Leveraging Machine Learning for Automated Detection of Check-Worthy Factual Claims

Simon Bross
University of Potsdam
bross@uni-potsdam.de

1 Introduction

In an era of information overload, the attention on (automated) fact-checking has gained considerable momentum since the 2016 presidential debate during which Donald Trump emerged victorious as the 45th President of the United States of America. It marked the advent of a pervasive crisis of truth that has challenged the traditional notions of falsehood and lead to a surge in accusations of fake news dissemination, not only if allegedly false factual claims are propagated but also particularly if statements do not align with mainstream discourse.

The term fact-checking was coined by the journalist community and refers to the process of identifying and verifying factual claims — assertions that can be proved or disproved by factual evidence — originally within media content. While this intricate task used to be manually performed by individuals or journalists in newsrooms, scientists and private organizations such as *FactCheck.org*, *PolitiFact*, and *ClaimBuster* started to build automated (live) systems that process textual and/or video content especially concerning politicians and public figures to (semi-)automatically assess the veracity of their statements. Automated fact checking is performed by a pipeline that includes different tasks depending on the specific modeling and implementation of the problem. In general, systems contain components for claim detection, check-worthiness and prioritization as well as evidence retrieval and veracity prediction.

This project is dedicated to the automated detection of check-worthy claims — i.e. claims that are both factual and regarded relevant for public verification — and discriminates them from non-factual claims and unimportant factual claims (factual but not relevant for public verification) that both would not be further processed in the fact-checking pipeline. It partially replicates the paper from the *ClaimBuster* group (cf. section 2.) and accordingly models the problem as a three-class supervised classification task using their dataset that contains spoken sentences from U.S. presidential election candidates.

The methodology involves exploring various classifiers with different feature combinations, comparing them to a baseline model. To gain deeper insights into the model performance and potential areas for improvement, an error analysis is conducted. This involves a detailed examination and comparison of two selected confusion matrices, offering a nuanced understanding of the specific challenges faced by the classifiers. By pinpointing where misclassifications occur, this analysis provides valuable information for refining the models and enhancing their accuracy.

The project concludes with an examination of ethical implications related to automated fact-checking, thereby aiming to foster a broader understanding of the societal impact and responsibility associated with deploying such technologies.

2 Hassan et al.: "Detecting Check-Worthy Factual Claims in Presidential Debates"

The original paper that this project is based upon was redacted by a subgroup of the *ClaimBuster* team (Hassan et al., 2015) using their self-established dataset (Arslan et al., 2020). The authors approach the detection of check-worthy factual claims as a three-class supervised classification problem at the sentence level, encompassing non-factual sentences (NFS), unimportant factual sentences (UFS) and check-worthy factual sentences (CFS) as classes. NFS refer to sentences devoid of factual claims, i.e. subjective sentences and many question constructions pertain to this class. UFS, while containing factual claims, are deemed irrelevant to the general public's interest in their truthfulness. Lastly, CFS as the focus of the classification task both contain factual claims and their truthfulness is of vital importance to the general public. Example sentences for the classes are listed in section 2 in Hassan et al.

The authors utilize the still-in-progress 2015 version of their dataset, with only 12 out of the 30 U.S. presidential debates held between 1960 and 2012 already labeled, specifically from 2004, 2008, and 2012. The data was labeled according to the aforementioned classes by journalists, professors and university students on a data collection website. After filtering out sentences spoken solely by presidential candidates, discarding sentences shorter than 5 words and answers by low-quality participants, they obtain a dataset size of 1571 sentences (882 NFS, 252 UFS, 437 CFS).

Extracting sentiment scores, sentence length in words, tf-idf word representations, count-vectorized POS tags, and entity type (e.g. "Bush" as "Person") as features, they perform 4-fold cross validation using Multinomial Naive Bayes Classifier, Support Vector Classifier, and Random Forest Classifier. Their results show that their best model achieves a precision and recall score of 0.85 and 0.65 in classyfing CFS, respectively. NFS and CFS show overall better accuracy scores compared to UFS by all models which is due to UFS' ambiguous nature and resulting classification intricacy. The weighted average scores considering all classes amount to a maximum of 0.71 for precision, 0.72 for recall, and 0.7 for F1.

3 Data and Feature Extraction

3.1 Data

Given that the 2015 version of the *ClaimBuster* dataset is no longer accessible and that the dataset size that resulted from the incomplete labeling in this version (1571 sentences) seems impractically small to approach the problem, I opted to use the latest 2020 version of the dataset that consists of 22501 labeled sentences. The class distribution is unbalanced towards NFS with the following distribution: 14685 NFS, 5413 CFS, and 2403 UFS instances. The ReadMe file from their dataset that lies in this project's data folder is inaccurate, as the data was sourced from *crowdsourced.csv*, described as containing only 1032 sentences. Attention should also be paid to the bias in the dataset : The authors exclusively enlisted individuals with academic backgrounds for data collection, potentially introducing a WEIRD bias. This bias persists through the exclusion of labelings from low-quality participants in their paper and contradicts the objective of capturing the interest of the general public in the rather subjective UFS class. What may appear insignificant to one person could hold significance for another. Therefore, labelings from a broad range of individuals would be necessary to obtain more representative data.

3.2 Feature Extraction

General preprocessing of the textual data was not necessary and only performed if the specific implementation to extract one of the seven features used required it. Feature values are collected into a single feature matrix for the model input, depending on the specified feature combination to be used. The features are described in the following.

3.2.1 Sentiment

Sentiment scores within the range [-1, 1] were computed for every sentence using the *TextBlob* library. Individual sentences are converted — without preprocessing — into *TextBlob* objects. They provide a method to retrieve the sentiment score. Sentiment contributes to the modeling of the problem insofar as neutral scores might either represent objective sentences that are CFS or UFS, whereas statements with highly polarized sentiments might be indicative of opinions, beliefs, or subjective content, i.e. NFS.

3.2.2 Subjectivity/Objectivity

The *TextBlob* object also allows the retrieval of a subjectivity score from a sentence ranging between [0, 1], where 0.0 is very objective and 1.0 very subjective. Sentences exhibiting higher scores might be indicative of NFS as they probably contain opinions, beliefs, or else, whereas lower ones of either CFS or UFS due to their more objective and neutral tone.

3.2.3 Binary Question Feature

Many questions pertain — reflected in their structure and performed speech act — to the NFS class. However, if they not only perform an interrogative speech act but also introduce information or facts (e.g. constructions like these: Given that X, ... ?, where X represents a claim), they can also appear as NFS or CFS. With the help of the other features, this feature tries to contribute to the discrimination between NFS questions and UFS/CFS questions by assigning the value 1 if a sentence contains a question mark, otherwise 0.

3.2.4 Mood

Mood refers to the grammatical category that expresses the speaker's attitude toward the action or state described by the verb. The four main moods are indicative, imperative, conditional, and subjunctive. They are expected to provide grammatical information to discriminate the classes. In order to get numerical feature values for the four types, sentences are parsed and lemmatized using the *Pattern* library. The output is then classified as one of the four moods using *Pattern* and a weighted mapping that I established myself is applied:

1. indicative: 2 (likeliest to express a factual claim; makes statements or asks questions about reality)
2. imperative: 1 (less likely to express a factual claim; indicates a desire or command for the listener)
3. conditional: 0 (statement is dependent on some condition, thus unfactual)
4. subjunctive: 0 (rather expresses opinions, feelings, or beliefs)

3.2.5 Modality

The *Pattern* library also allows for the extraction of modality, i.e. the semantic expression of possibility and necessity. Sentences are preprocessed in the same way as in 3.2.4. The modality value computed for a sentence indicates the degree of certainty and ranges from -1 to 1, where values greater than 0.5 are expected to represent facts (cf. De Smedt and Daelemans, 2012). By incorporating modality as a feature, the model can potentially distinguish between sentences that express a high degree of certainty (likely factual) and those with lower certainty (less likely factual).

3.2.6 Part-of-Speech Tags

Part-of-Speech tags are expected to provide grammatical information and syntactic patterns that are indicative of a certain class. For instance, a sentence containing cardinal numbers (such as 'I brought back 700.000 jobs') are very likely to be CFS, given that they make a verifiable claim about the quantity/quality of something. POS tags for a sentence are retrieved by means of the *NLTK* library using the Perceptron tagger and Penn Treebank tag set, involving tokenization as a preprocessing step. The POS tags are then converted into a count-vectorized matrix, where each row corresponds to a sentence, and each column represents the count of a specific POS tag in that sentence.

3.2.7 Sentence Embeddings

Sentence embeddings capture the semantic meaning and context of sentences in a continuous vector space, thus providing valuable information to distinguish between the classes. Embeddings are retrieved using the 'all-MiniLM-L12-v2' model from the *sentence transformers* library that converts a sentence into a dense vector representation of 384 dimensions per sentence.

4 Modeling

Four different feature combinations and six different classifiers (five + one baseline) are used to train and test the models, resulting in a total number of 21 models. For each of the models, stratified 5-fold cross validation is performed in order to examine the models' generalization capabilities across disjoint training and test splits and to mitigate the class imbalance. Each fold is evaluated using the *evaluation report* from the *scikit-learn* library. The resulting data is then used to compute the average performance across all folds.

While selecting the set of classifiers, it occurred that the feature values are continuous, discrete, and contain negative values. Therefore, some of the available classifiers (Multinomial Naive Bayes and Complement Naive Bayes, for instance) could not be considered due to incompatibility. Whereas the

negative values from the features sentiment and modality could in theory be rescaled to the range [0, 1], this is not applicable to sentence embeddings that also exhibit negative values.

4.1 Classifiers

The following supervised learning methods from the *scikit-learn* library are used:

1. DummyClassifier with the 'stratified' strategy, providing a baseline for comparison
2. K-Nearest Neighbors classifier with distance-weighted neighbors
3. Gradient Boosting Classifier with 300 estimators
4. Random Forest Classifier with 300 estimators and balanced class weights
5. Support Vector Machine classifier with balanced class weights
6. Multi-layer Perceptron classifier with 1000 hidden layers, SGD solver, adaptive learning rate, 500 maximum iterations, and early stopping

4.2 Feature Combinations

The four feature combinations for feature extraction are as follows:

1. All (Question, POS, Mood, Modality, Subjectivity, Sentiment, Embedding): Expected to provide the most holistic capturing of the sentences.
2. Only embeddings: Evaluates the effectiveness of embeddings in isolation. Expects the model to rely solely on distributed representations of sentences, ignoring syntactic and semantic features.
3. All but embeddings (Question, POS, Mood, Modality, Subjectivity, Sentiment): Expected to show the impact of embeddings.
4. All but embeddings and POS (Question, Mood, Modality, Subjectivity, Sentiment): Focuses on the impact of mood and modality along with sentiment without semantic information by the embeddings and syntactical/structural information provided by POS tags.

5 Results

In terms of accuracy, weighted precision, recall, and F1, the vast majority of models outperforms the baseline model which only performs at chance with values around 0.5, respectively (cf. experiment results table). The classifiers generally performed best using either all features or embeddings only (applies to SVM and K-Nearest Neighbor), with accuracy, weighted precision, recall, and F1 scores amounting to a maximum of 0.79 (Grad Boost all), 0.79 (SVM embedding only), 0.80 (Grad Boost all), and 0.78 (Grad Boost all and SVM embedding only), respectively. These values also slightly outperform the results from Hassan et al. who achieved weighted precision, recall, and F1 scores of 0.71, 0.72 and 0.70, respectively, without mentioning the accuracy scores. However, given that they used a relatively small dataset compared to the one used for this project — which is skewed towards and probably favoring the NFS class — this comparison should be exercised with caution.

Regarding CFS-specific performance, the models demonstrate precision, recall, and F1 scores of 0.86, 0.75, and 0.70, respectively. In contrast, Hassan et al. reported slightly lower values of 0.85, 0.65, and 0.67 in their results.

Particularly the MLP classifier seems to be rather sensitive to differing feature combinations, given its considerable decrease in classification performance when only embeddings or all features besides POS and embeddings are used, accounting for a feature-dependent necessity of finetuning its hyperparameters. Otherwise, using only embeddings either increases or slightly decreases the model performance, indicating that the semantic information encoded in the sentences' vector representations alone is, to a certain extent, able to discriminate the classes.

However, leaving out only the embeddings led to a performance improvement (F1 score) in the Random Forest Classifier and Multi-layer Perceptron classifier, indicating not only that the rest of the features can make up for the semantic information provided by the embeddings, but also that each classifier is more susceptible to a certain feature combination.

Excluding both embeddings and POS tags from the feature set results in the poorest performing models. This indicates that both semantic and syntactic information play a crucial role in discriminating the classes, emphasizing the need for a holistic approach that captures both the underlying meaning and grammatical structure of sentences. The feature combination question, mood, modality, subjectivity, and sentiment alone proves to be less suf-

ficient for a robust classification, highlighting the limitations of relying solely on these aspects. Even embeddings alone outperform this feature combination, underscoring their impact in this classification task.

Classifier	Features	P_NFS	P_UFS	P_CFS	R_NFS	R_UFS	R_CFS	F_NFS	F_UFS	F_CFS	ACC	P_WAVG	R_WAVG	F_WAVG
Baseline		0.65	0.10	0.24	0.66	0.11	0.22	0.66	0.10	0.23	0.50	0.49	0.50	0.50
K_Near	all	0.77	0.32	0.57	0.89	0.18	0.42	0.83	0.23	0.49	0.71	0.67	0.72	0.68
K_Near	embedd	0.79	0.47	0.66	0.91	0.23	0.54	0.84	0.31	0.59	0.75	0.72	0.75	0.73
K_Near	all\embedd	0.76	0.30	0.55	0.89	0.18	0.41	0.82	0.22	0.47	0.70	0.66	0.70	0.67
K_Near	all\{pos, embedd}	0.68	0.14	0.33	0.81	0.03	0.27	0.74	0.05	0.29	0.60	0.54	0.60	0.56
Grad_Boost	**all**	**0.83**	**0.60**	**0.72**	**0.93**	**0.29**	**0.65**	**0.88**	**0.39**	**0.68**	**0.79**	**0.78**	**0.80**	**0.78**
Grad_Boost	embedd	0.81	0.58	0.71	0.93	0.25	0.61	0.86	0.35	0.66	0.78	0.76	0.78	0.76
Grad_Boost	all\embedd	0.79	0.45	0.62	0.92	0.15	0.53	0.85	0.22	0.57	0.74	0.71	0.74	0.71
Grad_Boost	all\{pos, embedd}	0.67	0.11	0.46	0.96	0.00	0.11	0.79	0.00	0.18	0.65	0.56	0.65	0.56
Random_F	all	0.72	0.92	0.85	0.99	0.04	0.33	0.83	0.08	0.48	0.73	0.77	0.73	0.66
Random_F	embedd	0.70	0.90	**0.86**	0.99	0.03	0.27	0.82	0.05	0.41	0.71	0.76	0.71	0.64
Random_F	all\embedd	0.77	0.45	0.63	0.93	0.12	0.49	0.84	0.18	0.55	0.74	0.70	0.73	0.70
Random_F	all\{pos, embedd}	0.73	0.18	0.35	0.64	0.32	0.34	0.64	0.23	0.35	0.53	0.58	0.53	0.55
SVM	all	0.90	0.34	0.63	0.76	0.61	0.67	0.83	0.44	0.65	0.72	0.78	0.72	0.74
SVM	**embedd**	**0.90**	**0.44**	**0.65**	**0.82**	**0.54**	**0.75**	**0.86**	**0.49**	**0.70**	**0.77**	**0.79**	**0.77**	**0.78**
SVM	all\embedd	0.87	0.26	0.56	0.73	0.55	0.54	0.80	0.35	0.55	0.66	0.73	0.66	0.69
SVM	all\{pos, embedd}	0.77	0.16	0.34	0.49	0.47	0.37	0.60	0.23	0.36	0.46	0.60	0.46	0.50
MLP	**all**	**0.82**	**0.60**	**0.68**	**0.92**	**0.19**	**0.66**	**0.87**	**0.29**	**0.67**	**0.78**	**0.76**	**0.78**	**0.76**
MLP	embedd	0.65	0.00	0.00	1.00	0.00	0.00	0.79	0.00	0.00	0.65	0.43	0.65	0.52
MLP	all\embedd	0.77	0.37	0.63	0.93	0.05	0.50	0.84	0.09	0.56	0.73	0.69	0.73	0.69
MLP	all\{pos, embedd}	0.65	0.00	0.00	1.00	0.00	0.00	0.79	0.00	0.00	0.65	0.43	0.65	0.52

Table 1: Averaged experiment results across all folds and feature sets for the DummyClassifier (baseline), K-Neighbors Classifier, Gradient Boosting Classifier, Random Forest Classifier, Support Vector Machine, and Multi-layer Perceptron Classifier. Metrics include class-specific (METRIC_CLASS) values and weighted averages (METRIC_WAVG) for Precision (P), Recall (R), F1 Score (F), and Accuracy (ACC). The test data comprises approximately 1083 instances for CFS, 2937 for NFS, and 480 for UFS.

6 Error Analysis

Although the employed metrics are essential to measuring the performance, correctness, and quality of a system, they are only an approximation of the system's qualities as they are limited to their underlying criteria. The following error analysis aims to counteract the metrics' biases and drawbacks, namely by uncovering error patterns that the metrics hide under the guise of numerical values. The confusion matrices of the baseline and best performing Gradient Boost model will be analyzed and compared, which are attached in the appendix.

Both the baseline and Gradient Boosting models demonstrated their highest performance in accurately classifying NFS, with 1933 and 2757 correct predictions, respectively. Given that NFS represents the largest class, these values are not unexpected because the models might lean towards this class — despite efforts to mitigate bias. Nevertheless, the Gradient Boosting model showcased superior predictive ability by correctly predicting a total of 824 more instances compared to the baseline.

The baseline classified 662 CFS and 342 UFS wrongly as NFS. While misclassifying as NFS is not as severe, considering that NFS would not be further processed in the fact-checking pipeline, misclassifying as CFS is a more critical error, given that the pipeline would proceed with a non-factual or unimportant sentence. The Gradient Boosting model reduced this misclassifications to 146 (CFS instead of NFS) and 34 (UFS instead of NFS).

As for CFS, the baseline classified 251 instances correctly, while 712 are misclassified as NFS and 120 as UFS, i.e. around 77% of the instances were wrongly predicted. Misclassifying CFS instances is unfavorable because errors may lead to misinformation being overlooked or, conversely, non-factual statements undergoing unnecessary fact-checking, compromising the overall efficiency of the fact-checking system. The Gradient Boosting model performed better in that regard with 681 correct predictions, 355 instances wrongly classified as NFS and 47 instances wrongly assigned to the UFS class.

By nature, the UFS class represents the most challenging to classify, considering that it is supposed to capture whether the truthfulness of a sentence is irrelevant to a general public, raising questions as to how this irrelevance can be extracted from textual data only. This also mirrors in the confusion matrices, where only 45 correct predictions (out of 481) were made by the baseline and 146 by the Gradient Boosting model. The majority of the UFS instances were misclassified as NFS (325 for the baseline, 246 for the Gradient Boosting model) which for practical purposes is not severe as the fact-checking pipeline would not process them further. The 111 and 88 instances misclassified as CFS by the baseline and Gradient Boosting model, respectively, will however act as a disruptive factor in the pipeline.

7 Conclusion

The overall good experiment results that outperformed Hassan et al. with accuracy, weighted precision, recall, and F1 scores amounting to 0.79, 0.79, 0.80 and 0.78, respectively, indicate that it is worthwhile to dive deeper into the problem. Improvements might comprise a more balanced and less biased dataset and a more thoroughly conducted and broader feature extraction as well as hyperparameter optimization of the classifiers — especially the sensitive neural network exhibits the necessity of feature-dependent fine-tuning . Particularly both sentence embeddings and POS tags have been proven to be considerably beneficial to the model's performance by capturing semantic richness and providing insight into morpho-syntactic structures.

The main problem that occurred during the project was an initial incompatibility between the libraries that were used. The *pattern* library is only supported until Python 3.6, which is why I decided to use this outdated version for the sake of experimentation because I did not want to forgo its functionalities for feature extraction. However, this lead to a great number of compatibility problems among other libraries, resulting in an Odyssey of re-installing libraries.

Deducing from the findings in the error analysis, the discrimination between three classes might be redundant, given that only the CFS class is further processed in the fact-checking pipeline. Therefore, the problem could also be modeled as a two-class problem consisting of check-worthy and non-check-worthy sentences only. UFS are not necessary to draw this distinction, yet it seems interesting to explore to which extent the notion of UFS can be modeled. The underlying criteria for UFS is yet rather context specific and subjective to every individual, where the boundaries between relevance

and irrelevance blur.

As the classification operates at the sentence level rather than taking broader spans of text into consideration, claim span detection could be implemented as a preliminary step considering that claims may encompass multiple sentences, so that claims could then be classified in their entirety. However, if a claim is composed of several subclaims over several sentences, they could be of differing classes and thus incorporate noisy information into the span to be classifier, e.g. if a claim encompasses a CFS and NFS, it should be classified as a CFS even though it contains an NFS. The problem would thus be rendered more complex and an adapted dataset would be necessary.

8 Ethical Considerations

Fact-checking raises several societal and ethical issues resulting from its underlying nature of truth that the automation can even exacerbate. The pivotal problem lies in the existence or absence of a universally agreed-upon truth and thus affects the concepts of relativism and perspectivism. If misconducted, fact-checking can be used as a propaganda tool, undermining its intended purpose of promoting truth. As a consequence, fact-checking can have the opposite effect leading to the dissemination of misinformation and contribute to the erosion of public trust in fact-checking organizations.

While fact-checking should strive to be impartial and neutral, bias in the data used in automated fact-checking can negatively influence the assessment and should therefore be collected with caution. However, if this bias is mitigated, automated fact-checking might provide an ethical advantage over conventional (human-biased) fact-checking considering that machines do not deliberately favor individuals or political parties and might treat all claims equally.

Moreover, employing Machine Learning or even Deep Learning in the realm of fact-checking introduces a layer of opacity to the process as these methodologies do not inherently offer a comprehensible foundation for human decision-making.. It becomes imperative for developers to transparently elucidate the inner workings of their systems and furnish detailed insights into the datasets underpinning their algorithms.

References

Fatma Arslan, Naeemul Hassan, Chengkai Li, and Mark Tremayne. 2020. A Benchmark Dataset of Check-Worthy Factual Claims. In *14th International AAAI Conference on Web and Social Media*. AAAI.

Tom De Smedt and Walter Daelemans. 2012. Pattern en.

Naeemul Hassan, Chengkai Li, and Mark Tremayne. 2015. Detecting Check-Worthy Factual Claims in Presidential Debates. *Proceedings of the 24th ACM International Conference on Information and Knowledge Management (CIKM)*, pages 1835–1838.

A Appendix

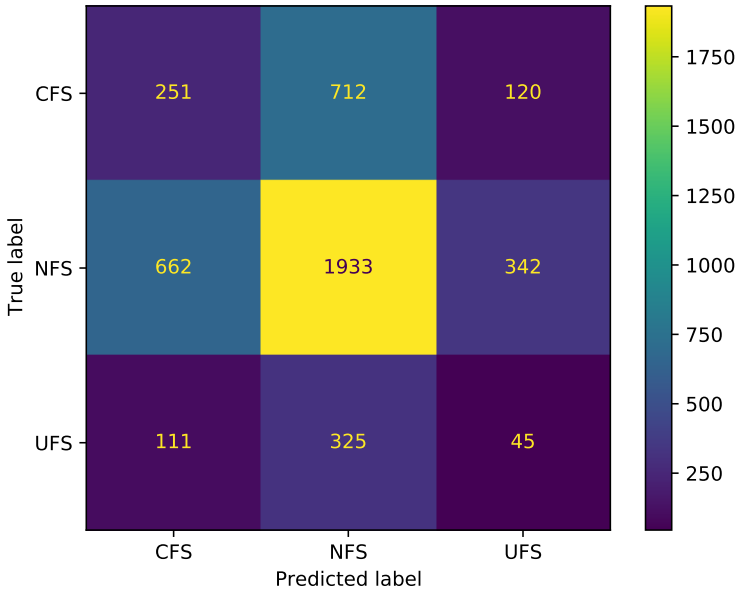

Figure 1: Confusion matrix for the baseline model

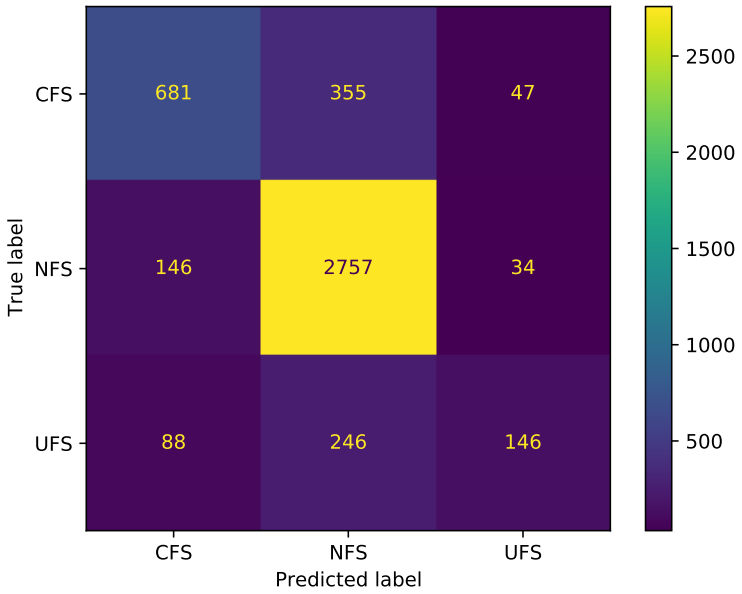

Figure 2: Confusion matrix for the Gradient Boosting model using all features, fold 3