Customer sentiment and trend analysis for Amazon, Yelp, IMDB, Twitter

Team members

Mohamed Talaat Abo Elftouh

Mohamed Mostafa Abdelhamed

Moaz Mohamed Tawfik

Amr Khaled Mostafa

Mahmoud Mohamed Abdelmawgoud

Mohamed Alaa Elsayed

Supervisor

ENG: Basma Reda

BY:

Digital Egyptian pioneers initiative (DEPI)

Overview:

The project analyzes customer sentiment and trend analysis of IMDB reviews, Twitter tweets and comments on various products, YELP reviews and amazon product reviews and ratings.

The project includes preprocessing and filtering data and using different machine learning models like: Random Forest, logistic regression, K-Nearest Neighbors (KNN), XGBoost and AdaBoost, Gradient Boosting on the data after cleansing the data and using deep learning models like: Long Short-Term Memory (LSTM) and BERT (Bidirectional Encoder Representations from Transformers), Multi-Layer Perceptron (MLP), Gated Recurrent Unit (GRU).

The project implements platforms like: Hugging face, MLflow and Streamlit, for model optimization tracking and deployment.

The project uses libraries like pandas, NumPy, matplotlib, seaborn, NLTK, Scikit-learn, TensorFlow to preprocess and analyze and clean the data and make operations and visualize the data, and training machine learning models and deep learning models on the data

Table of contents

Introduction

Customer sentiment and trend analysis is a critical component of modern business strategies and provides valuable insights into how customers perceive a brand, product or a service and tracking these sentiments enable the business to make data-driven decisions and improve customer satisfaction and it's also critical and very important because it allows us to improve customer experience through: analyzing negative sentiment and try to improve the customer experience regarding this and also enhancing positive experience and identify what customers appreciate the most and try to focus more on it and maintain its quality, it also allows the business to improve the product quality and keep up with trends through understanding the customer sentiment and trend analysis and it allows also for effective and successful marketing campaigns.

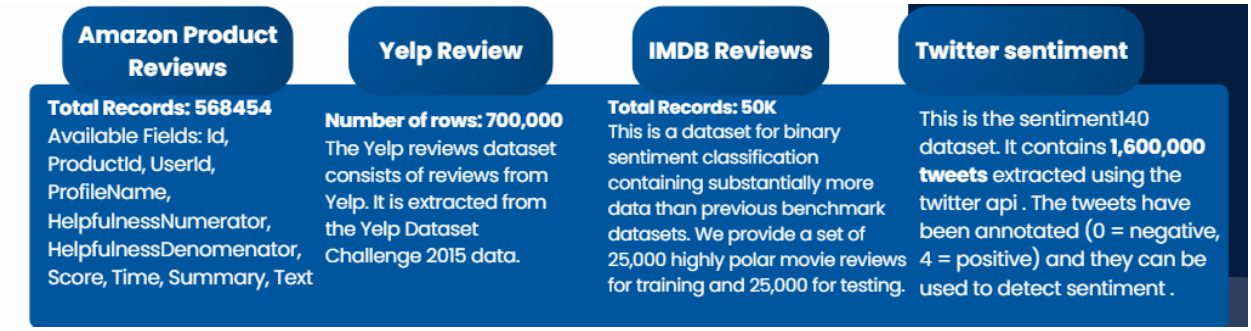
We made our project to fill this need as our project's objective is to understand customer sentiment trends over time for different organizations and analyze the trends over time for these companies to identify customer desires interests as we have collected reviews and comments for Amazon, IMDB, YELP, Twitter over different time periods and used it to understand customer interests and desires in different organizations in different periods of time.

Our project implements and uses different machine learning and deep learning techniques to analyze and understand customer behavior and sentiment across different platforms.

Data collection and preprocessing

Data sources

In our project we have collected data from different sources which are: Twitter, Amazon, YELP and IMDB. We have collected reviews for amazon products with about 500,000 records and collected reviews from YELP with about 700,00 records and we collected reviews from IMDB with about 50K records and finally we have collected reviews from twitter with about 1.6 million records extracted using Twitter API



The photo shows the fields, and the number of records collected from each source

Collection methodology

We have collected different types of data as follows:

YELP: we have collected five-star reviews and collected around 140,000 samples for each star divided into 130,000 samples for training and 10,000 samples for testing, which in total 700,000 samples divided into 650,000 samples for training and 50,000 samples for testing

IMDB: we have collected a dataset which contains 50,000 samples of data which are divided into 25,000 samples for training and 25,000 samples for testing

Twitter: we have collected from Twitter around 1.6 million records and we have chosen a certain sample and certain categories to work on from this data, and we have used twitter API for doing this

Amazon: Finally in amazon we have collected around 500k samples of data they were labeled from 1 to 5 we divided them into (3,4,5) for positive sentiment and (0,1,2) for negative sentiment

Preprocessing

We used techniques such as tokenization, normalization, and removal of stop words and special characters, duplicated words, converting characters to lower case, converting words to their base form, joining tokens and that aren't needed in the data to preprocess and clean the data and make it ready to train our models

Examples of our code and results:

```
# Initialize lemmatizer and stop words
lemmatizer = WordNetLemmatizer()
stop_words = set(stopwords.words('english'))
```

```
# Regular expressions for cleaning
    url_pattern = re.compile(r"(?:\@|https?\://)\S+|[^\w\s#]")
    repeat_pattern = re.compile(r"(.)\1{2,}")  # Matches characters
repeated 3 or more times
```

```
# Lemmatize, remove stopwords, and filter token length
    lemmatized_tokens = [lemmatizer.lemmatize(word) for word in
expanded_words if word not in stop_words and len(word) > 1]
```

Methodology

Sentiment analysis methods

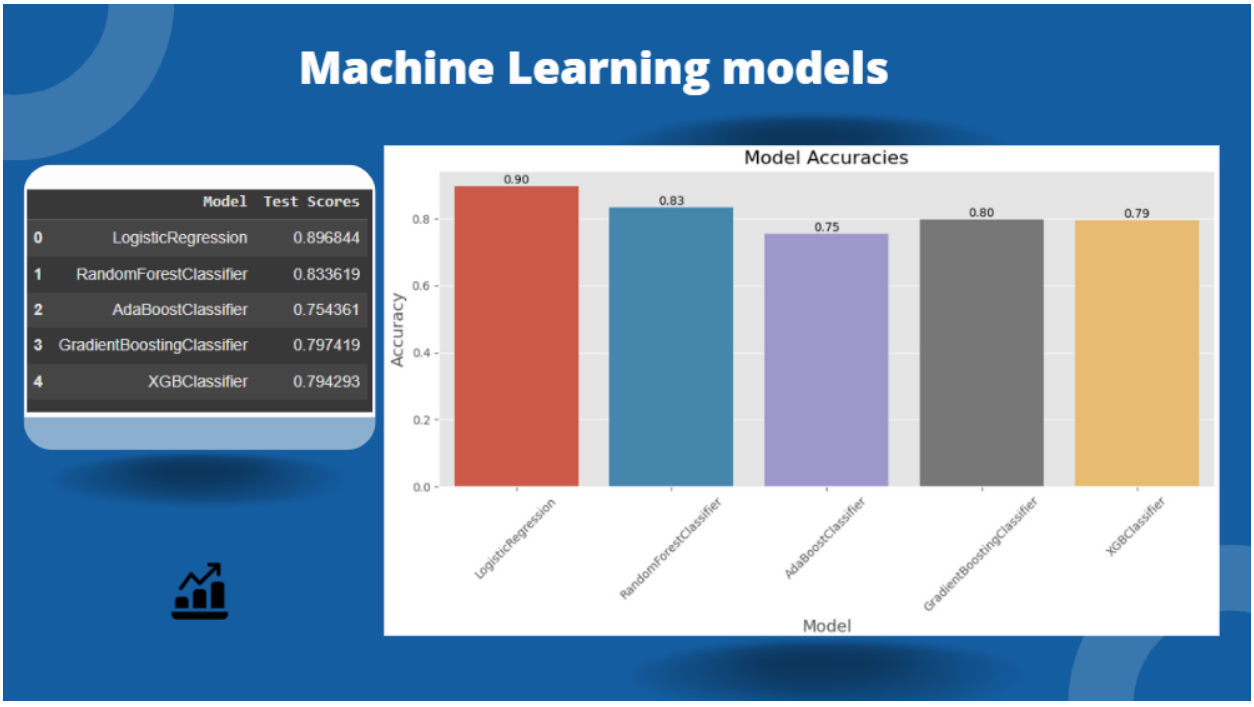
We have used various techniques and methods for sentiment analysis such as:

```
# taking a sample from the dataset and classifying it into 2 classes
df = df.groupby('sentiment').apply(lambda x: x.sample(75000,
random_state=42)).reset_index(drop=True)
```

We used also techniques like TF-IDF and glove to assess the importance of words in a document by comparing their frequency in that document to their rarity across a whole corpus, aiding in text analysis and search ranking.

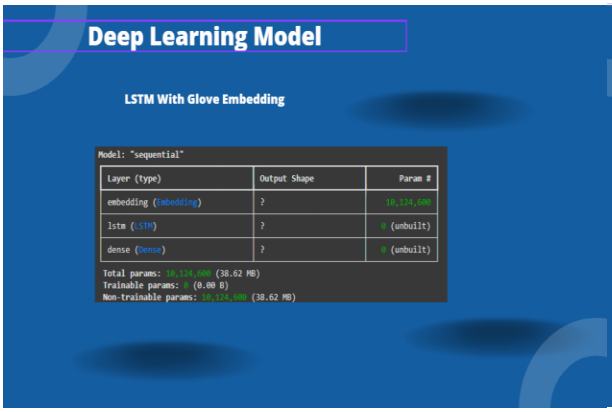
Models

We have tried a lot of machine learning and deep learning models to train and fit to our data and have achieved good results across different datasets

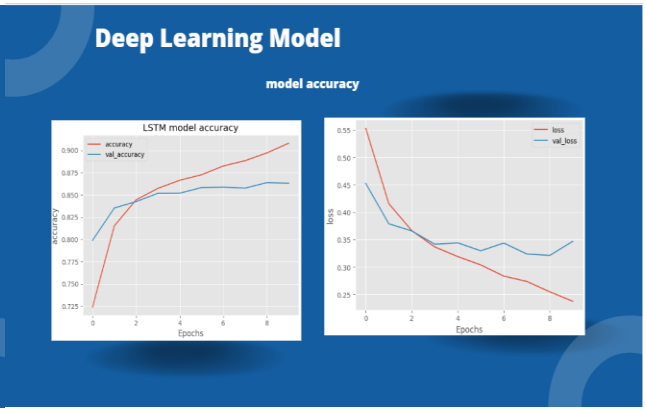


Results of different machine learning models on IMDB reviews

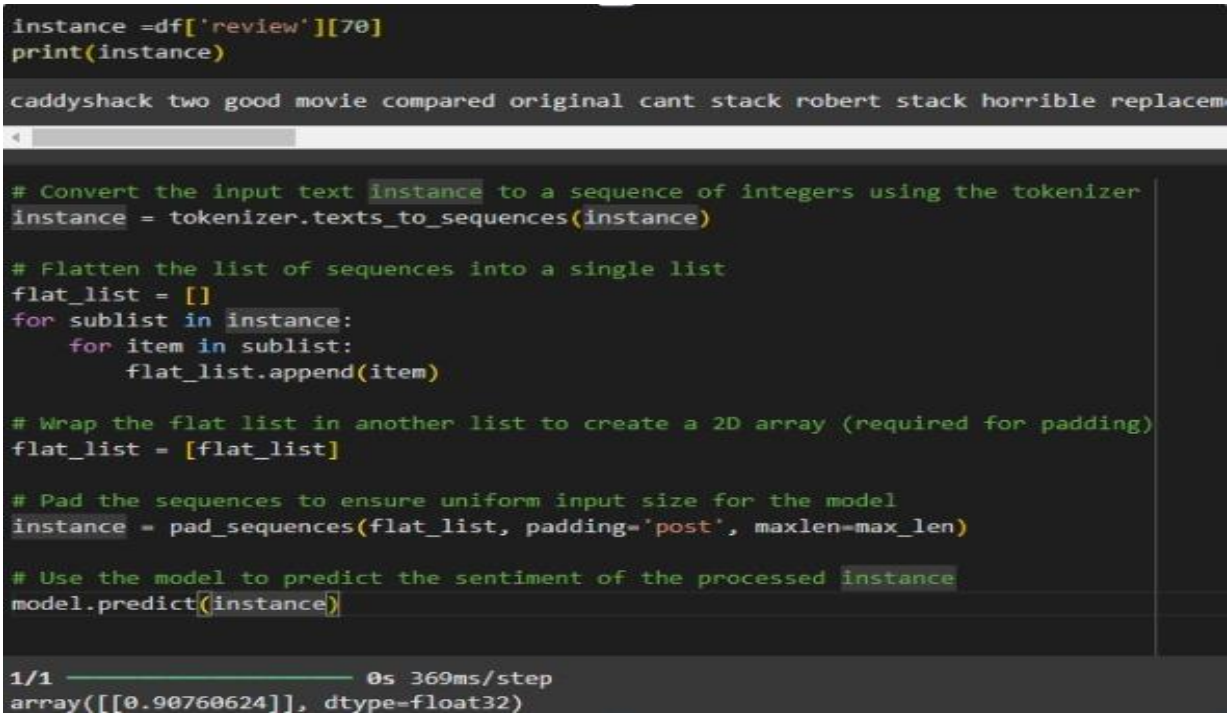
Also, we have used a lot of different deep learning techniques and models like: LSTM, GRU, bidirectional with embedding layer and libraries like: TensorFlow, PyTorch, Keras, matplotlib, pandas, seaborn to implement this



LSTM with Glove embedding



Accuracy of a deep learning model (LSTM)



```
instance =df['review'][70]
print(instance)
```

```
caddyshack two good movie compared original cant stack robert stack horrible replacem
```

```
# Convert the input text instance to a sequence of integers using the tokenizer
instance = tokenizer.texts_to_sequences(instance)

# Flatten the list of sequences into a single list
flat_list = []
for sublist in instance:
    for item in sublist:
        flat_list.append(item)

# Wrap the flat list in another list to create a 2D array (required for padding)
flat_list = [flat_list]

# Pad the sequences to ensure uniform input size for the model
instance = pad_sequences(flat_list, padding='post', maxlen=max_len)

# Use the model to predict the sentiment of the processed instance
model.predict(instance)
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 369ms/step
array([[0.90760624]], dtype=float32)
```

Our model making predictions

MLOps

After we finished training our models and achieved amazing results, we used multiple platforms like: Azure and stream lit to version and deploy our model and automate lifecycle of our ML model and to make our system more reliable and scalable and maintainable.

During MlOps to train and validate multiple versions of a model to ensure accurate tracking and comparison.

Also, MLOps help us to detect different issues in our model after deployment and launch such as: model drift or data drift. Also, MLOps provides frameworks for scaling up model training and deployment efficiently. This includes strategies for distributed training, inference, monitoring and maintenance.

Here are the steps that we have taken to deploy our model:

- Create handle to work space: create ml_client for a handle to the workspace and use ml_client to manage resources and jobs.

```
from azure.ai.ml import MLClient, Input
from azure.ai.ml.entities import (
    BatchEndpoint,
    ModelBatchDeployment,
    ModelBatchDeploymentSettings,
    Model,
    AmlCompute,
    Data,
    BatchRetrySettings,
    CodeConfiguration,
    Environment,
)
from azure.ai.ml.constants import AssetTypes, BatchDeploymentOutputAction
from azure.identity import DefaultAzureCredential
```

- Create training and registering the model as MLFlow model

```
mlflow.sklearn.log_model(
        sk_model=clf,
        registered_model_name=registered_model_name,
        artifact_path=registered_model_name,
```

```
    )

    # Saving the model and vectorizer to files
    model_dir = os.path.join(registered_model_name, "trained_model")
    os.makedirs(model_dir, exist_ok=True)
    mlflow.sklearn.save_model(
        sk_model=clf,
        path=model_dir,
    )
```

- Create Azure ML job

```
from azure.ai.ml import command

# Create the Azure ML job (no parameters passed)
job = command(
    code="./src/",  # The folder containing your main1.py script
    command="python main2.py",
    environment="azureml://registries/azureml/environments/sklearn-1.5/labels/latest",
    display_name="sentiment_analysis_logistic_regression6",
)

# Submit the job
ml_client.create_or_update(job)
```

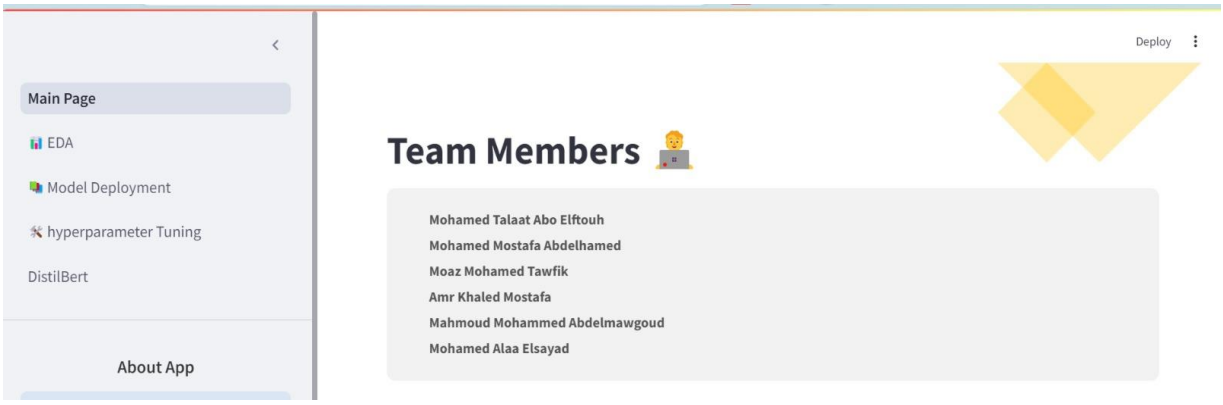
- Create a new Batch endpoint

```
from azure.ai.ml import MLClient
from azure.ai.ml.entities import BatchEndpoint, BatchDeployment

# Create a Managed batch Endpoint
endpoint = BatchEndpoint(
    name="depi-cloud-sentiment-endpoint",
    description="Cloud Endpoint for sentiment analysis using logistic regression",
)

# Create or update the endpoint
ml_client.batch_endpoints.begin_create_or_update(endpoint).wait()
```

- Deploy the model to the endpoint with the latest version of your model

```
deployment = ModelBatchDeployment(
    name="classifier-LogReg",
    description="A Sentiment classifier based on Logistic Regression",
    endpoint_name=endpoint.name,
    model=model,
    compute=compute_name,
    settings=ModelBatchDeploymentSettings(
        instance_count=2,
        max_concurrency_per_instance=2,
        mini_batch_size=10,
        output_action=BatchDeploymentOutputAction.APPEND_ROW,
        output_file_name="predictions.csv",
        retry_settings=BatchRetrySettings(max_retries=3, timeout=300),
        logging_level="info",
    ),
)
```

- Also, we deployed our model on stream lit and and made EDA also on stream lit and this is our model on stream lit after we have deployed it

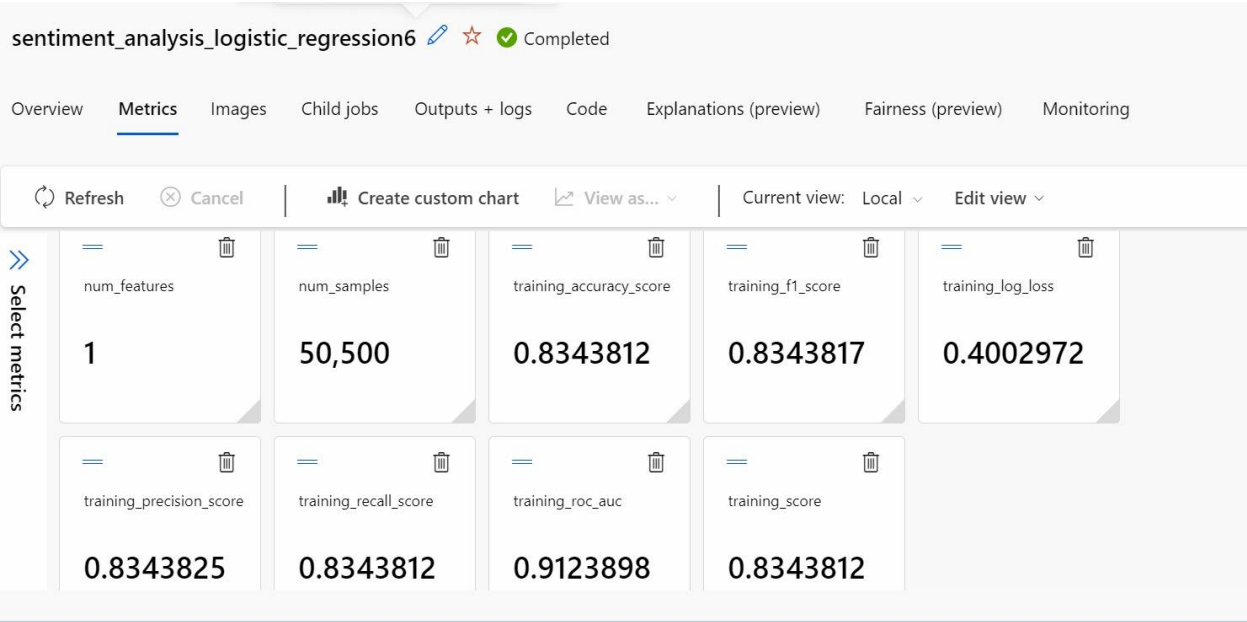


the main page on Streamlit after we deployed our model and made EDA

Conclusion

We have finished our model on our different data including Twitter data and IMDB data and Amazon data and YLP data and trained our models and deployed our model after making sure that it achieves high accuracy on Azure and Streamlit and these some of the results we have achieved after making our customer sentiment and trend analysis machine learning model

- The following photo shows the model and its performance metrics



After we deployed the model on Azure

- Training confusion matrix on azure

- ROC curve on azure

- Streamlit EDA