Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea in Informatica per il Management

DATA COLLECTION AND RECOGNITION OF ABSOLUTE ATTRIBUTES OF MOTORBIKE RIDES VIA MACHINE LEARNING

Relatore:
Dott.
FEDERICO MONTORI

Presentata da:
VITTORIA CONTE

Sessione II
Anno Accademico 2020/2021

Abstract

The recognition of the biker trough his or her way to drive the motorbike, in the activity recognition field, is a challenge that has never been tackled before, until now. In fact, in this thesis we designed and constructed an app, that combined with a script, allows us to recognise who is driving the motorcycle between two different bikers. The mobile application collects data from sensors integrated in the smartphone, that register the movement events, while the motorbike is ridden. The smartphone is positioned on the motorcycle and collects 5 different sensors' events. The data is stored in more than one CSV file. We elaborate the data files until we have a unique one with all the data recorded by the two bikers and the features calculated through an appropriate Python script. Each row must have a label that refers to one of the bikers (A or B). Subsequently, we feed the dataset to a machine learning algorithm. The purpose of such algorithm is to learn how to recognise the biker on top of this data. In order to do that we divide the dataset in two parts: with the first part, the training set, the algorithm learns how to classify the data on its own and, with the second part, the test set, the algorithm tests this new ability. The algorithm takes in input the test set (without considering the classification) then, using what it learnt previously, it outputs the class that it presumes the data belongs to. Comparing what the algorithm "thinks" the data represents with what it actually represents we have the accuracy of the algorithm. We also have the efficiency of the method that we used in order to recognise the biker from his or her riding style.

Contents

List of Figures

Introduction

In this thesis we wanted to establish if a biker could be recognised through his or her driving style. In fact, each biker drives in a unique way and we wanted to measure that way through empirical data. Even if the activity recognition is an expanding field, a research work like this one has never been done before. We had nothing to start with other than the activity recognition method itself. We studied the vehicle, the parameters that could influence the drive style, the sensors and the device. We defined the approach, the tools and the methods that we thought would fit better in order to solve this problem. Then, we used them: we implemented the app which collects the data, we used the app in the way that it has been design for (that is to allow us to collect correctly the data) and we elaborated the data with a script that allowed us to recognise the biker. All these steps will be explained in this thesis.
In the first chapter in fact, we're going to talk about the state of arts, the things that has been done before and the fields that this research work involves. In the second chapter we will see the architecture of this research work, how we proceeded in order to accomplish the target and why we did it that way. In the third chapter we will go deeper and see exactly what we did, the code, the calculation, the logic under the results that we obtained. In the end, in the last chapter, we will analyse the results obtained, make some conclusions on this research work and talk about future implications.

Chapter 1

State of the Art

1.1 Internet of Things

1.1.1 Introduction

Nowadays the majority of Internet connections are devices used directly by humans, such as computers and smartphones. The main form of communication is human-to-human. However, in future, it will be possible to communicate in other forms like human-to-machine or machine-machine(M2M). The networked interconnection of everyday objects is known as Internet of Things (IoT). This new term comes from the fact that the "things" can exchange information by themselves. Among the objects that can now do that we must remember that the wearable devices, that are becoming more and more common, are a large part of it. This open up to new way to see and make things of our everyday life and has many applications in many fields like agriculture or management [2][22][14].

1.1.2 Why now?

This new chapter of the Internet is now closer due to new technologies: all the everyday devices will be more often equipped with micro controllers, transceivers for digital communication, and suitable protocol stacks. That will make them able to communicate with one another and with the users. In this way they will become an integral part of the Internet. Is also useful that the memory and the data processing capability is constantly increasing [27].

1.2 Machine Learning

1.2.1 Definition

Machine Learning is the study of computer algorithms that improve their efficiency through experience. Machine learning algorithms build a mathematical model, based on the sample of data also known as "training set", in order to make decisions or predictions without being explicitly programmed to do so. Is closely related with fields like data mining, that is focused on the data analysis through the unsupervised learning, computational statistics that is focused on making predictions using computers and, of course, with predictive analysis [24][4][17].

1.2.2 Approaches, Methods and Models

There are several ways to do that. Basically, all the path we can take are based on a few points that must be followed in order to accomplish the target.
First thing first it must be clear that the quantity and the quality of the data determine the accuracy of the model. For this reason, is strongly recommended to verify the way the data were collected. Then the next step is to clean the data from errors, duplicates, missing values in order to have an error free dataset. Once obtained a sufficient level in terms of quality and quantity of the data, the dataset is ready to be used by the Machine Learning algorithm.

The Methods

The method chosen is important to achieve the goal. The methods go from Unsupervised Learning, where the algorithm tries to find a cluster from the given data, [6] to Supervised Learning, where the algorithm needs a classification of the given data to make a good prediction in future [6][18]. Is important to choose wisely because different problems need different methods[23].
For example, in order to recognise if a tumour is benign or not, which is better between the supervised learning and the unsupervised learning? The supervised learning is a type of machine learning algorithm that maps a previously classified set of data in order to be able to classify, in future, a not classified dataset. The unsupervised learning is a type of machine learning algorithm, that works with label less data as dataset, that draw interference from that data. The first needs a classification that can be made by a doctor (but he or she may made a human error), the second search for a cluster

(we're assuming that we're using the cluster model since is the most common for this type of method) that maybe can be easily found without any classification made before, maybe not.

The Models

The model used is the other choice that must be made thinking on which one would fit better to the problem that must be solved. Like the method, there are lots of models that can be used like the Random Forest, the Decision Tree or the Support Vector Machine. These are, of course, models which differ for many reasons. For example: the random forest is a method that construct a multitude of decision trees at training time and output the class. Is usually used for classification or regression tasks. The Support Vector Machine is also used to analyse data and do regression and classification tasks but it is a non-probabilistic binary linear classifier. The data in this case are marked as belonging to one or another category and the algorithm builds a model that assigns new examples to one category or the other.

1.2.3 Limits

Like every study, Machine Learning has its limits. The algorithm can predict only the cases it had the opportunity to study first. If a case is not present in the set of data it can't predict anything like it. In the case referred to above, if there is a certain type of tumour that the algorithm didn't had the opportunity to process before, then it can't classify that type of tumour. If the case we're analysing use an Unsupervised Learning method then the algorithm won't find a cluster if not present in the dataset.

1.2.4 Applications

There are many applications for machine learning. It's a vast and complex study area and it's only the beginning. It's present in activities like email filtering (e.g. spam, not spam) [13] and computer vision, activities where performing the needed task or develop a conventional algorithm is difficult or impossible. One of the main application it is used for is the activity recognition, which we will talk about in the next chapter.

1.3 Activity Recognition

1.3.1 Definition

The activity recognition is the problem of predicting the movements of a person thanks to sensors data that can be stored via smartphone sensors or wearable sensors. It is performed by capturing the contextual information while a user performs different activities.

1.3.2 How it works

The activity recognition, to predict the movements, needs precise sensors with which register the movements made, a structured and error free database and an accurate research on the methods and the data that must be used to make an accurate prediction. Knowing the problem is the key because, for each movement we need to register, there is a specific sensor and for each activity there is a device, or a group of devices, that is preferable to others [8].
Smartphone and Wearable devices are becoming more and more used for these types of researches because they're more and more present in our lives. Also, thanks to new technologies, the amount of these devices with sensors, and the accuracy of the sensors itself, is constantly increasing. To use them properly is necessary to exactly know what they do and how they work: there is a substantial difference between what a smartphone and other devices can do[25]. For example: if the movement we want to predict is predictable with only one sensor, then the device that must be used is the device that has that sensors in it and also that fit better for the area where there will be the movement. If the sensors needed are more than one and the movement is actually a complex activity then is necessary a study on what that activity consist (e.g. dancing needs a full body movement that is different than jogging) and the device or the devices to use to register the sensors needed. The sensors of the smartphone can register the events, if the device is on the left pocket of the trousers, but it may be useful if the data registered in that position can't later allow to know the difference between jogging and dancing. Also, a smartwatch on the wrist can record the data sensor that will later permit to recognise the movement of the hand but, for a two-hand activity like playing the piano, it may be necessary to use a second wearable that do the same job and add data to the total dataset[20].

Wearable Sensors

As mentioned before, Wearable provided with sensors have become very popular in many applications because they can be extremely useful in providing accurate and reliable information on people's behaviours and activities. The wearable sensor technology is revolutionising our life, our social interaction and our activities. Let's take, as an example, the fields were they became popular: medical, entertainment, security, sport and commercial fields. In all of them this new technology has led to new behaviours and activities. In the medical field now the blood pressure, the heartbeat, the temperature and other parameter can be detected with these sensors. This means that new procedures have been adopted to cure and supervise the patients: the wearable sensors can now automatically detect data that before were detected differently so new procedures are mandatory. These parameters mentioned above are useful also in the sport field where are used to control the reaction of the body under pressure (running, swimming etc...)[15][5].

Devices

Smartphones motion sensor embedded have provided a new platform for activity recognition. Using a large variety of data communication channels, they are able to exchange information with other devices and systems. These devices also provide built-in sensors and powerful processing units that were initially used for cell phone feature enhancement. That's why now they're used to detect everyday activities of the user or the device itself.
Is possible now to realize an activity recognition system that works exclusively on smartphones where the device automatically records the data and analyse them in order to recognise the activity made by the user[3], or where the device just record and recognise the activities with the parameter given [11][1]. It is also possible to use multiple devices to do the same thing if the activity requires it.

1.3.3 Applications

Activity Recognition can be applied to a large number of fields. This thanks to new technologies and studies like machine learning algorithm, sensors and wearable devices that are improving constantly [10][12][7][21]. The fields where it can be applied or that it exploit are changing themselves. Not only the procedures like the example of the medical field but also the technology: if a wearable or a smartphone is used for something such activity recognition then it needs to be improved to its best to collect the data correctly and do

all the steps require for that specific activity recognition.

Machine Learning for Activity Recognition

Since the activity recognition problem, to be solved, needs an analysis of the event to predict them, machine learning became one of the main tools used to do that prediction. In order to do that, a machine (in this case the device) need first some data in which to base its knowledge. That's when the machine learning came in: that type of analysis allow the device used to recognise a movement or a set of movement thanks to the similar data analysed before. There are a lot of research works about activity recognition using Machine learning algorithm. In the last years, a big part of data has been recorded through smartphone, or wearable [26]. This made possible a bigger and bigger quantity of activity recognition research and an increasingly accuracy in the final results. As time passes and the technologies improves, the possibility to use the right device to register the data increase.

IoT for Activity Recognition

The Internet of Things finds an application also in the activity recognition problem. Is a matter of facts that the device used to record the sensor's data are part of the network discussed previously. So, we can take advantage of this new interconnection and use it in this field to collect data or elaborate them.

How to?

If we look at all the previous research works, there is a pattern that almost everyone followed and that we will also follow in this thesis[16]. The first one is the collection of data. In this step we must consider all the activity type we want to recognise and use the right device and the right sensors do it as said before. So, there will be a device (or more), placed where the movements are, that will record all the sensor related event we previously decided to record. Usually the accelerometer [19] and the gyroscope are the main sensors used for this type of research. However, recording other type of data is not wrong. In this step we must remind that we can exploit a new network, the Internet of Things, to collect, elaborate or exchange data. More are the data higher is the possibility that the accuracy will be elevated. The second step is the elaboration of the data. Like the machine learning approach, the data must be cleaned from error, duplicates and so on. After that, we must find the features like the mean, the pitch and the standard deviation. The features are important data that represent important information that we need in

order to recognise correctly an activity. When we finally have a dataset with all the information taken by the sensors and the relatives features, we can finally start to study the data. This is where it comes in the machine learning algorithm[9]. It's not mandatory but it's strongly recommended because it's actually the better way to learn which sensor data correspond to a specific activity. There is, then, the choice of the method and of the algorithm to use. Since we have already all the data, we can use more than one of the methods and compare the accuracy of all them, then decide which one fits better.

In conclusion

The activity recognition is a complex yet not insuperable problem that requires different and advanced technologies and techniques. Since it can be applied to a large quantity of fields, and it can use various tools, it needs an accurate study on the activity that we want to recognise in order to choose the right way to do that. One of its purpose is to improve our lifestyle and recognise bad behaviour to avoid fatal error (e.g. car crash).

Chapter 2

Architecture

The figure below represents the architecture of this research work. In this chapter I'll explain in details these blocks.

Figure 2.1: Architecture

2.1 Data Acquisition

2.1.1 Introduction

The data acquisition is the first step we have to do in order to accomplish the target. It is necessary to do it in a way that ensures the integrity of the data.
In order to have a clean set of data the device used must be the same for every record. The device must also be positioned on the "thing" that makes the movements we want to recognise later (in our case we want to register the motorcycle movements so the device must be positioned on the motorcycle). For the data collection we also need to know the parameters defining the activity. Each activity has its variables so the activity recognition must know them in order to collect the right data.

2.1.2 Collection

As said above, the data collection is about recording the movements that defines the activity. To do this we need to carefully choose the device we will use. The device must be able to collect the data we need and it must be placed where the movement takes place. For example, if we want to recognise if someone is playing the piano, we can reasonably think that we must put a wearable on the wrist and a wearable on the ankle of the pianist. To play the piano the pianist needs to use both hands and feet so the devices used must be able to recognise the movements of those body parts. These wearables must be able to record the events generated by the gyroscope in the first place. It measures the orientation and the angular velocity and, if we think about a pianist playing the piano, we may presume that its measurements are required for such research work. However, if we want to recognise if a piano is played, we must follow another path. We're supposed to change the device position and also the sensors used because everything is different. In the first case we want to recognise a human activity from another, in the second the subject is an object.
Once decided which devices and which sensors must be used, the application must be settled. In order to correctly write the data in the database we must pay attention at the order in which we register the data and the units of measurement used. The data must be homogeneous or using them could be a problem later.
If the two previously steps were made correctly, and the devices are placed correctly, the collection of the data may start. The registrations must always be done in the same way and, like the activity recognition we're talking about, some parameters must remain unchanged. Let's use the previously example again. If we want to recognise the person who is playing the piano, we must use the same devices to collect the data. This is because, even if the mobile application is the same, different devices have a different accuracy. Collecting data with sensors that have a different accuracy is wrong. The data are collected to be used in sophisticated algorithm so every little thing may change the final result. For this reason, also other parameters of the registration must not change. The pianist must play the same piano every time and the score must be the same. If two people play different scores then the data won't be homogeneous and this will compromise the result.
The procedure is:

1. Study the parameters
   What defines the activity? What is used to perform it?

2. Study where the relevant events take place

How you do the activity?

3. Choose the sensors
   What are the movements that characterises the activity? What distinguish the activity or one person from another?

4. Choose the devices
   Which device that has the sensors I need? Which device that can collect the movements I recognise as important for the research work?

5. Settle the application
   How could be an app that helps me or other people to do all the passages above?

6. Collect the data with the devices correctly settled in the correct places

It's better to collect as much data as possible in order to have more data to work on later and be more accurate in the recognition.
In the activity recognition we are analysing in this thesis, this block of the architecture has been made as follows:

1. We studied the parameters that defines the ride of a motorbike in order to isolate those that depend on the rider and those that don't
   Among them there are: the way to bend, the acceleration, the average speed, the condition of the asphalt

2. We studied were the important events of riding a bike takes place
   In this case, to have a good set of data we decided to put the mobile phone on the motorbike

3. We choose the sensors which are to be used
   That are the accelerometer and the gyroscope

4. We choose the device to use
   We eliminated all the wearable so we used a mobile phone

5. We deployed the mobile application
   That is "Motorcycle Track", that we will explain better in chapter 3

6. We collected the data of two different people, with the same device, while they drive the same motorbike in the same circuit.

2.2 Features extraction

2.2.1 Introduction

The features extraction is a delicate phase. Its purpose is to elaborate the data in such way that we can use them later in the machine learning algorithm.

2.2.2 Features

The features extraction consists in calculating variables from the data collected in order to use them later in the machine learning algorithm. This means that we need to know the features needed. In a first phase we calculate features like:

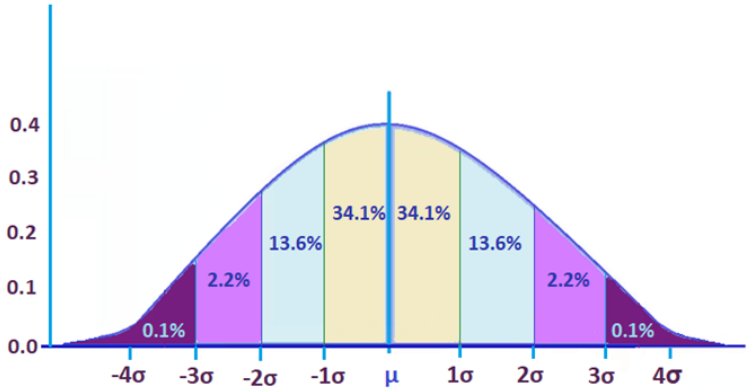
- Magnitude
- Magnitude over Couples of Axes
- Roll
- Pitch

Once we got them, we need to calculate the statistical features for the sensors' values of x, y, z and the features calculated previously. Before that we need to group the data for a unit of time. This unit may be smaller or bigger, it's a choice that must be made in accordance to the activity we want to recognise. Quicker are the movements of the activity, smaller is the unit of time we will use.
Once determined the unit of time, we just have to do the calculations and complete the database with the data. Usually these data are:

- Mean
  It's the mathematical mean of the data grouped
- Minimum
  It's the minimum value in the set of data
- Maximum
  It's the maximum value in the set of data
- Range
  It's is the range of values in which the data are (maximum - minimum)

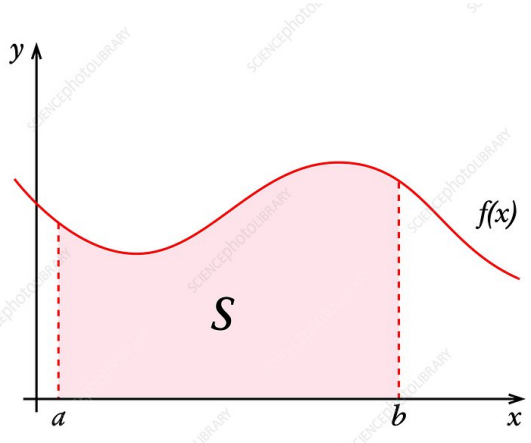
- Standard Deviation
  It's the amount of variation or dispersion of a set of values. If it is low the values tend to be close to the mean, while if it is high the values are spread out over a wider range.

Figure 2.2: Standard Deviation



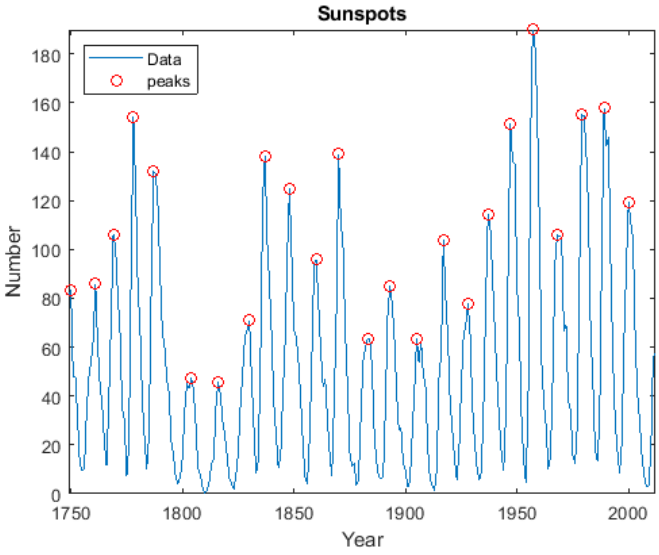
- Area Under the Curve
  It's the definite integral of a curve that describes the variation of a variable as a function of time.

Figure 2.3: Area under the curve



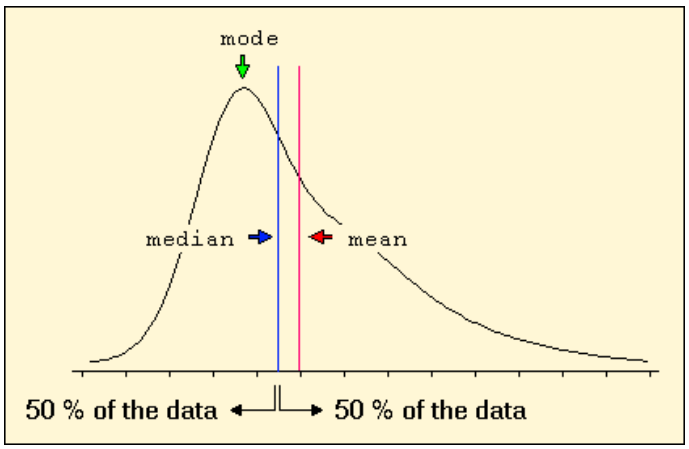
- Number of peaks

Figure 2.4: Number of peaks



- Median
  It's the value separating the higher half from the lower half of a data sample. We can consider it as the "middle value" of the dataset.

Figure 2.5: Median



In the research work we're discussing in the thesis this phase has been made exactly like above. We calculated the features, grouped the data per second and calculated the statistical features all thanks to a python script. We of course saved all these data in one file.

2.3 Machine Learning Algorithm

2.3.1 Introduction

The Machine Learning algorithm is the last step that must be made in order to recognise an activity. Its purpose is to learn how to recognise the activity.

2.3.2 Training and Test

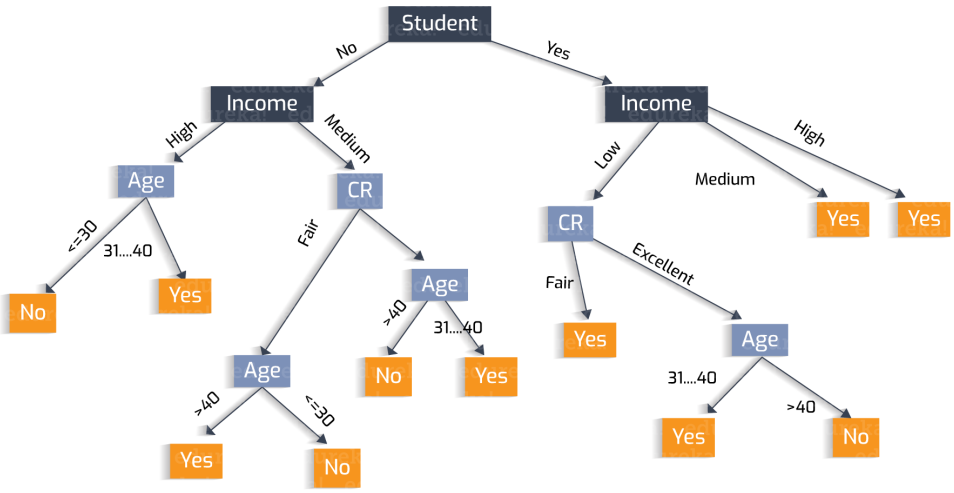
The Machine Learning algorithm needs to be trained before it is able to recognise the activity. In order to do this the dataset must be divided in two datasets: the training set and the test set. The first one is used from the algorithm to learn how to recognise the activity. It will learn how to recognise a cluster or a class, it will depend on the method used. Then, with the second dataset, we will test the accuracy of the algorithm.
In the activity recognition that we did we used a supervised learning algorithm. We classified the data as belonging to a class or another (biker A or biker B) and we divided the dataset in two files (70% of the total dataset was in the training set). We gave the algorithm both the dataset separately (the second free of classification) and we compared the actual results with the expected results.
We used three different algorithms:

Decision Tree

The Decision Tree is a decision support tool that uses a tree-like model of decisions and their possible consequences. That includes utility, resources cost and chance event outcomes.
It is a flowchart-like structure in which each internal node represents a "test" on an attribute, each branch represents the outcome of the test, and each leaf node represents a class label. The paths root-to-leaf represent the classification rules. The decision tree and the influence diagram, in decision analysis, are used as a visual and analytical decision support tool, where the expected values or utility of competing alternatives are calculated. A decision tree consists in three types of nodes:

- Decision nodes
- Chance nodes
- End nodes

Figure 2.6: Decision Tree



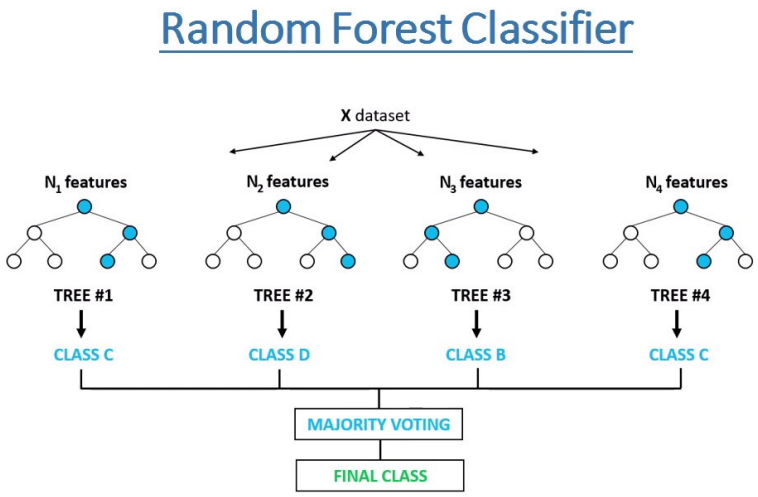
Random Forest

The Random forest is an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class. What a random forest does is to outperform decision trees with a lower accuracy than gradient boosted trees. The data characteristics can affect their performance.
It is used in Machine Learning problem because trees that are grown very deep tend to learn highly irregular patterns, they have low bias but very high variance and over fit their training sets. Random forests are a way of averaging multiple deep decision trees, trained on different parts of the same training set, with the goal of reducing the variance. This comes with some loss of interpretability and small increase in the bias, but it boosts the performance in the final model. Forests are like the pulling together of decision tree algorithm efforts. Taking the teamwork of many trees, improving the performance of a single random tree.

Support Vector Machine

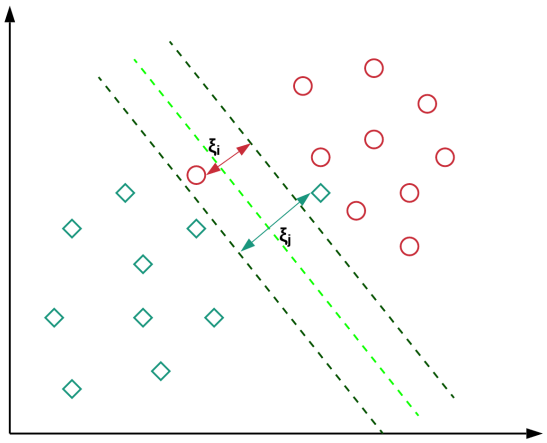
The support vector machines are supervised learning models with associated learning algorithms that analyse the data used for classification and regression analysis. It is one of the best and robust prediction methods. Given a

Figure 2.7: Random Forest



set of training examples, each classified as belonging to one of two categories, a SVM training algorithm builds a model that assigns new examples to one category or the other. The SVM, as matter of fact, is a non-probabilistic binary linear classifier. This model is a representation of the examples as points in space that are mapped in order to make the examples of the separate categories divided by a gap as wide as possible. New examples are then mapped into that same space and will belong to the category on the side of the gap on which they fall. When data are unlabelled it is required the unsupervised learning approach, which attempts to find natural clustering of the data to groups, and then map new data to these formed groups.

Figure 2.8: Support Virtual Machine



2.3.3 Conclusions

Once trained and tested the algorithms we should be able to see if we can actually recognise the activity.
In our case, after we trained and tested the three algorithms with the data in our possession, we were able to tell that it is possible to recognise the biker from his or her driving style and also with which algorithm we have a better accuracy.

Chapter 3

Implementation

3.1 Mobile Application

As said previously, the purpose of the app is to collect the data of the driving style of the biker. In order to do it we need a device, positioned on the motorbike, that collect those data. Recognising the drive style trough a device positioned on the motorbike instead on the biker may seem weird but, if we think about it, it's the best way to do this activity recognition. In fact, any device positioned on the biker would register even the slightest movement made by him or her. This means that the sensors' data would be compromised by the micro movements made by the biker. For example, if a biker is stopped at the stop sign, while he or she is waiting to go, he or she can do some movements. These movements would be recorded by the sensors and that would compromise the integrity of the data. Being stationary due to a sign is not part of the biker driving style. With a device positioned on the motorbike we're certain that this type of things won't compromise the integrity of the research work. The place chosen for this activity recognition then is on the top of the tank, screen facing up. There must be no changes of position nor changes of motorbike. The only difference between the registrations must be the driver.
The app then, must be able to start and stop the data collection whenever the user wants. It also must be able to collect all the sensors related the movements of an object. We're basically talking about the accelerometer sensor, the gyroscope sensor, the linear acceleration sensor, the gravity sensor and the location of the object. The app must also be able to save these data in a specific and organized dataset that stores not only the sensors' data but also the motorbike's data and the user's data.
Once determined all these features that the application must have, we can

start deploy it.

3.1.1 The map

An important feature that needs to be deployed in the application is certainly the map. This permits the user to know the path made while recording and where he or she is in that moment. In order to do that we need to access a google api that allow us to use google maps. Once obtained all the permissions needed to access the actual location and show it we can start deploying the fragment of the app that uses these permissions. This fragment must have:

1. A map that shows the actual location
2. A button that starts and stops the data collection
3. The list of the motorcycles owned
   In order to select one and use it
4. A method that draws the path made during the collection of the data
5. An item that allow the user to select if there is or there's not a passenger during the record

The points 1, 2, 4 and 5 were deployed first in order to have an app that draws a line where the user drove his or her motorbike. This decision made possible the test of the features that we deployed later. In fact, once ended the database deployment, the next step was immediately tested. As said above, this app was designed not only for this activity recognition but also for futures activity recognitions. For this reason, we improved two similar fragments, one called Route, for trips record, one called Circuit, for circuit record. Right now, they're similar, the only difference is that they record a different type of path.

```
1 public void onMapReady(GoogleMap googleMap) {
2         mMap = googleMap;
3         PolylineOptions polylineOptions = new PolylineOptions
    ();
4         polylineOptions.color(Color.CYAN);
5         polylineOptions.width(4);
6         gpsTrack = mMap.addPolyline(polylineOptions);
7         if (mLocationPermissionsGranted) {
8             getDeviceLocation();
9             if (ActivityCompat.checkSelfPermission(
    getActivity(),
```

```
            Manifest.permission.ACCESS_FINE_LOCATION)
                    != PackageManager.PERMISSION_GRANTED &&
                    ActivityCompat.checkSelfPermission(
    getContext(),
                    Manifest.permission.
    ACCESS_COARSE_LOCATION) !=
                    PackageManager.PERMISSION_GRANTED) {
                return;
            }
        }
    }
```

```
private void getDeviceLocation(){
        mFusedLocationProviderClient =
        LocationServices.getFusedLocationProviderClient(
    getContext());
        try{
            if(mLocationPermissionsGranted){
                final Task location =
                mFusedLocationProviderClient.getLastLocation
    ();
                location.addOnCompleteListener(new
    OnCompleteListener() {
                    @Override
                    public void onComplete(@NonNull Task task
    ) {
                        if(task.isSuccessful()){
                            Location currentLocation =
                            (Location) task.getResult();
                            moveCamera(new LatLng
                            (currentLocation.getLatitude(),
                            currentLocation.getLongitude()),
                                    DEFAULT_ZOOM);
                        }else{
                            location", Toast.LENGTH_SHORT).
    show();
                        }
                    }
                });
                mMap.setMyLocationEnabled(true);
            }
        }catch (SecurityException e){}
    }
```

The code above is the code used to show the map, the actual position and draw the path made on it.

3.1.2 The database

In order to store all the data about the user and the motorbike we need to deploy a database. The creation of a database allows the application to use these data whenever is necessary. First, we need to study which are the data needed. Then they must be correctly saved in order to retrieve them later. In the end, we must implement a user-friendly way to see and access them.

User

The user's data are not essential for this type of activity recognition but it's necessary to identify the user while classifying (biker A, biker B). For this reason, we created a fragment that shows the user's data and that allow the user to edit them.
Also, we structured a database with these data about the user:

- Email
  It is supposed it is unique for each user, will be the primary key to identify these data
- Name
- Surname
- Birthday date
  This may be needed for a future activity recognition that may need the age of the user in order to find a cluster.
  To insert these data correctly it has been implemented a specific class for the birthday date called "User Birthday".

```
@Entity(tableName = "User")
public class User implements Serializable {
    @PrimaryKey(autoGenerate = false)
    @ColumnInfo(name = "userEmail")
    @NonNull
    private String userEmail;
    @ColumnInfo(name = "userName")
    private String userName;
    @ColumnInfo(name = "userSurname")
    private String userSurname;
    @Embedded
    public UserBirthday birthday;
    public String getUserEmail(){
        return userEmail;
    }
    public void setUserEmail(String userEmail){
```

```
        this.userEmail = userEmail;
    }
    public String getUserName(){
        return userName;
    }
    public void setUserName(String userName){
        this.userName = userName;
    }
    public String getUserSurname(){
        return userSurname;
    }
    public void setUserSurname(String userSurname){
        this.userSurname = userSurname;
    }
    public UserBirthday getUserBirthday(){
        return birthday;
    }
    public void setUserBirthday(Integer year, Integer month,
    Integer day){
        UserBirthday birthday = new UserBirthday();
        birthday.year = year;
        birthday.month = month;
        birthday.day = day;
        this.birthday = birthday;
    }}
```

Then, once defined them, we created the methods needed to create, edit, delete and show these data in the mobile application. The files created to deploy all these features are:

- User
  That defines the user class and all its attribute
- DaoUser
  That implements the room and all the other methods to create, edit, delete and show the data

```
    @Dao
    public interface DaoUser {
    @Query("SELECT * FROM User")
    User getAllUser();
    @Insert
    void insertUser(User user);
    @Update
    void updateUser(User user);
    @Delete
    void deleteUser(User user);
}
```

- UserFragment
  That is the fragment that allow the user to see and open the editor to create or edit the user's data

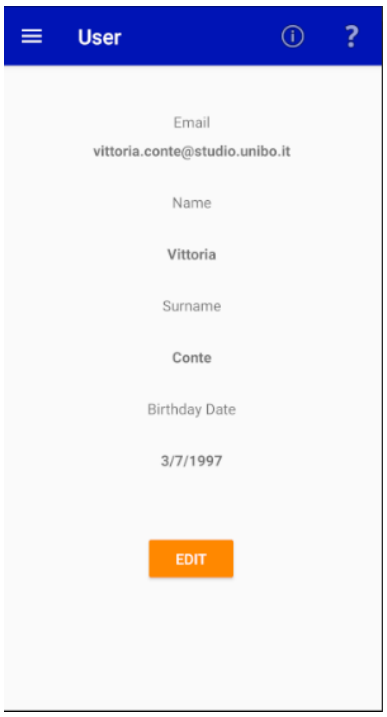


Figure 3.1: User fragment

- Edit_User
  That is the activity that allow the user to edit the data previously inserted

- Add_User
  That is the activity that allow the user to add his or her data in the app

Motorcycle

The motorcycle's data are not mandatory for this activity recognition problem but, in order to satisfy future requests for future activity recognition problems, I decided to add them too. For this reason, the motorbike's data are saved in the app and it is possible to insert more than one motorbike since lots of biker have more than on motorbike (however, is not possible to

insert a scooter as a motorbike, they're completely different vehicles).
Like the user's data, also the motorbike has its attribute:

- Plate
  It is unique for each motorcycle so we will use it as primary key
- Brand
- Model
- Register Year
  If in future will be necessary to know how old is the vehicle
- Displacement
- Wheels
  The dimension of the wheels
- Power Supply
- Customization
  Since a motorbike can be customized, we don't know if there will be a study in future that will need to know which type of customization has been made.

The code below is part of the Motorcycle class:

```
@Entity(tableName = "Motorcycle")
public class Motorcycle implements Serializable {
    @PrimaryKey(autoGenerate = false)
    @ColumnInfo(name = "plate")
    @NonNull
    private String plate;
    @ColumnInfo(name = "brand")
    private String brand;
    @ColumnInfo(name = "model")
    private String model;
    @ColumnInfo(name = "registerYear")
    public Integer registerYear;
    @ColumnInfo(name = "displacement")
    public Integer displacement;
    @ColumnInfo(name = "wheels")
    private String wheels;
    @ColumnInfo(name = "powerSupply")
    public Integer powerSupply;
    @ColumnInfo(name = "customization")
    private String customization;
    public String getPlate(){
```

```
22          return plate;
23      }
24      public void setPlate(String plate){
25          this.plate = plate;
26      }
27      public String getBrand(){
28          return brand;
29      }
30      public void setBrand(String brand){
31          this.brand = brand;
32      }
33      public String getModel(){
34          return model;
35      }
36      public void setModel(String model){
37          this.model = model;
38      }
39      public Integer getRegisterYear(){
40          return registerYear;
41      }
42      public void setRegisterYear(int registerYear){
43          this.registerYear = registerYear;
44      }
45      public Integer getDisplacement(){
46          return displacement;
47      }
48      public void setDisplacement(int displacement){
49          this.displacement = displacement;
50      }
51      public String getWheels(){
52          return wheels;
53      }
54      public void setWheels(String wheels){
55          this.wheels = wheels;
56      }
57      public Integer getPowerSupply(){
58          return powerSupply;
59      }
60      public void setPowerSupply(int powerSupply){
61          this.powerSupply = powerSupply;
62      }
63      public String getCustomization(){
64          return customization;
65      }
66      public void setCustomization(String customization){
67          this.customization = customization;
68      }
69  }
```

The fragment that allow the user to see the data of its motorbikes is different from the one with the user. In fact, a user is related to the smartphone but more motorcycles are related to the smartphone (and the user) in this application. So, in this fragment, it is used a list of items. Each item is defined to represent the motorbike's data. In this way the user is able not only to see the data of all his or her motorbikes but also to access them directly (to edit and to delete them directly).

The files used then to deploy this behaviour are:

- Motorcycle
  The class that defines the attribute of the Motorcycle

- DaoMotorcycle
  It implements the room and all the other methods to create, edit, delete and show the data

```
1     @Dao
2 public interface DaoMotorcycle {
3     @Query("SELECT * FROM Motorcycle")
4     List<Motorcycle> getAllMotorcycle();
5     @Query("SELECT * FROM Motorcycle WHERE plate =:plate
      ")
6     Motorcycle getSMotorcycle(String plate);
7     @Insert
8     void insertMotorcycle(Motorcycle motorcycle);
9     @Delete
10    void deleteMotorcycle(Motorcycle motorcycle);
11    @Update
12    void updateMotorcycle(Motorcycle motorcycle);
13    @Query("SELECT Motorcycle.Plate FROM Motorcycle")
14    List<String> getMotorcycle();
15 }
16
```

- Add_Motorcycle
  That is the activity where the user can add a new motorcycle to the database

- Edit_Motorcycle
  That is the activity where the user can edit the selected motorcycle

- MotorcycleFragment
  The fragment where the user can see all his or her motorbikes and decide which one edit or delete

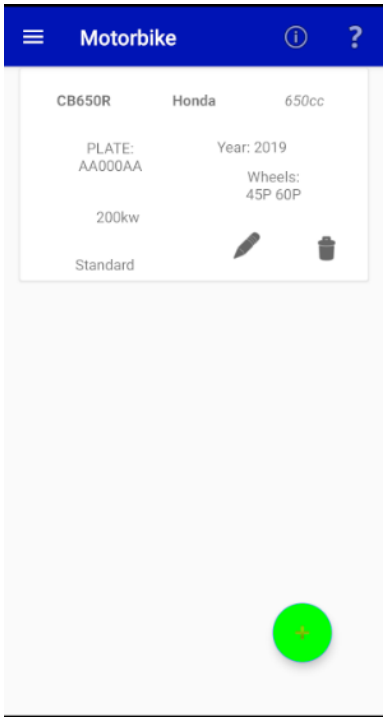


Figure 3.2: User fragment

- Adapter
  This is used to place the data in the item created to contain all the motorcycle's data. It is called in order to show the items needed with the data and to associate the item to the motorbike itself so that is possible to click on the delete or edit button and operate on that bike only. An example of the code used is below:

```
    public void onBindViewHolder
    (@NonNull RecyclerView.ViewHolder holder,
    final int position) {
        final Motorcycle motorcycleItem =
        listMotorcycle.get(position);
        ((Item) holder).txtModel.setText
        (motorcycleItem.getModel());
        ((Item) holder).txtPlate.setText
        ("Plate: " + motorcycleItem.getPlate());
        ((Item) holder).txtBrand.setText
        (motorcycleItem.getBrand());
        ((Item) holder).txtRegisterYear.setText
        ("Year: " +
        String.valueOf(motorcycleItem.getRegisterYear()))
   ;
```

```
        ((Item) holder).txtDisplacement.setText
        (String.valueOf
        (motorcycleItem.getDisplacement())+ "cc");
        ((Item) holder).txtWheels.setText("Wheels: " +
        String.valueOf(motorcycleItem.getWheels()));
        ((Item) holder).txtPowerSupply.setText
        (String.valueOf
        (motorcycleItem.getPowerSupply()) +"kw");
        if(motorcycleItem.getCustomization().
        equalsIgnoreCase("none") ||
        motorcycleItem.getCustomization().equals("/")){
        ((Item) holder).txtCustomization.setText("
    Standard");
        }
        else{
            ((Item) holder).txtCustomization.
            setText("Customized");
        }
        ((Item) holder).btnEdit.setOnClickListener
        (new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                Motorcycle motorcycle =
                listMotorcycle.get(position);
                Intent intent =
                new Intent(context, Edit_Motorcycle.class
    );
                intent.setFlags
                (Intent.FLAG_ACTIVITY_NEW_TASK);
                intent.putExtra
                ("motorcycle", motorcycle);
                context.startActivity(intent);
            }
        });
        ((Item) holder).btnDelete.setOnClickListener
        (new View.OnClickListener() {
            @Override
            public void onClick(View view) {
                if (onDeleteClickListener != null) {
                    onDeleteClickListener.
                    OnDeleteClickListener(listMotorcycle.
    get
                    (position));
                }}
        });}
```

Track

The track is a class created specifically to see how many records have already be written. The data are:

- Id
  The primary key
- Type
  Route or Circuit
- Data start
- Data stop
- Maximum speed
- Motorcycle used
  There may be more than one motorcycle registered in the database

How to use them outside their fragment

All these data shall be retrieved not only in the fragment used to visualize, edit and add them. The data stored in the mobile application shall be retrieved and selected also in the fragments were the user starts the records and save the data collected. It is another reason why these data are saved in the device.
Once deployed the structure needed to save the motorcycle's data, in the fragment were the map is shown we finally deployed another object. That object is a list where the user can select the motorbike that he or she is using. This is possible thanks to "listMotorcycle" and other functions deployed previously that allow us to save and retrieve the data.

```
1  private void listMotorcycle() {
2      class ListMotorcycle extends
3      AsyncTask<Void, Void, List<String>> {
4      @Override
5          protected List<String> doInBackground(Void... voids)
       {
6          motorcycle =
7          DbManager.getDatabase(getActivity().
8          getApplicationContext())
9                                  .daoMotorcycle()
10                                 .getMotorcycle();
11             return motorcycle;
```

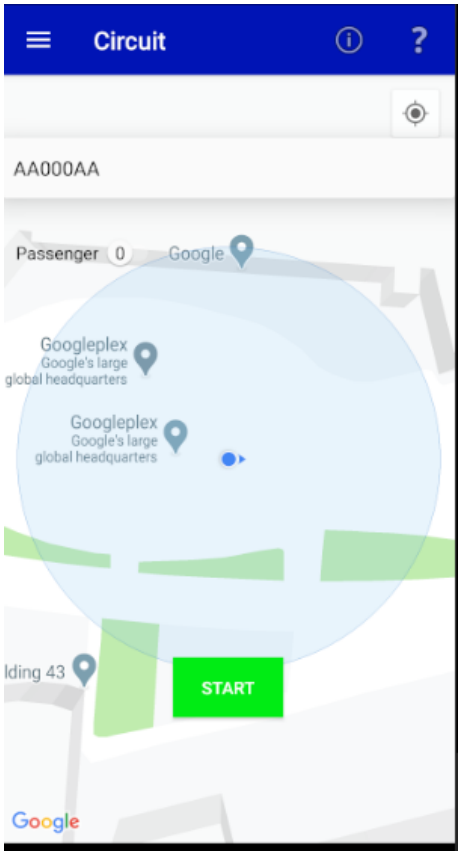


Figure 3.3: Circuit Fragment

```
12          }
13          protected void onPostExecute(List<String> motorcycle)
        {
14              super.onPostExecute(motorcycle);
15              adapter = new ArrayAdapter<String>(
16                      getActivity().getApplicationContext(),
17                      android.R.layout.
18                      simple_spinner_dropdown_item,
19                      motorcycle);
20              spinnerMotorcycle.setAdapter(adapter);
21          }
22      }
23      ListMotorcycle motorcycleL = new ListMotorcycle();
24      motorcycleL.execute();
25  }
```

3.1.3 Sensors and CSV

Once we implemented the previously features, we can start deploying the methods that write the sensors' data on the files. The format of the files is the csv format because it makes easier to parse the data and process them. To collect the data, we used the sensors integrated in the smartphone. In order to detect the values of the sensors we used the sensor manager. Once correctly obtained the values, and allocated these values to the relative variables, we started writing the csv files. The files must follow a structure:

- The file name must be formed by the track's id, the user's data, the motorbike's data, the name of the sensor(e.g."3_Route_vico1997@hotmail.it_3.7.1997_DV36308_CBF125_Honda_2011_125_46P 55P_8_none_false_ GYR.csv")
- There must be a file for each sensor (plus the location) and for each record made
- The data in the csv must be ordered in the same way for every record made

So, for every record made, we will have six separate csv files. We won't use them all but, as for some features implemented before, we're writing them for future activity recognition research work. The data were collected every millisecond. It is a short measure of time but we thought that even the slightest data could have been important so we used a measurement that allowed us not to lost important data.
Once we have all the files needed the app needs to be tested and, if it all works, we're done.

```
1  private void writeFile() {
2         getDataUser();
3         getDataMotorcycle((String) spinnerMotorcycle.
     getSelectedItem());
4         getIdTrack();
5         fileAcc=id +"_Route_" + email + "_" + birthday + "_"
     + plateM
6               + "_" + model + "_" + brand + "_" + registerYear
     + "_" +
7               displacement + "_"
8               + wheels + "_" + powerSupply + "_" +
     customization + "_"
9               + passenger +"_ACC.csv";
10        fileGyr=id + "_Route_" + email + "_" + birthday + "_"
      + plateM
11              + "_" +model + "_" + brand + "_" + registerYear +
      "_" +
```

```
            displacement + "_"
            + wheels + "_" + powerSupply + "_" +
    customization + "_"
            + passenger +"_GYR.csv";
        fileLin=id + "_Route_" + email + "_" + birthday + "_"
     + plateM
            + "_" +model + "_" + brand + "_" + registerYear +
     "_" +
        displacement + "_"
        + wheels + "_" + powerSupply + "_" + customization +
    "_"
        +passenger +"_LIN.csv";
     fileGra=id + "_Route_" + email + "_" + birthday + "_" +
    plateM
        + "_" +model + "_" + brand + "_" + registerYear + "_"
     +
        displacement + "_"
        + wheels + "_" + powerSupply + "_" + customization +
    "_"
        +passenger +"_GRA.csv";
     fileRot=id + "_Route_" + email + "_" + birthday + "_" +
    plateM
        + "_" +model + "_" + brand + "_" + registerYear + "_"
     +
        displacement + "_"
        + wheels + "_" + powerSupply + "_" + customization +
    "_"
        +passenger +"_ROT.csv";
     fileLoc=id + "_Route_" + email + "_" + birthday + "_" +
    plateM
        + "_" +model + "_" + brand + "_" + registerYear + "_"
     +
        displacement + "_"
        + wheels + "_" + powerSupply + "_" + customization +
    "_" +
        passenger +"_LOC.csv";
     String dataLoc= latitude + ", " + longitude+ ", " +
    altitude + ", "
        + speed + ", "+ timeLoc+ "\n";
        ...
}
```

Which sensor collect

The sensors collected are the sensors that catches the movements made while the motorbike is driven. The sensors collected in this case are:

- Accelerometer

The accelerometer is a tool that measures the proper acceleration. The proper acceleration is the acceleration of a body in its own instantaneous rest frame.

- Gyroscope
  A gyroscope is a tool used for measuring the orientation and the angular velocity. It is a spinning wheel or disc in which the axis of rotation (spin axis) is free to assume any orientation by itself. When rotating, the orientation of this axis is unaffected by tilting or rotation of the mounting, according to the conservation of angular momentum.

- Gravity
  The gravity sensor measures the acceleration effect of Earth's gravity on the device enclosing the sensor.

- Linear Acceleration
  The linear acceleration measures the acceleration effect of the device movement, excluding the effect of Earth's gravity on the device.

- Rotation
  Typically performed by rotating an object. Rotational sensor: A sensor that measures the turning movement of a wheel for purposes of calculating the distance travelled.

Only the accelerometer and the gyroscope are used in this research given that, like other studies has shown, these two sensors proved to be valid to recognise the activity. The other sensors are collected for a future research work that may need them.

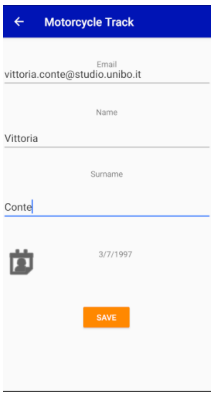
3.1.4 How it should be used and how it works

The app should be used as follows:

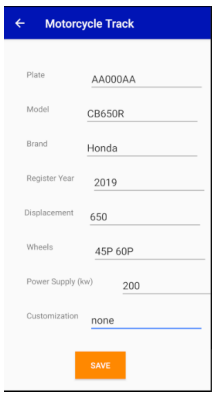
1. Open the app
2. Go in the user section from the menu on the left
3. Insert the data
4. Go in the motorbike section from the menu on the left
5. Insert the data of all the motorbikes owned
6. Go in the route or the circuit section from the menu on the left (it depends on the path that will be recorded

7. Let the app access the gps
8. Select the motorbike used
9. Select if there is a passenger or not
10. When ready click on Start
11. When done click on Stop
12. Check the last record on the main fragment

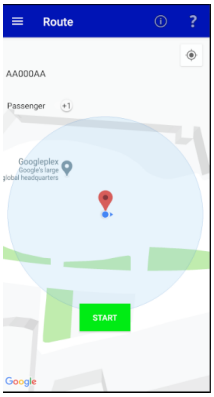
Here there are some example on how it works:



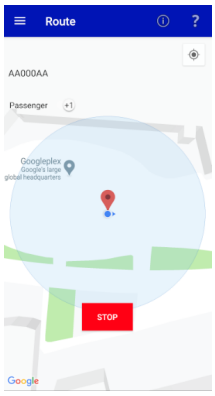
(a) User insert



(b) Motorbike insert



(a) Start Recording



(b) Stop Recording

3.1.5 Choices and evaluations

This app has been developed not only for this activity recognition research work but also to be useful for future research works. This app collect all the user's data and all the motorcycle's data because the data are a valuable goods nowadays and we don't know what we may need in future. For example: the birthday date could be used for a research that wants to determine the drive style of those born in the 90'. The motorcycle's data may be useful one day to recognise the motorbike or to do a market survey. The possibility to specify if there is a passenger or not may be useful to study how a passenger effects the drive style (we may presume that he or she effects the driving style of the biker but, like this study, we have to do an accurate research to prove it).

3.2 Data Acquisition and Processing

3.2.1 How the data were acquired

The data used in this activity recognition were collected as follows:

1. Place the smartphone on the tank, screen facing up
2. Go to the route fragment (or the circuit fragment, it depends on the trips that the user want's to make)
3. Select the motorbike
4. Click on start
5. Ride a path
6. Click on Stop
7. Do the same thing on the same path with another biker and than do that again with other routes until there are enough data for the final dataset

3.2.2 How the data were processed

After collecting enough data they must be processed. In this activity recognition we use only the accelerometer and the gyroscope values so the bikers have only two files each.

Features extraction

The first step to do is the feature extraction. It consist in calculating the magnitude, the magnitude of couples of axis, the roll and the pitch of every data in each file (roll and pitch are not calculated in the gyroscope file). We created a script that parse the files and calculate these features. The fact that the data were previously ordered means that this phase can be execute without too many problems. Then, once we got these values, we started to calculate the other features. We used the timestamp value to group the data, creating arrays of data. The features we calculated, like the mean, needs more than one value to be calculated. The arrays are now sets of data representing all the values a feature has in a second.
We calculated:

- Mean
- Standard Deviation
- Minimum
- Maximum
- Range
- Area Under the Curve
- Number of peaks
- Median

Here we have an example of the script used:

```
xMean=statistics.mean(xList)
xMax=max(xList)
xMin=min(xList)
dx = 5
xx = np.arange(1,100,dx)
yy = np.arange(1,100,dx)
xAuc=format(auc(xx,yy))
xH=peakdet(xList, 1)
xRange=(int(xMax)-int(xMin))
xMed=statistics.median(xList)
```

After that is necessary to merge the two files in order to have one file for each biker. We need then a second merge, we need one file that has all the data of one path. The file must have a label that determine, for each row, who was the biker when the record took place. In the end we have a file that stores

the feature of both bikers. The file correspond to a circuit or to a route. All the rows have a label that classify them as row of the biker A or row of the biker B.
The classification mentioned above is crucial because we decided to use a supervised learning algorithm. The data must then be classified or the algorithm won't work.

Supervised Learning

The supervised learning is the method that we decided to use for this activity recognition problem. It consists in classifying the data so the algorithm can learn how to classify the dataset on its own. In this particular case we have two type of data because we want to recognise two drive styles of two different bikers. Every row of our dataset has a label, "0" or "1", that identify the driver(biker A or biker B). Then we divide the dataset in two: the training set and the test set. The first, circa the 70% of the total dataset, is used from the algorithm to learn how to classify the data. The second in the other hand it is used to test the accuracy of the algorithm. It told us how well the algorithm learnt how to recognise the activity. We gave to the algorithm the second set of data without the label and we compared the results with the classification expected. This passage didn't only tell us the accuracy of the algorithm but also how good we made this research work and if we accomplished the target.

3.3 Predictive Algorithm

The predictive algorithm is the algorithm choose to recognise the activity. In this case three different algorithms were choose to be trained then tested in order to predict the biker who is driving the motorbike. These algorithms are:

- Decision tree
  Its purpose is to create a prediction model. This algorithm do the prediction by choosing the best features on which divide the dataset, making that feature a decision node, divide again the dataset in two and keep going until there are no more instances in the dataset or those are in the same category

- Random Forest
  That is a method performing decision trees

- Support Virtual Machine
  It is an algorithm on binary classification, so it operates only with two class(and we have only two classes). Its purpose is to find the optimal hyperplane that divides the classes using the margin.

All these algorithms took as an input the training set and elaborates it in their own way. Then they were all tested. They took as an input the test set and the prediction made were compared with the expectations. Their accuracy was then evaluated.
Using three different algorithm allowed us to establish if the result depends on the algorithm itself, if it depends on the method used or if depends on the fact that is not possible to recognise the biker by his or her driving style. For example: if we had a low accuracy for all the three algorithms than probably the driver can't be recognise by his or her driving style or the method used is completely wrong.
Here there is the python script we used to train and test the algorithms.

```
data=[]
data2=[]
with open('Totale.csv') as csv_file:
    line_count=0
    csv.reader = csv.reader(csv_file, delimiter=',')
    for row in csv.reader:
        if line_count == 0:
            line_count += 1
        else:
            data.append(row[:-1])
            data2.append(row[126])#126
            line_count += 1
X=data
y=data2
X_train, X_test,y_train, y_test = train_test_split(X, y,
                                    train_size=0.7,
                                    test_size=0.3)

clf = tree.DecisionTreeClassifier()
clf = clf.fit(X_train,y_train)
y_pred = clf.predict(X_test)
print("Accuracy Decision Tree:",metrics.accuracy_score(y_test
    , y_pred))
clf=RandomForestClassifier(n_estimators=100)clf.fit(X_train,
    y_train)
y_pred=clf.predict(X_test)
print("Accuracy Random Forest:",metrics.accuracy_score
(y_test, y_pred))
clf = svm.SVC(kernel='linear')
clf.fit(X_train, y_train)
```

```
29 y_pred = clf.predict(X_test)
30 print("Accuracy SVM:",metrics.accuracy_score(y_test, y_pred))
```

Chapter 4

Results & Conclusions

4.1 Results

The results in the table below represents the accuracy of each algorithm used to do the activity recognition.

Predictive Algorithm	Accuracy
Decision Tree	71,67%
Random Forest	90%
Support Vector Machine	81,67%

These results show that:

- The bikers are recognisable by their driving style: we didn't know it before this study because a research work like this has never be performed before
- The most accurate method is the Random Forest with a 90% of accuracy, so using that method means that the activity recognition is correct 9 times out of ten
- Even if we use the decision tree the accuracy is still high
- The two sensors used are the right ones in order to make an accurate prediction
- The two smartphone sensors used are enough accurate to make a prediction
- The supervised learning implemented for this activity recognition problem is efficient

Two biker drive the same motorbike, in the same path, with the same conditions in styles different enough to be recognised. A mobile application that collect the accelerometer and the gyroscope data allow the machine learning algorithm to have the data required to do an accurate prediction.

4.2 Conclusions

The purpose of this thesis was to know if we can recognise the rider from its riding style. We found out that with the data collected by the app we implemented and a supervised learning algorithm is possible.
The app "Motorcycle Track" correctly stores the movements of the motorbike while driven. All the values of the sensors integrated in the smartphone are neatly stored in csv files that allow us to process them. The application also stores all the user's data and all the motorcycle's data useful in order to recognize them. The application collects the data in a way that allow the machine learning algorithm to learn how to recognise the one who's driving. At the same time, the supervising learning combined with the random forest allow to recognise, with a high accuracy, the rider.
This result leads to new possibilities in the activity recognition field.
However, the mobile application is at its beginning. It stores more data than necessary and has more functions than necessary so it has more uses than the "app for the activity recognition of the driver".

4.3 Future applications

The mobile application "Motorcycle Track" may have lot of future different applications. It was deployed to record all the possible data concerning the drive style of a biker, the biker and the motorcycles used by the biker that downloaded the app.
This app, one day, could help recognise more and more activities like which motorbike is used, how many people are on the motorbike, how the motorbike should be driven in a specific path. It could, one day and with enough data, give some advises to the biker on how to improve his or her driving style. All these new activity needs different approaches from the approach we used in this thesis. There will be different features, different algorithms, different methods and maybe different devices. There is infinite opportunity for this research and for "Motorcycle Track".

Chapter 5

Thanks

Throughout the writing of this thesis I have received a great deal of support and assistance.
I would first like to thank my supervisor, Dott. Federico Montori, whose expertise is invaluable in formulating the research questions and methodology. Your faith in my potential, your patience, your feedback brought my work to a higher level.
I would like to thank my colleagues from the course of Informatica per il Management, my Karate buddies and my friends. These last years were unforgettable and crazy. You guys provided stimulating discussions as well as happy distractions to rest my mind outside of my studies.
I would like to thank my parents and my whole family for their wise counsel and their sympathetic ear. You are always there for me. (Mum and Dad, thanks for paying my feeds).
In addition, I would like to thank someone special. Thank you, Luca, for all the support, the patience, the comprehension, the fun and the love of these years.
Thank you to everyone who's been part of my life, it made me who I am now. Love you all.

Bibliography

[1] Alvina Anjum and Muhammad U Ilyas. Activity recognition using smartphone sensors. In *2013 ieee 10th consumer communications and networking conference (ccnc)*, pages 914–919. IEEE, 2013.

[2] Kevin Ashton et al. That 'internet of things' thing. *RFID journal*, 22(7):97–114, 2009.

[3] Martin Berchtold, Matthias Budde, Dawud Gordon, Hedda R Schmidtke, and Michael Beigl. Actiserv: Activity recognition service for mobile phones. In *International Symposium on Wearable Computers (ISWC) 2010*, pages 1–8. IEEE, 2010.

[4] A. Charleonnan, T. Fufaung, T. Niyomwong, W. Chokchueypattanakit, S. Suwannawach, and N. Ninchawee. Predictive analytics for chronic kidney disease using machine learning techniques. In *2016 Management and Innovation Technology International Conference (MITicon)*, pages MIT–80–MIT–83, 2016.

[5] L. Chen, J. Hoey, C. D. Nugent, D. J. Cook, and Z. Yu. Sensor-based activity recognition. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, 42(6):790–808, 2012.

[6] Y. Chen, J. Z. Wang, and R. Krovetz. An unsupervised learning approach to content-based image retrieval. In *Seventh International Symposium on Signal Processing and Its Applications, 2003. Proceedings.*, volume 1, pages 197–200 vol.1, 2003.

[7] Nhac Lu Dang. Mobile online activity recognition system-based on smartphone sensors. *Journal of Intelligent Computing Volume*, 8(1):17, 2017.

[8] Haluk Eren, Semiha Makinist, Erhan Akin, and Alper Yilmaz. Estimating driving behavior by a smartphone. In *2012 IEEE Intelligent Vehicles Symposium*, pages 234–239. IEEE, 2012.

[9] D. Guan, W. Yuan, Y. Lee, A. Gavrilov, and S. Lee. Activity recognition based on semi-supervised learning. In *13th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA 2007)*, pages 469–475, 2007.

[10] Ankita Jain and Vivek Kanhangad. Human activity classification in smartphones using accelerometer and gyroscope sensors. *IEEE Sensors Journal*, 18(3):1169–1177, 2017.

[11] Derick A Johnson and Mohan M Trivedi. Driving style recognition using a smartphone as a sensor platform. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1609–1615. IEEE, 2011.

[12] E. Kim, S. Helal, and D. Cook. Human activity recognition and pattern discovery. *IEEE Pervasive Computing*, 9(1):48–53, 2010.

[13] Nicholas Kushmerick and Tessa Lau. Automated email activity management: an unsupervised learning approach. In *Proceedings of the 10th international conference on Intelligent user interfaces*, pages 67–74, 2005.

[14] Xu Li, Rongxing Lu, Xiaohui Liang, Xuemin Shen, Jiming Chen, and Xiaodong Lin. Smart community: an internet of things application. *IEEE Communications magazine*, 49(11):68–75, 2011.

[15] Subhas Chandra Mukhopadhyay. Wearable sensors for human activity monitoring: A review. *IEEE sensors journal*, 15(3):1321–1330, 2014.

[16] Emmanuel Munguia Tapia. *Using machine learning for real-time activity recognition and estimation of energy expenditure*. PhD thesis, Massachusetts Institute of Technology, 2008.

[17] B. Nithya and V. Ilango. Predictive analytics in health care using machine learning tools and techniques. In *2017 International Conference on Intelligent Computing and Control Systems (ICICCS)*, pages 492–499, 2017.

[18] Mohammad Peikari, Sherine Salama, Sharon Nofech-Mozes, and Anne L Martel. A cluster-then-label semi-supervised learning approach for pathology image classification. *Scientific reports*, 8(1):1–13, 2018.

[19] Nishkam Ravi, Nikhil Dandekar, Preetham Mysore, and Michael L Littman. Activity recognition from accelerometer data. In *Aaai*, volume 5, pages 1541–1546, 2005.

[20] Muhammad Shoaib, Stephan Bosch, Ozlem Durmaz Incel, Hans Scholten, and Paul JM Havinga. Fusion of smartphone motion sensors for physical activity recognition. *Sensors*, 14(6):10146–10176, 2014.

[21] Xing Su, Hanghang Tong, and Ping Ji. Activity recognition with smartphone sensors. *Tsinghua science and technology*, 19(3):235–249, 2014.

[22] Lu Tan and Neng Wang. Future internet: The internet of things. In *2010 3rd international conference on advanced computer theory and engineering (ICACTE)*, volume 5, pages V5–376. IEEE, 2010.

[23] Jenine Turner and Eugene Charniak. Supervised and unsupervised learning for sentence compression. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 290–297, 2005.

[24] S. O. Uwagbole, W. J. Buchanan, and L. Fan. Applied machine learning predictive analytics to sql injection attack detection and prevention. In *2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pages 1087–1090, 2017.

[25] G. M. Weiss, J. L. Timko, C. M. Gallagher, K. Yoneda, and A. J. Schreiber. Smartwatch-based activity recognition: A machine learning approach. In *2016 IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI)*, pages 426–429, 2016.

[26] KI Wong, Yi-Chung Chen, Tzu-Chang Lee, and Sheng-Min Wang. Head motion recognition using a smart helmet for motorcycle riders. In *2019 International Conference on Machine Learning and Cybernetics (ICMLC)*, pages 1–7. IEEE, 2019.

[27] Andrea Zanella, Nicola Bui, Angelo Castellani, Lorenzo Vangelista, and Michele Zorzi. Internet of things for smart cities. *IEEE Internet of Things journal*, 1(1):22–32, 2014.