MS9007
NLP Capstone Project

Pauline Ng

Objectives

- This project aims to conduct binary sentiment classification on IMDB movie reviews using traditional machine learning models, such as logistic regression and support vector machines.
- This project also aims to compare DistilBERT's performance on raw text, showcasing its robustness to handle real-world data effectively.

Dataset

- The IMDB dataset is sourced from Hugging Face's datasets library and originates from StanfordNLP.
- The dataset used is a benchmark dataset for binary sentiment analysis with 50,000 movie reviews labelled as positive or negative (1 for positive; 0 for negative).
- https://huggingface.co/datasets/stanfordnlp/imdb
- The 50,000 labelled reviews were combined as a single dataset before splitting into an 80-20 train-test set.

Dataset Processing

Steps	Rationale	Code
1. Check for null values There were no null values.	Remove missing data to ensure data quality	null_counts = dataset.isnull().sum()
2. Drop duplicate reviews	Done at the beginning to prevent redundant operations on identical data points	df = df.drop_duplicates(subset=["text"])
3. Split dataset into train (80%) and test (20%)	Data is split first to ensure a fair comparison between DistilBERT, which can work on unprocessed data, and traditional ML models, which require preprocessing.	X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

Data Preprocessing (Logistic Regression and SVM)

These 6 steps are applied as a preprocessing pipeline:

Steps	Rationale	Code
4a. Remove hyperlinks	Remove irrelevant tokens	text = re.sub(r'http\S+', '', text)
b. Remove HTML tags	Remove noise that interfere with subsequent processing steps like contraction expansion and emoticon conversion	text = re.sub(r'<.*?>', '', text)
c. Expand Contractions	For better tokenization. Done before emoticon conversion ensures clear separation of words and emoticons.	text = contractions.fix(text)
d. Convert emoticons to text	Preserve sentiment-rich information as text	emoticons_dict = emot_obj.emoticons(text)
e. Remove non-ASCII characters	Special symbol irrelevant to sentiment analysis removed to reduce errors.	text = "".join(char for char in text if ord(char) < 128).strip()
f. Convert lowercase	Ensure consistency across tokens. Done later in this sequence to ensure no interference with emoticon conversion or contraction expansion which may be case-sensitive	text = text.lower()

Data Preprocessing (Logistic Regression and SVM)

Steps	Rationale	Code
5. Lemmatization using spacy	To reduce words to their base forms	Load SpaCy model, apply lemmatize_texts() with nlp.pipe() on dataset, then save the resulting CSV.
6. Stopword removal	Further refining the text by removing uninformative words. Done after lemmatization as stopword list are in base form	Load SpaCy model, apply remove_stopwords() using nlp.pipe() on the lemmatized dataset, then save the filtered CSV.
7. Feature extraction using Tf-idf	TF-IDF was selected for feature extraction as it emphasizes important words by balancing term frequency and document relevance, reducing noise in the data.	tfidf_vectorizer = TfidfVectorizer(max_features=5000) X_tfidf = tfidf_vectorizer.fit_transform(X)

Traditional ML (Logistic Regression)

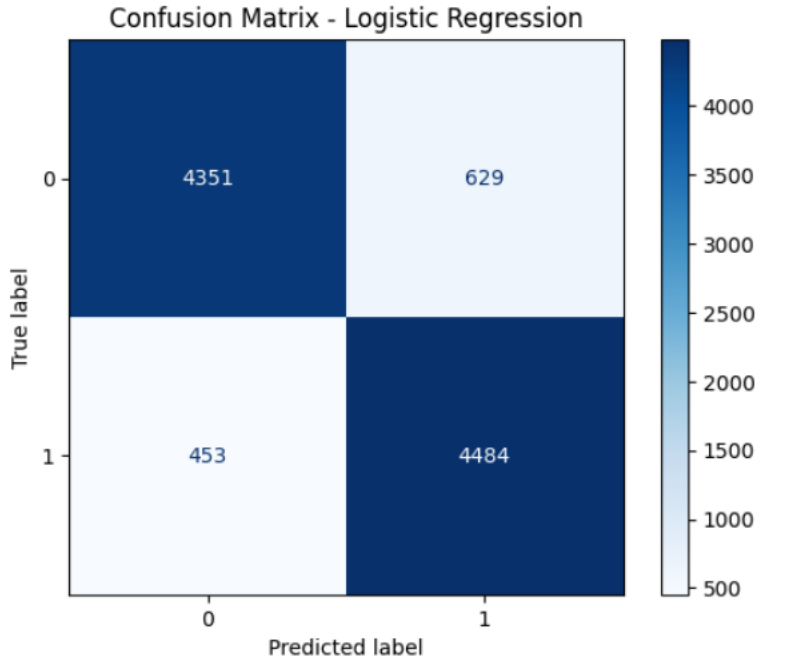
- Model selected as it can perform binary classification and is efficient.
- Trained with labels (positive/negative sentiment) as the target.
- model = LogisticRegression(max_iter=1000, random_state=42)

Traditional ML (Logistic Regression)

Logistic Regression Classification Report:

	precision	recall	f1-score	support
0	0.91	0.87	0.89	4980
1	0.88	0.91	0.89	4937
accuracy			0.89	9917
macro avg	0.89	0.89	0.89	9917
weighted avg	0.89	0.89	0.89	9917

Logistic Regression Test Report saved to: /content/drive/MyDrive/SPNLI



Test Results

- Balanced performance with an accuracy, precision, recall and F1-score of 0.89 for both classes
- This indicates it can handle both positive and negative sentiment prediction well.

Note: 1 for positive; 0 for negative

Traditional ML (SVM)

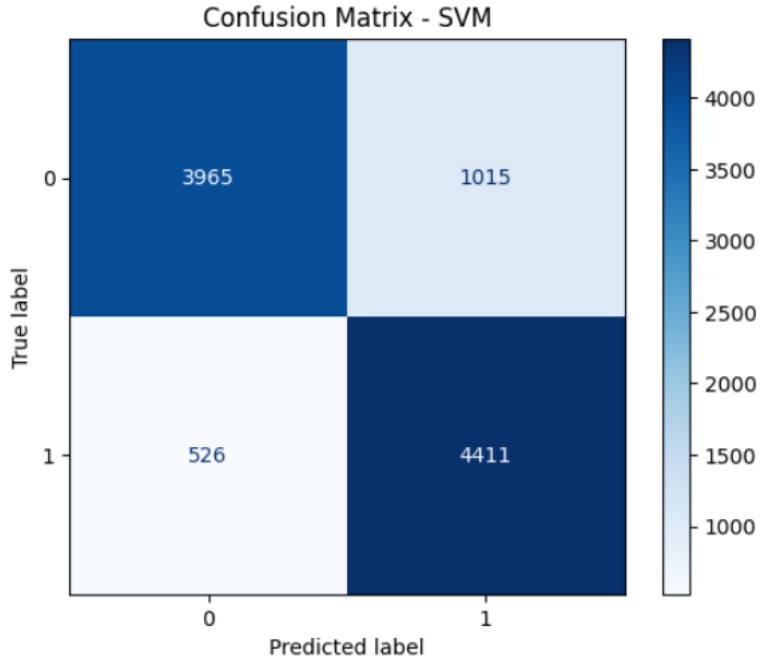
- Model selected as it can work with high dimensional features. Sentiments are inherently complex/non-linear, and SVMs can capture non-linear decision boundaries.
- SVM is sensitive to scale of features and is scaled using StandardScaler.
- Trained using default parameters (Kernel: 'rbf‘, C: 1.0, Gamma: 'scale‘)
- svm_model = SVC()

Traditional ML (SVM)

SVM Classification Report:

	precision	recall	f1-score	support
0	0.88	0.80	0.84	4980
1	0.81	0.89	0.85	4937
accuracy			0.84	9917
macro avg	0.85	0.84	0.84	9917
weighted avg	0.85	0.84	0.84	9917

SVM Test Report saved to: /content/drive/MyDrive/SPNLP/FINAL//svm_test



Test Results

- Balanced performance with an accuracy, precision and recall in the range 0.84 – 0.85 for both classes
- F1-score is 0.84 for both classes
- Results are slightly worse than logistic regression in classifying both sentiments

Note: 1 for positive; 0 for negative

Deep Learning with DistilBERT

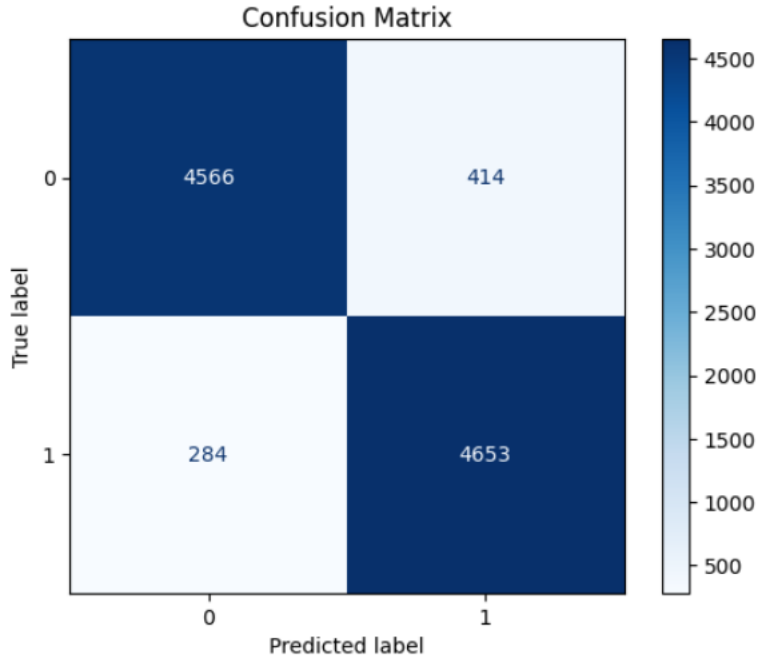
- DistilBERT training and evaluation uses raw text.
- Use PyTorch DataLoader for batching (batch size of 16)
- model = DistilBertForSequenceClassification.from_pretrained(model_name, num_labels=2)
- Optimizer used is AdamW for fine-tuning
- Trained for 3 epochs with loop to tokenize each batch (convert text into token IDs with padding, truncation, and max length of 512), forward pass (compute loss of prediction) and backward pass (update model weights using backpropagation).
- Loss decreased from 0.293 to 0.082, showing the model's improved performance in binary sentiment classification.

Deep Learning with DistilBERT

Classification Report:

	precision	recall	f1-score	support
0	0.94	0.92	0.93	4980
1	0.92	0.94	0.93	4937
accuracy			0.93	9917
macro avg	0.93	0.93	0.93	9917
weighted avg	0.93	0.93	0.93	9917

Accuracy: 92.96%



Test Results

- Balanced performance with an accuracy, precision, recall and F1-score of 0.93 for both classes
- Performance is better than traditional models due to its ability to adapt contextual embeddings to sentiment analysis.
- Best performance out of all models.

Note: 1 for positive; 0 for negative

Evaluation of all models

Model	Accuracy	Precision	Recall	F1 Score
1. Logistic Regression	0.84	0.84	0.84	0.84
2. Support Vector Machine	0.84	0.85	0.84	0.84
4. DistilBERT (raw text)	**0.93**	**0.93**	**0.93**	**0.93**

- DistilBERT without any preprocessing outperforms the traditional models which highlights its robustness in handling raw text and adapt contextual embeddings.