

**PAI Theory Assignment # 3**

**THE SUPERIOR UNIVERSITY**

**Submitted To:**

Sir Anser

**Name:**

Qaswa Chaudary

**Roll No:**

SU92-BSDSM-F23-002

**Section:**

BSDS-4A

**Sentiment Classification Using CNN - Code Documentation**

# 1. Importing Required Libraries

* Essential libraries are imported:
* **pandas, numpy:** Data manipulation.
* **matplotlib:** Plotting (though not used in current code).
* **sklearn:** Data preprocessing, feature extraction, evaluation.
* **nltk**: Natural Language Toolkit for text processing.
* **tensorflow.keras:** Building and training the deep learning model.

# 2. Downloading NLTK Resources

Downloaded three things from nltk:

* **stopwords:** Common words to ignore (e.g., “the”, “and”).
* **wordnet:** Needed for lemmatization.
* **punkt:** Tokenizer used for splitting text into words.

# 3. Load the Dataset

df = pd.read\_csv('dataset.tsv', delimiter='\t')

The dataset is a tab-separated .tsv file, likely with columns like review and sentiment.

# 4. Explore Dataset

**df.shape:** Checks number of rows and columns.

**df.isnull().sum():** Checks for any missing values.

# 5. Text Preprocessing

Following Preprocessing steps are applied during model training:

* Conversion to lowercase.
* Removal of punctuation.
* Tokenization
* Stopword Removal.
* Lemmatization
* Vectorization using TF-IDF vectorizer
* Label Encoding

# 6. Train-Test Split

* Splits the dataset:
* 80% training
* 20% testing
* Random state 42 ensures reproducibility.

# 7. CNN Model Architecture

1D Convolutional Neural Network (CNN) using Keras with following architecture:

* **Embedding Layer:** Converts words into 128-dimensional vectors.
* **Conv1D Layer**: Applies filters to detect local patterns in text.
* **GlobalMaxPooling1D:** Extracts the most important features from each filter.
* **Dropout:** Prevents overfitting by randomly dropping connections.
* **Dense Layer:** Fully connected layer with 10 neurons.
* **Output Layer:** Number of neurons equals number of sentiment classes, with softmax for classification.

# 8. Compile & Train Model

* **Loss: sparse\_categorical\_**crossentropy for multi-class classification with integer labels.
* **Optimizer:** Adam.
* **Metric:** Accuracy.
* **EarlyStopping** is used to avoid overfitting (stops training if validation loss doesn't improve for 3 epochs).

# 9. Model Evaluation

Model Evaluation is done on the following bases

* Predicts sentiment on test data.
* Converts predictions to class labels using argmax.
* Prints accuracy score.
* Prints class names.

# 10. Save Model & Tokenizer

* Model has been saved in .h5 format for future use.
* Tokenizer has been saved (word index) using pickle.