# CheckThat! 2024

## Task 2

Detecting subjectivity in sentences from news articles

# Introduction

Problem definition:

- binary classification
- given dataset
- language english
- Goal: create system that reliably classifies sentences from news articles as either subjective or objective

# Data

There are four datasets given:

- train_en.tsv (Columns: sentence_id, sentence, label, solved_conflict)
  830 samples, 36% SUBJ 64% OBJ, solved_conflict == True for 69 samples
- dev_en.tsv (Columns: sentence_id, sentence, label, solved_conflict)
  219 samples, 52% SUBJ 48% OBJ, solved_conflict == True for 20 samples
- dev_test_en.tsv (Columns: sentence_id, sentence, label)
  243 samples, 52% SUBJ 48% OBJ
- test_en.tsv (Columns: sentence_id, sentence)
  484 samples

# Data

- train_en.tsv is used as a training dataset
- dev_en.tsv is used as a testing dataset
- dev_test_en.tsv is used as a validation dataset
- test_en.tsv is used for the final evaluation and the leaderboard

# Preprocessing

The following steps are applied to preprocess the data:

- convert to lowercase
- remove links
- remove usernames
- remove special charachters except for ! and ?
- remove stopwords that don’t contain any info about subjectivity
- replace quoted sentences with <quote>
- replace numbers with <num>
- tokenize
- lemmatize

# Feature engineering

The following features are extracted:

- TF-IDF vectorizer. **Idea:** so the model can learn which words co-occur with which class. -> Might have not been the best choice.
- Word2Vec (pretrained google-news-300) for each word in given sentence. **Idea:** so the model can find similarities in the semantics for the two classes.
- Embeddings from SBERT transformer. **Idea:** extract the semantics of sentences. More complex than Word2Vec and also takes context of words into account.
- A feature selection of all the features out of Word2Vec and SBERT is done using only the k most meaningful features. $Chi^2$ is used to rank these features. **Idea:** filter out noisy/not important features.

# Model Selection

The following models were tried out and evaluated using the f1 macro:

- Logistic Regression Classifier
- SVM
- Random Forest
- Gradient Boost
- A simple Neural Network

=> Logistic Regression Classifier and SVM proofed to be the most promising models.

# Parameter Tuning

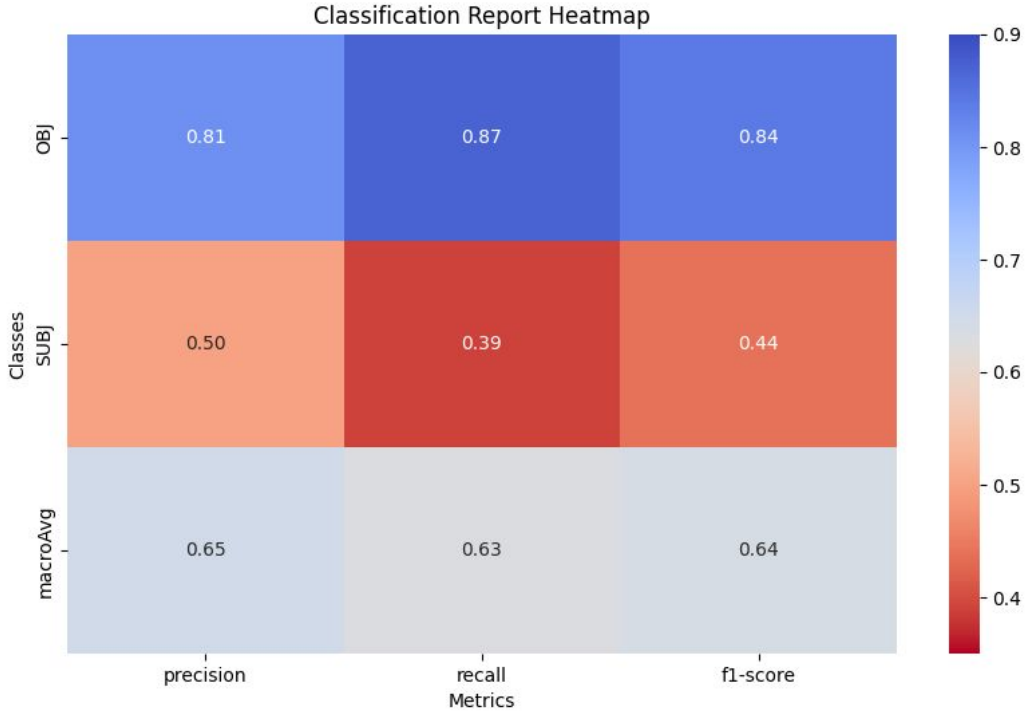
The parameters were tuned using bayesian optimization on the following hyperparameters:

- For Logistic Regression: C, solver and number of selected features
- For SVM: C, kernel, degree, gamma, coef0 and number of selected features

In the end, to avoid overfitting a Logistic Regression Classifier with the standard parameters was used since the best parameter-tuned model was only slightly better (f1 macro of 0.7076 vs 0.6983).

# Results

**Note:** 3 times as many samples are OBJ than SUBJ



# Leaderboard

Place 10 out of 16

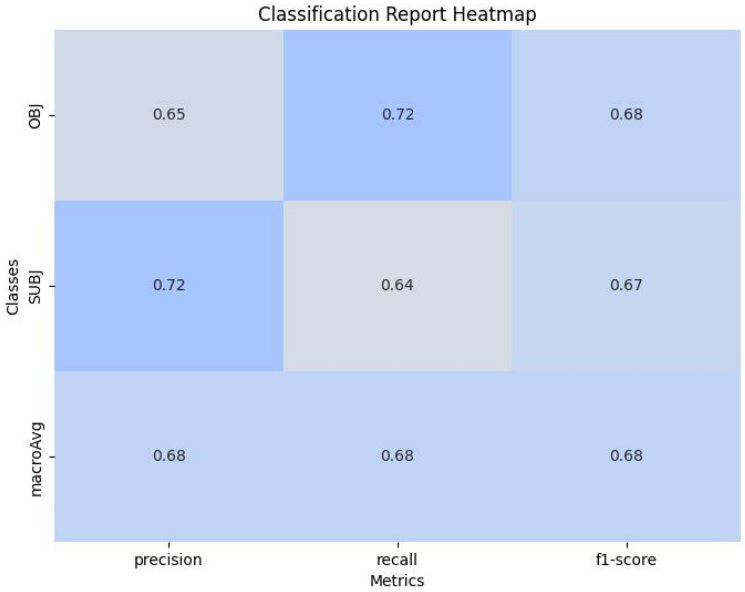
Macro F1: 0.6389
SUBJ F1: 0.44

**English**

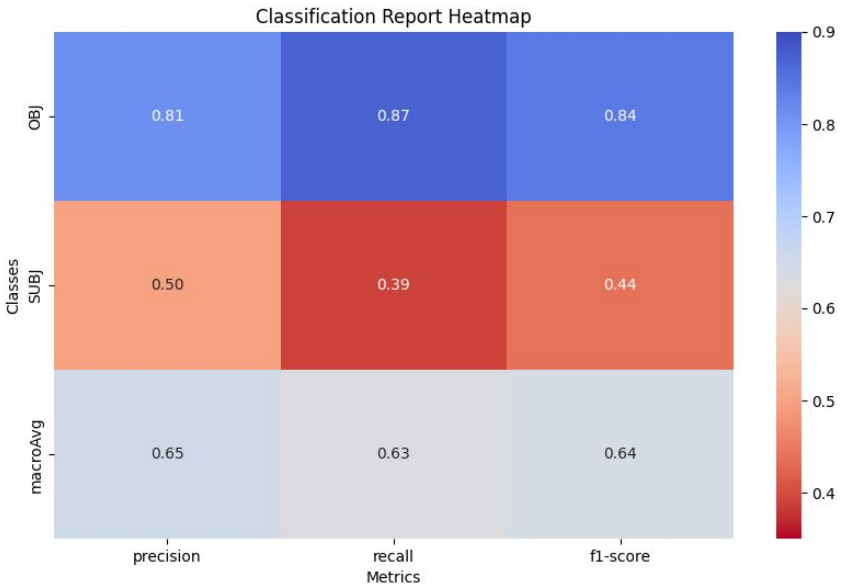
| | Team | Macro F1 | SUBJ F1 |
|---|---|---|---|
| 1 | Hybrinfox | 0.7442 | 0.6 |
| 2 | ToniRodriguez | 0.7372 | 0.58 |
| 3 | SSN-NLP | 0.7120 | 0.54 |
| 4 | Checker Hacker | 0.7081 | 0.54 |
| 5 | JK_PCIC_UNAM | 0.7079 | 0.55 |
| 6 | SINAI | 0.7035 | 0.53 |
| 7 | FactFinders | 0.6955 | 0.51 |
| 8 | Vigilantes | 0.6955 | 0.52 |
| 8 | eevvgg | 0.6955 | 0.52 |
| 9 | nullpointer | 0.6893 | 0.54 |
| 10 | Indigo | 0.6388 | 0.47 |
| 11 | (baseline) | 0.6346 | 0.45 |
| 12 | SemanticCuetSync | 0.6265 | 0.43 |
| 13 | JUNLP | 0.5598 | 0.36 |
| 14 | CLaC-2 | 0.4500 | 0.37 |
| 15 | IAI Group | 0.4491 | 0.39 |

# Interpretation of Results

## validation set




## final evaluation set




**Note:** both datasets were only run once

# Interpretation of Results

- the performance on the validation set is only a bit worse than on the training set (f1 macro of 0.68 vs 0.70) -> therefore there is likely no overfitting
- in general bad performance for SUBJ -> not enough SUBJ samples in trainset
- low SUBJ recall and low OBJ precision -> samples should be classified as SUBJ more often
- very unbalanced performance for the final prediction set this is also a contrast to performance on the validation set -> maybe set contains new patterns (espacially for SUBJ class) that were not seen before

# How to fix these issues?

- Not enough SUBJ samples in trainset?
  - join train and test set together and use cross validation
  - over-/undersampling
  - use data augmentation
- System classifies as SUBJ too seldom?
  - find the right threshold of the model with roc curve
- There are still unknown patterns?
  - use data augmentation
  - use different features to detect new patterns in existing data

# Summary

- the system is already good at the classification of sentences but there is still room for improvement
- it can already greatly assist humans at this task
- it struggles with identifying some cases of subjectivity
- but has potential to improve with more data