Classification of abnormal behavior in the utilization of cellular network with machine learning techniques

Aina Maki, *Master student, UPC* and Victor Roman, *Master student, UPC*

Abstract— **The purpose of this project is to detect abnormal behavior in the utilization of cellular network. From a labeled database of 36904 samples with 13 features, we train a machine learning system capable of classifying these samples, and later, we check the system with other unlabeled testing 9158 samples. The metric used to measure its performance is the F1-score. Firstly, from the original databases, we have performed some transformations in the non-numerical features in order to have a matrix with only numerical values. Different approaches have been tested in order to obtain the best F1-score, such as PCA and MDA as feature selectors and SVM and decision trees (also with ensemble classifiers as bagging and boosting) as classifiers. The best score was obtained by using decision trees with boosting.**

Keywords—Machine Learning, Training, Validation, Testing, PCA, MDA, SVM, Decision Trees, Bagging, Boosting

I. INTRODUCTION

In order to supply the need of dynamic management and configuration in order to adapt to the varying user demands in the most efficient way with regards to the energy savings and utilization of frequency resources, current research in mobile networks looks upon Machine Learning techniques.

The dataset has been obtained after analyzing two weeks of past activity gathered from a set of 10 base stations, each having different number of cells, every 15 minutes. Each row of the csv file of the dataset is a sample obtained from one particular cell at a certain time, and each data sample contains the features:

1. Time: hour of the day (in the format hh:mm) when the sample was generated.
2. CellName: text string used to uniquely identify the cell that generated the current sample.

All Uplink (UL) and downlink (DL) are measured separately.

3. PRBUsageUL and PRBUsageDL: level of resource utilization in that cell measured as the portion of Physical Radio Blocks (PRB) that were in use (%) in the previous 15 minutes.
4. meanThr_DL and meanThr_UL: average carried traffic (in Mbps) during the past 15 minutes.
5. maxThr_DL and maxThr_UL: maximum carried traffic (in Mbps) measured in the last 15 minutes.
6. meanUE_DL and meanUE_UL: average number of user equipment (UE) devices that were simultaneously active during the last 15 minutes.
7. maxUE_DL and maxUE_UL: maximum number of user equipment (UE) devices that were simultaneously active during the last 15 minutes.
8. maxUE_UL+DL: maximum number of user equipment (UE) devices that were active simultaneously in the last 15 minutes, regardless of UL and DL.
9. Unusual: labels for supervised learning. A value of 0 determines that the sample corresponds to normal operation, a value of 1 identifies unusual behavior.

The ML system to be designed has to be able to classify new entries of data as 0 if it has normal behavior or 1 if the current activity slightly differs from the behavior usually observed for that time of the day, in which case should trigger a reconfiguration of the base station.

II. FEATURE ANALYSIS

A. *Transformation of features*

As the original data is formed by data that has features with non-numerical values, as it is the case of Time, CellName and maxUE_ULDL, a transformation was needed in these features.

Time feature was converted into angles and then to 2D with sinus and cosines. This way, the system could recognize time intervals such as 23:45 and 00:00 as close data.

CellName was converted by creating a new feature for the number that identifies the base station and another feature for the cell within that base station, converted in its corresponding ASCII number.

maxUE_ULDL was converted in the train set in order to obtain only the numeric value without char characters (' ').

Also, in the beginning the samples with NaN or #¡VALUE! were discarded, but in order to train the system in case there was NaN or #¡VALUE! values in the test set, these samples were kept and set to 0.

B. *Features selection*

Even though the number of features is not high, in order to see if some features don't convey relevant information or are statistically dependent of others, we applied PCA and MDA.

1) *PCA*

PCA **¡Error! No se encuentra el origen de la referencia.** (Principal component analysis) is a statistical procedure that uses orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components. This transformation is defined in such a way that the first principal component has the largest possible variance (that is, accounts for as much of the variability in the data as possible), and each succeeding component in turn has the highest variance possible under the constraint that it is orthogonal to the preceding components.

When PCA was applied into the data using the matlab function pca(), the output matrix obtained was the same.

2) MDA

MDA (Multiple discriminants analysis) is a transformation that increases the inter-class distance and reduces the intra-class dispersion.

The problem with MDA is that the number of non-zero eigenvalues is c-1, where in this case c=2, and therefore, the transform W is composed by only one column, resulting in a much worse input for the ML system to classify.

Consequently, no feature selection was used.

III. CLASSIFICATION METHODS

Different classifications methods were used to solve this problem. As the density function of the classes is unknown and we have a dataset with labeled data for training, we have 3 options as classifiers: SVM, Neural networks and Decision Trees.

After trying SVM and Decision Trees, Neural Networks classifier was discarded as the results obtained with Decision Trees was really good and it is a classifier that requires less computational cost compared to the Neural Network classifier.

In this section it will be described the models that have been used, SVM and Decision Trees.

A. SVM

SVM **¡Error! No se encuentra el origen de la referencia.** (Support Vector Machines) is a supervised learning model with associated learning algorithms that analyze data used for classification and regression analysis. SVM builds a model with labeled training samples that assigns new examples to one category or the other, making it a non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped so that the examples of the separate categories are divided by a clear gap that is as wide as possible. New samples are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall.

B. Decision Trees

Decision trees [3] are predictive models that go from observations about an item (represented in the branches) to conclusions about the item's target value (represented in the leaves). Tree models where the target variable can take a discrete set of values are called classification trees; in these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels.

With decision trees it was used ensemble classifiers as well, such as Bagging and different methods of boosting

- Bagging:
    - Generates multiple versions of the training set with n'<n samples (with replacement)
    - Trains component weak classifiers of the same type, independently between them, reducing variance.
    - Generates a decision by voting, assigning the most frequent label among all classifiers outputs.
- Boosting:
    - Construction of successive classifiers from selected samples of the database that take into consideration the errors incurred by previous classifiers (dependent training). Improvement is obtained in terms of both bias and variance.

IV. CLASSIFICATION METHODS

The metric used to measure the performance of the ML system is F1-score.

As the main objective of the problem is to detect abnormal behavior, an important metric to take into account is the sensitivity or recall.

Also, as it has been seen, there is class imbalance (as it is much less probable to have unusual behavior than normal behavior). Using only performance metrics such as probability of error or specificity is not enough, as if the classification criterion decides that all the samples have normal behavior, with these metrics they would obtain good results while being a bad classifier.

F1-score is the best metric as it takes into account both precision and recall.

V. EXPERIMENTS

In order to estimate the performance of the classifier without submitting each time the outputs in the kaggle website, a validation set was created taking into account the proportion of each class of data. The F1-score obtained in the validation could be considered a good estimation of what we could obtain for the test set.

A. Random values

First of all, in order to see the correct operation of the creation and submission of the csv file in the kaggle competition, random label values of 0 or 1 were assigned to each test sample.

The F1-score obtained for the test set was 0.34113.

B. Linear SVM classifier

Linear SVM classifier seeks the hyperplane that best separates samples from the two classes.

The parameter to be validated is the penalization parameter P. However, even using different values of P, assuming non-separable classes, we couldn't obtain any good results.

C. Non-linear SVM classifier

Non-linear SVM classifiers use kernel functions in order to seek these decision boundaries.

The parameter to be validated in this case is the variance.

After trying different values of P and h, for P=4.8 and h=0.4 we obtain an F1-score for the training set of 0.9159. However, the F1-score for the test set is 0.24814. Therefore, we have overfitting, and it is not a good classifier to use. Varying the values of P and h, we either obtain results with overfitting, or bad results for both train and validation F1-score.

D. Decision Trees

The first results obtained with decision trees without changing any default value of the matlab function fitctree() was a F1-score of 0.9837 for the train set and 0.9299 for the test set.

We can see that there is a little bit of overfitting with these results, and therefore, different improving methods where applied, such as pruning (changing default values of minimum leaf size, alpha value and maximum number of splits) and cross-validation.

However, after observing the results we can see that the best performance is obtained when the default values are used.

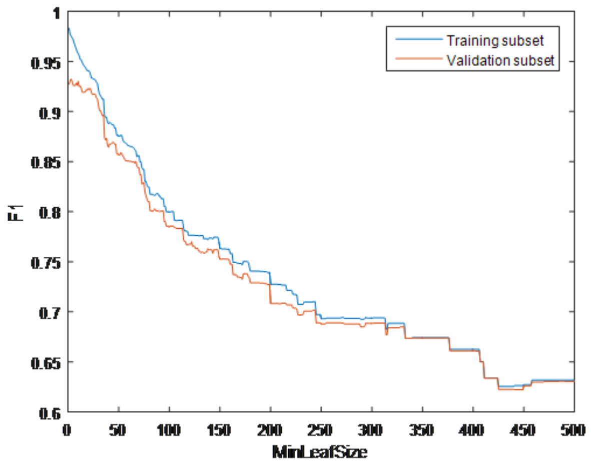

Figure 1. Plot of F1-score for different values of MinLeafSize

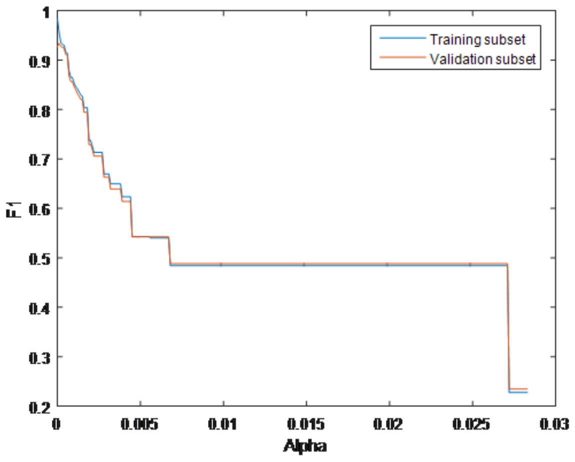

Figure 2. Plot of F1-score for different values of alpha

With cross-validation with 10 folds we obtain a F1-score of 0.9048 for the training set, being this value lower than obtained before.

Also, changing default parameters such as the split criterion for the type "deviance", we obtain a F1-score for the test set of 0.97763, which is an improvement.

After this, different ensemble classifiers were tried in order to outperform these results.

Firstly, bagging method was computed mannually (without using predefined matlab functions of ensembled classifiers). Using the training data it was observed that even using different numbers of weak classifiers, the F1-score of the training set never reaches the best F1-score obtained with previous methods. Therefore, this method was discarded.

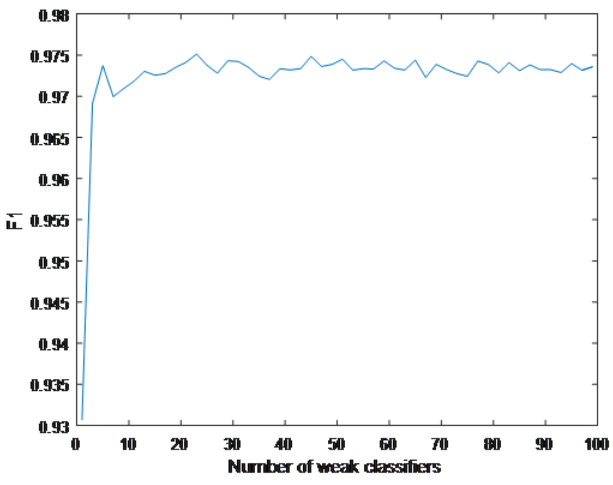

Figure 3. Plot of F1- score for diferent numbers of weak classifiers

Finally, using the function fitcensemble() of matlab, we tested different ensemble methods, testing bagging and checking that we obtain similar results as when it was done manually, and also different methods of boosting.
After trying all the fitcensemble methods, we saw that the best performance was obtained with "GentleBoost", obtaining an

F1-score of 1 for the training set and F1-score of 0.9906 for the testing set.
Finally, changing default values of this ensemble classifier, such as the number of maximum splits and the number of trees used to ensemble (as we realized that changing the other default parameters didn't improve the performance of the classifier), we obtained our final and best results.

VI. RESULTS

As it has been said in the previous section, the best results that have been obtained are using an ensemble classifier of boosting decision trees, changing the default values of maximum number of splits to 3 and the number of trees used to ensemble into 1500.

With these parameters, the F1-score for the training set is 1 and the F1-score for the testing set is 0.99425

VII. CONCLUSIONS

We can conclude that decision tree using boosting is the best classification method for this kind of data. We can say that it has sense in choosing this classifier as it is a binary classification problem with unbalanced data.

REFERENCES

[1] <<Principal component analysis>> , [online] Availabe at: https://en.wikipedia.org/wiki/Principal_component_analysis

[2] <<Support-vector machine>> , [online] Availabe at: https://en.wikipedia.org/wiki/Support-vector_machine

[3] <<Decision tree learning>> , [online] Availabe at: https://en.wikipedia.org/wiki/Decision_tree_learning.