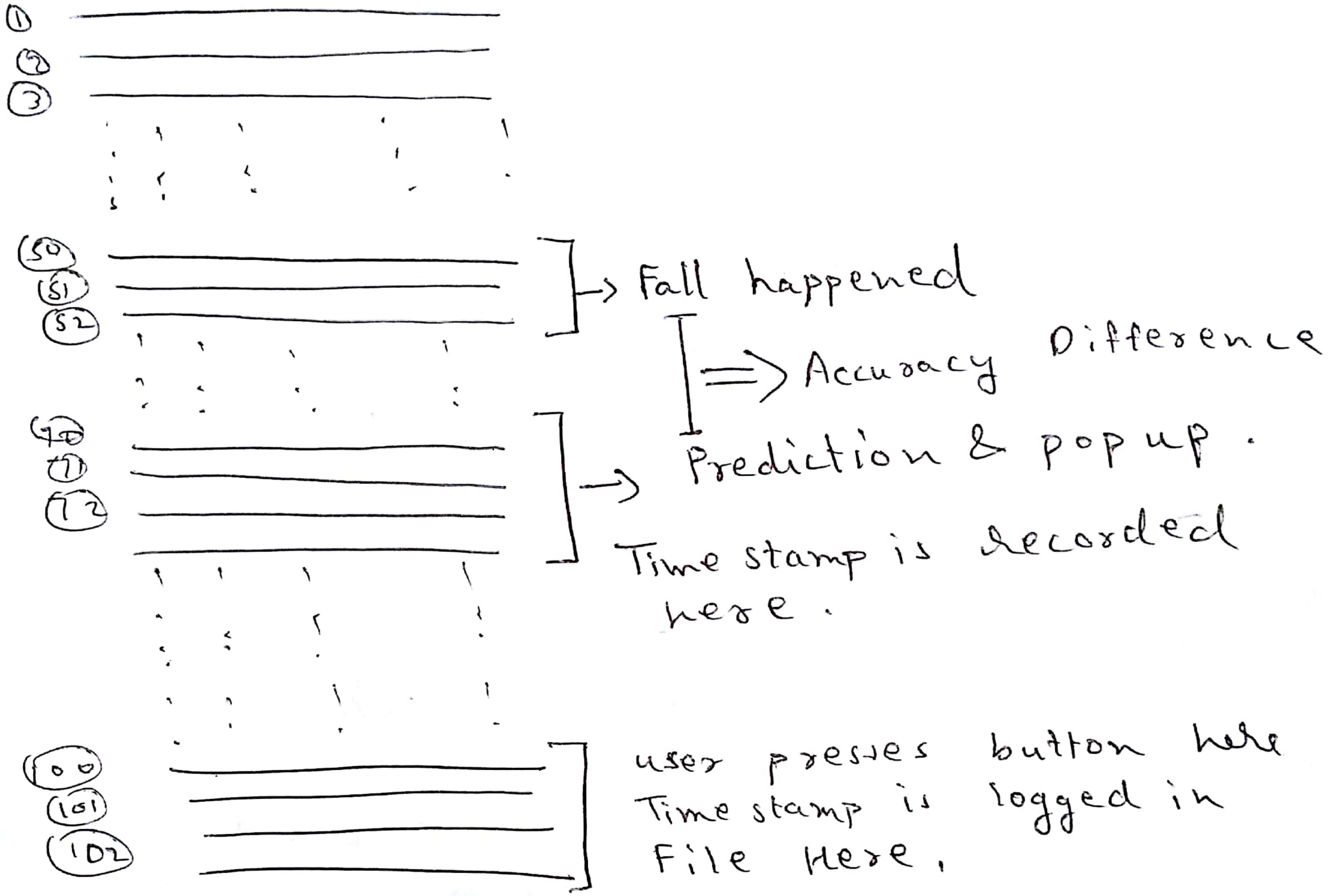
- Get false positive Time Stamp (by querying FalsePositive table)
- Get Sample ID of each Time Stamp from SampleMetaData table
- Retrieve 50 sampleId less than the current sample Id from SampleMetatData
- For each of the 50 sample id, match the correspoding sample ID to get all the accelermeter data (Ax, Ay, Az)
- Store the data in false_positive_timestamp.csv
- Go back to Step 2 as and repeat untill all the false positive TimeStamp is queried.

Cloud Server

- We have a php server with MySQL as the back end. There are two php scripts which are ready to accept the data in JSON format and insert it into MySQL database.
  - insertRawData.php
  - insertTimeStamps.php

Open Problems

- SQLite
  - Time Stamp is not accurate.
  - Use of client side database will help collect more accurate false positive data.
- System gets slow
  - Clear processed data file .
  - Circular queue data structure can be used.
- System should be Retrained with all the true positive and false positive data
- We haven’t retrained the model with sufficient data
- If user don’t want to upload the data the system will crash eventually



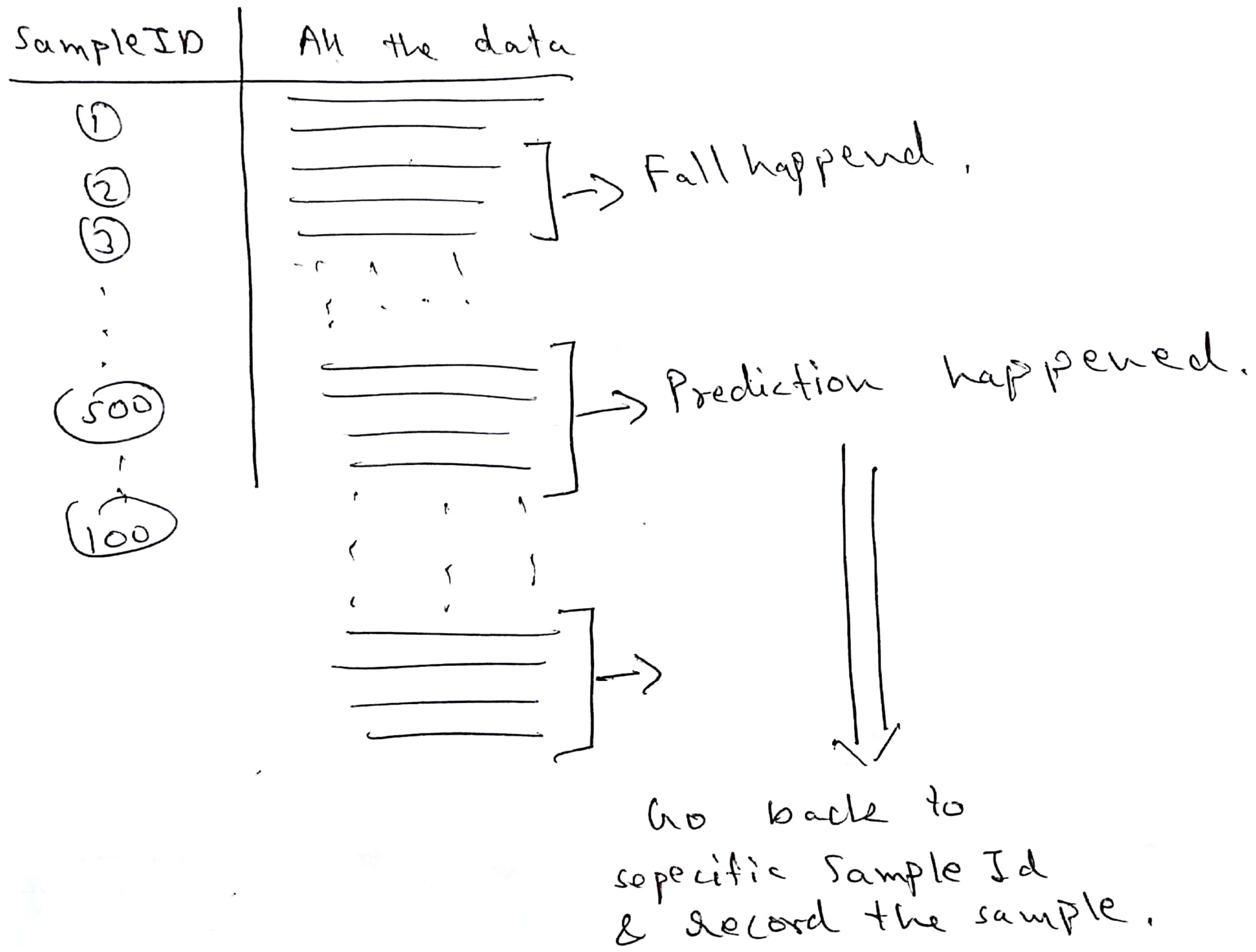
Fall happened

Accuracy Difference

Prediction & pop up.

Time stamp is recorded here.

user presses button here
Time stamp is logged in File Here,


SampleID
All the data
1
2
3
500
100
Fall happend.
Prediction happened.
Go back to sopecific Sample Id & record the sample.

Conclusion

- Chunking
  - Developed protocol to upload and archive data sensor data from phone to cloud robustly.
  - Data is divided into chunks and then uploaded.
- Time Stamps
  - Presented a method to reduce the amount of data to be uploaded.
  - Instead of uploading all the processed data only time stamps can be uploaded.

Thank you

-Manvick