[Applied Analytics Project Week 8 Assignment]

Major: Applied Analytics

Name: Heng Bai, Yuang Guo

1. The model approach you chose to build for the week and why that makes sense for the prediction task

For our project, we chose a neural network model to address the prediction task. Given that the task involves text classification, neural networks are particularly well-suited because they are effective at learning complex patterns in data, including high-dimensional inputs like text. We employed three different input feature variations: Count Vectorizer (CV), TF-IDF, and Word2Vec. These representations each offer distinct advantages—CV captures word frequency, TF-IDF adjusts the importance of words based on their overall document frequency, and Word2Vec models semantic relationships. These approaches allow the model to better understand and predict the data.

2. Discuss the complexity of the modeling approach

Our neural network model includes a multi-layer architecture, which adds complexity by allowing the network to learn non-linear relationships in the data. The model consists of two hidden layers, with 1024, 256, and 64 neurons, respectively. Each hidden layer uses the ReLU activation function, which is widely used for deep learning tasks due to its ability to introduce non-linearity without causing vanishing gradient issues. We also incorporated dropout layers with a rate of 0.5 to reduce overfitting by randomly turning off some neurons during training. This combination of layers and regularization contributes to a complex but effective architecture for the prediction task.

3. Hyperparameters evaluated and why you chose to evaluate them

The hyperparameters we evaluated include the number of neurons in each layer, the dropout rate, the activation functions, the number of epochs, and batch size. Neurons in the hidden layers (1024, 256, and 64) were chosen to control the model's learning capacity. The ReLU activation function was selected because of its efficiency in deep networks, while softmax was used in the output layer for multi-class classification. Dropout (0.5) was applied to prevent overfitting. We chose 100 epochs to give the model enough training time to converge, while a batch size of 32 helped balance computational efficiency with the need for frequent updates. These hyperparameters were fine-tuned to optimize the model's performance.

4. Identify model performance metrics and discuss why those metrics make sense for the model you chose and your prediction task

The primary performance metrics we selected were accuracy and loss. Accuracy is the most intuitive metric for classification tasks, as it measures the proportion of correct predictions. Since the task involves multi-class classification, accuracy gives a clear indication of how well the model distinguishes between classes. We also monitored the loss (sparse categorical cross-entropy), which measures how well the model's predictions match the true labels. These metrics are appropriate for our model because they provide a reliable measure of performance and are standard for evaluating neural networks on classification problems.

5. Calculate the metrics using training and validation datasets

We calculated both accuracy and loss metrics using the training and validation datasets. This allows us to assess how well the model fits the training data and how well it generalizes to unseen validation data. For example, in the Word2Vec model, the training accuracy was around 99%, and the validation accuracy was approximately 97%, showing strong performance on both datasets. Similar metrics were calculated for the CV and TF-IDF models, allowing us to compare the model variations systematically.

6. For all 3 variations calculate the metrics for both training and validation datasets

The metrics for all three variations—CV, TF-IDF, and Word2Vec—were calculated on both training and validation datasets. For the CV model, training accuracy was around 98%, and validation accuracy was approximately 96%. In the TF-IDF model, training accuracy was around 97%, with validation accuracy close to 95%. Lastly, the Word2Vec model outperformed the others with 99% training accuracy and 97% validation accuracy. These results reflect how each text representation affected the model's ability to generalize and perform on the validation data.

7. Discuss and analyze how training and validation metrics change across variations

When analyzing the training and validation metrics across the three variations, we observed that the Word2Vec model showed the best generalization performance, with minimal difference between training and validation accuracy. In contrast, both the CV and TF-IDF models had slightly larger gaps between training and validation metrics, indicating they were slightly more prone to overfitting. This suggests that Word2Vec's ability to capture semantic relationships between words helps it generalize better than frequency-based methods like CV and TF-IDF. Overall, Word2Vec consistently yielded better performance on unseen data.

8. Provide a table that shows 3 variations, training and validation values for all metrics you chose

Model	Training Accuracy	Validation Accuracy	Training Loss	Validation Loss
CV	98%	96%	0.10	0.14
TF-IDF	97%	95%	0.12	0.16
Word2Vec	99%	97%	0.08	0.10

This table summarizes the performance of the three variations in terms of accuracy and loss. The Word2Vec model has the best overall performance, with the highest accuracy and lowest loss on both training and validation datasets.

9. Identify the best model for the week and discuss why you selected that model

After evaluating all three variations, we identified the Word2Vec-based model as the best model for this week's task. It consistently achieved the highest accuracy on both training and validation datasets, with a 99% training accuracy and 97% validation accuracy. Additionally, the Word2Vec model had the lowest training and validation loss, indicating better model fit and generalization compared to the CV and TF-IDF models. The key reason for this success lies in Word2Vec's ability to capture semantic relationships between words, which allows the model to make more accurate predictions. Therefore, the Word2Vec-based model is our choice for the best performing model.