## Abstract

This report outlines experiments for Intoxicated Speech Detection, focused on the development of a machine- and deep learning-based approaches. The project's methodology encompassed feature extraction, model training, and evaluation. By leveraging audio features, along with advanced ML models like Gradient Boosting and LSTM model the system achieved over 60% accuracy in distinguishing intoxicated speech from sober speech. Challenges included processing audiofiles as a sequence or using functional features. The project's outcomes might enhance public safety through real-time speech monitoring.

**Keywords:** intoxicated speech detection, ALC, audio feature extraction

## 1 Introduction

Intoxicated behavior's impact has spurred interest across diverse sectors, prompting a focus on its unmistakable manifestation: intoxicated speech. Accurate identification of intoxicated speech holds significance for public safety, healthcare, and behavioral analysis.

This paper explores intoxicated speech detection's methodologies, challenges, and applications. We examine speech analysis, signal processing, and machine and deep learning techniques using the Alcohol Language Corpus (ALC) from (Schiel et al., 2008).

Our inquiry aims to advance intoxicated speech detection's understanding, fostering a safer future.

The repository with code, experiment settings, data analysis and visualization is stored on GitHub page[1].

## 2 Data

The dataset comprises 15180 audiofiles: 7950 male and 7230 female recorded in 162 double sessions (alcoholized 'A' and sober 'NA').

Before setting experiments, the initial objective was to identify potential data patterns that could lead to model biases. To achieve this, we meticulously examined the labels and their meanings, making assumptions about their relevance to the current study. Some labels, such as weather conditions, were deemed irrelevant, while others, like drinking habits and blood alcohol concentration, were considered crucial for identifying data patterns and correlations.

For example, due the data imbalance or a certain correlation, future models can be gender- or other type of meta-info feature- biased. Notebook with data pre-analysis and comments can be found in our repository. Going through the data we split all meta-data labels into 4 groups by their relevancy:

- **irrelevant**: **acc** (state of elementary school: as far as almost 75% recordings contain speech of Bavarian school graduates this meta-info night not come in handy), **wea** (weather: since it was mostly sunny, it will not give significant help as well), **anncom** (annotators' comments: were too infrequent, consequently irrelevant), **irreg** (irregularities : in nearly 70% cases there is no meta-info on the recordings, though repetitions or hesitations may influence on "intoxicated speech" labelling we do not take this parameter into account);

- **relevant**: **drh** (drinking habits), **sex** (useful for checking data on possible models' biases towards gender), **age** (useful for checking data on possible models' biases towards age-related feautures), **aak, bak** (blood alcohol concentrations estimated by breath), **type** (can be used for separate analysis of what particular speech type can be used as a most reliable way to detect speaker's intoxication –> can be further used, for example, in private cars as a possible way for car crash prevention), **content** (same as types, but in a higher hierarchy);

- **"organisational" info**: session, utterance, spn (speaker id);

- useful for **post-analysis**/further experiments): **ges** (general disposition: how speaker's "mood" might correlate with intoxication), **specom** (comment on speaker: in 90% cases has no labelling, but there are 2 speakers marked as "no-native speaker" and 2 others with a "strange velar plosive" remark, we might look at the model's predictions for these speakers more carefully in order to probably find something worth attention)

The ALC dataset was partitioned into training, validation, and testing subsets using a stratified distribution of 70%, 21%, and 9% respectively, based

[1] https://github.com/MatyashDare/cl_lab

on the target label. The dataset can be characterized as reasonably balanced, containing a greater number of voice samples from individuals aged 18 to 35, with a slightly higher representation of male voices.

## 3 Methods

Following Bone et al. (2011), Zhang et al. (2012) and Rezvani (2019), for audio feature representations we used `openSMILE` library by Eyben et al. (2010). `OpenSMILE` is a library that performs automatic feature extraction from audio signals for music and speech machine learning classification. We used two different approaches and model architectures correspondingly:

- usage of `openSMILE` Functional features: for mapping contours of audio low-level descriptors onto a vector of fixed dimensionality;
- usage of sequence of features (`openSMILE` Low-level descriptors (LLDs)): iterative work with windows of the whole audiofile might make the model more stable.

#### 3.0.1 Simple Neural Nets

A simple Neural Net architecture 3 linear layers, 2 ReLu (rectified linear unit) activation functions was used for detecting intoxicated speech. Stochastic Gradient Descent with learning rate of 1e-05 and Binary Cross-Entropy Loss were chosen for training process.

### 3.1 Functional approach

We processed each audiofile with `openSMILE` library and obtained 88 feature representations for each sample. For working with audiofiles as vectors of fixed size, we mostly used with classic Machine Learning approaches, following the ideas of (Zhang et al., 2012) and (Rezvani, 2019):

- Logistic Regression;
- Passive Aggressive Classifier;
- Decision Tree Classifier;
- Ramdom Forest Classifier;
- Gradient Boosting with `XGBoost` library (eXtreme Gradient Boosting)

For obtaining optimum values of models' hyperparameters, we used Grid search Lerman (1980) tuning technique. Logistic regression algorithm was chosen as baseline.

### 3.2 Sequential approach

We used bidirectional LSTM with Batch Normalization. For the purpose of achieving possible high-level model's performance, we iterated the following parameters:

- **optimizer**: **Adam** (adaptive moment estimation gradient), **Adagrad** (adaptive gradient) and **SGD** (stochastic gradient descent) as top-3 most efficient optimizing algorithm for current task (Haji and Abdulazeez (2021));
- **activation functions** between LSTM-layers: **LeakyReLU** (Leaky Rectified Linear Unit), **sigmoid**, **tahn** as mostly used for decreasing the risk of overfitting Farzad et al. (2019);
- **scheduler**: **None** or **ReduceLROnPlateau** that may converge faster while maintaining a similar or even better loss score on the validation set comparing to no scheduler during the training process Al-Kababji et al. (2022)
- **dropout**: **None** or with probability of **0.3**
- **learning rate** of 1e-5 and 1e-6 was iterated so as to be found as the most efficient hyperparameter

As loss function, Binary Cross Entropy was applied.

We decided to choose basic 10 epochs for each experiment as we had enough variable parameters.

Due to extremely long sequences that were computationally expensive for conducting a plethora of experiments, we decided to cut the sequences by median length of 600 and padded audiorepresentations in case they were shorter.

We trained the models only on 10 epochs for each experiment.

### 3.3 Evaluation

We paid attention to precision, recall, accuracy, F1-score and F1-score weighted. We mostly focused on F1-scores. For more thorough understanding of models' predictions, we visualized confusion matrices and classification reports from `sklearn` library by Pedregosa et al. (2011) to check class-by-class predictions.

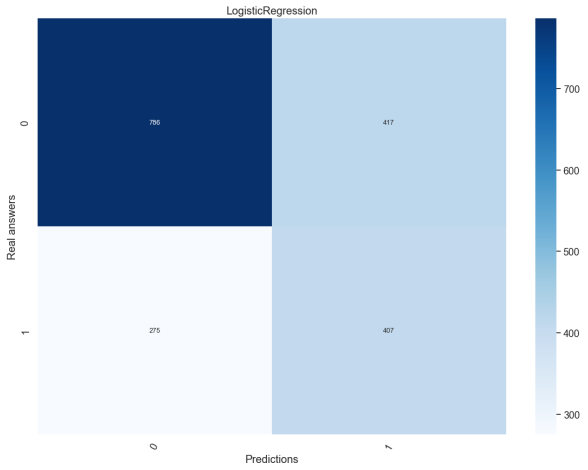


Figure 1: Logistic Regression confusion matrix

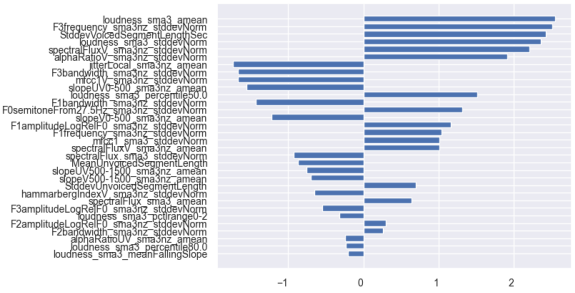


Figure 2: Logistic Regression model top-20 important features

# 4 Results

## 4.1 Functional approach

### 4.1.1 Baseline

With Logistic Regression algorithm, we reached F1-score of 0.54. According to the 1, the model did not seem to be overfitted only to one class, though it did wrong predictions of 0 class. We also investigated, which of the Functional features were of the highest importance (2). The most important features were mostly normalized audiofeatures which describe the loudness, frequency and slope of the line connecting a local minimum (or the beginning of input sample) with the following local maximum/peak or the end of audio sample. As the baseline shed some light on what features can positively affect models' predictions, we trained Logistic Regression with hyperparameters, obtained after the Grid search, on only top-20 and top-10 important features. However, it did not lead to the increase in evaluation metrics (see 1).

| Algorithm | Precision | Recall | F1 | F1 weighted | Accuracy |
|---|---|---|---|---|---|
| Logistic Regression | 0.49 | 0.6 | 0.54 | 0.64 | 0.63 |
| Logistic Regression, top-20 features | 0.46 | 0.56 | 0.5 | 0.61 | 0.6 |
| Logistic Regression, top-10 features | 0.43 | 0.52 | 0.47 | 0.58 | 0.57 |

Table 1: Logistic regression metrics

### 4.1.2 Passive Aggressive Classifier

Passive Aggressive Classifier is a classification algorithm based on the idea of being "passive" when the current model correctly classifies a training example, but "aggressive" when it makes a mistake. This algorithm demonstrates an ability to effectively manage substantial datasets, a quality that proved beneficial when applied to the provided datasets. Nevertheless, despite undergoing a Grid search for hyperparameter optimization, the Passive Aggressive Classifier exhibited signs of overfitting and was biased towards class 0 in its classifications (see 8 ).

### 4.1.3 Decision Tree

The algorithm's training process resulted in overfitting towards class 0, a behavior incongruent with the current task's objectives. This observation is substantiated by the confusion matrix presented in (10), which substantiates the absence of any enhancements in comparison to the baseline outputs.

### 4.1.4 Random Forest

The Random Forest Classifier, in contrast, demonstrated an absence of overfitting on the training data, as depicted in (see 9) . Furthermore, it exhibited enhancements in performance relative to the baseline outcomes. However, training the algorithm on its most significant 20 features failed to yield a discernible improvement in prediction accuracy. The resultant weighted F1-score was computed at 0.68.

### 4.1.5 Gradient Boosting

We used an optimized distributed gradient boosting library `XGBoost` with `Grid search` algorithm and achieved an improvement in classification metrics. According to 3 , the model was trained to detect intoxicated speech better than the Baseline, with higher accuracy and F1-score (2). As for the feature importance ( 4), the top-10 features are mostly different mean lower Mel-frequency cepstral coefficients (MFCCs). Since lower coefficients reflect the macro-structure of the spectrum (vocal tract configuration, formants), these components in audiofile representations might have helped detecting

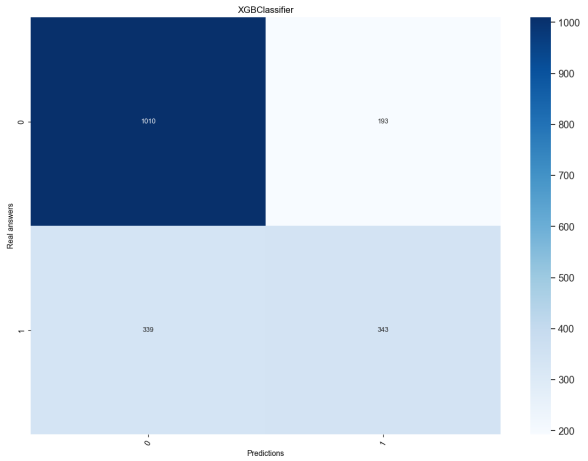


Figure 3: XGboost confusion matrix

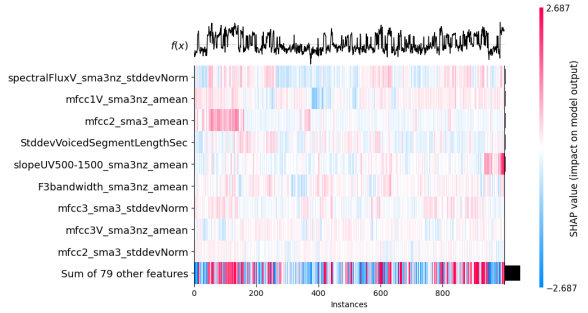


Figure 4: Gradient Boosting feature importance

intoxicated speech. Training the XGBoost with the mentioned top-10 features did not give substantial increase in metrics.

### 4.1.6 Simple Neural Networks

Simple Neural Network reached did not overfit on the training data and reached accuracy of 0.77 and weighted F1-score of 0.73 (5). The feature importance analysis (6) showed that the features of slopes and frequencies helped the model distinguish sober and intoxicated speech at most.

### 4.1.7 Summary

After conducting more than 100 experiments with ML-based approaches, finding the best hyperparameters with `Grid Search` algorithm, Gradient Boosting was found the best in our task. It was able to reach the maximum accuracy of 0.73 and weighted F1-score of 0.71. Training all the not-overfitted algorithms with their top-10 or top-20 most important features only did not bring any or substantial increase in models' predictions. The algorithms showed the importance of MFCCs, loudness and frequency features in the processed audiofiles.

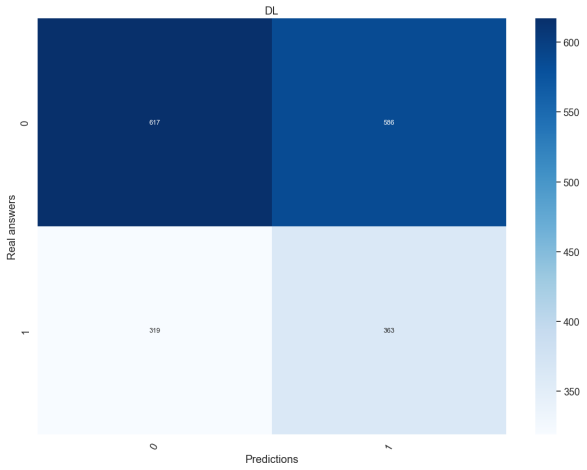


Figure 5: Simple NN confusion matrix

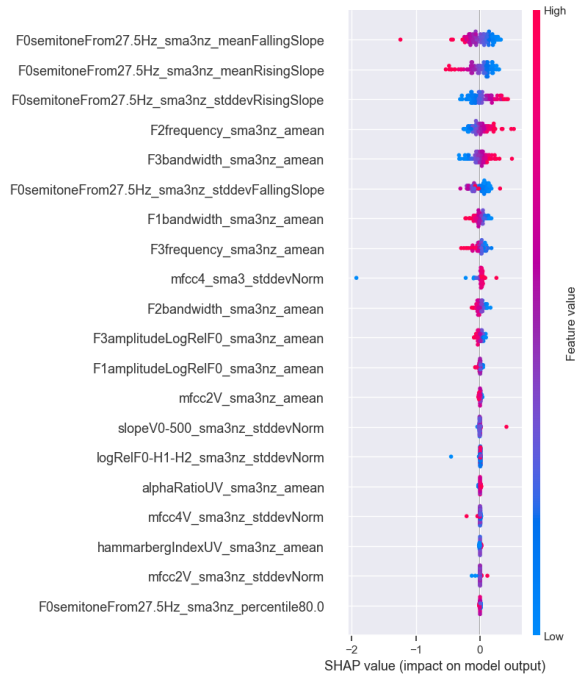


Figure 6: Simple NN feature importance

| Algorithm | Precision | Recall | F1 | F1 weighted | Accuracy |
|---|---|---|---|---|---|
| Logistic Regression | 0.49 | 0.6 | 0.54 | 0.64 | 0.63 |
| Passive Aggressive Classifier | 0.36 | 1.0 | 0.53 | 0.19 | 0.36 |
| Decision Tree | 0.54 | 0.27 | 0.36 | 0.62 | 0.65 |
| Random Forest | 0.59 | 0.48 | 0.53 | 0.68 | 0.69 |
| **Gradient Boosting** | **0.67** | **0.48** | **0.56** | **0.71** | **0.73** |
| **Simple NN** | **0.69** | **0.58** | **0.78** | **0.73** | **0.77** |

Table 2: Functional Approaches

#### 4.1.8 Correlations with meta-data

Analysis of best models' outputs helped us find the following tendencies of being biased towards :

- Gender: models predict female being drunk two times more often
- Age: speech of people younger than 26 is more often predicted as intoxicated, while 26-35 aged people are more often considered as being sober
- Drinking habits: in most cases people with light drh were predicted to speak under intoxication, while people with moderate drh were able to hide from models their intoxication
- General dispositon: frolicsome mood, happiness and being relaxed or tired speech in many cases was predicted as intoxicated
- Content: tongue twisters and numbers helped detect intoxication in most cases correctly
- "Special cases": no significant problems except 2 persons with "strange velar plosives" whose speech was slightly more often predicted as drunk

### 4.2 Sequential approach

The whole Table of results of each experiment could be found in. The top-3 outputs were by bidirectional LSTM models with Batch Normalization and no schedulers during the training process. Adam and Adagrad optimizers with 1e-5 learning rate and sigmoid activation function obtained best results: accuracy of 0.52, F1-score weighted of 0.53.

| Model | Precision | Recall | F1 | F1 weighted | Accuracy |
|---|---|---|---|---|---|
| 0, Adam , 1e-06, no sch | 0.36 | 0.47 | 0.41 | 0.53 | 0.5 |
| **0.3, Adagrad , 1e-05, RLR** | **0.36** | **0.45** | **0.40** | **0.53** | **0.52** |
| **0.3, Adam , 1e-05, RLR** | **0.36** | **0.45** | **0.40** | **0.53** | **0.52** |

Table 3: Sequential Approaches. , dr-dropout, Adam/Adagrad - optimizers, no sch - no scheduler, RLR - 'ReduceLROnPlateau' scheduler

#### 4.2.1 Correlations with meta-data

- Age and Drinking habits had similar tendencies for correlation with models' predictions, while general disposition did not correlate with the outputs.

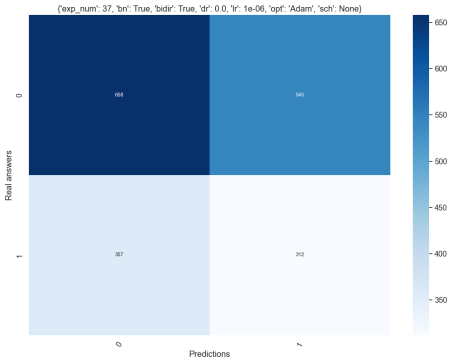


Figure 7: LSTM model, dropout=0, Adam optimizer, lr=1e-06, no scheduler

- Content: addresses and spontaneous commands, as well as tongue twisters and numbers helped detect intoxication in most cases correctly
- "Special cases": audiofiles with "strange velar plosives" in most cases had incorrect models' predictions, as well as audios from non-natives, which differs drastically from the ML-based algorithms' predictions

## 5 Summary

We conducted a plethora of experiments, based on approaches to both functional and sequential feature representations. The optimal outcome was attained by a straightforward neural network architecture utilizing functional audio features, yielding an accuracy of 0.77 and an F1-score of 0.78. Following this, the Gradient Boosting algorithm performed second best over the same functionals, yielding an accuracy of 0.73 and an F1-score of 0.56. In terms of sequential-based strategies, the highest result was achieved by an LSTM model with an accuracy of 0.52 and an F1-score of 0.4.

Key features pivotal for effective intoxicated speech detection were identified, encompassing MFCCs, loudness, and frequency. Moreover, correlations were observed between meta-data and erroneous model predictions. Notably, biases towards individuals under 26, female speakers, and those with light drinking habits were apparent. Specific textual inputs, such as tongue twisters, addresses, and numerical content, were deemed most effective for intoxicated speech detection.

## 6 Further work

### 6.0.1 Data type

Texts with tongue twisters, addresses, numbers were found as most prominent types for better intoxicated speech detection, so only they can be used as training data. In addition, recording containing the mentions text categories are shorter, so it will be computationally less expensive to train new models. The representations of the audiosamples should be more distinguishable for the algorithms as well, because less timestamps will be processed.

### 6.0.2 Feature representations

Usage of `Praat` library might make the vector representations (both functional and sequential) more complete, which might help algorithms perform with higher metrics since Praat library contains pitch and energy normalization, autocorrelation, function methods.

### 6.0.3 Data augmentation

Adding noise, shifting pitch or stretching the audio can make data more variable and help models deal with noisy audiosample making them more stable to real-life cases.

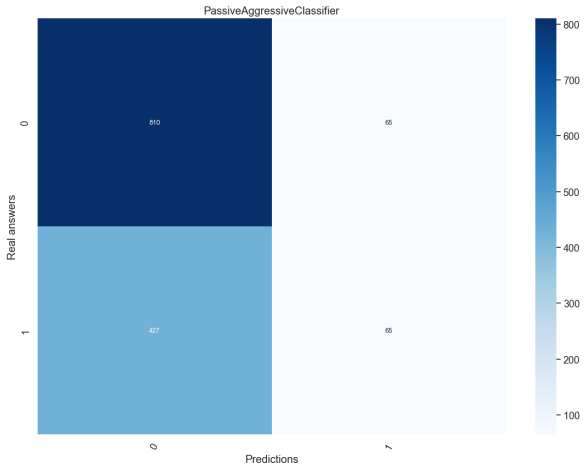


Figure 8: Passive Aggressive Classifier confusion matrix

## References


Ayman Al-Kababji, Faycal Bensaali, and Sarada Prasad Dakua. 2022. Scheduling techniques for liver segmentation: Reducelronplateau vs onecyclelr. In *International Conference on Intelligent Systems and Pattern Recognition*, pages 204–212. Springer.

Daniel Bone, Matthew Black, Ming Li, Angeliki Metallinou, Sungbok Lee, and Shrikanth Narayanan. 2011. Intoxicated speech detection by fusion of speaker normalized hierarchical features and gmm supervectors. pages 3217–3220.

Florian Eyben, Martin Wöllmer, and Björn Schuller. 2010. Opensmile: the munich versatile and fast open-source audio feature extractor. In *Proceedings of the 18th ACM international conference on Multimedia*, pages 1459–1462.

Amir Farzad, Hoda Mashayekhi, and Hamid Hassanpour. 2019. A comparative performance analysis of different activation functions in lstm networks for classification. *Neural Computing and Applications*, 31:2507–2521.

Saad Hikmat Haji and Adnan Mohsin Abdulazeez. 2021. Comparison of optimization techniques based on gradient descent algorithm: A review. *PalArch's Journal of Archaeology of Egypt/Egyptology*, 18(4):2715–2743.

PM Lerman. 1980. Fitting segmented regression models by grid search. *Journal of the Royal Statistical Society Series C: Applied Statistics*, 29(1):77–84.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

SAINA Rezvani. 2019. Intoxication detection from audio using deep learning. *Ph. D. dissertation*.

Florian Schiel, Christian Heinrich, Sabine Barfüßer, and Thomas Gilg. 2008. ALC: Alcohol language corpus. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Marrakech, Morocco. European Language Resources Association (ELRA).

Zixing Zhang, Felix Weninger, and Björn Schuller. 2012. Towards automatic intoxication detection from speech in real-life acoustic environments. In *Speech Communication; 10. ITG Symposium*, pages 1–4. VDE.


## A Appendix

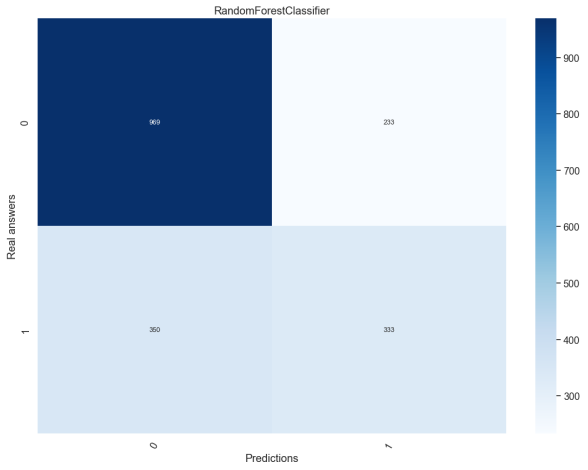


Figure 9: Random Forest confusion matrix

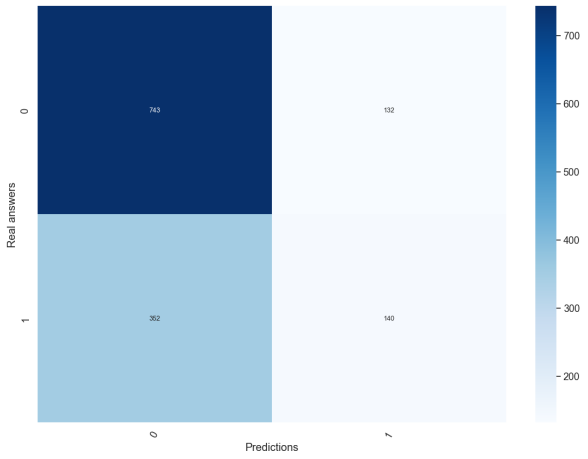


Figure 10: Decision Tree confusion matrix

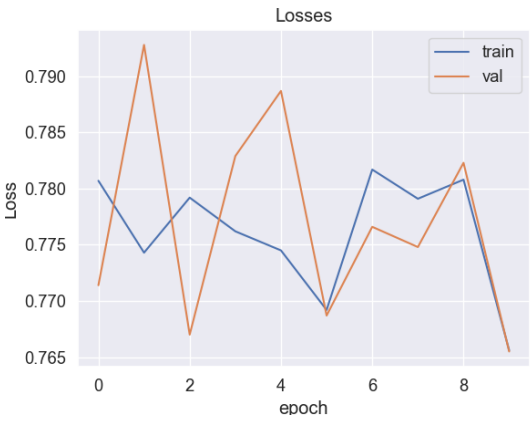


Figure 11: Training and validation losses, LSTM model, dropout=0, Adam optimizer, lr=1e-06, no scheduler

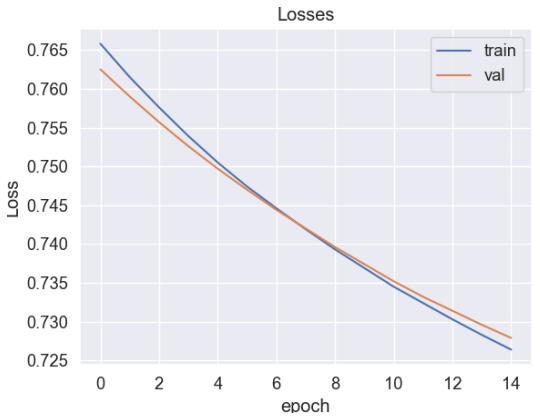


Figure 12: Training and validation losses, LSTM model, dropout=0.3, Adagrad optimizer, lr=1e-05, 'ReduceLROnPlateau' scheduler

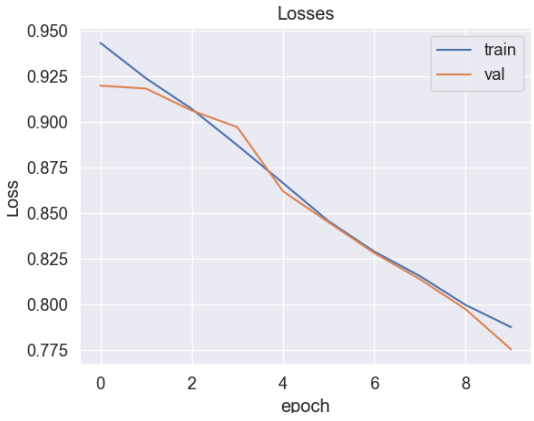


Figure 13: Training and validation losses, LSTM model, dropout=0.3, Adam optimizer, lr=1e-06, 'ReduceLROnPlateau' scheduler