Natural Language Processing - Sentiment Analysis

GitHub

Team: Vikramaditya Singh, James Baker, Taha Boty **Project Mentor TA:** Brian Chen
https://github.com/Tbot101/CIS-5190-Project

1) Abstract

For this project, we study the problem of sentiment analysis regarding fine food reviews on Amazon. This is an important problem because of the integral part that reviews have in dictating future customer purchases and how they interplay with customer behavior. This dataset can uncover important metrics for food producers in order to help them improve their products and target customers better. Our overall implementation regarding this problem was to run the dataset on a logistic regression model, a pretrained BERT model and a feed-forward neural network.

2) Introduction

We are training and testing our data on the summary as well as the longer text of the review. This will be preprocessed - tokenized, lemmatized, cleaned - and then fed into the models. We also changed the reviews that were originally on a scale of 1-5, to instead be a binary 0 or 1 where a review score of 3+ is a positive (1) review and negative (0) otherwise. The output of the model was a prediction on the sentiment.

Implementation: We created a baseline model that would predict a positive sentiment on all inputs. We then extended this by comparing it to logistic regression. Furthermore, we used a pre-trained BERT model as well as a BERT model layered with a logistic regression on top of it.

Evaluation: We tried different hyperparameters for the models such as changing L1 and L2 norms for logistic regression and adjusting the architecture for our feed-forward neural network. Furthermore, we added data set shifts in order to test the robustness of our models.

3) How We Have Addressed Feedback From the Proposal Evaluations

The project mentor's feedback centered around trying time shifts to see the robustness of our models as well as encouraging us to use F1 scores as a metric to see how well our models perform. Furthermore, the mentor asked us to double check if the tokenizer we used for the BERT model was the same as the tokenizer that it was pre-trained on. We incorporated all this feedback by performing a shift in the dataset by inputting more negative reviews in order to see how the model reacted. Furthermore, we used F1 scores as well. Finally, we changed our tokenizer to match the one that was used in the pre-training stage and we noticed that our model improved after this change.

4) Background

As many of the most successful submissions to the Amazon Find Food Review Kaggle dataset were from a range of about 4-8 years ago, our main motivation with this project is to apply one of the latest Natural Language Processing (NLP) models and compare it to models we have learned in class. Our goal with this project is to analyze the limitations of models in a varying range of complexity and modernity, exploring Logistic Regression, the Feed-Forward Neural Network, and finally the state-of-the-line BERT for NLP.

For context, BERT is a Transformer model (learns contextual relations between words in a text) with a large number of parameters. It generates a non-directional language model (learns context of a word based on all surrounding words) using the encoder mechanism of the Transformer. The model was tuned on many diverse data sets to enhance generalization across different types of texts (reviews, tweets, etc.).

After additionally using their tokenizer on our data we believe we can have a strong model that is able to correctly predict the sentiment of reviews.
There have been other notebooks that have conducted analysis with basic models[1], but there were no known submissions using the BERT model. Sentiment analysis on the dataset by Kaggle user DLAO was a great foundation for where we could start our work from, employing various vectorization strategies before applying basic models.

After Milestone 2, we implemented some baseline models and preprocessed the dataset. We then worked on improving the models we employed (Logistic Regression, FNN and BERT) by fine-tuning parameters[2] and changing around the architecture.

Among the above, the most relevant work to our project is:

I. Kaggle submission: Amazon fine food review - sentiment analysis.
https://www.kaggle.com/code/laowingkin/amazon-fine-food-review-sentiment-analysis.
This work uses logistic regression to predict review sentiment scores (positive/negative) and provides three different logistic regression models; on word count, TFIDF (term frequency-inverse document frequency), and TFIDF + n-grams respectively. The main takeaway we want to get from this source is using logistic regression to predict sentiments of food reviews.

II. Existing sentiment analysis model on Hugging Face.
https://huggingface.co/siebert/sentiment-roberta-large-english.
This is the state of the art model that we are using as a benchmark to compare our own implementations against.

5) Summary of Our Contributions

We created models that should be able to predict the sentiment of a given review - positive or negative - to roughly 90% accuracy. We summarize the contributions as follows:

1. Implementation contribution(s)

For implementation, we explored a variety of models of ranging complexity. As a baseline, we have a majority label predictor, which already outputs an accuracy of 0.78 on the initially unbalanced dataset. We utilized traditional machine learning methods such as Logistic Regression in combination with vectorization methods such as CountVectorizer and TF-IDF. We also explored deep learning methods including a Feed-Forward Neural Network and BERT - a state-of-the-art language model.

2. Evaluation contributions

The aim of our experiments was to create a model that was effective in classifying sentiment. In terms of evaluation, we compared our baseline against a Dummy Majority classifier using key metrics such as accuracy and F1 score. We also analyzed two key dataset shifts - firstly by balancing out the labels of our test dataset, and secondly by training and testing on different time segments.

---

[1] Kaggle submission: Amazon fine food review - sentiment analysis.
https://www.kaggle.com/code/laowingkin/amazon-fine-food-review-sentiment-analysis.

[2] Existing sentiment analysis model on Hugging Face.
https://huggingface.co/siebert/sentiment-roberta-large-english.

6) Detailed Description of Contributions

6.1 Implementation Contributions

The most standard classification model is Logistic Regression. As a classical solution to NLP tasks, there are a lot of options on how to tokenize the text. We chose two standard options - a Bag-of-Words model and a TF-IDF model. In order to implement these two tokenizers, we relied on the sklearn library:

- Bag-of-Words: CountVectorizer
- TF-IDF: TfidfVectorizer

The tokenized inputs were then passed into the Logistic Regression Model. We experimented on a variety of Logistic Regression models, varying the type of tokenizer and adjusting the type of regularization used.

As an experiment, we also explored the capabilities of a feed-forward neural network (FFNN) in NLP tasks. Modern deep-learning solutions to NLP tasks use specific architectures (RNNs, transformers and even CNNs), however we wanted to evaluate the efficacy of a straight-forward FFNN. This is motivated by trying to understand if FFNNs have any role to play in NLP tasks, and understanding their limitations. We opted to continue using the CountVectorizer as the input to the network in order to better compare its performance to logistic regression. The FFNN struggled with overfitting, and we undertook various experiments to try and solve this problem:

- Number of layers: ranged from 2-4 fully connected layers
- Dimensions per layer: first hidden layer varied from 128 to 1024 neurons
- Dropout: probability of dropout varied from 0 to 0.8
- Activation function: ReLU v/s sigmoid
- Learning rate: 0.001, 0.01, 0.1
- Number of epochs: 5 - 10
- Batch size: 16 - 256

Our final architecture ended up being 2 fully-connected hidden layers (input_dim x 256, 256 x 32) and a fully-connected output layer (32 x 2) with a ReLU activation function and a dropout of 0.8 on both the hidden layers. We ran for 8 epochs, with a batch size of 256 and a learning rate of 0.001 using the Adam optimizer and Cross-Entropy Loss as our loss function.

We also used the BERT language model as a state-of-the-art solution. Imported from Hugging Face, we used the *sentiment-roBERTa-large-english* tokenizer and model. This model has been trained on sentiment data from Twitter, and therefore has highly transferable value. There were no hyperparameters to tune or architecture decisions involved with this. We also attempted transfer learning on this model. Unfortunately, given the size of this model (355M trainable parameters), running `.train()` on the model resulted in our instance crashing (GPU ran out of vRAM). Perhaps it would have been possible to train this model on a large AWS instance, but we suspect that this would have been beyond the capabilities of a standard EC2 instance.

Instead of that, we wrapped the outputs of BERT in a logistic regression model. Our BERT model outputs logits (the inputs to a softmax classification function), and we use these logits as inputs to a logistic regression model. The hypothesis behind this experiment is that the logits represent probabilities of the input being of either class, and perhaps the model uses a lighter/heavier definition of “positive”/ “negative”, resulting in a slight overprediction of a particular class. This logistic regression model was trained on the logits of the training set, and tested on the logits produced by the text of the test set.

6.2 Evaluation Contributions

Model	Base Dataset		Label Shift		Time Shift	
	F1 Score	**Accuracy**	**F1 Score**	**Accuracy**	**F1 Score**	**Accuracy**
Dummy Classifier	0.88	0.78	0.67	0.50	*untested*	
Logistic Regression - No regularization, CountVectorizer	0.92	0.88	*untested*		0.90	0.84
Logistic Regression - L2 Regularization, CountVectorizer	0.93	0.89	0.84	0.84	0.91	0.86
Logistic Regression - L2 Regularization, TF-IDF and n-grams	0.93	0.88	*untested*		0.91	0.85
FFNN	0.94	0.90	0.85	0.83	0.91	0.86
BERT	0.96	0.93	**0.91**	**0.91**	*untested*	
BERT + Logistic Regression	**0.97**	**0.95**	0.90	0.90		

The key aim of our experiments is to create a model that is effective in classifying sentiment. We've explored various attempts at this. We measure our success using key metrics such as F1 score (ideal for our imbalanced dataset) and accuracy, as well as by testing the model's performance on dataset shifts. Our baseline is a Dummy Classifier, that simply predicts the majority label (positive) for all text. In our imbalanced dataset, this gives an already decent F1 score and accuracy of 0.88 and 0.78 respectively. We used various test set sizes depending on the model. Given the dramatically different speed of prediction between Logistic Regression and BERT, we used the entire dataset to train/test Logistic Regression, 80% of the dataset to train the FFNN and 10% of the dataset for BERT and BERT + LR. This 10% still includes over 3,000 training points, and it takes over 10 minutes for BERT to simply run predictions on this sized dataset.

We considered two main dataset shifts. Firstly, a label shift, which adjusted the test set so that it included 50% positive reviews and 50% negative reviews. Given the imbalanced nature of our dataset, this shift is necessary in ensuring that the models aren't overpredicting positive sentiment. The second shift we considered was a time shift. Models were trained on data points before the median time of review, and tested on data points that were after the median time of review. This was to determine if there has been a change in the nature of reviews over time, and to evaluate if our models could cope with that. Note that the ratio of positive/negative reviews has stayed largely the same over time, which is a convenient independent variable.

Perhaps the most interesting conclusions can be drawn from the FFNN, and the comparison to Logistic Regression. Given an input dimension of ~ 40,000 from the CountVectorizer, the FFNN has around 10M trainable parameters. Naturally, the vast majority of these parameters are in the initial layer of the neural network and are likely to be zeroed out, but this is an obviously more complex model than Logistic Regression. However, for all that increase in complexity, the gains of the FFNN over Logistic Regression are minimal. This suggests that, despite our efforts to prevent overfitting, the model is still likely unnecessarily complex, and there are inherent limitations to the CountVectorizer that means that arbitrarily complex FFNNs cannot increase performance much. This helps us understand why FFNNs are not used much in NLP, as the performance gains over significantly simpler methods are not major.

It is evident that BERT is a superior model. It has better accuracy and F1 than any of our methods, and this is especially significant with the Label shift. This makes sense as the BERT model is not trained on any of our data, and therefore the shifts have less of an impact. BERT's architecture is significantly more complex than any of our methods, and has been trained on orders of magnitude more data. Despite not being able to fine-tune our model as we would have liked, adding the Logistic Regression layer was an interesting experiment. Whilst it may seem like a 1-2% increase, at the state-of-the-art level that is a significant improvement. It is evident that there is some mis-classification by BERT due to differing boundaries of positive/negative sentiment, and adding a regression layer does seem to have some impact. The fact that this experiment does underperform against the default BERT model when there's a label shift suggests there is some degree of overfitting in the Logistic Regression, but there is future potential in tuning this layer to perform better.

The label shift does highlight the problem with more primitive models, which suffer from overfitting. The drops in performance are significant and suggest an over-eagerness to predict the majority label. The time shift seems to not have had a significant impact, suggesting that the nature of reviews largely does not change over time, and that these models should be resilient to future data. It should be noted that certain models were not tested against certain shifts. The dummy classifier would perform the same with the time shift as it did with the base. Certain logistic regression models were not tested against shifts as all the models seemed to perform almost identically. Finally the BERT-based models were not tested against the time shift as it is apparent that the time shift does not have a significant impact and testing BERT is a time-consuming process.

We can conclude that logistic regression and BERT are the two most sound models for sentiment analysis. Logistic Regression offers simplicity and fast training/prediction, whereas BERT offers state-of-the-art prediction capabilities. The FFNN is better than logistic regression, but it is uncertain if those minor performance improvements warrant the increased complexity. BERT + LR is an interesting new idea that may be potentially better than the base model, but given the slightly worse performance on the label shift, there is still uncertainty over its true benefit.

7) Compute/Other Resources Used

We used the Google collab interface in order to collaboratively work together and run our code. We used the CUDA GPU architecture in order to improve our compute resources. We utilized the HuggingFace platform for accessing the pre-trained BERT model.

8) Conclusions

Outcomes:

From this project we can draw conclusions on the efficacy of various models on sentiment analysis. We established that classical machine learning models such as Logistic Regression are a solid baseline for sentiment analysis, especially when used in combination with vectorizers such as CountVectorizer and TF-IDF. We also realized that, whilst FFNNs do perform marginally better than classical models, this increase in performance does not necessarily justify the increased complexity of a neural network. We also explored state of the art models such as BERT, which we discovered was clearly superior to our hand-crafted methods. Finally, our experiment with BERT + Logistic Regression was interesting, and whilst we cannot be confident that Logistic Regression made a significant impact, it did have some impact that warrants further investigation.

From this project, we can draw important conclusions that could be valuable. We firstly establish that Logistic Regression is sufficiently good enough to implement as a simple solution. We also establish that

FFNNs are not necessarily a good architecture choice for NLP problems. Finally, our exploration with BERT + Logistic Regression could be applied to other similar NLP classification problems.

In hindsight which parts of your original proposal plan were you unable to execute?
In hindsight, it was difficult to load and train the dataset in an appropriate manner for the models that we employed. For example, we ran into RAM difficulties when trying to train the BERT model because of the large amount of space it would use. Furthermore, it was frustrating to create our own Neural Network because of the difficulties in overfitting. We created a Feed Forward Neural Network and it overpredicted the training data. We then tried to change the parameters but this proved to be difficult and not as effective as we would have liked. The feedback from our project mentor did enable us to improve our accuracy on the BERT model as we used the same tokenizer that BERT was pre-trained on.

For the Future:

We would extend this by finding better data that includes more features. For example, the only other relevant feature our dataset had was a timestamp of the review. The timeshift in the data was interesting to run and see how the models were able to execute after undergoing this shift. It would have been interesting to see how the ethnic demographics or socio-economic differences between user ratings would impact their sentiment towards products. We would be able to better understand different consumer segments and make better predictions, this can help companies improve their product offerings. From a technical perspective, it would have been interesting to implement more complex CNN or RNN models that would be able to offer a different conclusion on the dataset. However, it is evident that the BERT model would perform the best due to its significantly stronger capabilities.

Ethical Considerations, and Broader Social and Environmental Impact:
We noticed that we used a lot of RAM in order to process our data. We wanted our BERT to have better training accuracy however, we were limited by compute power. This is because the virtual RAM of the GPU ran out when we switched the BERT model to the training mode. This shows that in order to scale up our solution we would need significantly more computer power. This shows that even though this data is very useful it may be difficult for us to process it and run models because of physical limitations. Furthermore, ethical considerations include the use of identifying data that could be dangerous with the rise of Big Data. We can see this being a real threat when usernames, demographic data and other such revealing information could be a part of review data.

9) Roles of team members:

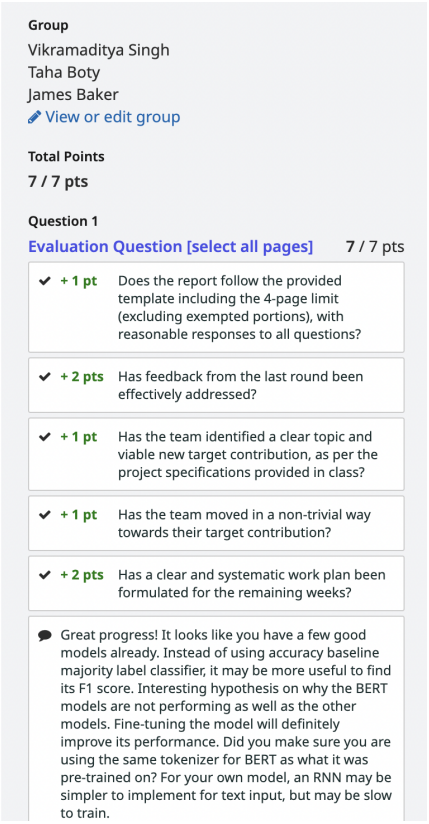
Vikramaditya Singh: Vikram worked on creating the Feed Forward Neural Network as well as the data shift on the positive and negative sentiment reviews. He also helped with the BERT model.

James Baker: James worked on the pre-processing steps as well as creating the Logistic Regression model and all its variants. He also attempted to make an RNN model but was unsuccessful.

Taha Boty: Taha implemented the BERT model and worked on training and testing the data using the tokenizer provided by the pre-trained BERT.

Main Links:
GitHub Project, Baseline, Logistic Regression and FFNN, BERT, Time Shift
(These links are to a custom Notebook viewer that provides a better viewing experience, however, if it shows a 404 error then the code is also available on Github.)
(Exempted from page limit) Other Prior Work / References (apart from Sec 3) that are cited in the text:

(Exempted from page limit)



Group

Vikramaditya Singh
Taha Boty
James Baker
View or edit group

Total Points

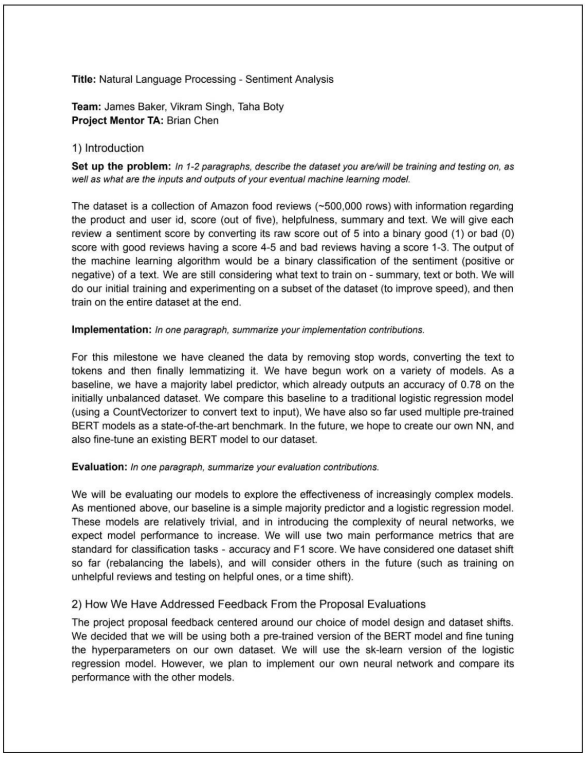
7 / 7 pts

Question 1

Evaluation Question [select all pages] **7** / 7 pts

- ✔ **+ 1 pt** Does the report follow the provided template including the 4-page limit (excluding exempted portions), with reasonable responses to all questions?
- ✔ **+ 2 pts** Has feedback from the last round been effectively addressed?
- ✔ **+ 1 pt** Has the team identified a clear topic and viable new target contribution, as per the project specifications provided in class?
- ✔ **+ 1 pt** Has the team moved in a non-trivial way towards their target contribution?
- ✔ **+ 2 pts** Has a clear and systematic work plan been formulated for the remaining weeks?

Great progress! It looks like you have a few good models already. Instead of using accuracy baseline majority label classifier, it may be more useful to find its F1 score. Interesting hypothesis on why the BERT models are not performing as well as the other models. Fine-tuning the model will definitely improve its performance. Did you make sure you are using the same tokenizer for BERT as what it was pre-trained on? For your own model, an RNN may be simpler to implement for text input, but may be slow to train.



Title: Natural Language Processing - Sentiment Analysis

Team: James Baker, Vikram Singh, Taha Boty
Project Mentor TA: Brian Chen

1) Introduction

Set up the problem: *In 1-2 paragraphs, describe the dataset you are/will be training and testing on, as well as what are the inputs and outputs of your eventual machine learning model.*

The dataset is a collection of Amazon food reviews (~500,000 rows) with information regarding the product and user id, score (out of five), helpfulness, summary and text. We will give each review a sentiment score by converting its raw score out of 5 into a binary good (1) or bad (0) score with good reviews having a score 4-5 and bad reviews having a score 1-3. The output of the machine learning algorithm would be a binary classification of the sentiment (positive or negative) of a text. We are still considering what text to train on - summary, text or both. We will do our initial training and experimenting on a subset of the dataset (to improve speed), and then train on the entire dataset at the end.

Implementation: *In one paragraph, summarize your implementation contributions.*

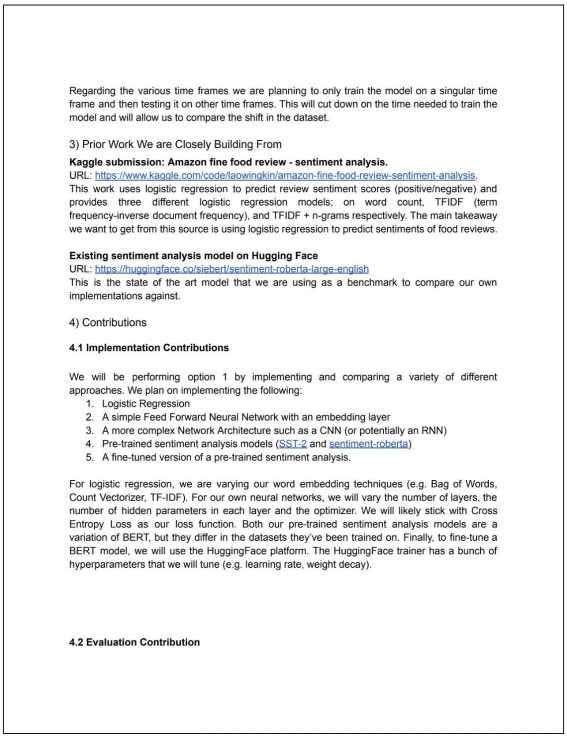
For this milestone we have cleaned the data by removing stop words, converting the text to tokens and then finally lemmatizing it. We have begun work on a variety of models. As a baseline, we have a majority label predictor, which already outputs an accuracy of 0.78 on the initially unbalanced dataset. We compare this baseline to a traditional logistic regression model (using a CountVectorizer to convert text to input), We have also so far used multiple pre-trained BERT models as a state-of-the-art benchmark. In the future, we hope to create our own NN, and also fine-tune an existing BERT model to our dataset.

Evaluation: *In one paragraph, summarize your evaluation contributions.*

We will be evaluating our models to explore the effectiveness of increasingly complex models. As mentioned above, our baseline is a simple majority predictor and a logistic regression model. These models are relatively trivial, and in introducing the complexity of neural networks, we expect model performance to increase. We will use two main performance metrics that are standard for classification tasks - accuracy and F1 score. We have considered one dataset shift so far (rebalancing the labels), and will consider others in the future (such as training on unhelpful reviews and testing on helpful ones, or a time shift).

2) How We Have Addressed Feedback From the Proposal Evaluations

The project proposal feedback centered around our choice of model design and dataset shifts. We decided that we will be using both a pre-trained version of the BERT model and fine tuning the hyperparameters on our own dataset. We will use the sk-learn version of the logistic regression model. However, we plan to implement our own neural network and compare its performance with the other models.



Regarding the various time frames we are planning to only train the model on a singular time frame and then testing it on other time frames. This will cut down on the time needed to train the model and will allow us to compare the shift in the dataset.

3) Prior Work We are Closely Building From

Kaggle submission: Amazon fine food review - sentiment analysis.
URL: https://www.kaggle.com/code/laowingkin/amazon-fine-food-review-sentiment-analysis.
This work uses logistic regression to predict review sentiment scores (positive/negative) and provides three different logistic regression models; on word count, TFIDF (term frequency-inverse document frequency), and TFIDF + n-grams respectively. The main takeaway we want to get from this source is using logistic regression to predict sentiments of food reviews.

Existing sentiment analysis model on Hugging Face
URL: https://huggingface.co/siebert/sentiment-roberta-large-english
This is the state of the art model that we are using as a benchmark to compare our own implementations against.

4) Contributions

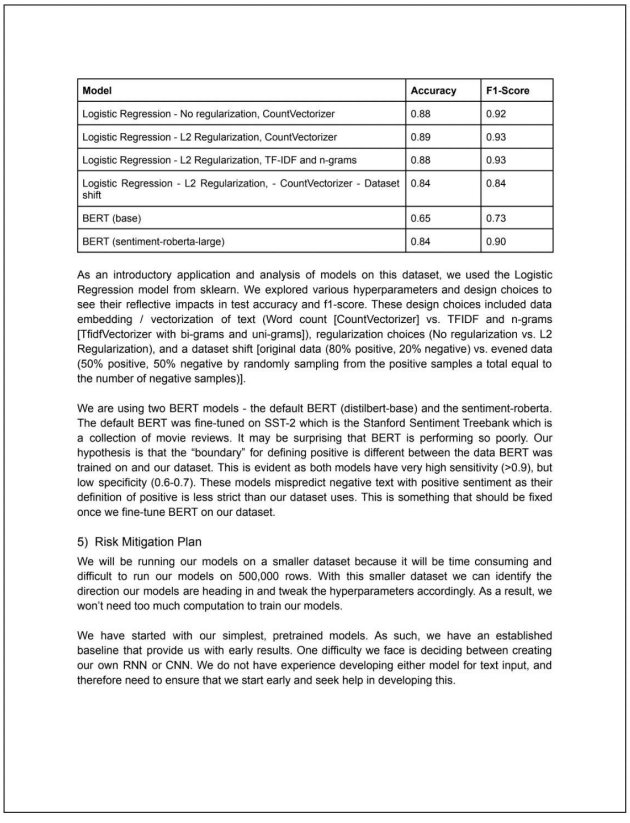
4.1 Implementation Contributions

We will be performing option 1 by implementing and comparing a variety of different approaches. We plan on implementing the following:

1. Logistic Regression
2. A simple Feed Forward Neural Network with an embedding layer
3. A more complex Network Architecture such as a CNN (or potentially an RNN)
4. Pre-trained sentiment analysis models (SST-2 and sentiment-roberta)
5. A fine-tuned version of a pre-trained sentiment analysis.

For logistic regression, we are varying our word embedding techniques (e.g. Bag of Words, Count Vectorizer, TF-IDF). For our own neural networks, we will vary the number of layers, the number of hidden parameters in each layer and the optimizer. We will likely stick with Cross Entropy Loss as our loss function. Both our pre-trained sentiment analysis models are a variation of BERT, but they differ in the datasets they've been trained on. Finally, to fine-tune a BERT model, we will use the HuggingFace platform. The HuggingFace trainer has a bunch of hyperparameters that we will tune (e.g. learning rate, weight decay).

4.2 Evaluation Contribution



Model	Accuracy	F1-Score
Logistic Regression - No regularization, CountVectorizer	0.88	0.92
Logistic Regression - L2 Regularization, CountVectorizer	0.89	0.93
Logistic Regression - L2 Regularization, TF-IDF and n-grams	0.88	0.93
Logistic Regression - L2 Regularization, - CountVectorizer - Dataset shift	0.84	0.84
BERT (base)	0.65	0.73
BERT (sentiment-roberta-large)	0.84	0.90

As an introductory application and analysis of models on this dataset, we used the Logistic Regression model from sklearn. We explored various hyperparameters and design choices to see their reflective impacts in test accuracy and f1-score. These design choices included data embedding / vectorization of text (Word count [CountVectorizer] vs. TFIDF and n-grams [TfidfVectorizer with bi-grams and uni-grams]), regularization choices (No regularization vs. L2 Regularization), and a dataset shift [original data (80% positive, 20% negative) vs. evened data (50% positive, 50% negative by randomly sampling from the positive samples a total equal to the number of negative samples)].

We are using two BERT models - the default BERT (distilbert-base) and the sentiment-roberta. The default BERT was fine-tuned on SST-2 which is the Stanford Sentiment Treebank which is a collection of movie reviews. It may be surprising that BERT is performing so poorly. Our hypothesis is that the "boundary" for defining positive is different between the data BERT was trained on and our dataset. This is evident as both models have very high sensitivity (>0.9), but low specificity (0.6-0.7). These models mispredict negative text with positive sentiment as their definition of positive is less strict than our dataset uses. This is something that should be fixed once we fine-tune BERT on our dataset.

5) Risk Mitigation Plan

We will be running our models on a smaller dataset because it will be time consuming and difficult to run our models on 500,000 rows. With this smaller dataset we can identify the direction our models are heading in and tweak the hyperparameters accordingly. As a result, we won't need too much computation to train our models.

We have started with our simplest, pretrained models. As such, we have an established baseline that provide us with early results. One difficulty we face is deciding between creating our own RNN or CNN. We do not have experience developing either model for text input, and therefore need to ensure that we start early and seek help in developing this.

ria

evaluation):

N/A