Ming Hsieh Department
of Electrical Engineering

EE660
Machine Learning from Signals: Foundations and Methods
Final Project Report

Speaker Classification for South Park TV Series

Nikolaos Flemotomos
flemotom@usc.edu
USC I.D.: 7149389176

December 4, 2017

Abstract

Speaker Classification (SC) is an important issue in Speech and Language Processing. In this work,various classification approaches are compared for text-based SC, applying the different techniques in the scripts of the popular animated TV series *South Park*. In particular, two distinct problems are adressed. Firstly, a binary problem is constructed where giving an utterance to the classifier, the desired output is whether it has been said by one of the main characters of the series or not. Seconldy, a multiclass problem is contstructed where the goal is to classify the four main characters of the series. The best performance, in terms of the F_1 score, is achieved by the logistic regression algorithm which yields $F_1 = 0.6471$ for the binary problem with and $F_1 = 0.1434$ for the multiclass problem.

1 Problem Statement & Goals

The aim of this project is to apply feature engineering and Machine Leraning techniques for text-based Speaker Classification (SC), also known as Speaker Identification. The problem can be thought of as an effort to find linguistic patterns that have the potential of distinguishing different utterances with respect to the speaker they correspond to. The dataset used is the *South Park Dialogue*[1], which provides more than 70896 lines (where each line is an utterance) of the popular animated TV series *South Park*, annotated with speaker labels. The goal is to use commonly used language features in order to predict the speaker of a specific utterance, by knowing what (s)he has said. By that perspective, it is a supervised learning problem in the area of Natural Language Processing.

The dataset contains 70896 utterances from 3950 unique characters. Lots of the characters, however, appear in just one episode or even speak just once, making it impossible for a system to be trained to predict them. Additionally, an accurate and robust system would depend heavily on the language features used. In order to avoid devoting most of the time looking for suitable sophisticated features, since that would be out of the scope of this class, and to be able to compare different classification methods, while preserving as much interpretability as possible, two 'lighter' versions of the problem were constructed and addressed. Out of all those characters appearing in the show, only 4 of them are the main ones (Cartman, Stan, Kyle and Kenny)[2]. So, the first problem is a binary one, where the goal is to predict whether an utterance has been said by a main character or not. The second problem uses only the utterance from those 4 characters and the goal is to predict who is the character who said a specific utterance. It is, thus, a multiclass classification problem with 4 classes.

The problems addressed are inherently difficult for a variety of reasons. First of all, significant pre-processing of the available data is required, since just raw text is provided. Additionally, the n-grams, which are the most commonly used language features for such tasks, lead to a very high-dimensional and sparse feature space which requires careful handling. Finally, human language is one of the most complicated mechanisms and trying to find distinguishable patterns between different users of this mechanism is a really challenging task.

2 Prior & Related Work

I have worked on another project in the past[2], involving similar feature extraction techniques for document classification. However, the application domain, as well as the classification approach, was completely different.

[1]https://www.kaggle.com/tovarischsukhov/southparklines
[2]https://en.wikipedia.org/wiki/South_Park

3 Project Formulation & Setup

For both the problems addressed (binary and multi-class) the general steps followed are the same. The overall procedure is summarized in Figure 1.

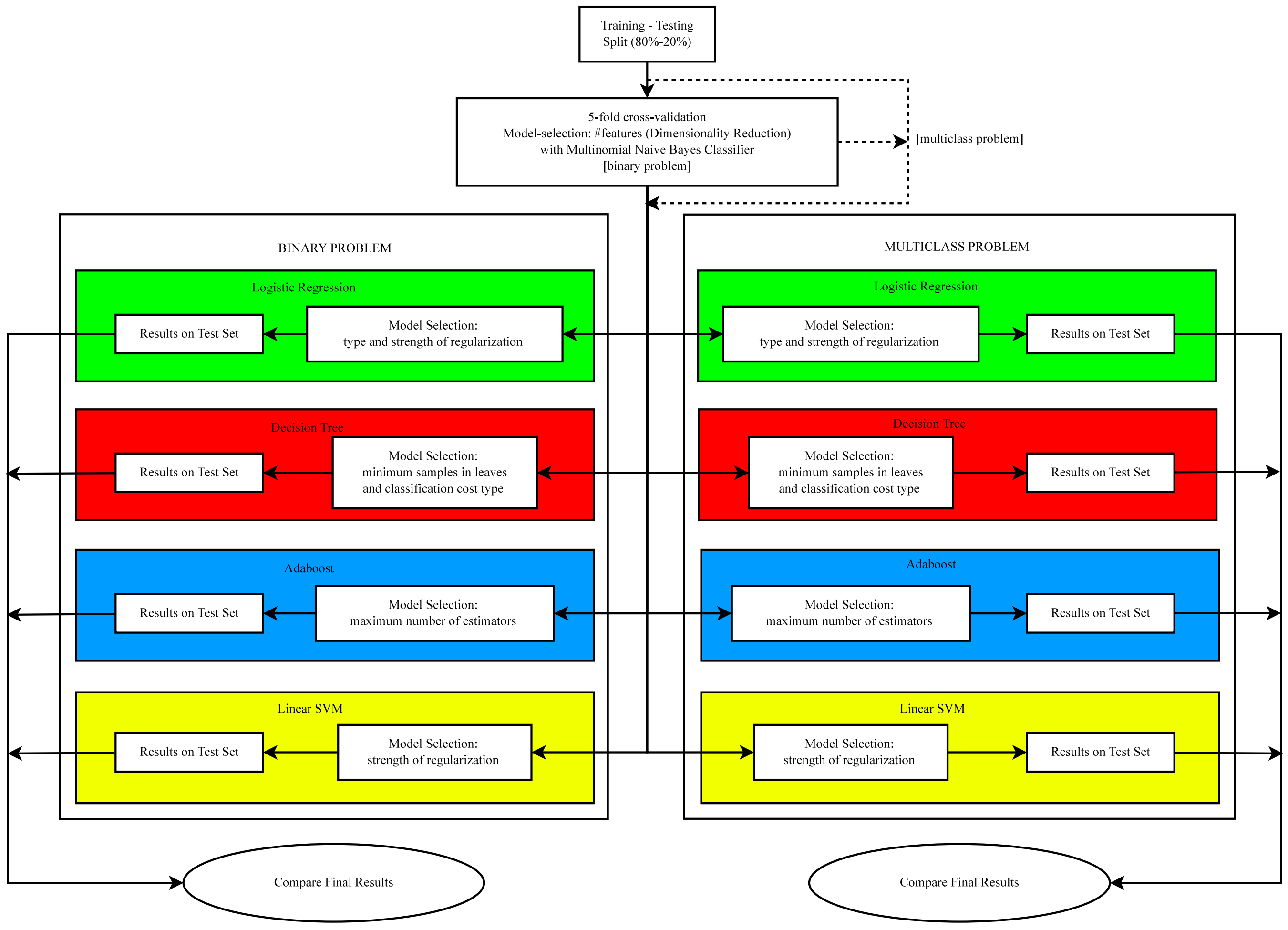

Training - Testing Split (80%-20%)
5-fold cross-validation Model-selection: #features (Dimensionality Reduction) with Multinomial Naive Bayes Classifier [binary problem]
[multiclass problem]
BINARY PROBLEM
MULTICLASS PROBLEM
Logistic Regression
Model Selection: type and strength of regularization
Results on Test Set
Decision Tree
Model Selection: minimum samples in leaves and classification cost type
Adaboost
Model Selection: maximum number of estimators
Linear SVM
Model Selection: strength of regularization
Compare Final Results

Figure 1: Flowchart of the project formulation and setup. All the 'Model Selection' modules are based on a 5-fold cross-validation on the training set.

First of all, the entire dataset is split into training and test set based on a 80-20 split. It should be noted that the initial dataset is different for the two problems. Specifically, just a subset of the data is used for the multiclass problem, since only the utterances spoken by one of the main characters are taken into consideration. The test set is held aside and all the subsequent computations are based on the training set. The first step is to find the number (and type) of features to use. More details on that module will be given in Section 4 and Subsection 5.1. This is done only for the binary problem and the result obtained is also used for the multiclass problem. Ideally, this model selection would be incorporated in the subsequent model selection schemes so that an optimal number of features would be deduced for each classifier. However, it was decided to follow this approach in order to reduce the overall complexity and to compare all the classifiers when applied on the same features.

Then, for each of the two problems, 4 different classifiers are applied and compared to each other:

logistic regression, decision trees, adaboost with decision stumps, and linear Support Vector Machines (SVMs). Each one of them has a number of parameters to be tuned and the tuning is done each time through a 5-fold cross-validation. A brief mathematical formulation of the classifiers is given below. In the case that the classifier is inherently targeted to binary classification problems, only the binary formulation is given. The extension to the multiclass problem is done using the One-Vs-Rest approach.

The objective function for binary-class logistic regression can be differentiated depending on the regularization used. In this work, l_1- and l_2-regularization are applied and compared. The minimization problem for the l_2 regularized logistic regression is formulated as

$$\min_{\mathbf{w},w_0} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N} \log\left\{\exp\left(-y_i\left(\mathbf{x}_i^T\mathbf{w}+w_0\right)\right)+1\right\} \tag{1}$$

while for the l_1 regularized logistic regression the corresponding formula is

$$\min_{\mathbf{w},w_0} ||\mathbf{w}||_1 + C\sum_{i=1}^{N} \log\left\{\exp\left(-y_i\left(\mathbf{x}_i^T\mathbf{w}+w_0\right)\right)+1\right\} \tag{2}$$

where N is the number of training samples, $y_i \in \{-1, 1\}$, and C is a parameter which affects the strength of regularization (the smaller the value of C, the greater the regularization effects).

For the linear SVM the optimization problem is given as

$$\min_{\mathbf{w},w_0,\boldsymbol{\zeta}} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\zeta_i \tag{3}$$

$$\text{such that } y_i\left(\mathbf{x}_i^T\mathbf{w}+w_0\right) \geq +1-\zeta_i \tag{4}$$

$$\text{and } \zeta_i \geq 0, \quad i = 1, 2, \cdots, N \tag{5}$$

The parameter C which appears in equations (1), (2) and (3) needs to be tuned.

For the decision tree, the algorithm used is CART. In order to achieve an optimal performance of the algorithm, an early stopping criterion usually needs to be applied, in order to avoid overfitting. In this work, the approach taken involves the search for the optimal value of the minimum number of samples $\tilde{M}$ required in each leaf node of the tree. From that viewpoint, the algorithm does not split any node if immeiatly after the split, the children are going to define an area in the feature space where less than $\tilde{M}$ samples exist. Apart from finding the best value for $\tilde{M}$, the model selection here also involves the decision of which classification cost, between the Gini index and the cross-entropy, should be used.

Decision trees comprise an adaptive basis function model which usually yields non-robust results. This is why we also try a boosting technique. Specifically, the algorithm applied is the Adaboost, where the weak estimators are decision stumps, that is one-stage CART trees. The maximum number of weak estimators used in the iterative procedure of the algorithm is again decided after a model selection module.

4 Methodology

The dataset available includes 70896 lines of dialogue between different characters of *South Park*. For the binary problem, the entire dataset is used, where each lines gets the label 'main character' or 'not main character'. For the multiclass problem, only the 25434 lines corresponding to the 4 main characters of the series are considered and each line gets the label 'Cartman', 'Stan', 'Kyle' or 'Kenny'. The number of samples corresponding to those classes are reported in Table 1.

First of all, for both probelms, the dataset is randomly shuffled and 80% is used for the training of the various systems. The rest of the samples are held aside in order to be used for the final evaluation and

character	number of samples (lines)
Cartman	9774
Stan	7680
Kyle	7099
Kenny	881
not main character	45462

Table 1: Number of samples per class. For the multiclass problem, only the samples corresponding to the 4 main characters are used. For the binary problem, all the samples are used with the 4 main characters being merged into one class.

the estimation of the out-of-sample performance. That process leaves us with 56717 training samples for the binary probelm and 20348 training sampes for the multiclass problem.

The features which are going to be used are not known a priori, since initially only the raw text is available. Maybe the most commonly used feature set for document representation is the set of n-grams, calculating either the occurrences or the frequency of occurrence of each n-gram (sequence of n words). It is obvious that is the general case the number of features is not known, since it depends on the training set. So, if cross-validation is used at some point of the process, the feature extraction model has to be fit at each fold in the new training set, because the words (or n-grams) existing in the particular set of samples may be different than in other sets. In order to ensure that we compare differenct classifiers when applied to a dataset with a fixed number of features, we have an initial model selection scheme to choose the final number of features to be used.

The dimensionality reduction is done based on a univariate F-test[3] with a 5-fold cross-validation scheme, after which only the K best features (with the highest F-score) are kept. The reason I chose a univariate test instead of a more robust statistical test, such as a forward or backward selection was to reduce the computational complexity, which is already high when dealing with such high-dimensional spaces. On the other hand, the reason I chose a statistical test instead of a commonly used dimensionality reduction technique in Natural Language Processing, such as Truncated SVD (known as Latent Semantic Analysis [4]) is because it was desired to preserve the interpretability of the features (words), avoiding changing the coordinates of the feature space.

As for the order of n-grams, it is assumed a priori that unigrams (just words) should be used, as explained in Subsection 5.1. However, for completeness (and for validation of the assumption), the performance is also evaluated with higher order n-grams, as well as with the Global Vectors for Word Representation (GloVe) [5]. GloVe is a set of 300-dimensional features for semantic vector space representation of words, pretrained on 840B tokens found in the Web. In order to derive an utterance embedding, the mean of the word embeddings (for the words in the utterance) is computed.

The feature selection procedure is only done for the binary problem and the result is also used for the multiclass problem.

After the decision about the features to be used, the 4 classifiers introduced in Section 3 (Logistic Regression, Decision Trees (CART), Boosting (Adaboost), Linear SVM) are trained. Before training on the whole training set, a model selection is applied based on a 5-fold cross validation. The parameters yielding the best results (averaged on the 5 folds) are used for the final training and the systems are evaluated on the test set.

Except for the classifiers, performance is also estimated for a baseline system, where the majority class is always the predicted one.

At each case, the main metric used for evaluation is not the accuracy, but the F_1 score, which is much more reliable, especially for unbalanced classes, as in the problems addressed in this work.

5 Implementation and Interpretation

The implementation of the project was done using the `sklearn` library in Python. Most of the plots were built using the Python library `matplotlib`, but some of them were built using MATLAB. The code used for the project (apart from some very simple MATLAB illustrations) is given in separate files together with this report.

5.1 Pre-processing and Feature Extraction

The assumption made is that the best option would be to use first order n-grams (unigrams), because the utterances are in general very small, so higher order n-grams cannot capture important information. Just to verify the assumption, I ran a 5-fold cross validation on the training set for the binary problem. The classifier used for this experiment is a simple Multinomial Naive Bayes classifier The results are shown in Figure 2 as a function of the number of features. As observed, the assumption is verified. Additionally, it is obvious that after some point, more features do not contribute to much better results. So, I chose to use the 250 most informative unigrams for all the subsequent experiments. Having in mind the rule of thumb that the number of samples should be at least equal to 5-10 times the number of features for classification tasks, the data samples available are more than enough.

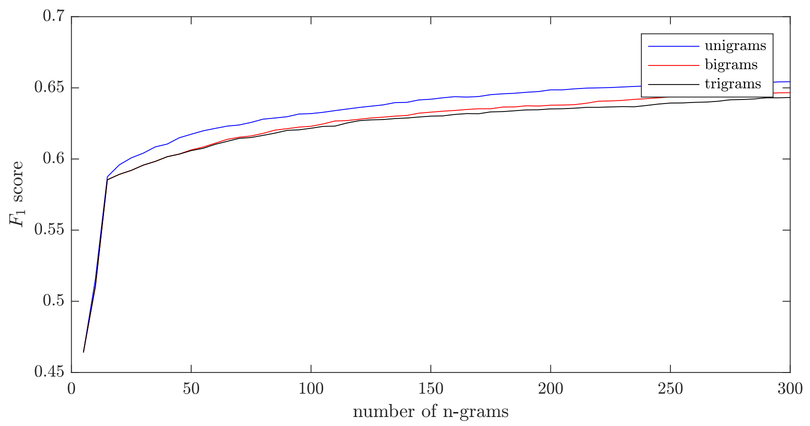


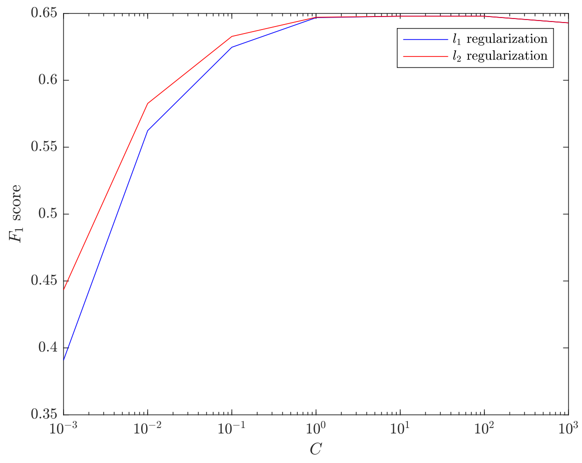
Figure 2: F_1 score as a function of number of n-grams used after a feature selection based on a univariate F-test.

For completeness, the same experiment (without any feature reduction) was done with standarized GloVe features, but with a Gaussian Naive Bayes classifier. The corresponding cross-validation F_1 score was 57.78, yielding a worse performance than the n-grams.

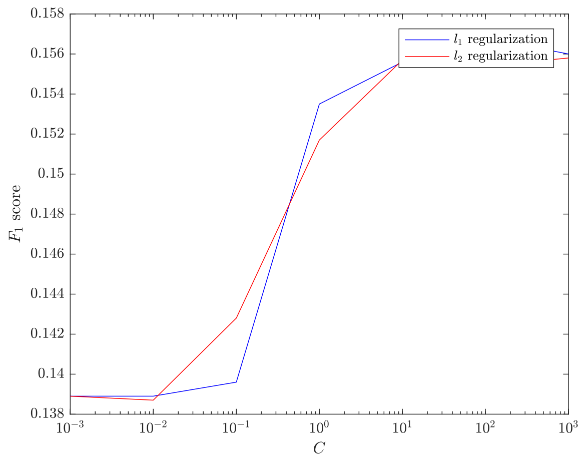
5.2 Logistic Regression

In Figure 3, the cross-validation F_1 score is plotted as a function of the regularization parameter C for the two regularization approaches. Based on the results, the training on the entire training set was done using $C = 100$ with l_2 regularization for the binary problem and $C = 100$ with l_2 regularization for the multiclass problem. Great regularization does not help the system. For example, in the case of l_1 regularization that promotes sparsity, this behavior was expected because I have already discarded most of the features through the feature selection.

For better visualization, I first trained the system using only the 2 most informative unigrams. The resulting decision regions are plotted in Figure 4. It is interesting to observe the decision boundaries



(a) Binary problem.



(b) Multiclass problem.

Figure 3: F_1 score as a function of the regularization parameter C for logistic regression.

for the binary problem with the help of the plotted histograms. Even though for both classes the two particular unigrams are not observed most of the time (as expected - it would be really weird if a specific word existed in every utterance), there is a significant amount of utterances for the class 'main character' where one of the two words exists. The decision boundary is then drwan by the algorithm based on that.

5.3 Decision Trees

In Figure 5, the cross-validation F_1 score is plotted as a function of the minimum number of samples allowed in the leaf nodes. Based on the results, the training on the entire training set was done using $\tilde{M} = 12$ with cross-entropy classification cost for the binary problem and $\tilde{M} = 4$ with Gini index for the multiclass problem. In general, early stopping is indeed required to avoid overfitting, but, as shown, we don't have to prune a lot (greater $\tilde{M}$ means greater pruning).

As in the case of logistic regression, the decision boundaries when using only 2 features are plotted in Figure 6. The results are similar, but as expected the regions are with boundaries parallel to the axes of the feature space.

Finally, in Figures 7 - 10, the actual trees constructed are given.

5.4 Adaboost

In Figure 11, the cross-validation F_1 score is plotted as a function of the number of weak estimators E (decision stumps). Based on the results, the training on the entire training set was done using $E = 200$ for the binary problem and $E = 350$ for the multiclass problem. It is made obvious that increasing the number of weak estimators doesn't necesserily lead to improved performance.

5.5 Linear SVM

In Figure 12, the cross-validation F_1 score is plotted as a function of the regularization parameter C for the two regularization approaches. Based on the results, the training on the entire training set was done using $C = 100$ with for the binary problem and $C = 0.001$ for the multiclass problem. The different behavior of the two problems in that aspect is really interesting. However, we should have in mind that the differences in performance for the multiclass problem are very subtle.

6 Final Results

The final results, after training the classifiers on the entire training sets with the parameters selected through the model selection and evaluating them on the corresponding test sets, are reported in Tables 2 and 3. It is noted that all the metrics used (both here and previously) are macro-averaged across the different classes. With the particular approach we have followed, since the test set was set aside from the very beginning, those test errors actually reflect reliable estimations for the corresponding out-of-sample errors of each algotithm. In particular, the out-of-sample error is less than the test error reported plus a tolerance of $\sqrt{\frac{1}{2N}\ln\frac{2}{\delta}}$ with probability $1-\delta$, where $N = 56717$ for the binary problem and $N = 20348$ for the multiclass problem.

	accuracy	precision	recall	F_1 score
logistic regression	**72.28**	**72.59**	**64.34**	**0.6471**
decision tree	70.42	67.44	62.31	0.6232
Adaboost	72.14	72.42	64.16	0.6449
linear SVM	71.65	71.90	63.42	0.6358
baseline	64.07	32.04	50.00	0.3905

Table 2: Final results for the binary problem.

	accuracy	precision	recall	F_1 score
logistic regression	37.99	**38.89**	25.06	**0.1434**
decision tree	**38.32**	13.75	**25.13**	0.1415
Adaboost	37.99	19.02	25.06	0.1430
linear SVM	37.89	22.96	24.99	0.1427
baseline	**38.32**	9.58	25.00	0.1385

Table 3: Final results for the multiclass problem.

For the binary problem, logistic regression gives the best results , with Adaboost, however, being really close. It is interesting that Adaboost (with uses a big number of the simplest possible CART trees) yields better results than the CART algorithm itself. All the algorithms implemented yielded significantly better results than the baseline.

For the multiclass problem, on the other hand, which is a significantly more difficult problem, the algorithms implemented did not manage to give great imrovements with respect to the baseline, as far as the F_1 score or the accuracy are concerned. However, it is interesting, that the precision is significantly improved, especially using logistic regression.

7 Summary and Conclusions

In this work, four different classification algorithms -Logistic Regression, Decision Trees (CART), Adaboost, and Linear SVM - were implemented and compared for two different problems, a binary one and a multiclass one. Special focus was given The final goal was to see how those algorithms can perform in the difficult task of text-based Speaker Classification, for which reason the scripts of the TV series *South Park* were used. Logistic Regression yielded the best results for both problems, even though the overall improvement was not big compared to the baseline for the multiclass problem. However, it was shown that even very simple linguistic features, such as the unigrams, can give valuable hints concerning the speaker information.

References

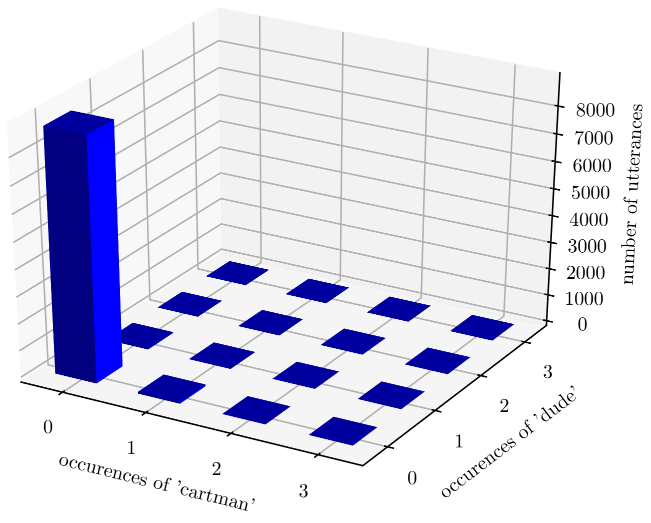
[1] K. P. Murphy, *Machine learning: a probabilistic perspective.* MIT press, 2012.

[2] N. Flemotomos, V. Martinez, D. Atkins, T. Creed, and S. Narayanan, "Language features for end-to-end automated evaluation of cognitive behavior psychotherapy sessions," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP) 2018 (submitted).*

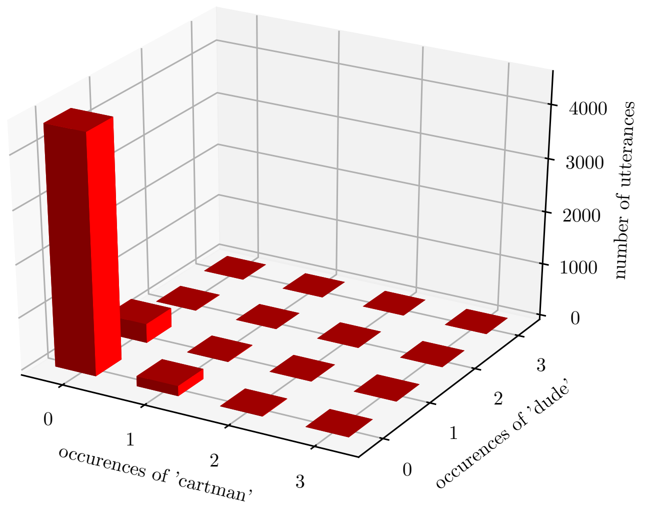
[3] W. Mendenhall and T. Sincich, *Statistics for Engineering and the Sciences.* Dellen Pub. Co.. Collier Macmillan Canada. Maxwell Macmillan International, 1991.

[4] T. K. Landauer, P. W. Foltz, and D. Laham, "An introduction to latent semantic analysis," *Discourse processes*, vol. 25, no. 2-3, pp. 259–284, 1998.

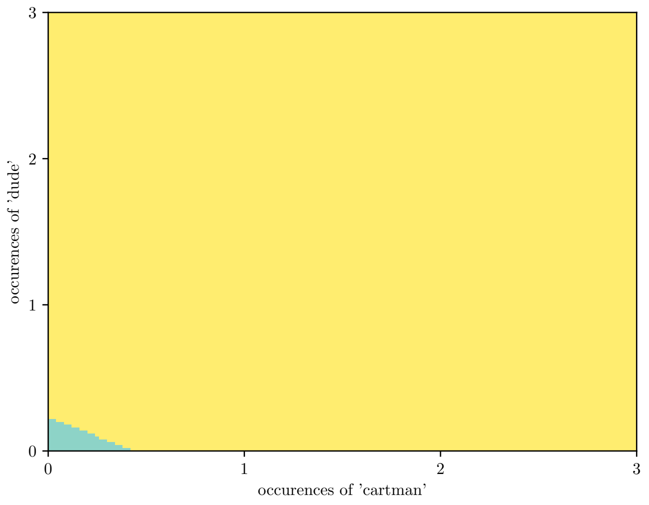
[5] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543, 2014.



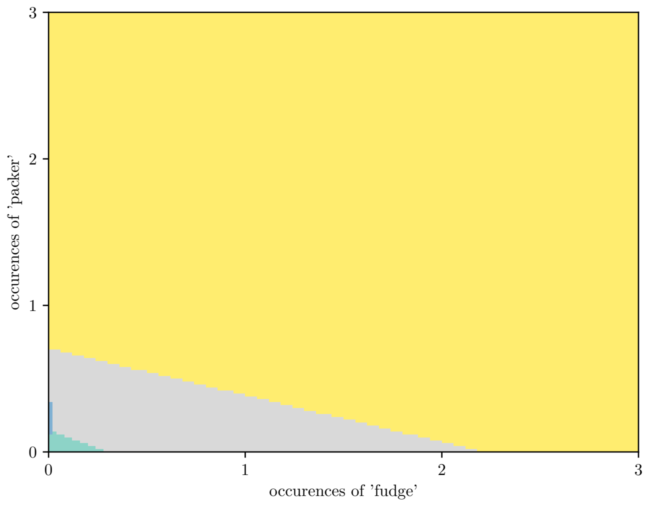
(a)



(b)

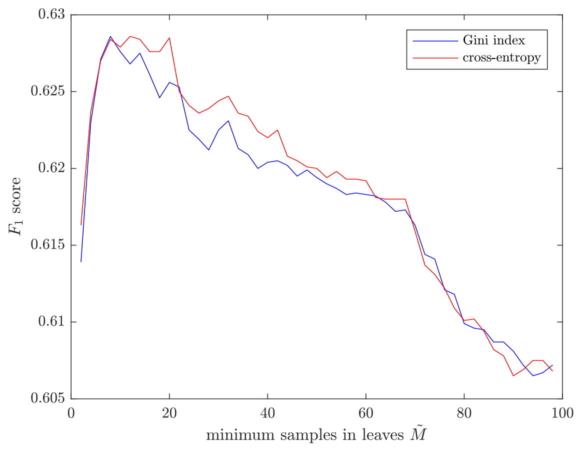


(c)

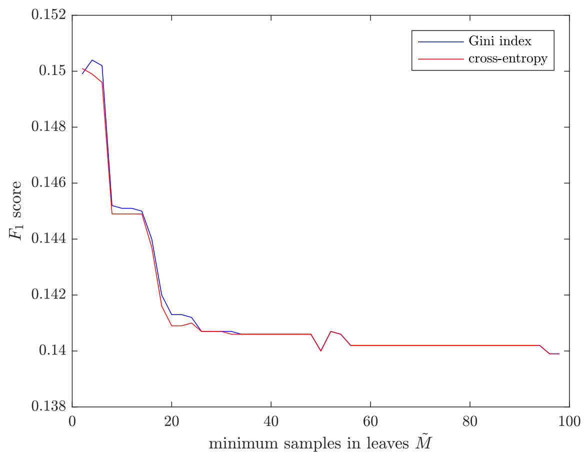


(d)

Figure 4: (a) histogram for the class 'not main character', (b) histogram for the class 'not main character', (c) decision regions for the binary problem (green: 'not main character', yellow: 'main character'), (d) decision regions for the multiclass problem (green: 'Kyle', yellow: 'Stan', grey:Cartman).

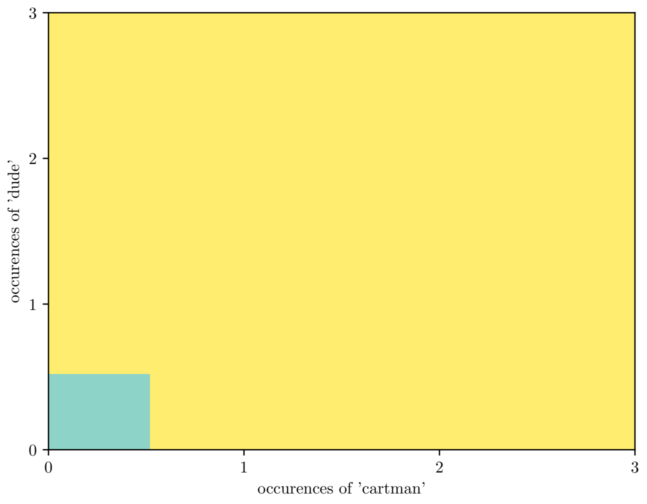


(a) Binary problem.

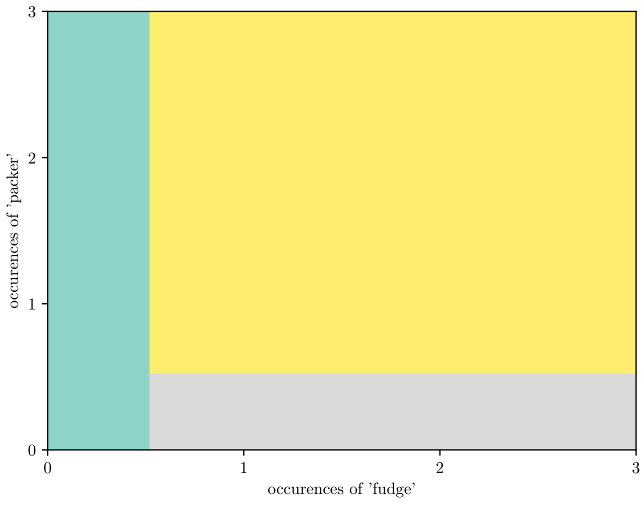


(b) Multiclass problem.

Figure 5: F_1 score as a function of the minimum number of samples allowed in leaf nodes $\tilde{M}$ for the CART algorithm.



(a)



(b)

Figure 6: (a) decision regions for the binary problem (green: 'not main character', yellow: 'main character'), (b) decision regions for the multiclass problem (green: 'Kyle', yellow: 'Stan', grey:Cartman).

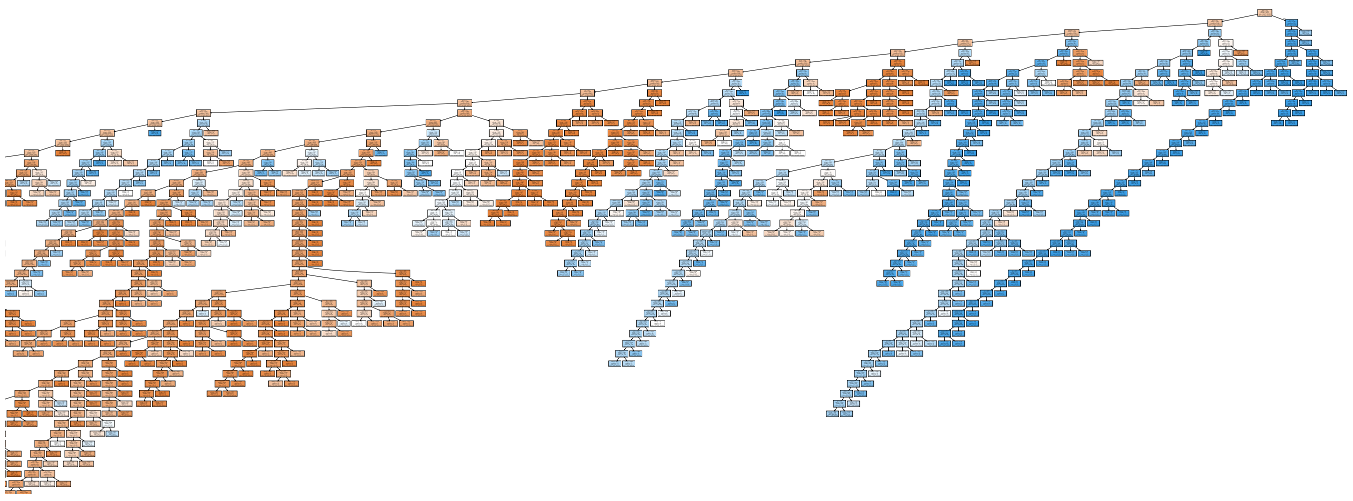


Figure 7: Part of the decision tree constructed for the binary problem.

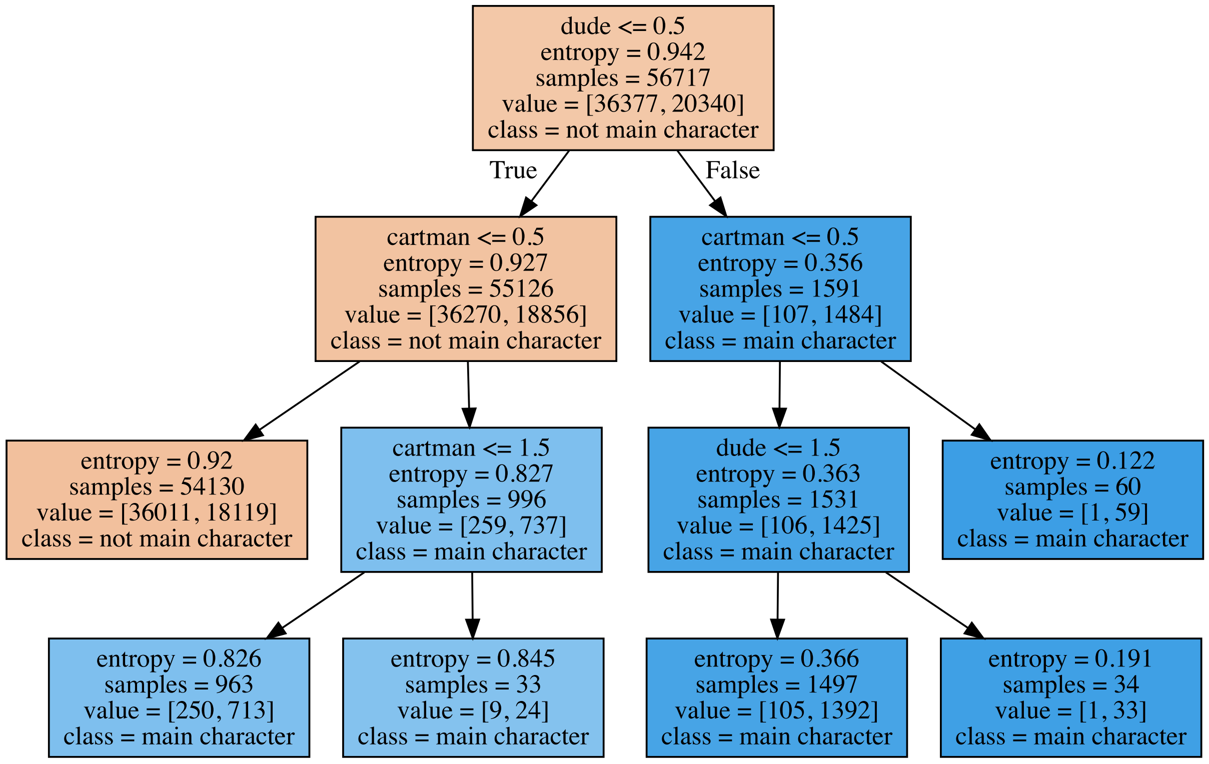


Figure 8: Decision tree constructed for the binary problem when only the 2 most informative features were used.

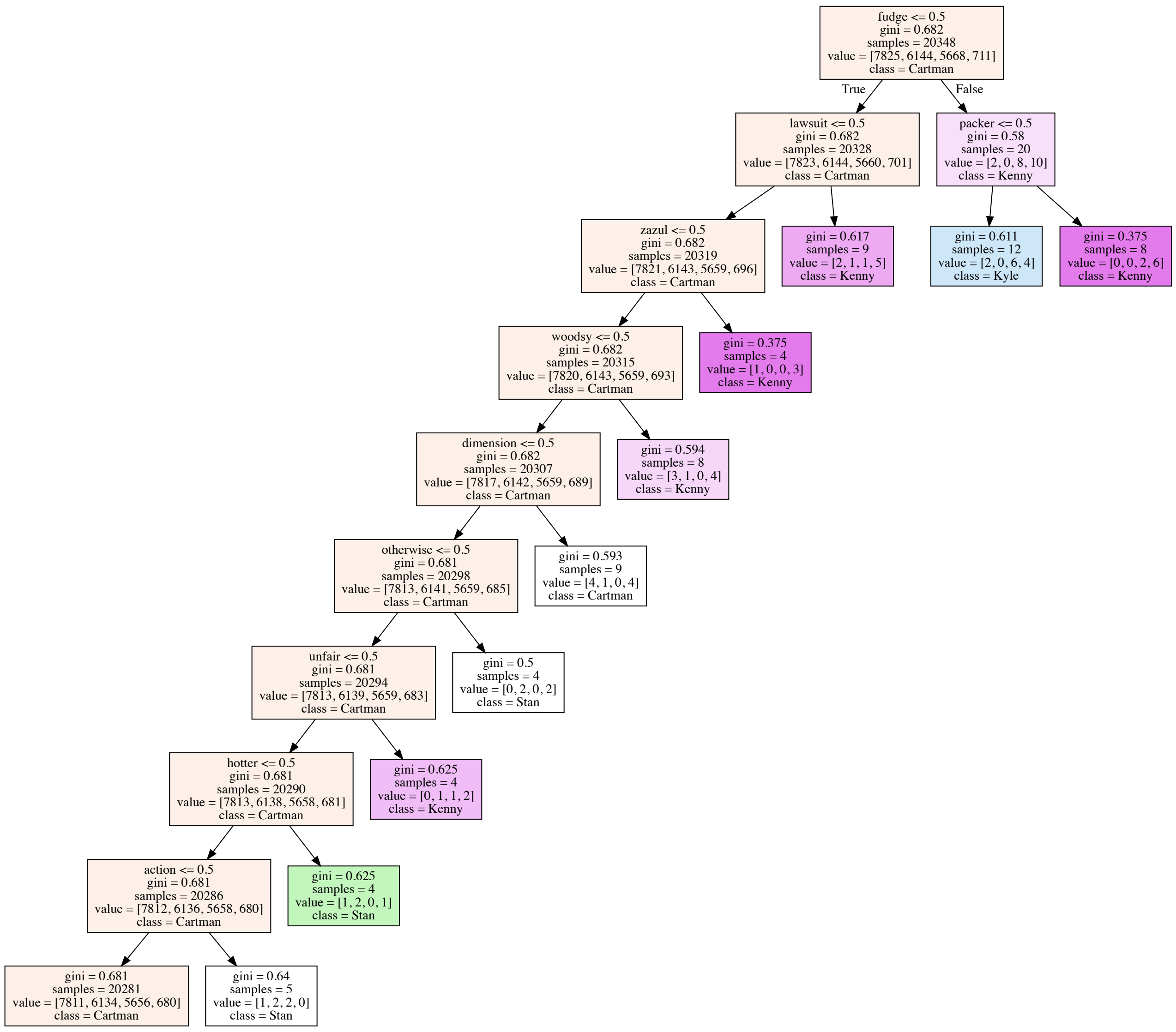


Figure 9: Decision tree constructed for the multiclass problem.

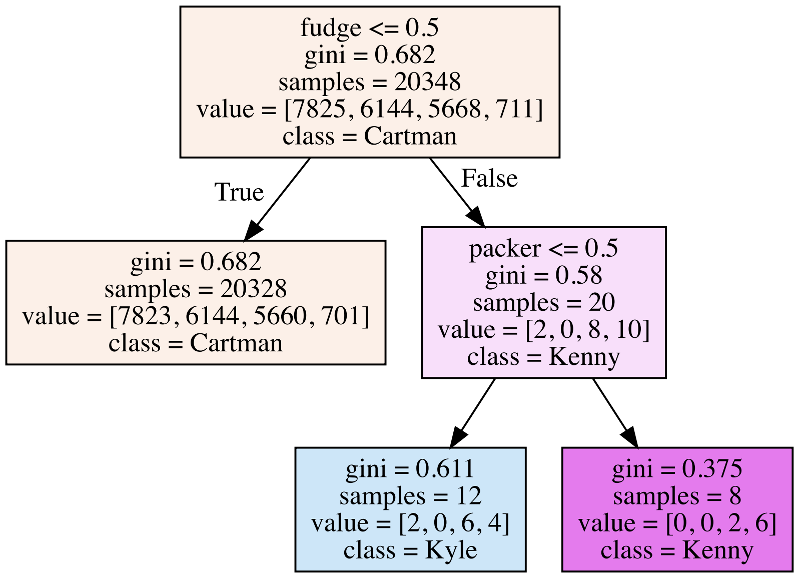
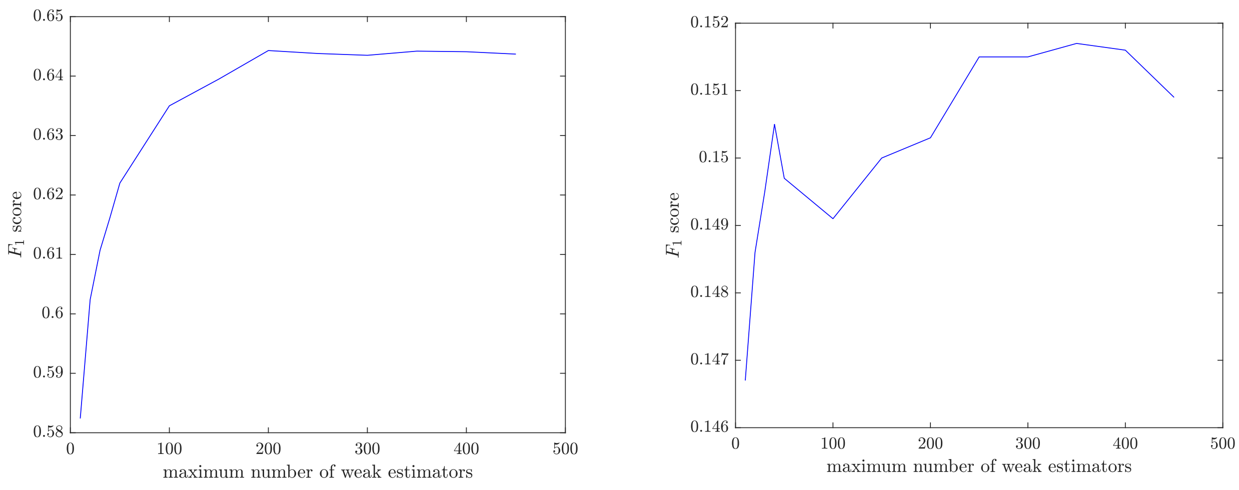


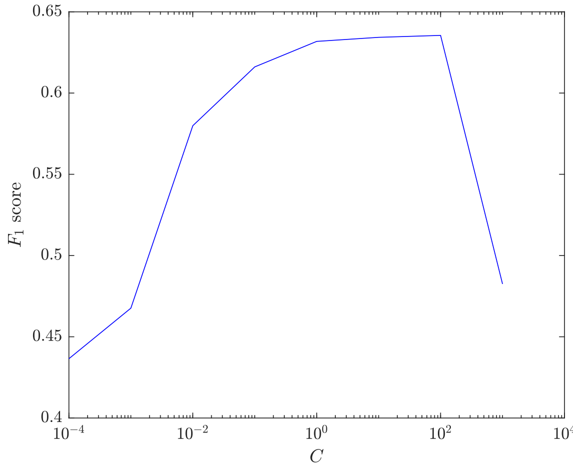
Figure 10: Decision tree constructed for the multiclass problem when only the 2 most informative features were used.



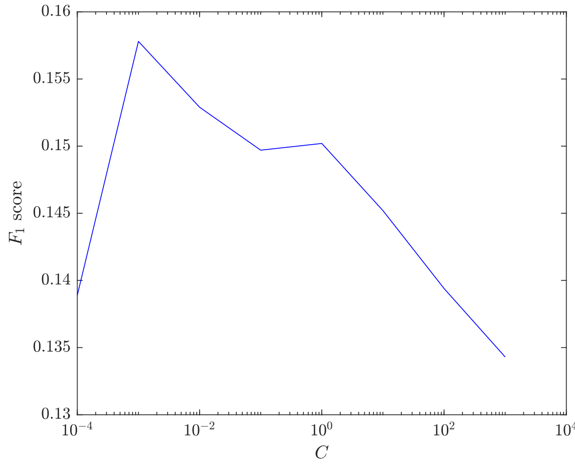
(a) Binary problem.

(b) Multiclass problem.

Figure 11: F_1 score as a function of the maximum number of weak estimators for the Adaboost algorithm with decision stumps.



(a) Binary problem.



(b) Multiclass problem.

Figure 12: F_1 score as a function of the regularization parameter C for the linear SVM.