KENNESAW STATE
UNIVERSITY

GROUP PROJECT

IT 7143 CLOUD ANALYTICS TECHNOLOGY

GROUP MEMBERS:

DRASHTI PATEL

QUANN SHEARS

REUVEN MULLER

SIBGHA AJMAL

ACKNOWLEDGEMENT

We would like to thank Professor Jyothsna Dinadayalane for providing the course on cloud and analytics technology which contributed to the development of this work.

ABSTRACT

With the exponential growth of online shopping, vast amounts of data on consumer behavior and satisfaction are generated, offering valuable insights for businesses. However, the manual curation of this data is impractical due to its volume. The project focuses on creating a model that accurately classifies reviews into positive and negative categories, emphasizing the automatic detection of user complaints. By experimenting with various machine learning models and hyperparameter configurations, the goal is to optimize accuracy in identifying negative feedback. The resulting system will enable businesses to efficiently address customer concerns and improve their services through automated review analysis.

INTRODUCTION

In the current digital age, online shopping has become a major source of data on consumer behavior and customer satisfaction. Analyzing this data can provide invaluable insights for making informed business decisions. However, the vast amount of data generated from user feedback presents challenges for businesses to manage; manually curating this large amount of data is often impractical.

This project focuses on the analysis of user reviews and ratings of the Amazon Shopping App using machine learning techniques. The challenge with analyzing user reviews and ratings is accurately and automatically filtering out bad ratings and complaints from large volumes of feedback. The primary goal in this project is to train a model to classify reviews into good and bad categories, with a particular emphasis on filtering out user complaints. By tackling this problem with machine learning, businesses can use the resulting models to automate the process of identifying bad reviews and complaints and then effectively attend to all their customers' needs and concerns, allowing them to improve their services.

In this project we will experiment with different models and under different hyperparameter configurations. Our research aims to train a model that yields the most accurate results.

LITERATURE REVIEW

In early days, researchers primarily relied on knowledge-based techniques and simple statistical methods. These approaches include lexicon-based methods, which used predefined lists of words associated with specific sentiments such as “happy”, “sad”, or “afraid.” Rule-based systems were also common, classifying text based on a set of manually crafted rules. Additionally, basic statistical methods like latent semantic analysis and support vector machines were employed to analyze sentiments.

Modern sentiment analysis has evolved significantly, incorporating more sophisticated techniques driven by advancements in machine learning and natural language processing. Current approaches include machine learning-based methods that use supervised and unsupervised learning algorithms to classify sentiments. Deep learning techniques, employing neural networks and other advanced AL models, allow for more nuanced analysis. Aspect-based sentiment analysis has emerged, focusing on identifying sentiment towards specific aspects or features within a text. Furthermore, emotion detection techniques have been developed, going beyond simple polarity to detect specific emotions like happiness, frustration, and anger.

Today, two approaches dominate the field of sentiment analysis: machine learning and rule-based methods. The machine learning approach uses algorithms trained on large datasets to classify sentiment. This method can adapt to new data and improve over time, capable of handling complex language nuances and context. Examples of machine learning techniques include support vector machines, naive Bayes, and various deep learning models. On the other hand, the rule-based approach uses predefined rules and lexicons to determine sentiment. Often based on natural language processing techniques, this method can be more interpretable and easier to adjust for specific domains. It typically involves creating lists of positive and negative words or phrases and applying them to the text.

Both machine learning and rule-based approaches have their strengths and are often used in combination for more robust sentiment analysis systems. Machine learning approaches tend to be more accurate and adaptable, especially for large-scale analysis, while rule-based approaches can be more transparent and easier to implement for specific use cases. The field of sentiment analysis continues to evolve, with ongoing research into more advanced techniques like deep learning-based approaches and multimodal sentiment analysis, which incorporates visual and audio data alongside text.

METHODS

In this section we list the models that we chose, and the methods applied for training.

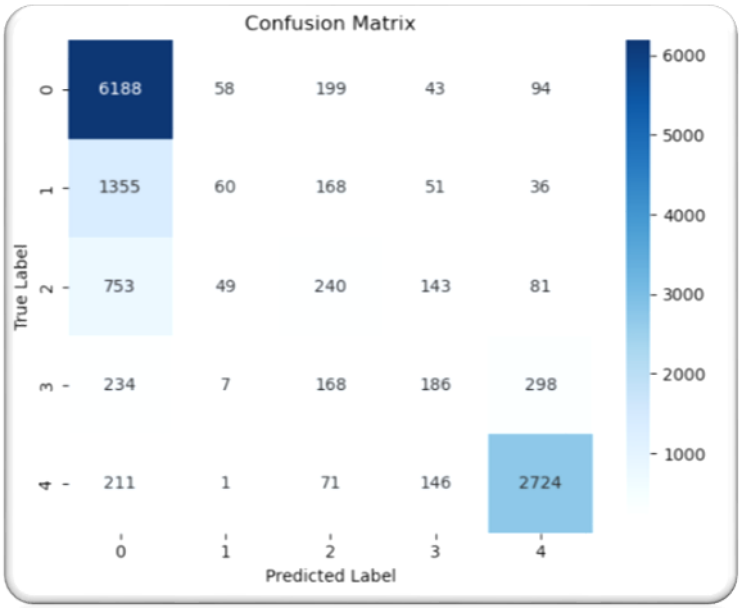
BERT Model: This AWS-based model involves several main components and steps, starting with the environment setup. It required the installation of libraries such as transformers, torch, datasets, and s3fs for accessing AWS S3. Additionally, environment variables were configured to ensure that parallelism in tokenizers is set to false for optimized performance.

For data handling, the model integrated with AWS S3 for storing and retrieving datasets and scripts. A CSV file containing Amazon reviews was loaded from S3 into a pandas DataFrame, which was then split into training and validation sets. These sets were subsequently uploaded back to S3 for training purposes.

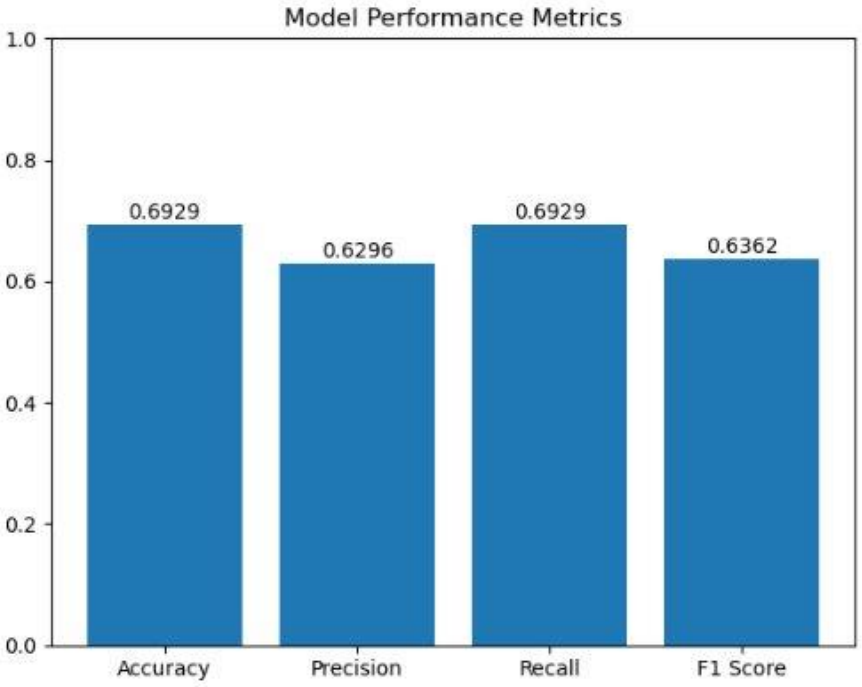
In terms of model configuration and training, the process began with the initialization of the tokenizer and model using BertTokenizer and BertForSequenceClassification from the Hugging Face transformers library, specifically a pre-trained BERT model (prajjwal1/bert-tiny). A PyTorch estimator was then configured within SageMaker, specifying hyperparameters such as the number of epochs, batch size, and instance type. The model was trained on the training dataset and validated on the validation dataset stored in S3.

Following training, the model was downloaded from S3 and extracted for deployment and evaluation. The fine-tuned model and tokenizer were loaded to predict sentiment for new texts. Evaluation metrics such as accuracy, precision, recall, F1 score, and confusion matrix were used to assess the model's performance. These metrics are visualized using plots for better understanding.

The confusion matrix visualizes the performance of the classification model by showing the true versus predicted labels.



For visualization, a metrics bar chart displays accuracy, precision, recall, and F1 score.



Overall, this model leverages the powerful BERT architecture fine-tuned on a specific dataset for sentiment analysis. The use of AWS SageMaker allows for scalable and efficient training and evaluation, while comprehensive evaluation metrics and visualizations provide clear insights into the model's performance.

Key metrics from the evaluation include accuracy, which measures the overall correctness of the model; precision, which indicates the number of true positive predictions out of all positive predictions; recall, which represents the number of true positives out of all actual positives; and the F1 score, which is the harmonic mean of precision and recall, providing a single metric to evaluate the model's performance.

GPT2 For sequence classification: GPT2 is a large language model by OpenAI. GPT stands for generative pretrained transformer, and as the name suggests is based on the transformer architecture. See the paper "Attention is All You Need" for details [1]. GPT2ForSequenceClassification is a GPT2 model with an additional head designed for sequence classification problems.

The training process works as follows: first the data needs to be tokenized. This refers to the process of converting text data into tokens that the model being trained can interpret it. We then feed the data along with the labels to the model in a training loop. In the training loop a forward pass is done through the model and based on its prediction and the true value from the labels a loss is calculated. The loss is then used to update the model internally so that it can learn to make better predictions.

In our training setup, we used 30000 review rating pairs, which we split into 0.8 and 0.2 train and test datasets. We had a batch size of 16 items per training loop and we trained on the data 5 times (5 epochs).

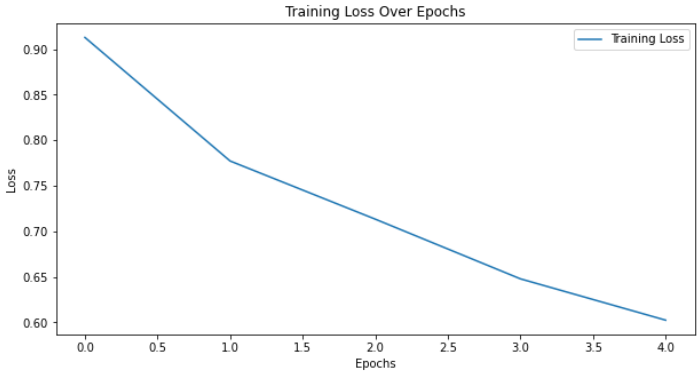
These were the results after training:

```
Final Evaluation Loss: 0.8413001093864441
Final Evaluation Accuracy: 0.6858333333333333
Precision: 0.648050124379942
Recall: 0.6858333333333333
F1 Score: 0.6605251839236552

Classification Report:
              precision    recall  f1-score

           0       0.75      0.89      0.81
           1       0.34      0.23      0.28
           2       0.36      0.22      0.28
           3       0.42      0.35      0.38
           4       0.86      0.87      0.86

    accuracy                           0.69
   macro avg       0.55      0.51      0.52
weighted avg       0.65      0.69      0.66
```

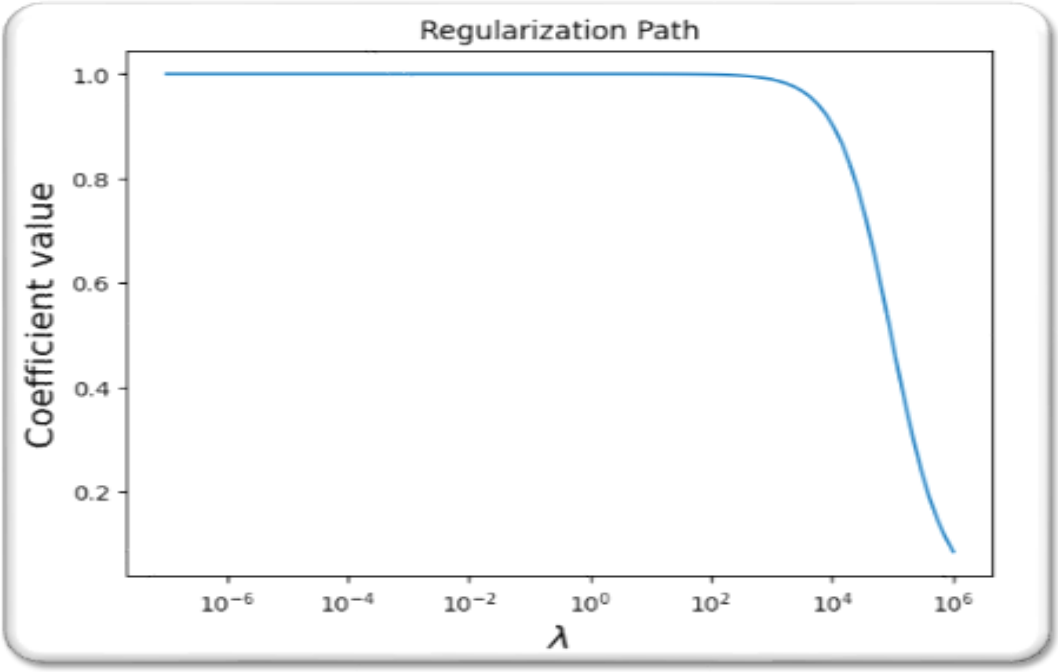


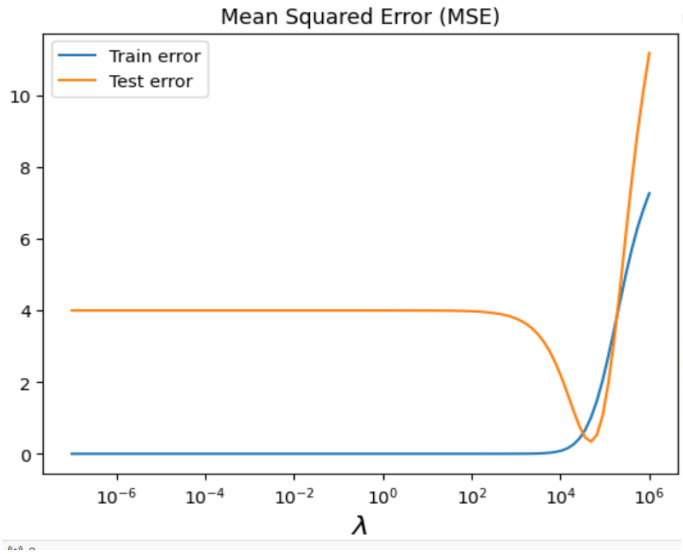
We want to focus on accuracy and f1 scores. Having an accuracy score and f1 score that are close to each other indicate that the training is not causing the model to overfit the data. In our case, the average accuracy score was 0.69 and the average F1 score 0.66. This means that the model predicts the correct answer 69% of the time.

Ridge Regression Model: Ridge regression is a statistical regularization technique used for estimating the coefficients in scenarios where the independent variables are highly correlated. The preprocessed amazon dataset includes multicollinear variable ratings ranging from 1 to 5 that fit the beneficiary rigors of the ridge regression model. As the environment was set differing libraries were installed ranging from sklearns model section for train test split. linear ridge models and pandas. Functions for processing the loss, regularization factor and mean squared error were implicated for the integration of calculated

comparisons and predictor variables. The comma separated value dataset was loaded and reshaped into a panda dataframe prior to creating train test split sets. These sets were then fit into a regressor to properly calculate predicted test scores based off the test set.

A predicted rating was calculated and an accuracy score was measured to compare similarity between initial rating scores and predicted rating scores used within test field which gave us a 99% accuracy score. The dimensions of the array were then defined before appending the lambda values, train errors and test errors for more precise measurement with the ridge parameter. The mean squared error represents the average squared residual. As the data points fall closer to the regression line, the less statistical error there was in the model. Since lambda produces generalization models for new data, we used it with estimated coefficients and continued for different set values of lambda to plot the regularization path. The same set lambda values to plot the difference in the models for train error and test error and calculated the difference between the test to receive a mean square error (MSE) of a little under 7.3. The overall accuracy score between the predicted test set and the actual test set was 99% and this model proved to be efficient and effective for analyzing the data especially regarding the multicollinear variables in the amazon reviews and ratings





Score: 0.999999971315096

	Score	Predicted Scores
0	2	2.000074
1	1	1.000244
2	3	2.999905
3	4	3.999736
4	1	1.000244
...	...	...
43398	1	1.000244
43399	1	1.000244
43400	3	2.999905
43401	3	2.999905
43402	1	1.000244

Distilbert-Base-Uncased: Based on this analysis, we can see that the dataset contains 52,901 Amazon reviews with their corresponding scores. The scores range from 1 to 5, with a mean score of about 2.44. Interestingly, only about 29.85% of the reviews are positive (score >= 4), indicating that there are more negative or neutral reviews in this dataset.
The score distribution graph shows that there's a higher concentration of reviews with lower scores (1 and 2), which aligns with the relatively low percentage of positive reviews.

1. Accuracy Score: The accuracy score, which represents the percentage of reviews with a score of 4 or 5, is approximately 29.85%. This means that about 29.85% of the reviews in the dataset are considered positive or highly positive.
2. Score Distribution:
    - 1 star: 48.23% of reviews
    - 2 stars: 12.54% of reviews
    - 3 stars: 9.37% of reviews
    - 4 stars: 6.67% of reviews
    - 5 stars: 23.18% of reviews
3. Mean Score: The average score is approximately 2.44 out of 5, which indicates a generally negative sentiment in the reviews.
4. Median Score: The median (50th percentile) score is 2, confirming that more than half of the reviews have a score of 2 or lower.
5. Standard Deviation: The standard deviation of 1.65 suggests a significant spread in the scores.

These statistics paint a picture of a dataset where negative reviews (1 and 2 stars) dominate, making up about 60.77% of all reviews. Positive reviews (4 and 5 stars) account for 29.85% of the total, which matches our accuracy score. The remaining 9.37% are neutral (3-star) reviews.

This distribution explains why the mean score (2.44) is lower than what one might expect from a balanced set of reviews. This data set clearly contains a higher proportion of negative feedback, which could be valuable for identifying areas of improvement or common issues reported by customers.

	score
count	54253.0
mean	2.4403627449173317
std	1.653889831658395
min	1.0
25%	1.0
50%	2.0
75%	4.0
max	5.0
accuracy	29.854570254179492

DATA COLLECTION

The data set was taken from Kaggle and contains reviews and ratings of Amazon Shopping App by users. It also includes information on the relevancy of reviews and the date of posting the review.

Date of data set: 06/16/2024

Rows: 52901

Features: 6 (ID, Reviewer, Review, Rating, Likes, App version, Date)

Data processing: We removed all non-text values in the review section as well as all non-English text in the review section. We removed all null values in the ratings section. Lastly, we dropped all columns except for "Review" and "Ratings".

RESULTS

Model	Accuracy Score
GPT2 for classification	0.68
Ridge regression	0.99
Bert	0.69
Distilbert-Base-Uncased	0.29

SUGGESTIONS AND FUTURE RESEARCH

Suggestions for improvement are to determine overfitting, especially in the case of ridge regression. Furthermore, to explore how much models improve with larger datasets.

For future research, one could consider combining machine learning and deep learning techniques to include the strengths of both approaches.

DISCUSSION AND CONCLUSION

In our project, we used two distinct major architectures: ridge regression and two deep learning models based on the transformer architecture (GPT2 for sequence classification and Bert).

The first architecture, ridge regression, is a linear regression model used together with a regularization component. Ridge regression is much simpler than the second architecture we used. It is well-suited for tasks where the data can be approximated linearly. Due to its simplicity, it requires fewer resources for training. However, for sequence classification tasks, using ridge regression requires converting the text data into numerical embeddings before applying the model.

The second architecture for both GPT2 and Bert is a deep learning model based on the transformer architecture. Transformers have a complex internal structure that allows them to capture complex relationships in the data. They are useful for complex relationships such as in language. However, transformers require a lot of computational resources for training.

Our findings suggest that ridge regression performed the best, but it may be because the model was overfitting the data.

REFERENCES

[1] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., Polosukhin, I., & others. (2017). Attention is all you need (Version 7). arXiv. https://arxiv.org/abs/1706.03762v7

[2] "Stack Overflow." Stack Overflow, https://stackoverflow.com. Accessed 14 July 2024.

[3] "AWS re:Post." AWS re:Post, https://repost.aws. Accessed 18 July 2024.

[4] Hugging Face. (n.d.). Sequence Classification. Retrieved July 22, 2024, from https://huggingface.co/docs/transformers/en/tasks/sequence_classification

[5] Thematic. "Sentiment Analysis | Comprehensive Beginners Guide." https://getthematic.com/sentiment-analysis/

[6] Towards Data Science. "Sentiment Analysis: Concept, Analysis and Applications." https://towardsdatascience.com/sentiment-analysis-concept-analysis-and-applications-6c94d6f58c17