Lê, Minh Truyền

minhtruyen.le.89@gmail.com

Grab ai for sea

Safety Challenge

Table of Contents

[Introduction 2](#_Toc10801260)

[Background 2](#_Toc10801261)

[Data 2](#_Toc10801262)

[Decision Tree Classifier 2](#_Toc10801263)

[Pre-processing 2](#_Toc10801264)

[Convert time 2](#_Toc10801265)

[Magnitude Convert 2](#_Toc10801266)

[Change Calculate 2](#_Toc10801267)

[Lines of best fit 3](#_Toc10801268)

[GPS KMeans Center 3](#_Toc10801269)

[Approach 3](#_Toc10801270)

[Test Run Result 3](#_Toc10801271)

[Evaluate 3](#_Toc10801272)

[Conclusion 3](#_Toc10801273)

[Reference 4](#_Toc10801274)

# Introduction

This paper document the solution for constructing a method to predict the safeness of a grab trip. This includes the data constructing method and the creation of the class to train the “Ride Safely” data set.

# Background

## Data

The “Ride Safely” data set is a combination of two data frames. The first one is the features. In features, it contains all the data measured by the two sensors accelerator and gyroscope, which is an x, y, z 3D-dimension vector, and data from GPS includes accuracy, bearing, and speed. The data measure during the trip separates into many time periods which is recorded. The separation makes an example contains many rows which use booking ID to reference. The second data frame is the labels which have a booking ID column as a reference and a label column. Booking IDs in this data frame are unique so that it can conclude there are about 20000 examples.

## Decision Tree Classifier

The labels of this data set are binary 0 and 1 so it easier to a Classifier Algorithm than ant other. Decision Tree is the algorithm used to train the model. It is a non-parametric supervised learning method used for classification. It aims to create a model that can predict the target’s values by learning simple decision rules. DecisonTreeClassifier from the sklearn.tree library is implemented for training the model. [1]

# Pre-processing

Decision Tree Classification cannot take multiple-rows-example as attributes so that they must be combined into one-row-example. Clustering, Linear Regression models are used for preprocessing all the data. Clustering generalizes all the GPS associated data, while Linear Regression gives a line of best fit which can use to conclude the rate of change of sensors data.

## Convert time

The time column shows all time length measuring the data. The convert generates all the time point when a specific sensor data is measured. It will use as the attribute for constructing the line of best fit.

## Magnitude Convert

All the sensors data are in the form of vector dimension so that it may lower the accuracy of the prediction since vector measurement can be affected by the position of the device. Therefore, calculating the magnitude with removing the error made by the device position.

## Change Calculate

Vector measurement can be affected by environment vector. For example, the acceleration vector measured is affected by the gravitational acceleration. Therefore, the vector measured form acceleration sensor is not the device acceleration but the total vector of device acceleration and the gravitational acceleration, which of course base on the orientation of the device. Calculate the change will remove all the environment vector which in fact exist in all vector measurement of the same sensor.

## Lines of best fit

All the data computed (magnitude, change, and time) can create lines of best fit by using Linear Regression model. Time is the independent variable of the line where change/ or magnitude is the dependent one. The lines make hundred of rows combine into a representative one for acceleration and gyroscope. LinearRegresssion [2] from sklearn.linear\_model is used to produce the lines.

## GPS KMeans Center

All GPS data do not go with any pattern so that it can not use a mathematical method to construct new attributes. Using KMeans Clustering can generalize no pattern data into a representative one by using KMeans form sklearn.cluster.[3]

# Approach

As discuss the label for this data set is binary 0 and 1 so that it is easier to use a Decision Tree Classifier. A SafetyDecisionTree class is created by applying it and the discussed pre-processing method. \_\_time\_converter is a private function for converting time. \_\_change\_converter is a private function for change calculation. \_\_magnitude\_convert is a private function for converting to magnitude. \_\_change\_reg and mag\_reg are for fitting the change, magnitude, and time to create regression models. \_\_gps\_cluster for clustering all the GPS data using KMeans. \_\_features\_preprocessing create new data frame base on all the preprocessing methods. The two main functions of the class are fit and predict. Fit takes the two standard data frame of attributes and labels as parameters. The function includes preprocessing then training the tree. Predict takes a standard attributes data frame and a data frame contains only the column for booking ID. The predicting process is all so pre-processing than predicting. The function returns the array of all the prediction. All the tree parameters can be initialized when calling the class of using the standard value.

# Test Run Result

The test run base on the given data set with the split 80 for training and 20 for testing. There is also a 10-cycle-KFolds run for cross-validation with 90/10 data split.

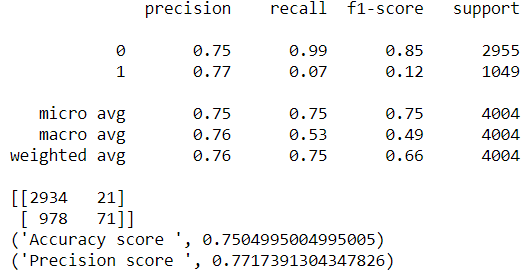


Figure 1 Evaluation Result

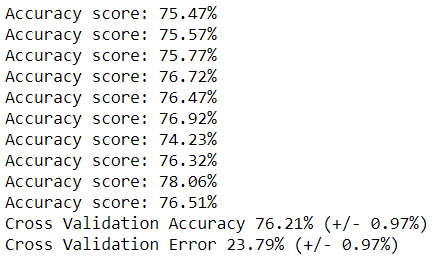


Figure 2 KFold Cross-Validation Result

# Evaluate

This model produces quite a good prediction overall. The accuracy always stays around 74% to 75% and reach a peak of 78%. The KFolds Cross-Validation has proven that the model does not overfit the data set. However, the time conducting a pre-processing procedure is quite long.

# Conclusion

This model performs very well in term of fitting the data set which gives a prediction at around 75% accuracy.

# Reference

[1] "1.10. Decision Trees — scikit-learn 0.21.1 documentation", *Scikit-learn.org*, 2019. [Online]. Available: <https://scikit-learn.org/stable/modules/tree.html#tree>. [Accessed: 10- June- 2019].

[2] "sklearn.linear\_model.LinearRegression — scikit-learn 0.20.3 documentation", *Scikit-learn.org*, 2019. [Online]. Available: <https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html#sklearn.linear_model.LinearRegression>. [Accessed: 10- June- 2019].

[3] "sklearn.cluster.KMeans — scikit-learn 0.21.2 documentation", *Scikit-learn.org*, 2019. [Online]. Available: <https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html#sklearn.cluster.KMeans>. [Accessed: 10- Jun- 2019]