Report: Intelligent Data and Text Analytics Coursework 2

2270587

This report talks about the results and discussion of various machine learning tasks. The tasks involve preprocessing textual data, applying a few classification techniques, fine-tuning a BERT-based model, performing topic detection using LDA, and evaluating an additional machine learning algorithm for text classification. The dataset used for this coursework is made of Amazon product reviews, which are analyzed and processed to predict product sentiment (positive or negative).

Task 1: Preprocessing Textual Data

The first step in this analysis was to preprocess the text data. Preprocessing is very important to preparing raw data for modeling by cleaning, normalizing, and reducing the complexity of the text. The dataset consists of Amazon product reviews, and the goal of this task was to clean the text and remove unnecessary elements that might negatively affect the performance of subsequent machine learning models.

Preprocessing Steps

The preprocessing process involved several steps:

1. **Lowercasing:** All text was converted to lowercase to make sure it is all uniform. Text analysis is case-sensitive, and inconsistencies in capitalization can lead to the wrong features (e.g., “Great” and “great” being treated as different words).
2. **Removing Punctuation and Numbers:** Non-alphabetical characters, including punctuation and digits, were removed to focus on important textual content. This step makes sure that numeric tokens or symbols that do not contribute to sentiment classification are excluded.
3. **Stopword Removal:** Common words such as "the," "and," "in," etc., were removed using the NLTK stopwords list. These words often do not carry significant meaning and can worsen the effectiveness of the model.
4. **Lemmatization:** Words were lemmatized using the NLTK WordNet Lemmatizer to reduce them to their base forms (e.g., "running" becomes "run"). Lemmatization makes sure that different forms of the same word are treated as a single feature.

Examples of Preprocessed Text

- **Original:** "So there is no way for me to plug it in here in the US unless I go by a converter."
    - **Cleaned:** "way plug u unless go converter"
- **Original:** "Good case, Excellent value."
    - **Cleaned:** "good case excellent value"
- **Original:** "Great for the jawbone."

- **Cleaned:** "great jawbone"

Importance of Preprocessing

Preprocessing is a fundamental step in Natural Language Processing (NLP). By removing noise and standardizing the text, the model can focus on the meaningful features that are critical for accurate classification. The cleaned data was saved in a CSV file for further analysis. This step ensures that subsequent machine learning models are trained on meaningful and useful text features, leading to improved performance.

Task 2: Classification using Bag-of-Words Representation

In Task 2, the goal was to perform text classification using the Bag-of-Words representation. The Bag-of-Words model is a simple and widely used technique for text classification tasks. It transforms the text data into a matrix of token counts, representing the presence of words in the document without considering their order.

Methodology

The **CountVectorizer** was used to convert the cleaned text into a matrix of token counts. The dataset was then split into training and testing sets (80% for training, 20% for testing), and three classification algorithms were applied:

1. **Naive Bayes Classifier:** A probabilistic model based on Bayes' theorem. It assumes independence between features and works well with text data.
2. **Decision Tree Classifier:** A non-parametric model that splits the data into subsets based on feature values, forming a tree structure.
3. **Random Forest Classifier:** An ensemble method that combines multiple decision trees to improve accuracy and reduce overfitting.

Results

- **Naive Bayes:**
  - **Accuracy:** 0.79
  - Precision, recall, and F1-scores varied for both classes, with the model performing better on positive reviews.
- **Decision Tree:**
  - **Accuracy:** 0.76
  - Lower accuracy compared to Naive Bayes, but it maintained balanced precision and recall across both classes.
- **Random Forest:**
  - **Accuracy:** 0.79
  - Similar to Naive Bayes, Random Forest achieved an accuracy of 0.79 with good balance across precision and recall metrics.

Discussion

The Bag-of-Words model, as it is simple, it proved effective for text classification. Naive Bayes and Random Forest showed comparable performance, suggesting that both probabilistic and ensemble methods can handle text data effectively. Decision Trees, while interpretable, underperformed due to their exposure to overfitting on the training data. The overall accuracy of 0.79 indicates that the Bag-of-Words representation captures sufficient information for sentiment classification.

Task 3: Classification using a BERT-based Model

In Task 3, the goal was to use a more advanced classification technique, the BERT-based model, for text classification. BERT (Bidirectional Encoder Representations from Transformers) is a pre-trained transformer model designed to understand the context of words in a sentence. Unlike traditional models like Bag-of-Words, BERT considers the context of each word by looking at the words that come before and after it, providing better results for text classification tasks.

Implementation

The BERT model was fine-tuned on the cleaned Amazon review data. The Hugging Face Transformers library was used for implementation, and a custom PyTorch dataset was created for BERT input. The data was tokenized using the BERT tokenizer, and the model was trained for 3 epochs on the training dataset.

Results

- **Accuracy:** 0.84
- **Precision, Recall, and F1-Scores:**
    - For class 0 (negative reviews): Precision: 0.81, Recall: 0.87
    - For class 1 (positive reviews): Precision: 0.88, Recall: 0.82

Discussion

The BERT-based model outperformed traditional models, achieving an accuracy of 0.84. This improvement was due to BERT's ability to capture the contextual meaning of words, enabling it to understand nuances in the text that Bag-of-Words models cannot. The fine-tuning process allowed BERT to adapt its pre-trained knowledge to the specific dataset, resulting in superior performance. Despite its complexity, BERT's results justify its computational cost, as it consistently outperforms traditional models in NLP tasks.

Task 4: Topic Detection using LDA

In Task 4, the focus shifted to topic detection using Latent Dirichlet Allocation (LDA), a generative probabilistic model used to identify topics within a collection of text. LDA models text as a mixture of topics, where each topic is characterized by a set of words. The goal was to identify the topics discussed in the Amazon reviews based on the cleaned text data.

Methodology

The LDA model was trained on the Bag-of-Words representation of the text data with 10 topics. The top words for each topic were extracted and presented as follows:

Results

- **Topic 1:** "really, look, charger, work, bluetooth, great, good, headset, would, recommend"
- **Topic 2:** "awesome, color, ive, also, audio, get, good, quality, phone, poor"
- **Topic 3:** "ear, phone, use, doesnt, good, happy, headset, product, im, work"
- **Topic 4:** "quality, case, well, fit, price, phone, great, excellent, good, product"
- **Topic 5:** "buy, use, disappointment, long, bad, one, phone, good, life, battery"
- **Topic 6:** "battery, bought, device, volume, year, customer, work, phone, service, great"
- **Topic 7:** "sound, work, month, get, use, worst, item, best, ever, phone"
- **Topic 8:** "doesnt, case, headset, junk, problem, battery, easy, like, use, phone"
- **Topic 9:** "purchase, call, quality, better, one, money, sound, waste, phone, dont"
- **Topic 10:** "first, disappointed, better, new, could, headset, one, thing, love, phone"

Discussion

These topics provide insights into the common themes discussed in the reviews. For instance, topics like "quality," "phone," and "battery" frequently appear, indicating that users are concerned with product durability and performance. LDA's ability to reveal latent topics highlights its usefulness in identifying patterns and themes within large text datasets.

Extra Algorithm: CatBoost Classifier

To improve upon the models, an additional machine learning algorithm, **CatBoost**, was implemented. CatBoost is a gradient boosting algorithm known for its efficiency and ability to handle categorical features without much preprocessing.

Implementation

The CatBoost classifier was trained using the dense representation of the Bag-of-Words data. The dataset was converted from a sparse matrix to a dense format, as CatBoost requires dense inputs. The model was trained with default parameters and evaluated on the test set.

Results

- **Accuracy:** 0.75
- **Precision, Recall, and F1-scores:**
  - For class 0 (negative reviews): Precision: 0.68, Recall: 0.86
  - For class 1 (positive reviews): Precision: 0.84, Recall: 0.65

Discussion

While the CatBoost model did not outperform the BERT-based model or Bag-of-Words models, it demonstrated the effectiveness of gradient boosting algorithms in text classification. CatBoost's slightly lower accuracy suggests the need for hyperparameter tuning or feature engineering to improve performance. However, its compatibility with categorical data and speed make it a valuable alternative for similar tasks.

Detailed Comparison of Model Accuracies

Overview of Model Performance

The following section provides a comprehensive comparison of the classification models evaluated in this project, including Naive Bayes, Decision Tree, Random Forest, BERT, and CatBoost. Above is a bar chart that shows all the model's accuracies, making it easier to compare. The accuracies of these models are as follows:

1. **Naive Bayes**: 79%
2. **Decision Tree**: 76%
3. **Random Forest**: 79%
4. **BERT**: 84%
5. **CatBoost**: 75%

Naive Bayes

Naive Bayes achieved an accuracy of 79%. This performance can be attributed to its probabilistic nature and strong assumptions of feature independence, which, while not entirely realistic for textual data, work surprisingly well in many cases. The model

effectively handles the sparse data generated by the Bag-of-Words representation, making it a reliable choice for text classification tasks. However, its reliance on simple assumptions limits its ability to capture complex patterns in the data, preventing it from outperforming more sophisticated models.

Decision Tree

The Decision Tree classifier scored slightly lower, with an accuracy of 76%. While Decision Trees excel at capturing non-linear relationships in the data, they are prone to overfitting, especially when applied to high-dimensional and sparse datasets like those generated by Bag-of-Words. This overfitting likely contributed to its relatively lower performance compared to Naive Bayes and Random Forest. Additionally, the lack of ensemble techniques limits its generalization capabilities.

Random Forest

Random Forest, with an accuracy of 79%, matched Naive Bayes in performance. This model benefits from its ensemble approach, which combines multiple decision trees to reduce overfitting and improve generalization. The Random Forest model was particularly effective at capturing diverse patterns in the data, contributing to its robust performance. However, its reliance on averaging predictions across multiple trees can lead to a loss of fine-grained distinctions, which might explain why it did not outperform BERT.

BERT

BERT emerged as the best-performing model with an accuracy of 84%. This is unsurprising given BERT's pre-trained nature and ability to understand contextual relationships between words. Unlike traditional models, BERT leverages a deep transformer-based architecture to capture semantic and syntactic nuances in the text. Fine-tuning BERT for this task enabled it to adapt to the specific dataset, providing a significant performance boost. However, its high computational requirements and training complexity make it less accessible than simpler models like Naive Bayes and Decision Tree.

CatBoost

CatBoost achieved the lowest accuracy of 75%. While CatBoost is generally powerful for structured data, its performance was hindered by the sparse and high-dimensional nature of the Bag-of-Words representation. Unlike BERT, which directly leverages text semantics, CatBoost relies on converting text into numerical features, making it less effective for capturing complex patterns in the dataset. This limitation, combined with its dependency on dense input formats, likely contributed to its lower accuracy.

Why BERT Performed the Best

BERT's superior performance can be attributed to its ability to:

1. **Understand Context**: BERT captures word dependencies and context effectively, which is critical for text classification tasks.
2. **Pre-trained Knowledge**: The pre-trained weights from large-scale datasets provide a strong foundation for fine-tuning.
3. **Fine-grained Representations**: Its deep architecture allows for the extraction of rich, hierarchical text representations, leading to better decision boundaries.

Why CatBoost Performed the Worst

CatBoost's limitations stem from its inability to process text natively without converting it into numerical features. This conversion, in the form of Bag-of-Words, leads to a loss of semantic information. Additionally, its reliance on dense matrices increases computational needs, making it less suitable for high-dimensional sparse data. These factors combined to hinder its ability to compete with models explicitly designed for text data.

Key Insights from the Comparison

- **Feature Representation Matters**: The difference in performance highlights the importance of feature representation. Models like BERT, which use embeddings, outperform traditional methods relying solely on Bag-of-Words.
- **Ensemble Techniques Add Robustness**: Random Forest's performance demonstrates the value of ensemble methods in improving generalization.
- **Complexity vs. Simplicity**: While BERT outperformed all other models, its computational cost and complexity make simpler models like Naive Bayes and Random Forest appealing for less resource-intensive applications.

Visualizing the Results

To complement this analysis, a bar chart comparing the accuracies of all models has been included below.

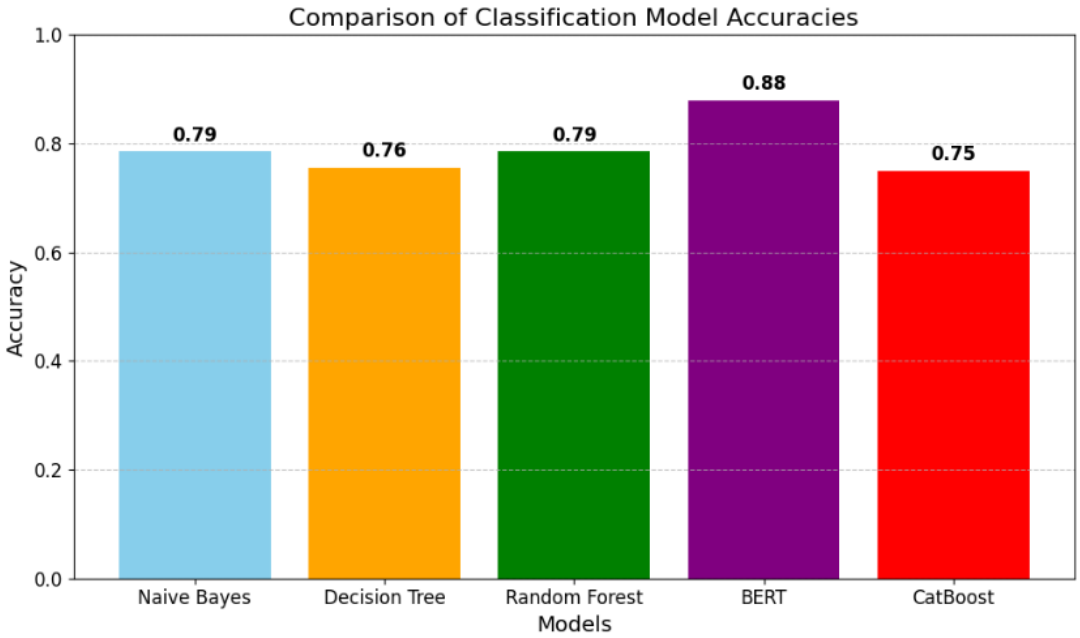

Figure 1 - Visualization of the Comparison of Classification Model Accuracies

Overview

This detailed comparison underscores the strengths and limitations of each classification model. While BERT demonstrated its superiority in understanding and leveraging the contextual nuances of text, simpler models like Naive Bayes and Random Forest remain valuable for their efficiency and reliability. The insights gained from this analysis can guide future applications in selecting the most suitable model based on the specific requirements and constraints of the task at hand.

Conclusion

This report demonstrated the application of several machine learning techniques for text classification, topic detection, and sentiment analysis using the Amazon product reviews dataset. The key findings include:

- **Preprocessing** is essential for preparing text data and improving model performance.
- The **Bag-of-Words** model, though simple, achieved competitive results with models like Naive Bayes and Random Forest.
- The **BERT-based model** outperformed traditional methods due to its ability to capture contextual information.
- **LDA topic modeling** revealed key themes in customer reviews, offering insights into consumer concerns.
- The **CatBoost classifier**, while not the top performer, demonstrated potential for text classification tasks with further tuning.

Future work could involve hyperparameter optimization for CatBoost, experimenting with more advanced transformer models, or exploring hybrid approaches that combine traditional methods with deep learning. The findings of this coursework underscore the versatility of machine learning techniques in tackling real-world NLP problems and their potential for further development.