FINAL PRESENTATION

NATURAL LANGUAGE PROCESSING

CODE: HTTPS://GITHUB.COM/JOSEFWEIBEL/NLP-MEDICAL-PROJECT

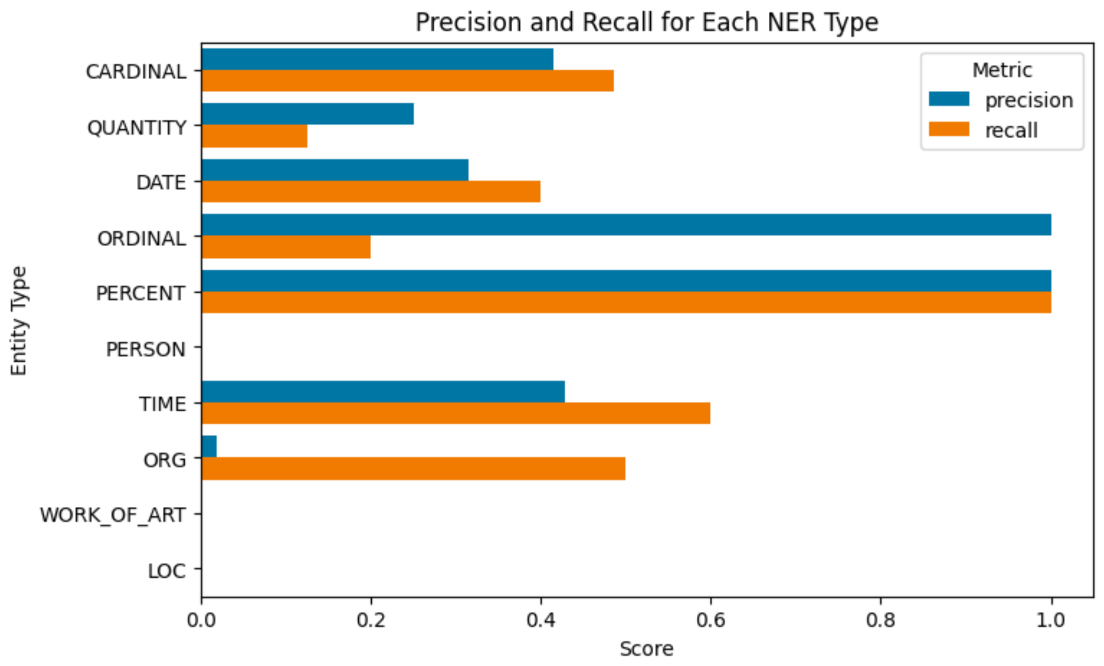
Rebecka Fahrni, Joseph Weibel

AGENDA

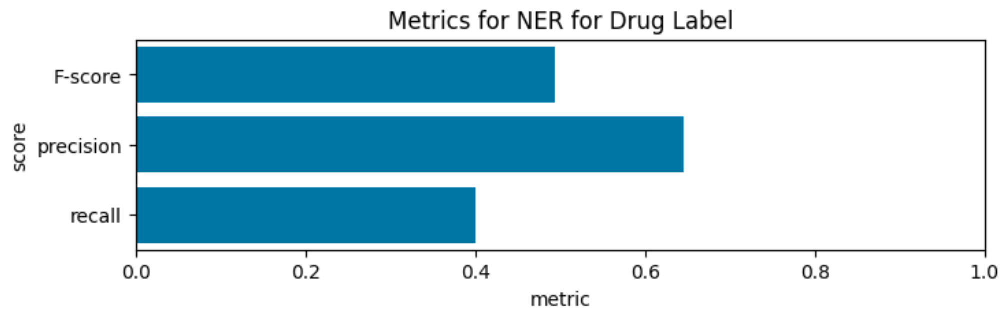
- Short Recap of Task 2
- Recap Goal for Task 3
- Dataset Overview
- Evaluation Metric of Models
- Baseline Models
- Applying BERT-like and GPT-like models

TASK 2

Results for default spaCy NER:



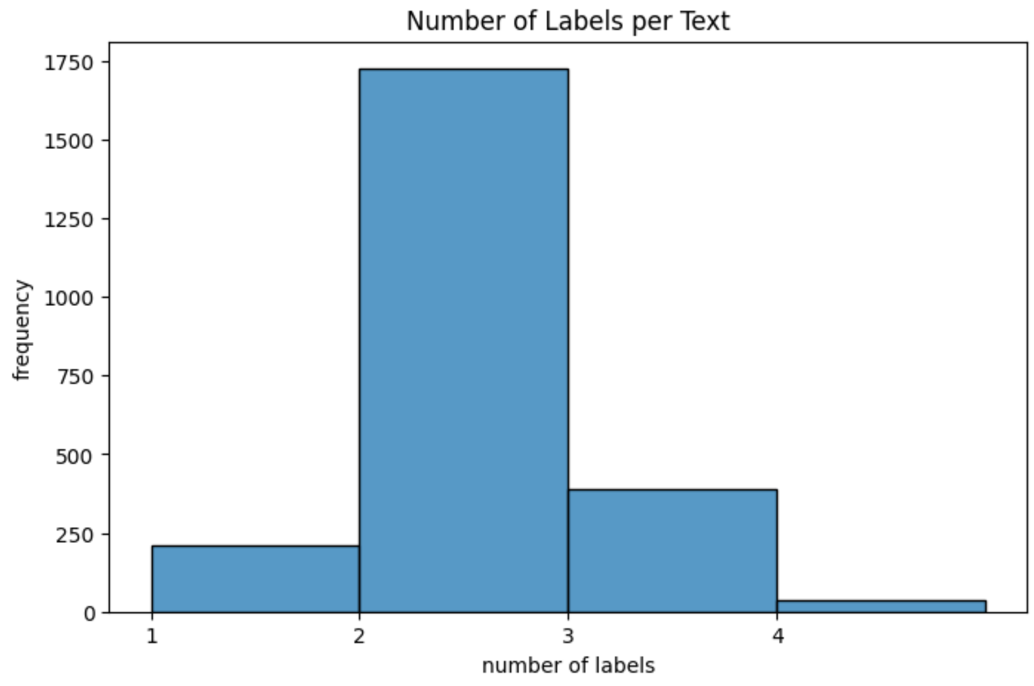
Results for own label «drug»:



RECAP GOAL FOR TASK 3

Classification of medical transcriptions into medical specialties

DATASET OVERVIEW



- Medical transcriptions for various medical specialties
- Highly unbalanced dataset
- Different text styles: very long text and shorter ones
  - Letters vs. Autopsy reports
- Multi-Label - most text have at least 2 labels
- Some classes have only very few samples

EVALUATION STRATEGY

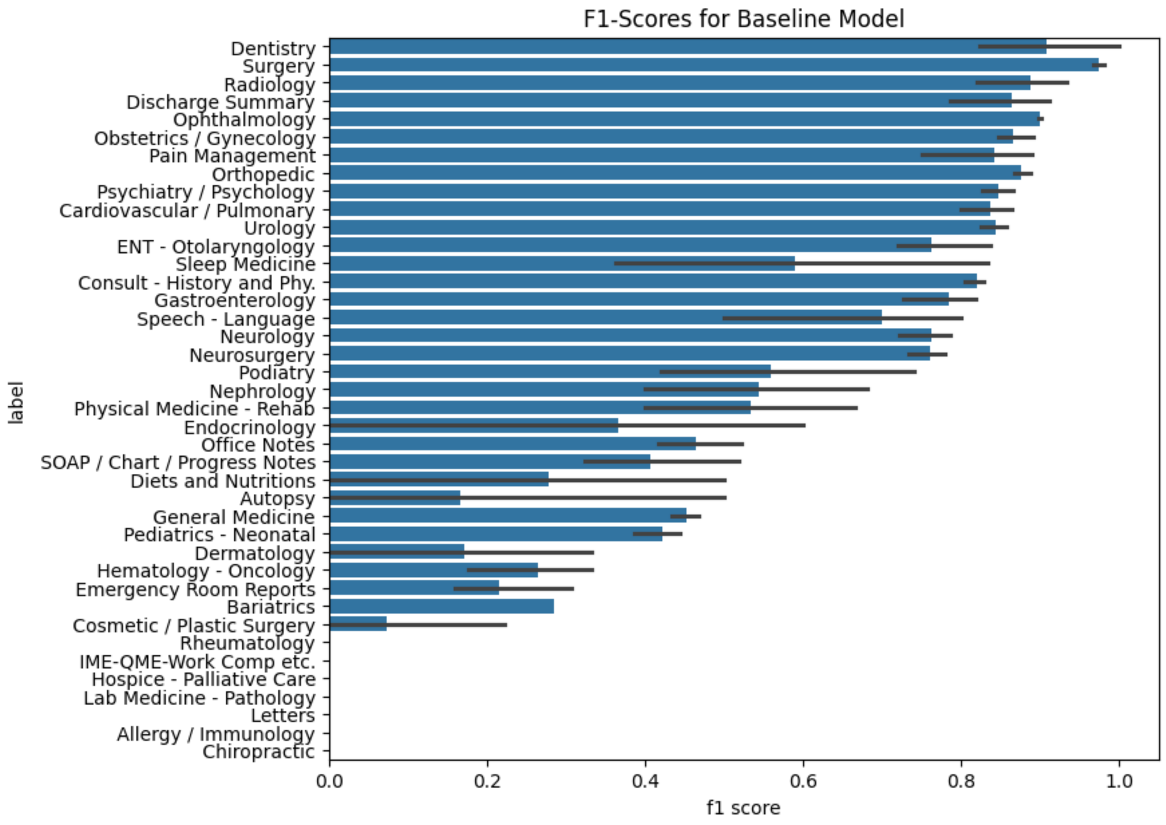
- Evaluation of method: 5-fold cross-validation to ensure robust model assessment using iterative stratification.
- Comparing advanced models to simpler machine learning models to evaluate improvements.
- Accuracy does not provide a complete picture of model performance, especially where class imbalance in the dataset.
- Primary Metric: F1-scores, balancing precision and recall for a holistic performance view
  - Harmonic mean of precision and recall
  - F1-score crucial for datasets with imbalanced classes.
  - High F1 score: well-balanced performance
  - Low F1 score: model trouble striking balance
- Precision and Recall for better understanding of the predictions

BASELINE MODELS

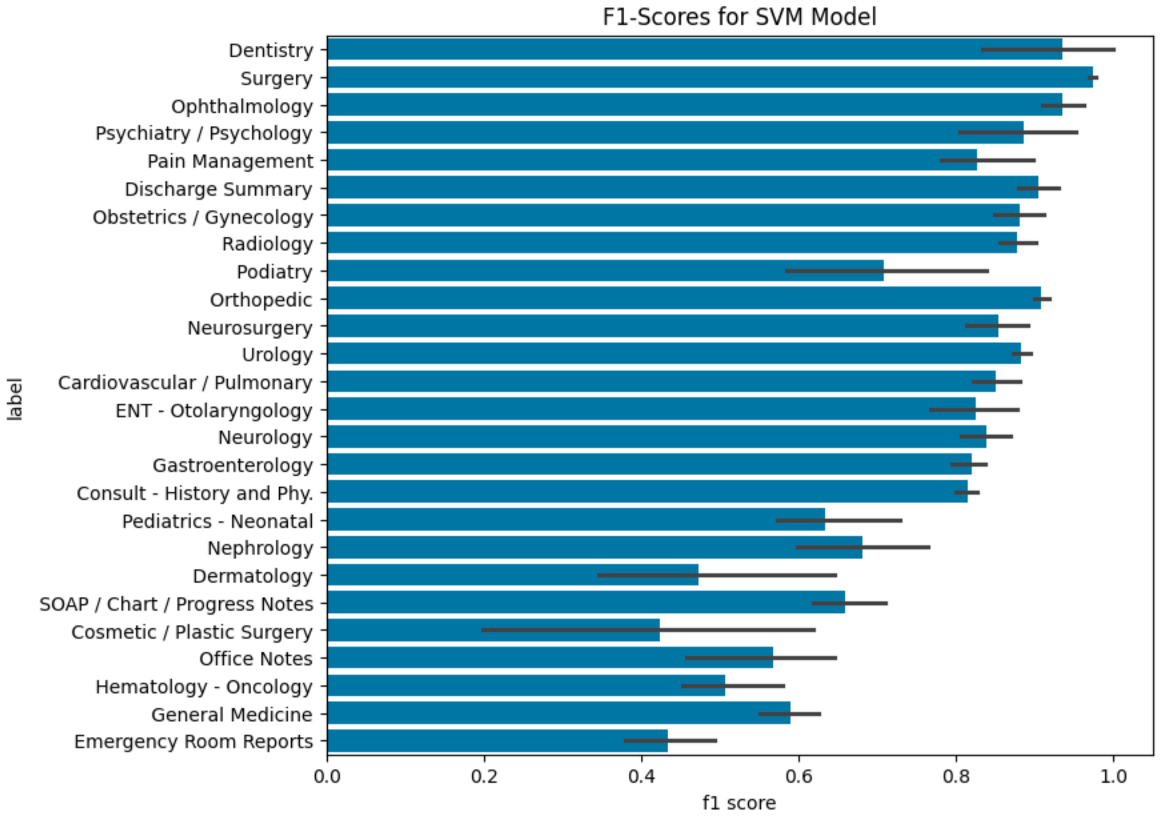
- Pipeline:
  - TF-IDF
  - Pre-processing:
    - standarize text length (512 tokens)
    - Lowercasing
    - Stemming (Snowball) / (Lemmatization – did not apply)
    - Punctuations, numbers removal
    - Label preprocessing: filtering out labels with fewer than 25 texts to focus on more significant labels
    - Balancing dataset (model specific, can't be done for every model in a multilabel setting)
  - Tested: SVM, NB, RF, XGBoost,

F1-SCORES FOR BASELINE MODEL - SVM

SVM – unbalanced class weights

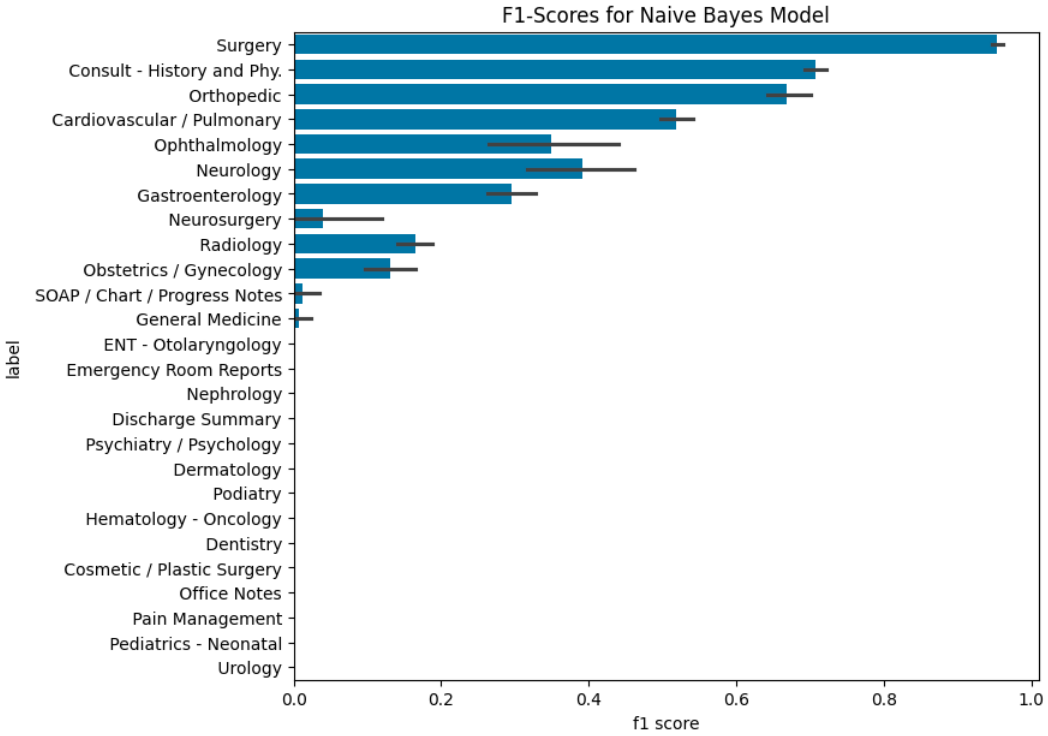


SVM – balanced class weights (only labels with >25 samples)



F1-SCORES FOR BASELINE MODEL - NAIVE BAYES

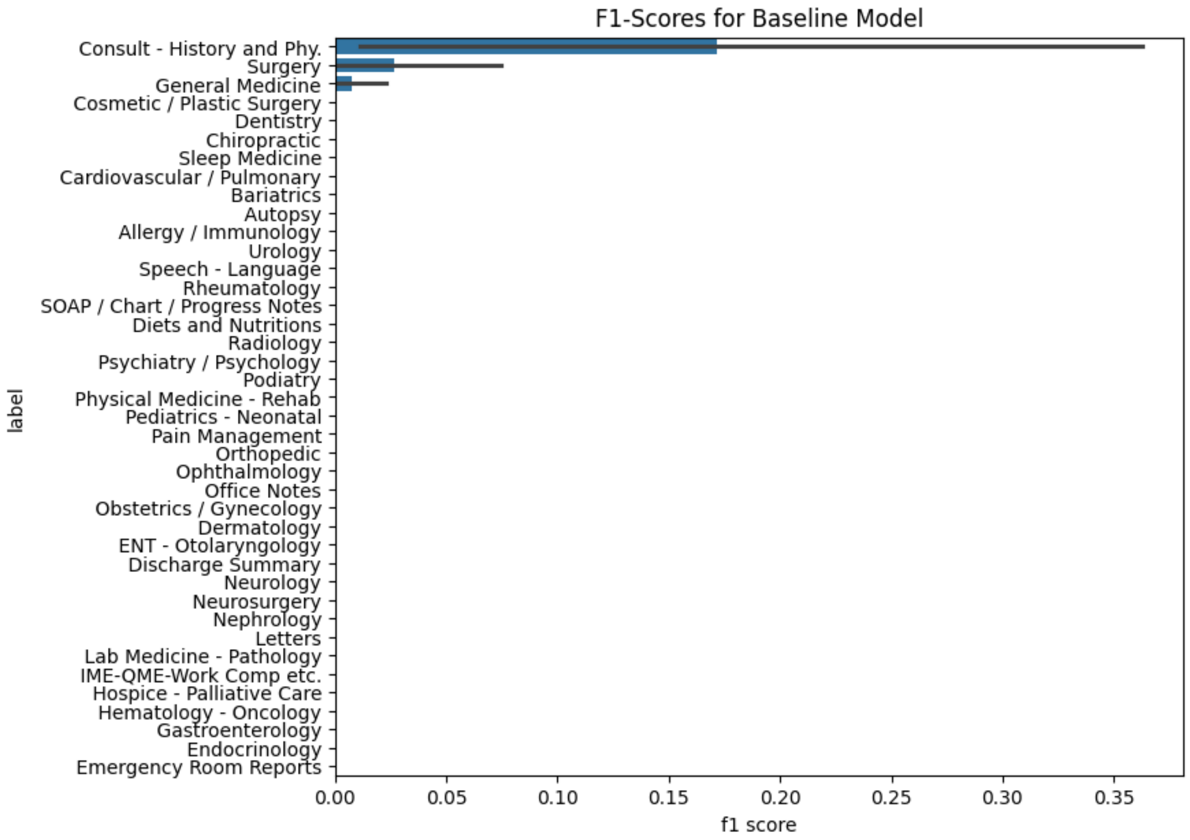
NB - unbalanced class weights (only labels with >25 samples)



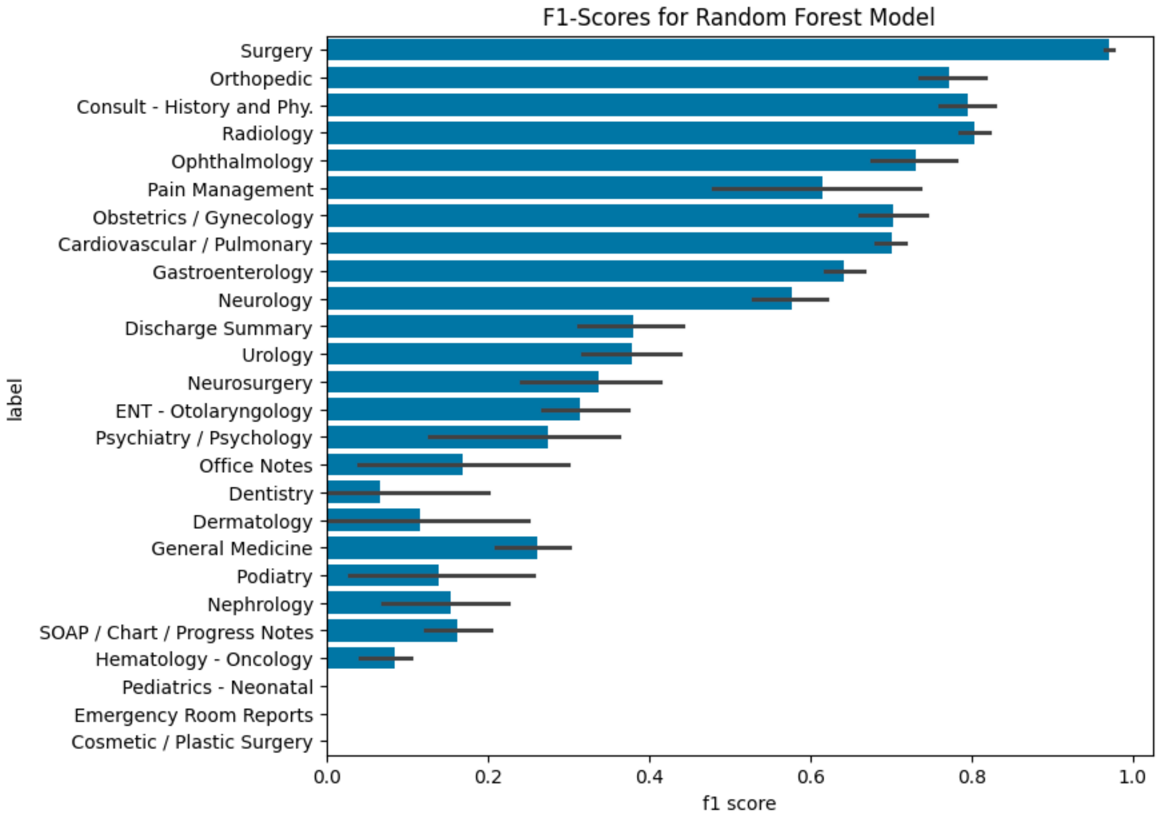
Other models can handle imbalanced data better. Random Forests or Gradient Boosting Machines can be more resilient to class imbalance.

F1-SCORES FOR BASELINE MODEL – RANDOM FOREST

RF- unbalanced class weights

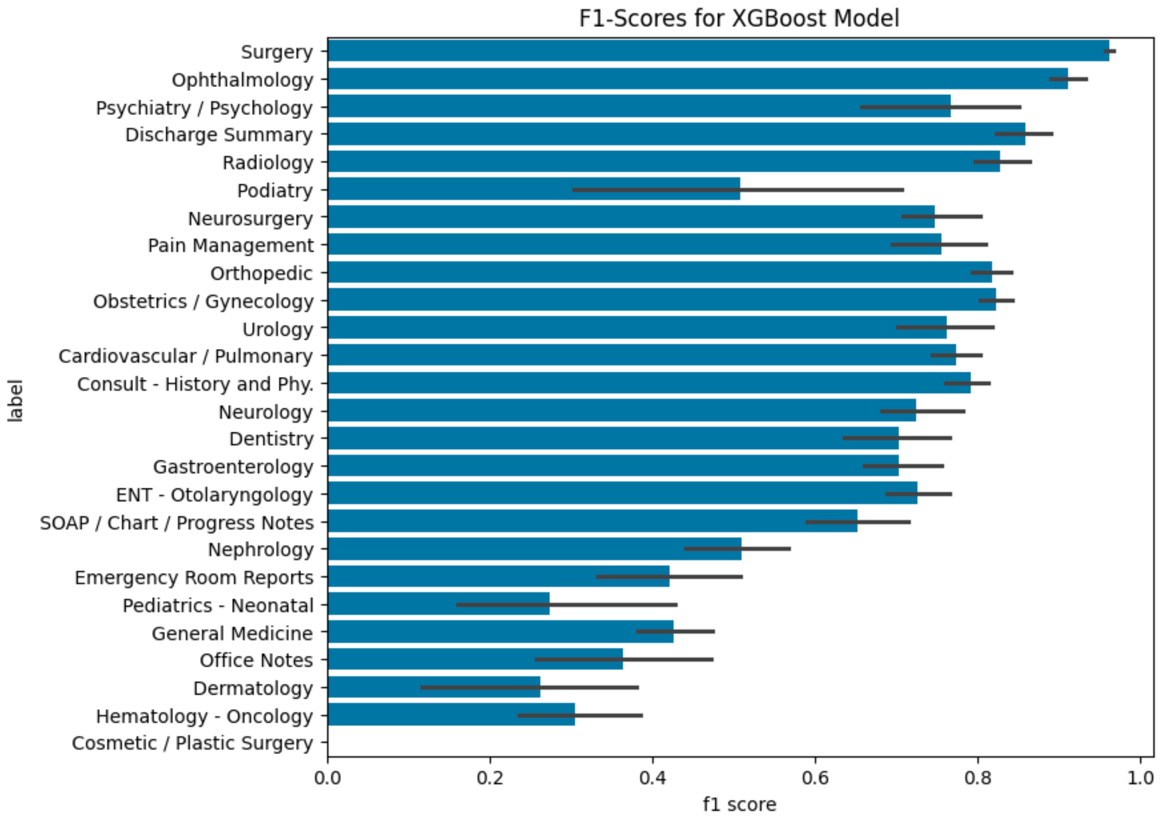


RF - balanced class weights (only labels with >25 samples)



F1-SCORES FOR BASELINE MODEL – XGBOOST

XGBoost - unbalanced class (only labels with >25 samples)



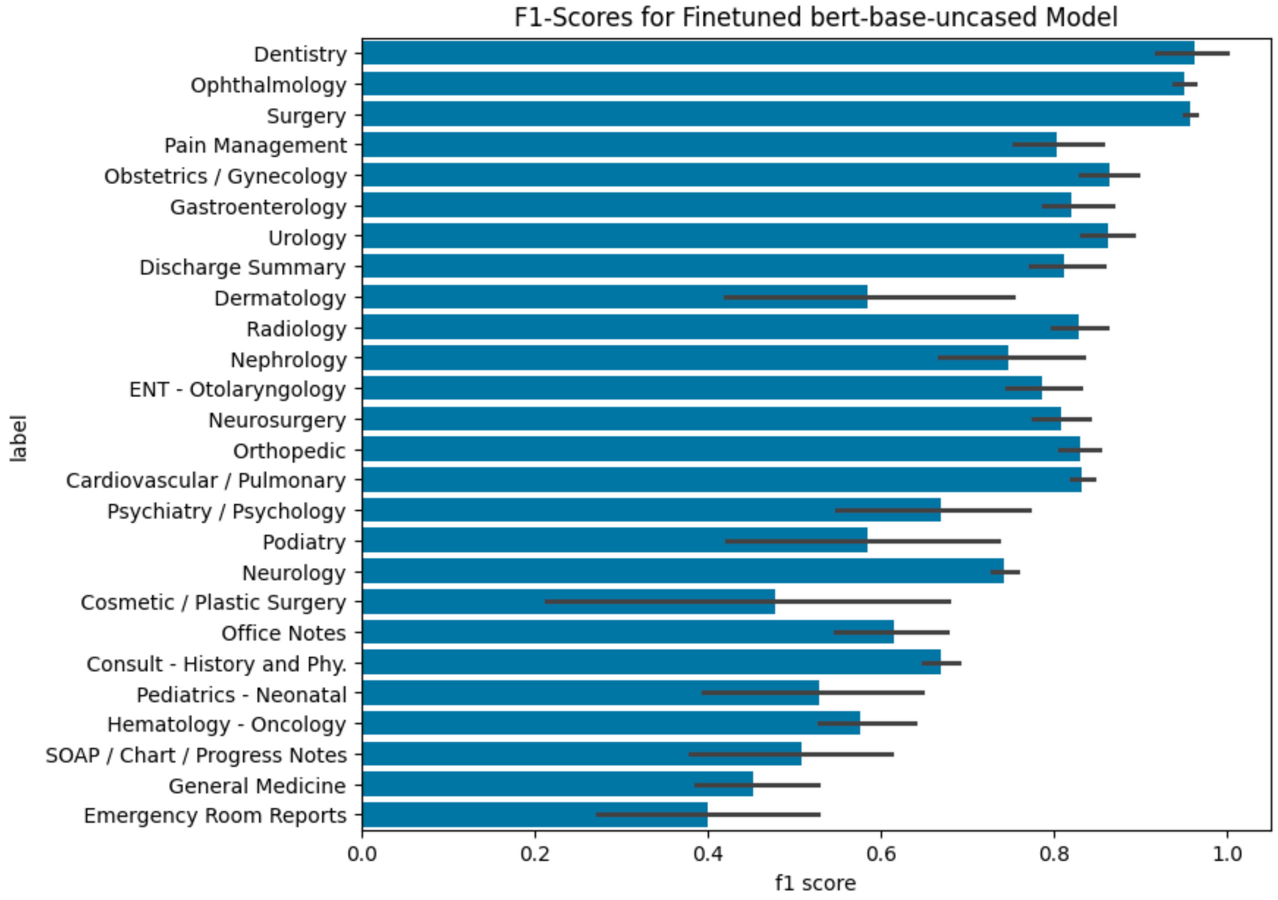
BASELINE MODEL

- Variability in F1 Scores: Variance in F1 scores across different medical categories maybe because data is very diverse or label contain too few samples to generalize
- Impact of Class Balance:
  - Random Forest: balance class weights in Random Forest models has led to a more uniform distribution of F1 scores across categories – positive impact of handling class imbalance for model performance
  - SVM: Balanced class weights improved performance on several minority classes
  - NB and XGBoost: These models do not inherently support class weights and show more disparity in performance across categories. NB struggles more with the class imbalance, gradient boost is more robust.
- Random Forest is the only classifier predicting all classes

BERT

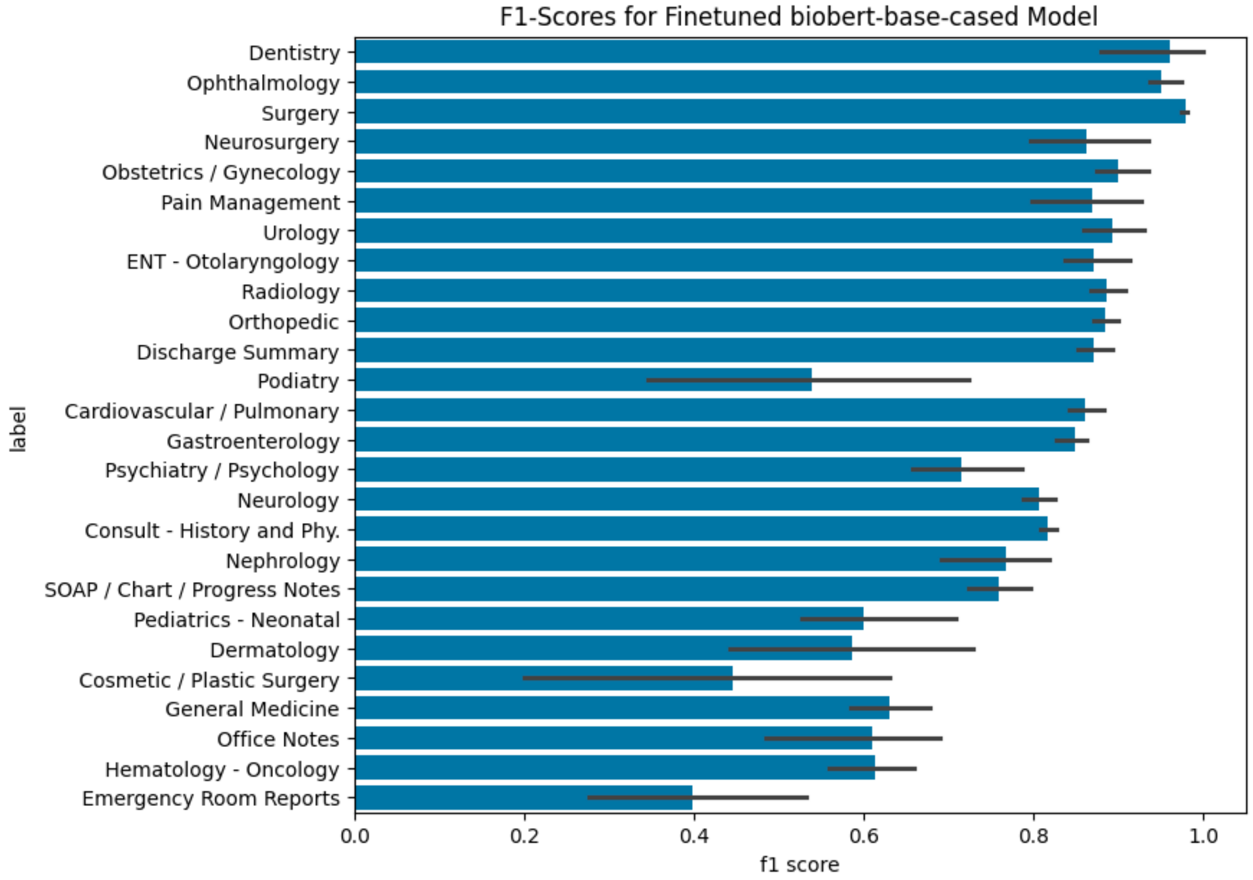
- Using pre-trained language models
- Additional linear layer with sigmoid activation function for classification
- Classification based on embedding for [CLS] token of last hidden state in BERT model
- No preprocessing for BERT
- Reducing token length to 256 to reduce compuational complexity
- Weighted cross-entropy loss used
- BERT Variants:
  - bert-base-uncased
  - dmis-lab/biobert-base-cased-v1.2
  - medicalai/ClinicalBERT (DistilBERT)
- epochs=10, weight_decay=0.1, lr=1e-4, batch_size=32

ADVANCED MODELS: BERT-BASE-UNCASED



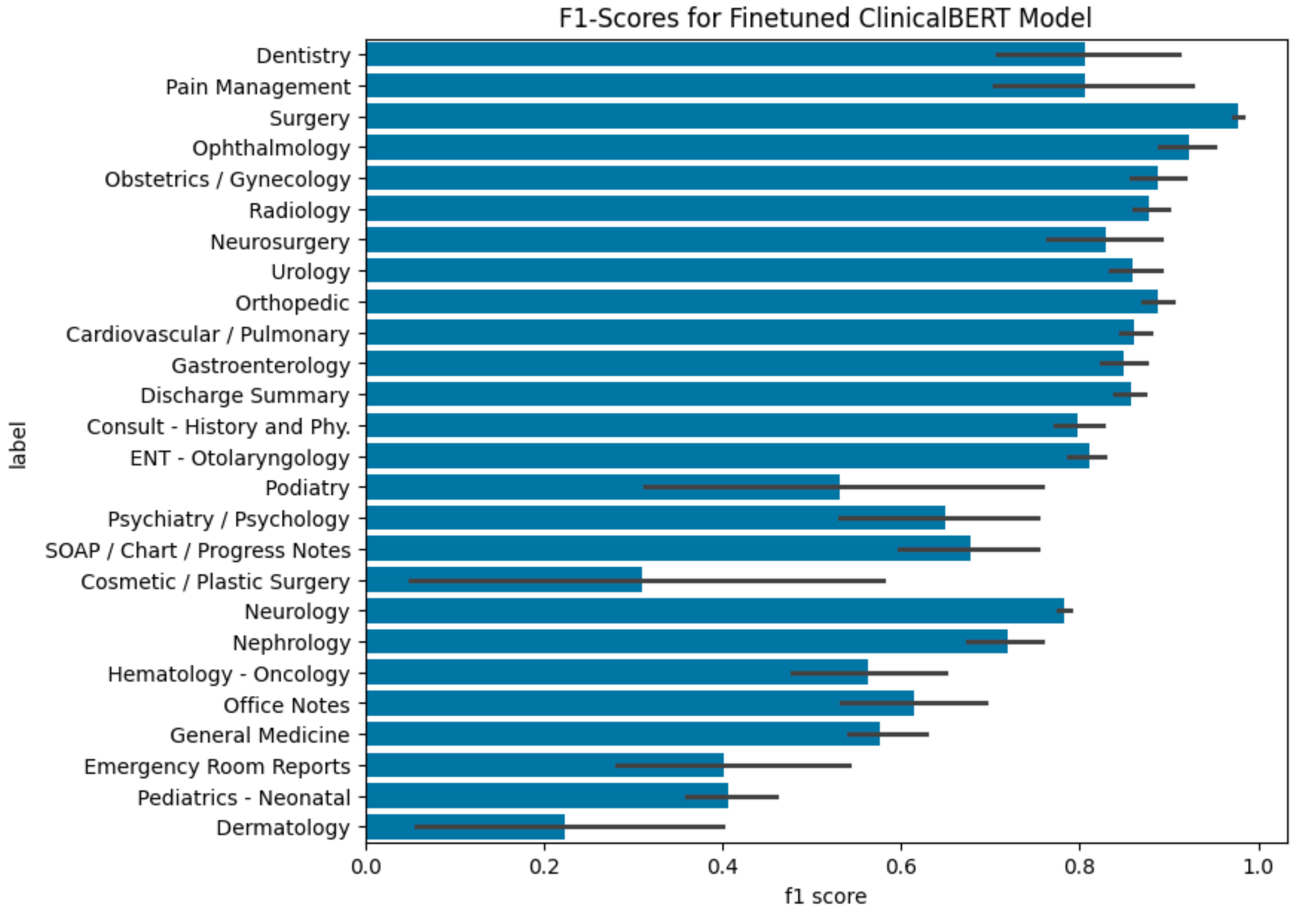
- all classes are predicted
- classes with many samples perform better and have lower variance across folds

ADVANCED MODELS: BIOBERT-BASE-CASED



- similar but slightly better results

ADVANCED MODELS: CLINICALBERT



- also similar, but slightly worse results

BERT MODELS

- Scores for all classes are above 0
- Top-Classes Dentistry, Surgery and Ophthalmology perform well on all three models
  - Surgery is the most common class (1088 samples), however Dentistry has only 27 samples, Ophthalmology has 83 samples
  - Texts for Dentistry and Ophthalmology might consist of some specific and unique words or token combinations for their subject
- Regular bert-base-uncased performs good
- Medical adaption BioBERT is slightly better
  - might be due to refined embeddings for medical vocabulary during pre-training
- ClinicalBERT has the lowest metrics of all BERT models
  - might be due to the lower accuracy of DistilBERT models in general (has fewer parameters than BERT)
- In general: BERT models achieve higher scores for classes with fewer samples
  - might be due to the knowledge existing in the pre-trained models
- Uncertainties vary greatly across classes
- Quality might improve when using 512 instead of 256 tokens

GPT

- Tried different pre-trained Llama 2 variants (without finetuning)
  - meta-llama/Llama-2-7b-chat-hf
  - meta-llama/Llama-2-13b-chat-hf
  - meta-llama/Llama-2-70b-chat-hf
  - epfl-llm/meditron-7b
  - TheBloke/meditron-7B-chat-AWQ
  - mistralai/Mistral-7B-Instruct-v0.2
- Asked models whether text belongs to a single class
  - resulted in 2357*40 prompts
  - single prompt asking for all 40 classes per text perfomed poorly
- Testing different prompts
  - Zero-Shot
  - Few-Shot using different text of prompted classes
- Set low temperature (0.01) for mostly deterministic responses with few hallucinations
- Asking model to answer with yes or no
- Determining binary response by counting occurences of yes and no in text reponse by model
- Difficulties with some models and prompts
- No cross-validation as no training data was needed

ZERO-SHOT PROMPT

System: You are a helpful assistant responding to the user's classification requests. Answer in a single word: yes or no.

User: Do you think the following text can be classified as "{category}"? Answer yes or no.

{text}

Assistant:

FEW-SHOT PROMPT WITH 1 EXAMPLE SAMPLE

System: You are a helpful assistant responding to the user's classification requests. Answer in a single word: yes or no.

User: Do you think the following text can be classified as "{category}"? Answer yes or no.

{sample_text}

Assistant: Yes

User: Do you think the following text can be classified as "{category}"? Answer yes or no.

{text}

Assistant:

ZERO-SHOT PROMPT FOR MEDITRON (NOT INSTRUCTION-FINETUNED VARIANT)

Answer in a single word: yes or no. Do you think the following text can be classified as "{category}"? Answer yes or no.

{text}

Answer:

ZERO-SHOT PROMPT FOR MISTRAL (NO SYSTEM MESSAGES)

User: Do you think the following text can be classified as "{category}"? Answer in a single word: yes or no.

{text}

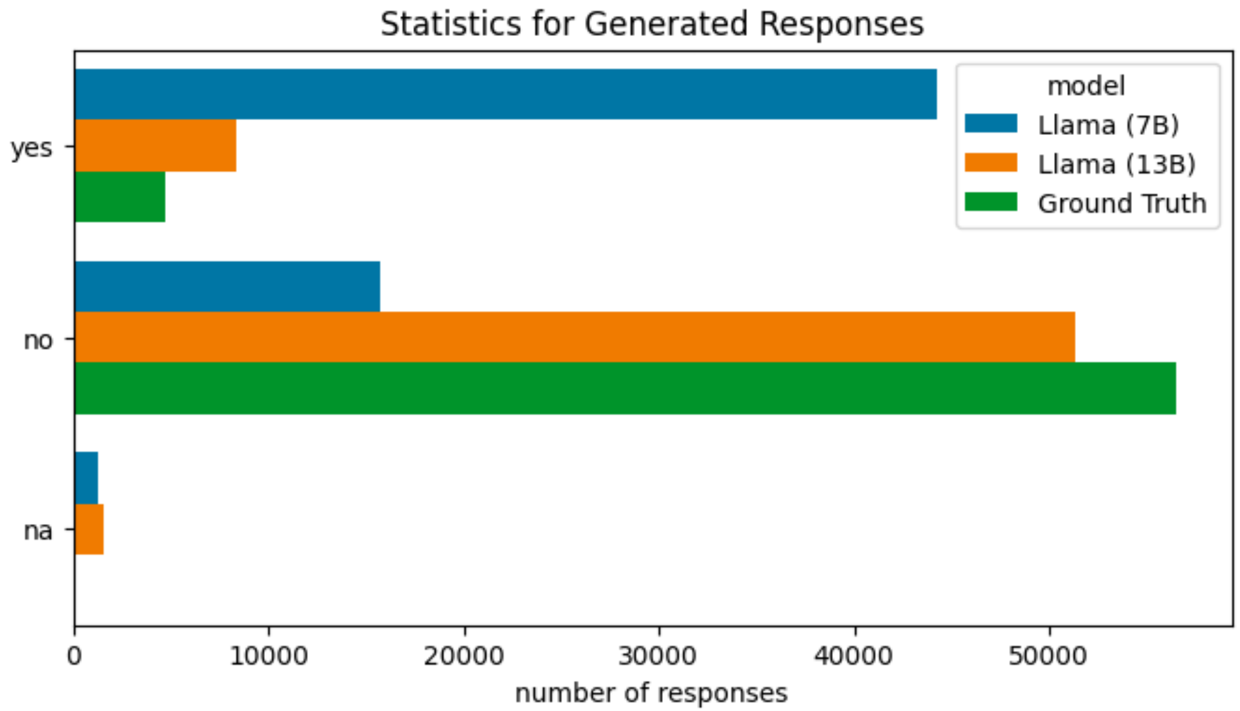
Assistant:

ISSUES

- Llama 2 model with 70B parameters is too large for used infrastructure
  - could be loaded on four GPUs, but inference took too long (several days for all prompts)
- Few-shot prompt has too many tokens and could not be tested
  - (tried with Llama 2 7B and 13B) inference took too long (several days for all prompts)
- Chain-of-Thought prompt could not be tested
  - inference would have taken too long as well
- Mistral model could not be loaded
  - because of issues with transformers package
- Meditron models produced unusable results
  - we tried multiple prompts
  - model generated text endlessly and generated many answers for the questions
  - might first need finetuning on a downstream task

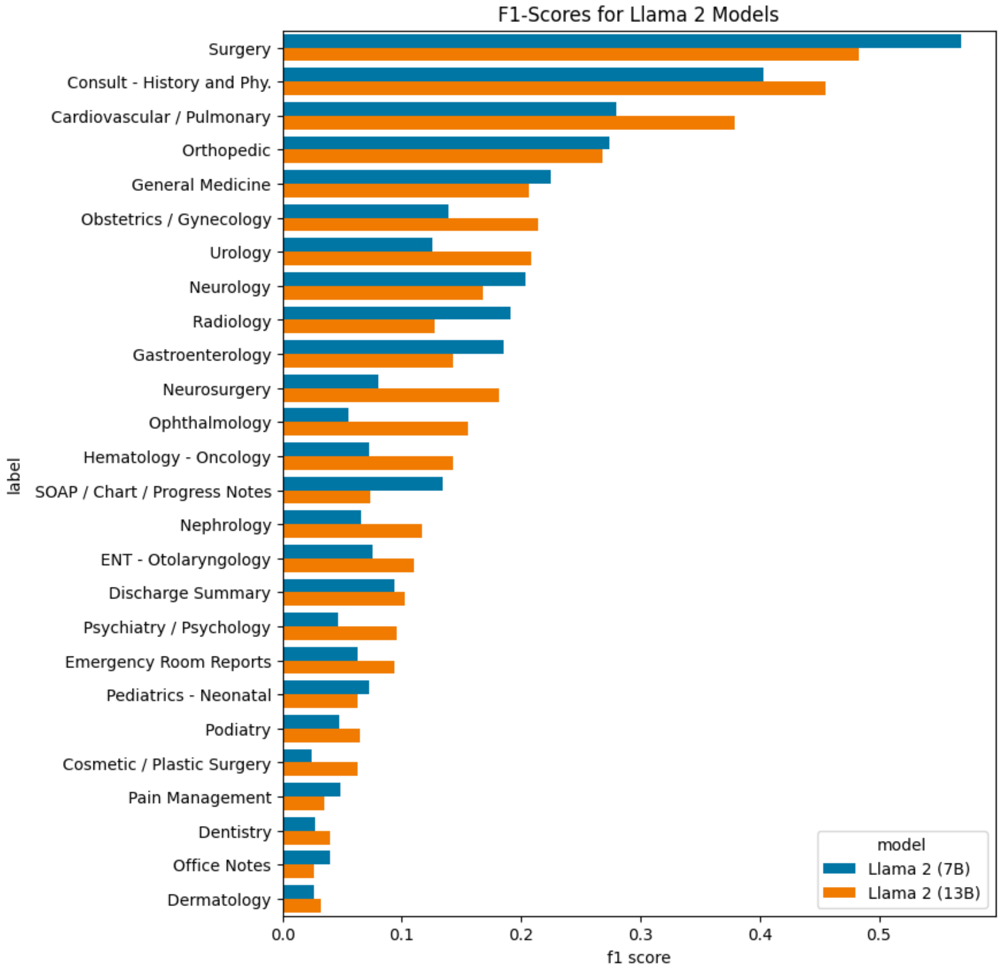
However, zero-shot prompt with Llama 2 7B and 13B did work ☺

STATISTICS FOR GENERATED RESPONSES



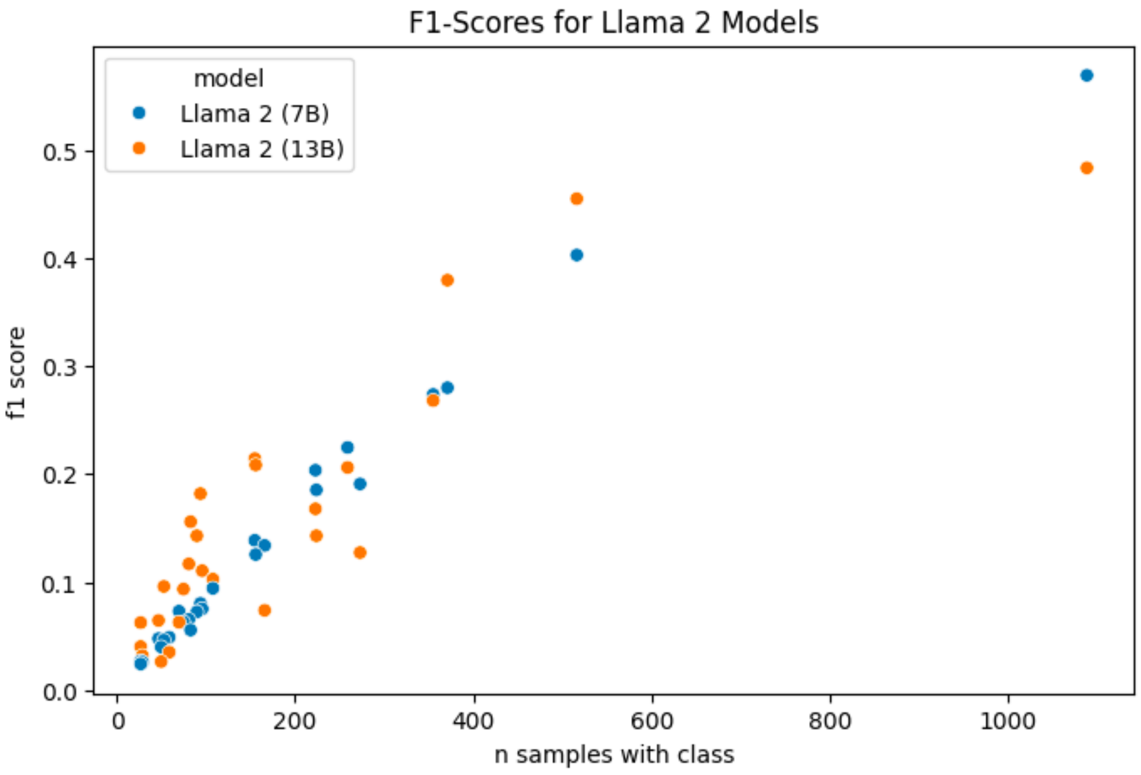
- Most responses (7B: 64%, 13B: 63%) consisted only of either "yes" or "no"
- Counting yes and no in response and deciding whether the model predicts yes or no
- 7B model concludes way more often with yes than 13B model
- Both models are more optimistic to yes answers than yes answers exist in the ground truth
- only a few responses for which no yes or no could be identified (either because the model did not want to decide or it used different words than yes or no)

F1-SCORES FOR LLAMA 2 MODELS



- In general: scores worse than for BERT models (mostly lower than 0.5)
- For most classes model with 13B params performs better than with 7B
- Many classes perform very poorly with both models

F1-SCORES FOR LLAMA 2 MODELS

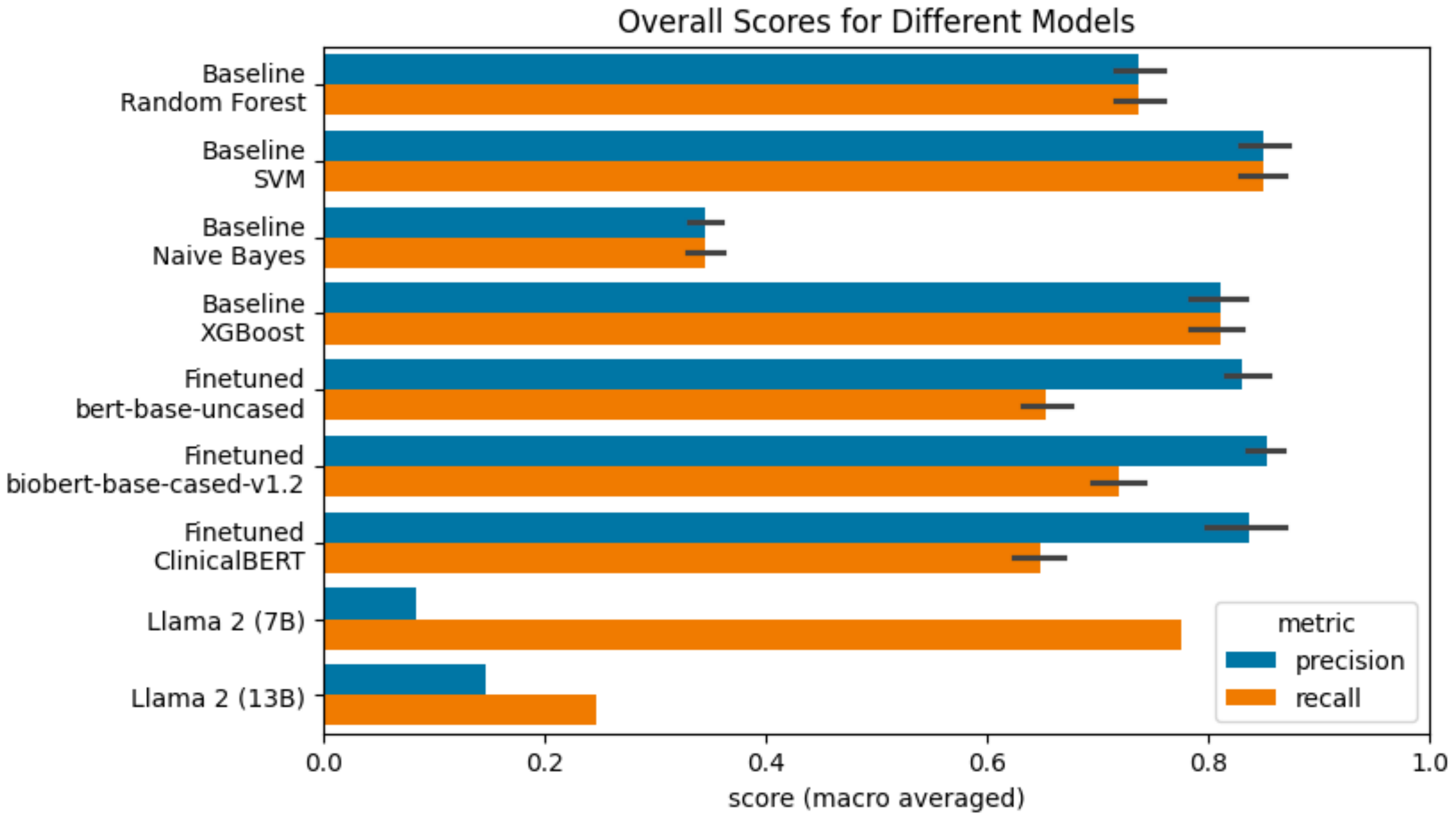


- There seems to be a dependency between number of samples per class and F1-score of both models
- More samples result in higher scores
- This is interesting as each sample was queries separately and no training with that data happened
  - As the dataset is public data, the model might have been pre-trained using that data
  - If Llama 2 is asked whether it knows the argilla/medical-domain dataset, it confirms and lists some characteristics of the dataset
  - In that case, achieved scores can not be transferred to unseen data

CONCLUSION

- Comparison between baseline, BERT and GPT models
- Using precision and recall (decide metric on use case)

OVERALL SCORES FOR ALL MODELS



CONCLUSIONS

- Precision and recall are very similar for each baseline model
  - Seem to have a good balance
- Finetuned BERT models perform equally well when comparing precision
- Llama 2 models perform worse than all other models, including baseline models.
  - Finetuning (regularly, LoRA or QLoRA) might improve the results
  - Few-shot prompt might improve predictions → increases required compuation resources
- Recall of Llama 7B is very high; however, recall is very low
  - Because it mostly predicts yes
- Uncertainty between folds is relatively low for all models
- SVM model offers a good balance between quality and computational costs
- Quality varies greatly between the classes
  - more training data for underrepresented classes needed

THANK YOU

REBECKA FAHRNI,   JOSEPH WEIBEL

FOR DETAILS, SEE CODE IN GITHUB REPOSITORIES