Sentiment Analysis Workshop – Customer Feedback Classification

Introduction

In this interactive workshop, participants will explore how to leverage AI for sentiment analysis. Through a hands-on approach leveraging Cursor, attendees will learn to preprocess text data, build and train a basic sentiment analysis model, and evaluate its performance. The workshop is designed to demonstrate how to pull actionable insights from customer reviews and emphasizes practical applications. By the end, participants will leave with a functional model and the skills to refine and adapt it for real-world business challenges, enabling enhanced decision-making.

Requirements

1. Install Conda
   1. Download and install one of the options below for Conda (Miniconda or Anaconda) if not already installed:

Option 1.) Miniconda (lightweight option): <https://docs.conda.io/en/latest/miniconda.html>

Option 2.) Anaconda (includes more tools): <https://www.anaconda.com/>

Option 3.) Alternatively, you can install Python directly, but this won’t have the dependencies loaded.

1. Install Cursor
   1. <https://www.cursor.com/downloads>
2. Download zip files from GitHub. The main file is customer\_feedback.csv, and the requirements.txt has all necessary packages. All files are located at [bit.ly/3PIK4RU](https://bit.ly/3PIK4RU).

# Outline

1. Introduction to Sentiment Analysis

* Definition, Business Value, and Examples.

2. Data Overview and Setup

* Dataset Description and Characteristics.
* Initial Setup in Cursor.

3. Data Cleaning

* Removing Duplicates and Handling Missing Values.
* Preprocessing Text (lowercase, punctuation removal, stop words, lemmatization).

4. Data Visualization

* Visualizing Sentiment Distribution.

5. Model Training

* Training
* Evaluating with metrics (accuracy, precision, recall, F1-score).

6. Model Adjustment

* Fine-tuning hyperparameters and saving the best model.

7. Deployment

* Deploying a FastAPI application for sentiment prediction.

8. Testing

Validating the API and resolving issues.

9. Wrap-Up, Q&A, and Takeaways

Recap of Key Steps

* Introduction: Covered the concept, business value, and examples of sentiment analysis.
* Data Preparation: Loaded, cleaned, and preprocessed the dataset for training.
* Model Development: Trained and evaluated a Hugging Face sentiment model with key metrics.
* Deployment: Built and deployed a FastAPI application for real-time predictions.

Q&A

* Misclassification: Refine preprocessing, adjust the dataset, or tweak hyperparameters.
* Handling Longer Text: Increase the sequence length during tokenization.
* Improving Accuracy: Use data augmentation, advanced models (e.g., DeBERTa), or ensembles.

Takeaways

* Skills Gained: Text preprocessing, model fine-tuning, and API deployment.

Key Insights:

* High-quality data preparation is crucial.
* Combining Positive and Neutral classes simplifies classification but reduces detail.
* Deployment enables practical applications like live sentiment monitoring.

Next Steps

* Advanced Techniques: Optimize hyperparameters with tools like Optuna.
* Real-World Use: Apply the model to specific tasks like support ticket analysis or product reviews.

# Introduction to Sentiment Analysis

Sentiment analysis is a natural language processing (NLP) technique used to determine the emotional tone behind text data. It classifies text into categories such as positive, negative, or neutral, helping businesses and individuals understand opinions at scale.

Sentiment analysis provides businesses with actionable insights by analyzing customer feedback, reviews, and social media content. It helps identify trends in customer satisfaction, prioritize issues, improve products or services, and enhance customer experience. Automating sentiment analysis saves time and resources while delivering valuable insights.

Examples of sentiment analysis include evaluating customer reviews on e-commerce platforms, monitoring social media sentiment about a brand, and categorizing support tickets by tone to prioritize urgent issues. For instance, a company can use sentiment analysis to gauge customer responses to a new product launch or identify dissatisfaction trends in service feedback.

# Data Overview and Project Setup

The dataset contains 1,000 rows of customer feedback data. It includes three columns:

**Review\_ID**: A unique identifier for each review.

**Review\_Text**: The actual text of the customer review, describing their opinions or experiences.

**Sentiment**: The sentiment classification of the review, labeled as "Positive," "Negative," or "Neutral."

**Key Characteristics**:

The dataset has a mix of sentiments to provide a balanced training dataset for sentiment analysis models.

* It includes intentional duplicates and missing values to simulate real-world data cleaning challenges:
* Duplicates: Some reviews are repeated, requiring removal during preprocessing.
* Missing Values: A few entries in the Review\_Text and Sentiment columns are missing, necessitating handling for robust analysis.

## Project Steps

1. Complete the quick start steps in Sentiment Analysis Workshop Quick Start.docx.
2. If you haven’t set up the conda environment: Open a command prompt, navigate to the directory with the downloaded files using the command cd and type:

conda env create -f environment.yml

conda activate workshop\_env

A grey rectangular sign with white text

Description automatically generated

1. Open Cursor.
2. File > Open Folder – open the folder containing the csv.
3. To open the chat in Cursor, press **Ctrl + L (on Windows)** or **Command + L (on Mac).**

A screenshot of a computer

Description automatically generated

1. In the chat type:

Create a python jupyter notebook to load customer\_feedback.csv

A screenshot of a computer program

Description automatically generated

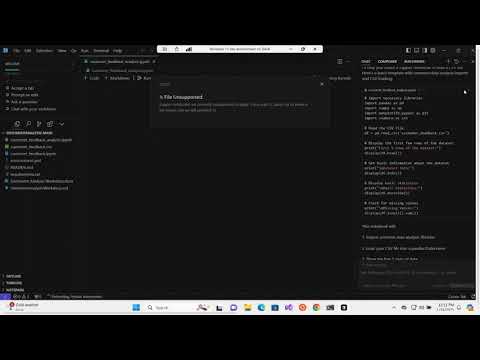
1. Click the Select Kernel button, then choose your conda python environment (workshop\_env).

A screenshot of a computer

Description automatically generated



Note: A terminal is also available in the terminal menu within Cursor.

[](https://www.youtube.com/embed/aChxzCWhYUY?feature=oembed)

<https://youtu.be/aChxzCWhYUY>

1. Save the file with **Ctrl + S**.
2. Click the **play** button to run the cell and see the output.

A screen shot of a computer

Description automatically generated

1. Scroll to the bottom of the cell output and click the **+ Code** button.

A black rectangular object with white lines

Description automatically generated

1. Add an additional code block and ask Cursor to clean up the data.

Write Python code to replace the existing data:

- Remove duplicate rows from the dataset and keep the first occurrence.

- Drop rows where "Review\_Text" or "Sentiment" columns have missing values.

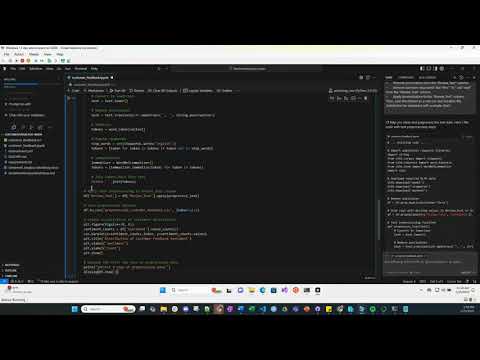
- Convert all text in the "Review\_Text" column to lowercase.

- Remove punctuation from the "Review\_Text" column.

- Remove common stop words like "the," "is," and "and" from the "Review\_Text" column.

- Apply lemmatization to the "Review\_Text" column.

Then, save the dataset as a new csv and visualize the distribution by sentiment with a simple chart.

[](https://www.youtube.com/embed/6DliH4buK6o?feature=oembed)

<https://youtu.be/6DliH4buK6o>

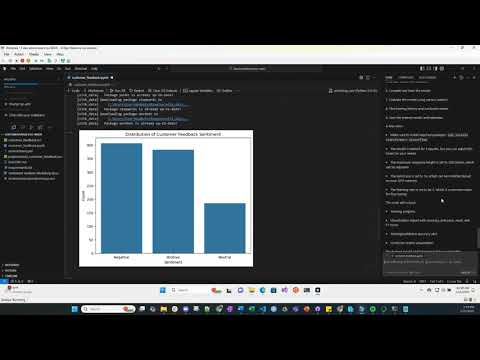
1. Ask cursor to:

Use Hugging Face Transformers and tensorflow to train a sentiment classification model:

- Model: distilbert-base-uncased-finetuned-sst-2-english.

- Group Positive and Neutral sentiments together.

- Provide metrics to evaluate: Accuracy, Precision, Recall, F1-score.

[](https://www.youtube.com/embed/v96I9n8_5vI?feature=oembed)

<https://youtu.be/v96I9n8_5vI>

1. Ask cursor ‘how might I adjust the model to capture more of the positive values’ or ‘is this model overfitting.’

Remember to:

* Adjust the batch size based on your available memory
* Experiment with different learning rates
* Consider using a validation set during fine-tuning
* Monitor for overfitting
* Save the best model weights during training

1. Ask cursor to deploy the code (Ensure this is using the same version of python. This will also give you the option to create the files directly.):

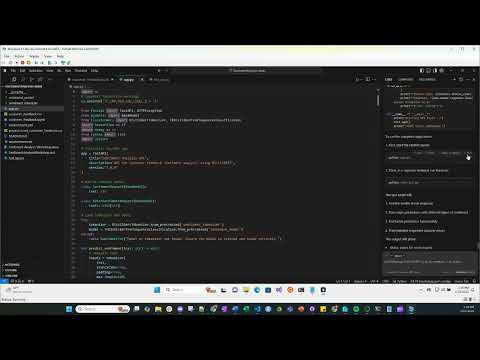
Deploy a simple FastAPI application for sentiment analysis with:

1. TensorFlow integration.

2. Environment variables to suppress TensorFlow warnings.

3. Endpoint: "/predict" to classify sentiment.

4. A test script.

[](https://www.youtube.com/embed/HVagt427NzE?feature=oembed)

<https://youtu.be/HVagt427NzE>

1. Run the test. When running commands in the terminal such as python app.py select an interpreter in the lower right corner.

A computer screen shot of a program

Description automatically generated

# Conclusion

During this project, we explored the end-to-end process of sentiment analysis, from understanding its concept and business value to building and deploying a real-time application. We started by introducing sentiment analysis as an NLP technique to categorize text as positive, negative, or neutral, with applications like monitoring customer feedback or prioritizing support tickets. After preparing the dataset by cleaning, preprocessing, and balancing the sentiment categories, we trained a Hugging Face distilbert-base-uncased-finetuned-sst-2-english model and evaluated its performance using metrics such as accuracy, precision, recall, and F1-score. Finally, we deployed the model using FastAPI, creating a functional API capable of providing real-time sentiment predictions.

The Q&A session addressed common challenges and solutions, such as handling misclassified data by refining preprocessing steps or adjusting hyperparameters. For longer text inputs, extending the sequence length during tokenization ensures the model captures the full context. Suggestions for improving accuracy included leveraging advanced models like DeBERTa, experimenting with data augmentation, and combining models through ensemble techniques. These adjustments offer practical ways to enhance the model’s performance and adaptability to various use cases.

Key takeaways from this project include the importance of high-quality data preparation, which directly impacts the effectiveness of machine learning models. Additionally, grouping sentiment categories, such as combining Positive and Neutral classes, simplifies classification but may reduce granularity. By completing this workflow, you’ve gained practical skills in data preprocessing, model fine-tuning, and API deployment, enabling you to apply sentiment analysis to real-world tasks like customer feedback analysis, support ticket categorization, or product review monitoring. Moving forward, exploring hyperparameter optimization, retraining with real-world data, and expanding use cases will further refine and extend the value of your sentiment analysis capabilities.