Capstone Proposal

Arunabh Saikia
20-Oct-2017

To find an Optimum Machine Learning Model for Human Activity Recognition Using Smartphones Sensor Data

Domain Background:

Nowadays smartphones have become an inseparable part of daily human life. May the purpose be for communicating with others, playing games, constantly be on social network or getting news updates about what is going on around the globe. But still the all the possible uses of smartphone are not exhausted. One major study regarding smartphone is how to understand human activities using smartphones and use it for health monitoring. Nowadays all the smartphones are equipped with built in sensors such as gyroscope, GPS, accelerometer, compass, barometer etc. Smartphone-based activity tracking is an active area of research because they can lead to new types of mobile applications. Applications that can record the user's regular activities and reports when there is any anomaly. It can also help in elder care support and rehabilitation assistance, thus saving huge number of resources that are being used currently.

The problem was discussed by **Davide Anguita, Alessandro Ghio, Luca Oneto, Xavier Parra, and Jorge L. Reyes-Ortiz** in their paper titled "*Human Activity Recognition on Smartphones using a Multiclass Hardware-Friendly Support Vector Machine*" published in 2012, which uses a support vector machine classifier to tackle the problem. Also, a number of other open source analysis has been performed using various other machine learning algorithms. The dataset we are considering here is an open source dataset from UCI Machine Learning Repository and can be found under Kaggle's datasets section (Link: Kaggle: https://www.kaggle.com/uciml/ human-activity-recognition-with-smartphones)

Problem Statement:

The problem in question is a multi-class Classification problem. The input data consists 3-axial linear acceleration and 3-axial angular velocity readings from the gyroscope of a Samsung Galaxy S II at a constant rate of 50Hz accelerometer. The input data consists of 561 which we need to analyze and our goal is to predict if a user is doing any of the following activities: WALKING, WALKING_UPSTAIRS, WALKING_DOWNSTAIRS, SITTING, STANDING, LAYING from their smartphone activity.

I will be approaching this as any other multi-class classification problem and after analyzing the dataset and doing all the needful pre-processing on the dataset, I will be applying Naive Bayes Algorithm to do an initial classification to check how the algorithm works on the dataset. After that I plan to use a Logistic Regression Classifier, a Support Vector Classifier followed by an ensemble using LightGBM and finally a Keras Neural Network to check which model performs best for the problem at hand.

Datasets and Inputs:

The dataset I am going to use is an open source dataset which can be found in UCI Machine Learning repository as well as under Datasets section in Kaggle. (Link: https://www.kaggle.com/uciml/human-activity-recognition-with-smartphones/data).

The data set was collected from an experiment which was carried out with a group of 30 volunteers within an age bracket of 19-48 years. They performed a series of activities standing, sitting, lying, walking, walking downstairs and walking upstairs. The experiment also included postural transitions that occurred between the static postures which are: stand-to-sit, sit-to-stand, sit-to-lie, lie-to-sit, stand-to-lie, and lie-to-stand. All the participants were wearing a smartphone (Samsung Galaxy S II) on the waist during the experiment execution. Data was captured with 3-axial linear acceleration and 3-axial angular velocity at a constant rate of 50Hz using the embedded accelerometer and gyroscope of the device. The experiments were video-recorded to label the data manually. The obtained dataset was randomly partitioned into two sets, where 70% of the volunteers was selected for generating the training data and 30% the test data.

(Reference: http://archive.ics.uci.edu/ml/datasets/Smartphone-Based+Recognition+of+Human+Activities+and+Postural+Transitions).

The data-set consists of the following features:

- A 561-feature vector with time and frequency domain variables.
- Its associated activity labels (WALKING, WALKING_UPSTAIRS, WALKING_DOWNSTAIRS, SITTING, STANDING or LAYING).
- An identifier of the subject who carried out the experiment.

The Activity feature of the dataset is categorical as well as the subject feature. All the other features of the dataset are normalized and bounded within [-1,1]. The dataset has no missing values. We will remove the subject feature from our dataset and use all the remaining features to train a model and predict the activity a user is performing.

Solution Statement:

The solution will be classifying a user's activity to any of the following six activities: (WALKING, WALKING_UPSTAIRS, WALKING_DOWNSTAIRS, SITTING, STANDING or LAYING based on the input feature vector from his smartphone's sensor readings.

My approach is to find an optimal model which can solve the task in hand accurately and quickly. So, I will be using few classification algorithms on the dataset to see which one performs better and then use the one that provides the best output and work on that model further to fine tune its parameters to reach an optimum model.

My first step after importing the dataset would be a quick visualization of how the records are distributed against the Activity labels. This will allow us to visualize if the distribution of the dataset based on activities is balanced or some activities are preferred over the others. The next step I would consider would be pre-processing the dataset before feeding it into our algorithms. In our case, the dataset doesn't contain any missing values. Also, the dataset is normalized in the range [-1, 1]. So, I would look for any feature that is not necessary for predicting the activity and I will remove those form the dataset. Then I will store the labels separately by encoding the activities to numerical categories by using Label Encoder.

Once the pre-processing is done, I will fit a Naïve Bayes model to check the prediction score. After that I will iterate through the following models to find the one that provides the highest prediction accuracy and then work on that to improve its performance.

- Logistic Regression.
- SVC.
- LightGBM.
- Keras Neural Network.

Logistic regression is basically a regression method with the exception that it provides probability output of each class rather than actual predicted class. Classification using logistic regression is a supervised learning method, and therefore requires a labeled dataset. Our problem is a multiclass classification problem. Here, I will use the "one-vs-all binary logistic regression" approach and then fine tune its parameters to attain maximum F1 and lowest Log Loss for our problem. A one-vs-all binary logistic regression works on the assumption that for **N** output classes there are **N** independent classification, one for each class.

SVC or Support Classifier is another supervised algorithm. I will use the Radial Basis Function(RBF) as the kernel for our classification problem. For RBF kernel, SVC has two parameters that needs tuning, gamma and C. Gamma can be thought of as the 'spread' of the decision region. When gamma is low, the 'curve' of the decision boundary is very low and thus the decision region is very broad. When gamma is high, the 'curve' of the decision boundary is high, which creates islands of decision-boundaries around data points. Whereas, the C parameter is the penalty for misclassifying a data point. When C is small, the classifier is okay with misclassified data points (high bias, low variance). When C is large, the classifier is heavily penalized for misclassified data and therefore bends over backwards avoid any misclassified data points (low bias, high variance). Tuning gamma and C by using “Grid Search CV” method we can improve the model accuracy.

LightGBM is a fast gradient boosting framework based on decision tree algorithm, used for classification problems. Light GBM grows the tree vertically (leaf wise) while other algorithms grow trees horizontally (level wise). I am choosing LGBM over XGBoost for its higher speed. Even though basic implementation of LGBM is easy, it has a huge number of parameters which needs tuning. I will try to tune the parameters using Grid Search CV to obtain the highest accuracy for our model.

Keras Neural Net is a high-level neural networks API with fast prototyping which supports both convolutional networks and recurrent networks, as well as combinations of the two. We will use the simplest type of keras model which is the Sequential model, a linear stack of layers.

Benchmark Model:

We already have many solutions for our problem. However, the model I am taking as a benchmark is a SVC which obtains an Accuracy: 96.34, Precision: 96.44 and Recall: 96.34.
(Link: https://github.com/gornes/Human_Activity_Recognition). I will try to exceed this score using various machine learning models.

Evaluation metrics:

To measure the model performance, it is very important to choose the right evaluation metrics for the problem. As our problem is a classification one, we will consider our model is “Right” when it correctly predicts the class/label of a test instance, and "wrong" when it incorrectly predicts a class/label. To measure this, there are a variety of accuracy metrics that implement this idea in one way or another, such as precision, recall, F1, AUC scores. For our problem, we will use F1 Score as our evaluation metrics. F1 takes into consideration both the precision and the recall of the test to compute the score. Precision is the number of correct positive results divided by the number of all positive results, and Recall is the number of correct positive results divided by the number of positive results that should have been returned. The F1 score is the harmonic average of the precision and recall, where an F1 score reaches its best value at 1 (perfect precision and recall) and worst at 0. Another accuracy metric we are going to use is Logarithmic Loss which measures the probability for each class, rather than just the most likely class. Log Loss heavily penalizes classifiers that are confident about an incorrect classification. Minimizing the Log Loss is basically equivalent to maximizing the accuracy of the classifier.

Project Design:

Before pre-processing our dataset, I would first fit a Naïve Bayes model to see how much accuracy we can obtain without any pre-processing and how much runtime is required so that after pre-processing we can compare the results and see if the pre-processing has helped us in increasing the accuracy or in reducing the model train time or not. For the pre-processing step, as our dataset is already normalized and scaled in the range [1, -1], we can move on to feature selection step. We can see that the ‘subject’ column which is the subject who has taken reading for this dataset has no real effect on the output activity as all the subjects has performed the same activities and predicting an activity is not dependent on the subject in any way, we can discard that feature. Before starting to work on the model, I will first do some visualization on the dataset to see which features are more important and strongly related. After removing ‘Subject’ and storing ‘Activity’ separately, we will be left with 561 features, and I plan to use all of these features while training and predicting with our model.

To train the models, initially I plan to choose different models to compare their performance. A few approaches in my consideration would be SVC, Logistic Regression, LightGBM and a Neural Network. First, I will use SVC and Logistic Regression Classifier, and LightGBM and compare the F1 and Log Loss score of these three classifiers after doing Grid Search for their parameters. Then I will select the one which gives a better F1 score and proceed with building the Neural Network.

The Neural Network I am going to consider is a Keras Model for Multiclass Classification. Our sequential model will have a dense and an activation layer. Below is the basic architecture of our model. The very first layer of our model is a Dense layer with 64 Filters and Activation as relu. As this is the first layer, input shape is also provided using ‘input_dim = 561’. Then we add a dropout layer using the “.add” method. A dropout layer prevents overfitting. The Dense(64) is a fully connected layer with 64 hidden units. The output layer is a dense layer with 6 nodes which corresponds to our 6 class outputs. Activation used in this layer is ‘softmax’ which will give the probability of each class. The optimizer I will use is a stochastic Gradient Descent Optimizer (SGD) with Categorical Cross-entropy as loss function and Accuracy as evaluation metric.

```
model = Sequential()
model.add(Dense(64, activation='relu', input_dim=561))
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(6, activation='softmax'))
```

After the initial prediction, I will use GridSearchCV from sklearn to find an optimum epoch and batch size which gives maximum accuracy for our model.

Finally, I will compare the performance between these two and take the one with the better F1 Score and low Log Loss value. This will be our optimum model.