Machine Learning Engineer Nanodegree

Capstone Proposal

Sebastian Nietardt
December 30, 2016

Proposal of Human Activity Recognition with Smartphones

Domain Background

Most people, nowadays, own a smartphone and carry it close to their body most of the time. A smartphone has several sensors in order to measure its movement. This sensor values can be used for guessing what the user is doing. This information can be valuable in several situations. One can use it to guess the calories burned by knowing how many steps its owner took, how fast the steps were taken and if a staircase or a slope was involved. To name another example, I could imagine the data could also be used by the police or insurance companies to figure out if its owner was using the smartphone while driving. In this project, we will try to use the sensor data of the accelerometer and gyroscope to guess what kind of activity the smartphone's owner is doing. The six activities which we are trying to guess are WALKING, WALKING_UPSTAIRS, WALKING_DOWNSTAIRS, SITTING, STANDING and LAYING.

Problem Statement

This project is about recognizing human behavior with a smartphone. A smartphone is constantly measuring its movement and when its owner is carrying it, these measurements can be used for guessing its owner's activity. In this project, we have six activities which we are trying to predict. These activities are WALKING, WALKING_UPSTAIRS, WALKING_DOWNSTAIRS, SITTING, STANDING and LAYING. This problem is quantifiable, since it has a lot of sensor data which are expressed in numbers and exactly one activity assigned to it. It is measurable, because we have a defined outcome and data to train and test on. And it is replicable, because it can be reproduced with every smartphone.

Datasets and Inputs

The dataset was obtained from kaggle. It has been released under the CC0: Public Domain License and can therefore be used without any restrictions. The dataset has been provided by UCI Machine Learning.

The complete description of the dataset can be found on kaggle. The following paragraph is a short summary of the description which UCI Machine Learning provided together with the dataset. Link to the description and the dataset.

30 volunteer subjects participated in this project. The data is already split into training data (containing the data of 21 subjects) and testing data (containing the data of 9 subjects). The signals of two sensors (accelerometer and gyroscope) have already been pre-processed by applying noise filters and have been sliced into 2.56 second intervals with 50% overlap. The gravitational and body motion components of the acceleration signal where separated, into body acceleration and gravity, using a Butterworth low-pass filter. The final dataset consists of 561 features which were obtained by calculating variables from each interval. The training data contains 7,352 examples and the testing data contains 2,947 examples. Each example has a particular activity as well as an anonymous subject assigned to it. Since we have a lot of features in the dataset, there is a high risk that the applied algorithms will suffer from the curse of dimensionality and tend to overfit. We will probably need to extract the most important features by applying feature reduction algorithms. But with a total amount of more than 10,000 examples, I am confident to get a model out of this project which will perform well.

Solution Statement

The solution to the problem can be obtained by several machine learning techniques. Since we have labeled data, we can use a supervised learning algorithm. The output we are trying to predict is discrete. That means it is a classification problem. Some models which could be used for this problem are e.g. logistic regression, support vector machines, k-nearest neighbors, naive Bayes, neural networks, linear discriminant analysis, AdaBoost, random forests or decision trees.

The provided data has a lot training and testing data, but it also provides a lot of features which needs to be separated. That means the algorithms will suffer from the curse of dimensionality and the chance of overfitting is very high. The best models might be models like AdaBoost, random forest or neural networks. We also need to do some pre-processing like feature reduction or feature scaling. Feature reduction would help to lower the dimensionality of the dataset and could help lowering the chance of overfitting. Feature scaling could make the features more comparable to each other. In the end the model will be evaluated by using the accuracy score. The accuracy score will show the percentage of the correct guesses.

Benchmark Model

The dataset has already been used for some data analysis on kaggle. The user PedramNavid got a good accuracy of 0.9471 out of his project. I will take this accuracy as a benchmark model and will try to outperform it or at least to get a similar result.

Link to PedramNavid's project.

Evaluation Metrics

The problem has exactly six outcomes which the algorithms are trying to predict. There are no close answers. Each prediction is either right or wrong. This means the best way to evaluate the predictive power of the model is an accuracy score. The accuracy score counts all the right predictions and divides the result by the total amount of predictions made. This calculation leads to a number between 0 and 1 which expresses the percentage of correct predictions.

Example:
Total predictions: 20
Correct predictions: 17
Accuracy Score: 17 / 20 = 0.85 = 85 %

Project Design

The first step in this project will be to load the data and take a look at it. It is already known that the data is already divided into a training set and a testing set. The division has been done by subjects. The training data contains 21 subjects (70 percent of the subjects) and the test data contains 9 subjects (30 percent of the subjects). This means that the test set is independent from the training set and we should probably keep it that way.

After the first look at the data, we might do some feature scaling and outlier detection. If the data is logarithmically spread, we will scale it with a logarithm first. Then we will look for outliers by applying an multiplicator to the interquartile range and add it to the upper quartile or subtract it from the lower quartile. The removal of outliers should be done thoughtful, though. It is possible that outliers are actual measurement results from particular activities. By removing these outliers, we could also remove significant data which is needed by a model to detect an activity. After the outlier removal, we will also try to use a min max scaler in order to resize the features to the same dimension.

Now that outlier removal and feature scaling have been done, we will try to find some correlations and do feature reduction. One way to do feature reduction is a random forest classifier which calculates the importance of each feature. With the knowledge

about the importance of the features, we might be able to decide how many features we really need and let us extract the most important ones.

Now the data should be prepared for applying an algorithm on it to train a model. We will use AdaBoost and a multi-layer perceptron (a neural network) on the data. We should have enough training data to get a good performance out of these two models and they probably perform well even if we still have a lot of features left. The trained models will be applied on the test dataset and the performance of each model will be measured by calculating the accuracy score of the test data prediction.

The model with the best accuracy score will be tuned in several reruns with different parameter values.