Real Time Sentiment Analysis using AutoML

Overview

This document presents the technical design for the dummy customer. The demo project utilises the GCP AutoML Service for sentiment analysis of IMDB reviews. The solution aims to provide data-driven analytics, enabling businesses to make informed decisions based on valuable insights extracted from movie reviews. Its concise architecture, data flow, and API integration ensure an efficient and powerful sentiment analysis system.

Contents

1. Code Repository & Dataset

Dataset= https://www.kaggle.com/datasets/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews
Code Repository = https://github.com/Pritam-N/whitepapers/tree/main/automl

2. Use Case

The client operates a platform where users submit reviews for movies and TV series. They are keen on enhancing user engagement and satisfaction by implementing a sentiment analysis system customised to their platform. This system will analyse the sentiment of user reviews to optimise movie recommendations, refine marketing strategies, and provide valuable feedback to producers and directors about audience perceptions.

Historically, the client has relied on standardised sentiment analysis tools or third-party APIs, which lacked the specificity required for their unique dataset. However, they now aim to leverage machine learning techniques to develop a tailored solution capable of accurately analysing sentiments from their dataset. It's worth noting that the client's team lacks expertise in machine learning and natural language processing (NLP), underscoring the importance of a user-friendly solution that can be seamlessly integrated and operated by their staff.

To address this, the proposed solution entails building a machine learning model trained on the client's labelled dataset. By harnessing Google Cloud Platform (GCP) AutoML services, the model will be fine-tuned to predict sentiment from new user reviews effectively. The goal is to provide the client with a user-friendly web application interface, enabling them to input user reviews and promptly receive precise sentiment analysis results.

In essence, the client seeks to implement a bespoke sentiment analysis system that aligns seamlessly with their platform, empowering them with actionable insights derived from user-generated content while mitigating the need for specialised ML expertise within their organisation.

3. Data Exploration

We utilised a dataset from IMDB that comprises movie and TV series reviews for our analysis. This dataset features approximately 50,000 entries across two primary columns: 'review' and 'sentiment.' The 'review' column is an independent variable, consisting of user-submitted critiques in string format. The 'sentiment' column, a dependent binary variable, categorises these reviews as either 'Positive' or 'Negative.'

```
import pandas as pd

df = pd.read_csv('IMDB Dataset.csv')
filtered_df = df[df['review'].str.len() < 500]
filtered_df.head(5)
```

S. No.	Review	Sentiment Obtained
1	If you like original gut wrenching laughter you will like this movie. If you are young or old then you will love this movie, hell even my mom liked it. Great Camp!!!	positive
2	This a fantastic movie of three prisoners who become famous. One of the actors is george clooney and I'm not a fan but this roll is not bad. Another good thing about the movie is the soundtrack (The man of constant sorrow). I recommand this movie to everybody. Greetings Bart	positive
3	What an absolutely stunning movie, if you have 2.5 hrs to kill, watch it, you won't regret it, it's too much fun! Rajnikanth carries the movie on his shoulders and although there isn't anything more other than him, I still liked it. The music by A.R.Rehman takes time to grow on you but after you heard it a few times, you really start liking it.	positive
4	The plot is about the death of little children. Hopper is the one who has to investigate the killings. During the movie it appears that he has some troubles with his daughter. In the end the serial killer get caught. That's it. But before you find out who dunnit, you have to see some terrible acting by all of the actors. It is unbelievable how bad these actors are, including Hopper. I could go on like this but that to much of a waste of my time. Just don't watch the movie. I've warned you.	negative
5	Protocol is an implausible movie whose only saving grace is that it stars Goldie Hawn along with a good cast of supporting actors. The story revolves around a ditzy cocktail waitress who becomes famous after inadvertently saving the life of an Arab dignitary. The story goes downhill halfway through the movie and Goldie's charm just doesn't save this movie. Unless you are a Goldie Hawn fan don't go out of your way to see this film.	negative

3.1. Exploratory Data Analysis

We employed open-source Python libraries, including NLTK, Seaborn, Matplotlib, Scikit-learn, WordCloud, and Plotly, for Exploratory Data Analysis (EDA).

1. Pie Chart: This chart illustrates the distribution of records across different classes in the dataset. By using Matplotlib, we generated a pie chart to show the percentage of records belonging to each class, distinguishing between positive and negative sentiments.

```
# Generate a pie chart to visualize the proportion of 'Positive' and 'Negative' reviews
in the dataset.
sns.set(font_scale = 1.2)
plt.figure(figsize=(6,6))
plt.pie(
        df['sentiment'].value_counts(),
        labels=["Positive", "Negative"],
        autopct='%1.0f%%',
        colors=sns.color_palette('Set2'),
        startangle=90,
        counterclock=False,
        explode=[0.05, 0.05])
plt.show()
```

2. **Box Plot:** A box plot provides a quick visual summary of the variability of values within a dataset, highlighting the median, quartiles, and potential outliers. This visualisation helps us understand the distribution and spread of data points.
3. **KDE Plot:** Kernel Density Estimate (KDE) plots offer a way to visualise the distribution of observations, similar to histograms but smoother. These plots use a continuous probability density curve to represent the data.

3.1.1 Data Pre-processing and Feature Engineering

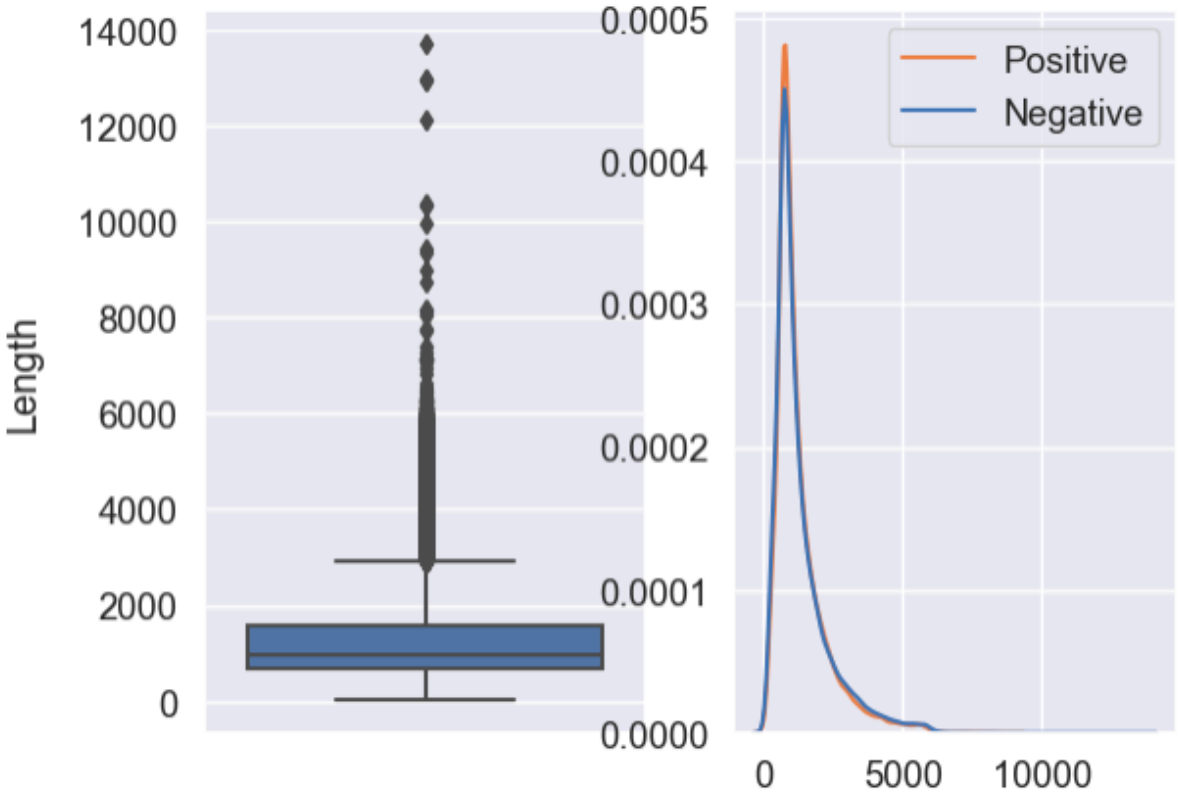
We performed several pre-processing steps to prepare the dataset for analysis:

- Calculating the length of reviews and the count of words per review.
- Determining the average word and sentence length within each review.
- Mapping sentiment ratings to categorical labels (Positive/Negative).
- Cleaning the review texts to remove special characters and stop words, and to lowercase the text.

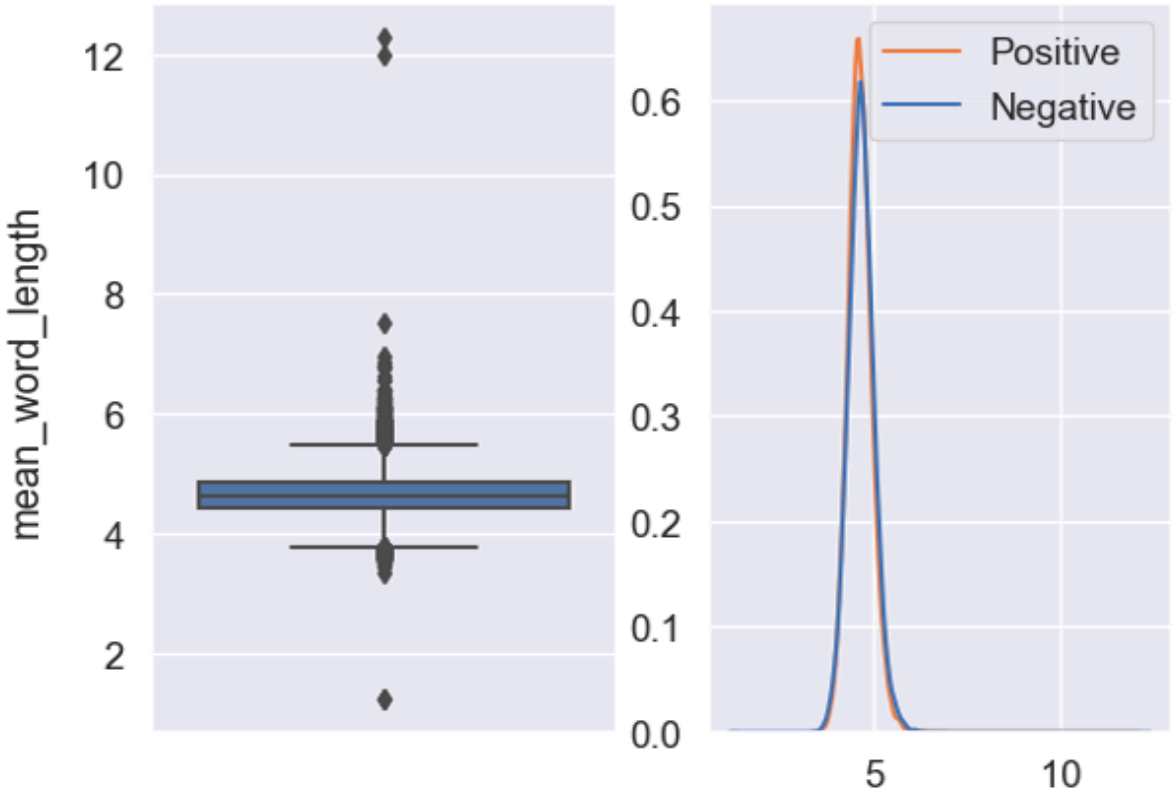
3.1.2 Visualisation of Text Data Features

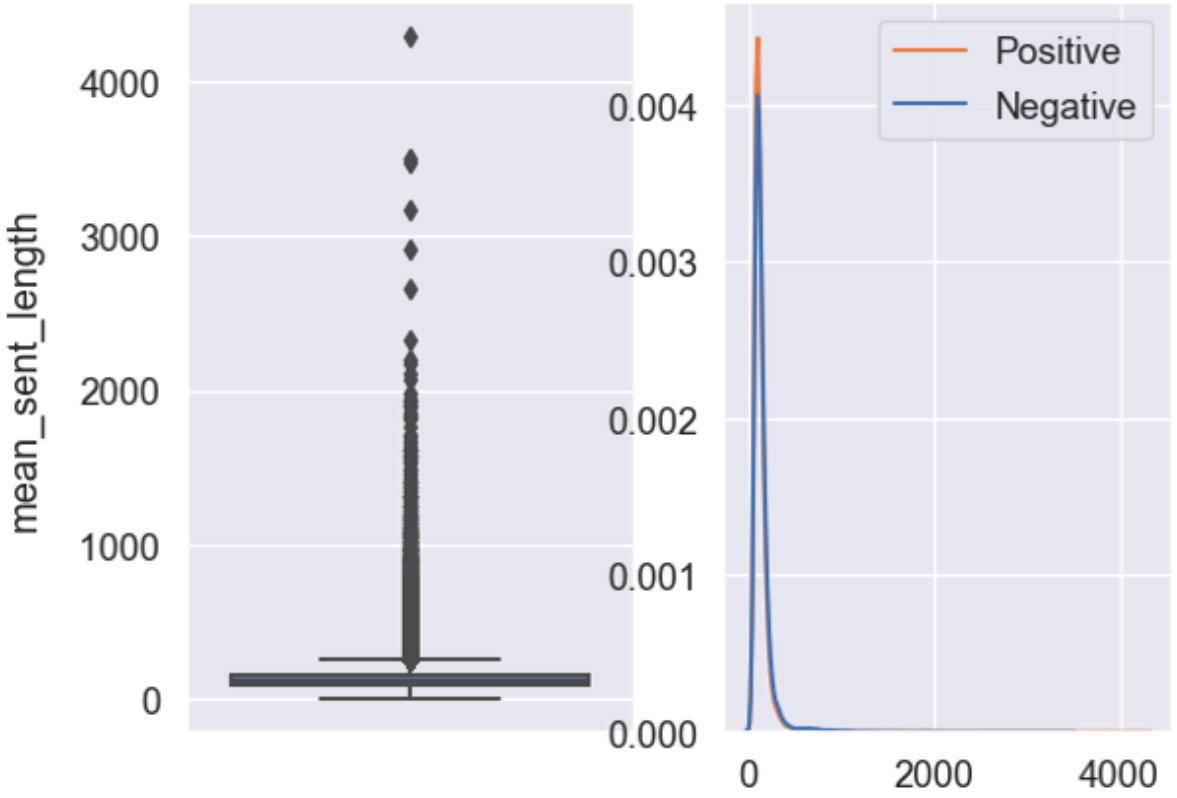
For each engineered feature, we generated visualisations to explore the data further:

- **Distribution of Review Lengths:** Using box plots and KDE plots to examine the variability in character and word counts.

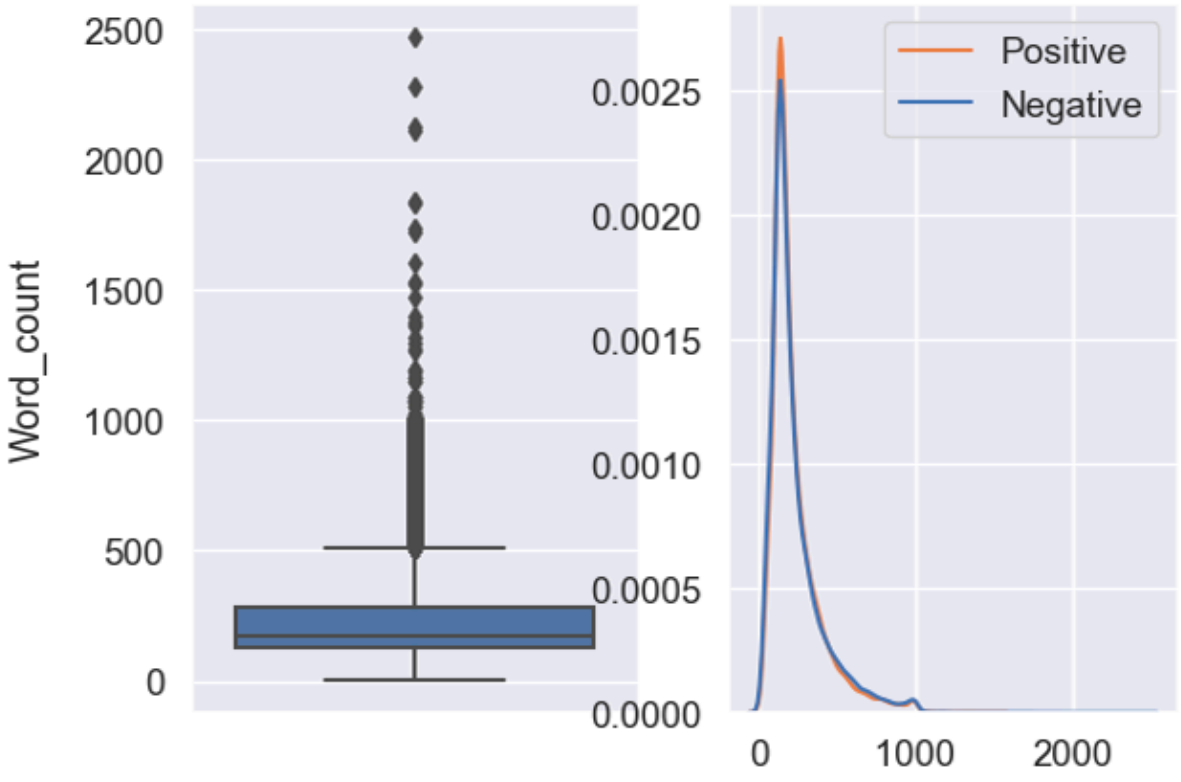


- **Word and Sentence Length Distributions:** Visualising the average word and sentence lengths in reviews to understand text complexity.





- **Frequency Analysis:** Creating bar plots for the most frequently occurring words, bigrams, and trigrams in the dataset. This analysis helps identify common themes and topics.



3.1.3 Word Cloud and N-Gram Analysis

- **Word Clouds:** To visualise the most common words in the dataset, providing a graphical representation of word frequency.

```
wordcloud = WordCloud(
        background_color='white',
        stopwords=stopwords,
        max_words=100,
        max_font_size=30,
        scale=3,
        random_state=1)

wordcloud=wordcloud.generate(str(data))
fig = plt.figure(1, figsize=(12, 12))
plt.axis('off')
plt.imshow(wordcloud)
plt.show()
```

- **Bigram and Trigram Analysis:** Identifying the most common sequences of two and three words to understand common phrases and context within the dataset.

```
df1 = pd.DataFrame(common_words, columns = ['ReviewText' , 'count'])
df1.groupby('ReviewText').sum()['count'].sort_values(ascending=False).plot(kind='bar',
title='Top 20 Bigrams in reviews ')
```

3.1.4 Part-of-Speech Tagging

Analysing the frequency of different parts of speech in the reviews, which helps in understanding the grammatical structure and composition of the text.

3.1.5 Summary of EDA Findings

The EDA revealed that the dataset is balanced in terms of labelled classes, indicating no need for algorithms tailored for imbalanced datasets. The analysis supports the use of a binary classification model. Despite observing outliers in review lengths, these do not impact the chosen model, indicating no requirement for their removal. The exploration of common words, bigrams, trigrams, and parts of speech emphasises the importance of context in text analysis.

3.2 Modelling Decisions

As per our EDA, we need a model that can perform binary classification as the target column has two unique values. It also indicates we need to use NLP techniques that can extract insightful information such as context, emotions, and sentiments from unstructured data.

Before directly implementing the Machine Learning model to capture sentiments, we used Cloud NLP API but the issue with the NLP API output for our use case is, being a generic API, it does not work well with business domain terms which are specific to industry.

On the other hand, AutoML Natural Language uses machine learning techniques to analyze the structure and meaning of documents. We can train a custom machine learning model to classify documents, extract information, or understand the sentiment of authors. A sentiment analysis model inspects a document and identifies the prevailing emotional opinion within it, especially to determine a writer's attitude as positive, negative. It helps us by easily building high quality custom machine learning models that are tailored to our business needs, and then integrate those models into your applications and web sites with limited machine learning expertise needed.
Due to the above reasons, we decided to go with the AutoML services.

4. Feature Engineering

In the fast-evolving field of sentiment analysis, the efficiency of model training is paramount. Traditional machine learning approaches require meticulous feature engineering, often involving an elaborate process of transforming raw text into a machine-readable format. However, this paradigm has been revolutionised by the advent of AutoML services, particularly the Google Cloud Platform's AutoML.

In our implementation, we harnessed the power of GCP AutoML to streamline the feature engineering process. We distilled our feature engineering to a singular yet crucial step: encoding the target 'sentiment' column. Given the AutoML sentiment analysis service's requirement for numerical representation, we applied label encoding to convert categorical sentiment labels into a binary format. This encoding facilitated the translation of 'Negative' and 'Positive' sentiments into 0s and 1s, respectively.

```
from sklearn import preprocessing

le = preprocessing.LabelEncoder()
df['sentiment'] = le.fit_transform(df.sentiment.values)
```

Leveraging GCP's AutoML service, we circumvent the need for extensive feature engineering on independent columns. AutoML employs sophisticated natural language processing techniques to discern and interpret the meaning embedded within text data. It automatically transforms this raw input into features that are primed for machine learning algorithms, optimising the feature representations to refine the model's training process.

In summary, our strategic decision to streamline feature engineering in tandem with utilising GCP's AutoML has resulted in a more agile and focused development process. It has not only bolstered the accuracy of our sentiment analysis model but also set a precedent for resource optimization in machine learning workflows.

5. Preprocessing and Data Pipeline

Data processing is a crucial part of building a machine learning solution. The application deployed extracts the data from reviews in a certain amount of batch size. This data is then cleaned using certain data preprocessing steps and the sentiment of the review is obtained. Since we are using the AutoML model we don't have to explicitly preprocess data as AutoML takes care of that for us.

Sentiment analysis model is trained on a text dataset and a flag indicating the sentiment of the text. We used public Kaggle dataset and extracted the reviews and label field which consists of sentiment for that review and converted the same into a csv file dataset. Further this csv was used to import data in the Vertex dataset of type text.

```
le = preprocessing.LabelEncoder()
df['sentiment'] = le.fit_transform(df.sentiment.values)

df["review"] = df["review"].str.replace('\n', ' ')
df.to_csv("reviews.csv", index=False)

dataset_created = aiplatform.TextDataset.create(
        display_name=config.DATASET_DISPLAY_NAME,
        gcs_source=config.GCS_LOCATION + config.REVIEWS_CSV_FILE,

import_schema_uri=aiplatform.schema.dataset.ioformat.text.single_label_classification,
        )
        dataset_created.wait()
        logging.info(f'Dataset resouce name is {dataset_created.resource_name}.')
```

6. Machine Learning Model Selection

6.1 Revolutionising Text Analysis with AutoML NLP

Google Cloud Platform (GCP) offers a suite of AutoML NLP services, designed to simplify and enhance the analysis of textual data. These services include:

1. **AutoML Classification**: For categorising text into predefined labels.
2. **AutoML Entity Extraction**: To identify and extract various entities within the text, such as names, places, or specific terms.
3. **AutoML Sentiment Analysis**: Specialised in discerning the attitudes and opinions expressed in English-language text.

Each of these models serves distinct use cases, ranging from organising data thematically to extracting relevant information and understanding emotional undertones.

6.2 The Case for AutoML Sentiment Analysis

In an age where data is predominantly unstructured, especially in the form of customer reviews and feedback, sentiment analysis emerges as a critical tool. It enables businesses to decipher the complex layers of customer sentiment and respond with agility. GCP's AutoML Sentiment Analysis service automates this intricate process, tapping into the subtleties of language to unveil a more detailed understanding of customer reviews.

Our client necessitated a robust solution that minimises the need for deep NLP expertise and offers customization capabilities to suit specific industry domains. Additionally, ease of use and deployment were pivotal considerations. The AutoML Sentiment Analysis service, with its intuitive interface and advanced modelling capabilities, was the optimal choice. It promises not only to elevate the precision and reliability of sentiment interpretation but also to significantly reduce the time and resources expended in traditional model development.

6.3 Business Impact and Model Suitability

Selecting the AutoML Sentiment Analysis model aligns seamlessly with our operational objectives and business imperatives. It offers a tailored fit for our use case, promising to deliver insights that are both actionable and impactful.

7. Machine learning model training and development

7.1 Balancing and Automating ML Workflows with Vertex AI and Kubeflow Pipelines

In our commitment to fostering equitable and unbiased machine learning models, we have meticulously curated our initial dataset to ensure equal representation across classes. This deliberate sampling technique is crucial to mitigate bias and maintain the integrity of the classification process, particularly in sentiment analysis tasks where imbalanced data can skew the model's performance.

7.2 Leveraging Cutting-Edge Python Libraries

To orchestrate our model training process, we utilised state-of-the-art Python libraries, such as **google-cloud-aiplatform**, **kfp**, and **Google-cloud-pipeline-components**. These tools collectively form the backbone of our machine learning pipeline, allowing for scalable and reproducible workflows on the cloud.

7.3 Constructing Kubeflow Pipelines for Sentiment Analysis

We architected a Kubeflow pipeline that seamlessly integrates with Vertex AI, utilising the **AutoMLTextTrainingJobRunOp** component to initiate the training of our text-based model. This model is attuned to discern sentiment from movie reviews, classifying them into binary categories represented by **0** for Negative and **1** for Positive sentiment.

```
ds_op = GetVertexDatasetOp(dataset_resource_name=dataset_name)
# Model training component that receives the dataset from the previous step
training_job_run_op = AutoMLTextTrainingJobRunOp(
        project=config.PROJECT_ID,
        display_name=config.TRAINING_MODEL_DISPLAY_NAME,
        prediction_type="sentiment",
        dataset=ds_op.outputs["dataset"],
        model_display_name=config.TRAINING_MODEL_DISPLAY_NAME,
        training_fraction_split=0.8,
        validation_fraction_split=0.1,
        test_fraction_split=0.1,
        sentiment_max=1)
```

The above code snippet delineates the integration of Vertex AI's capabilities within the Kubeflow pipeline, showcasing the ease of transitioning from dataset retrieval to model training.

7.4 Dataset Randomization and Split

Our approach harnesses the AutoML service's built-in functionality to randomise and divide the dataset into training, validation, and test sets, adhering to an 80-10-10 split ratio. This feature eliminates the necessity for explicit data partitioning, streamlining the model training process. Upon splitting, our dataset yielded 39,674 training records, 4,962 validation records, and 4,945 test records, setting a solid foundation for robust model training.

8. Machine learning model Evaluation

8.1 Evaluating Sentiment Analysis Models with F1-Score

8.1.1 Importance of Model Evaluation Metrics

In sentiment analysis, which is intrinsically a binary classification problem, the choice of evaluation metrics is critical to ensure the model's efficacy. Our business objective is to accurately classify movie reviews as "POSITIVE" or "NEGATIVE." To this end, we rely on the F1-score, which harmonises the balance between precision and recall, making it a robust metric for binary classification tasks.

- **Precision** measures the accuracy of positive predictions, defined as the ratio of true positives (TP) to the sum of true positives and false positives (FP).
- **Recall** assesses the model's ability to correctly identify all positive instances, calculated as the ratio of true positives to the sum of true positives and false negatives (FN).
- **F1-Score** combines precision and recall into a single metric, offering a balanced mean that requires both precision and recall to be high for the F1-score itself to be high.

8.2 Custom Component for Model Evaluation

To automate the evaluation process, we have developed a custom component within our machine learning pipeline:

```
@kfp.dsl.component(packages_to_install=["google-cloud-aiplatform"],
base_image='python:3.9')
def sentiments_model_evaluation(model: Input[Artifact], api_endpoint: str =
"us-central1-aiplatform.googleapis.com"):
        """
        This custom component is used to evaluate the performance of the model.
        Args:
        model: Model trained in the Trainer Component
        api_endpoint: Endpoint for Google public APIs.
        """
        from google.cloud import aiplatform
        model_resource_path = model.metadata["resourceName"]

        # Get the trained model resource
        model = aiplatform.Model(model_resource_path)
        evaluations = model.list_model_evaluations()
        for eval in evaluations:
        print("f1-score", eval.metrics["f1Score"])
```

This component, which utilises the **google-cloud-aiplatform** library, is designed to retrieve model evaluations and print the F1-score, thus facilitating an understanding of the model's performance.

8.3 Sentiment Analysis Model Performance

We assessed the model on a test dataset of 4,945 samples, yielding the following results:

- For the Negative class, the model achieved a precision of 93.8%, a recall of 90.8%, and an F1-score of 92.2%.
- For the Positive class, the model showed a precision of 90.68%, a recall of 94.3%, and an F1-score of 92.4%.

These metrics underscore a high level of accuracy in sentiment classification, with an overall accuracy rate of 92.5% across the test dataset. The balance achieved between precision and recall, as evidenced by the F1-scores, confirms the model's robustness in predicting sentiment accurately.

9. Model Deployment and Inference

9.1 Model Deployment on Vertex AI Endpoint

Following the successful training of the AutoML model, the next critical phase is deployment. The model is deployed onto a Vertex AI endpoint as part of the Kubeflow Pipelines (KFP) workflow. This deployment process involves creating an endpoint where the model can receive and process review data to infer sentiment.

The deployment is executed using the following KFP operations:

```
# create endpoint and deploy the model.
create_endpoint_op = EndpointCreateOp(
        project=config.PROJECT_ID,
        display_name="reviews-semantic-analysis-endpoint"
)

ModelDeployOp(
        model=training_job_run_op.outputs["model"],
        endpoint=create_endpoint_op.outputs['endpoint'],
        automatic_resources_min_replica_count=1,
        automatic_resources_max_replica_count=1
)
```

The **EndpointCreateOp** operation initialises a new endpoint, while the **ModelDeployOp** operation is responsible for deploying the trained model to this endpoint. This process ensures that the model is served in a scalable and managed environment, with the infrastructure automatically adjusting to the load while keeping the resource utilisation efficient.

9.2 Inference and Sentiment Prediction

The deployed model becomes accessible through an application hosted on Google App Engine, which acts as the user interface. Users can submit text reviews via this interface, triggering real-time sentiment analysis. The sentiment prediction is obtained by making a **predict** call to the endpoint with the review content:

```
response = endpoint.predict(instances=[{"content": input_text}], parameters={})
sentiment = ""
for prediction_ in response.predictions:
        if prediction_['sentiment'] == 0.0:
        sentiment = "Negative"
        else:
        sentiment = "Positive"
```

The **predict** method sends the review content to the model and receives a prediction response. The loop iterates through the predictions, interpreting the numeric sentiment value returned by the model (**0.0** for Negative, presumably **1.0** for Positive) and translates it into a human-readable form.

9.3 Business Integration and User Experience

The integrated system—spanning from model training on Vertex AI, deployment through KFP, to inference via App Engine—creates a seamless pipeline that not only automates the sentiment analysis process but also enhances the end-user experience. Users interact with a simple and intuitive interface, submitting reviews and receiving immediate sentiment feedback, which is powered by sophisticated machine learning models in the background.

The approach highlights the synergy between scalable cloud services and machine learning, offering businesses a powerful tool to glean insights from customer feedback without the need for deep technical expertise in deploying and maintaining AI models.

10. Callable Application Deployment

To operationalize sentiment analysis in a user-friendly manner, we have developed and deployed an intuitive User Interface (UI) on GCP's App Engine Service. This UI allows users to input text reviews and receive sentiment evaluations in real-time. The application is structured with both front-end and back-end components, where the back-end communicates with the deployed AutoML model endpoint to process and return the sentiment analysis.

*Users can access the application through the generated link.

11. Customizable Application

Flexibility and adaptability are key features of our sentiment analysis application. Users have the liberty to modify the prediction code according to their specific requirements. Additionally, the UI can be customised to match the aesthetic and functional preferences of the user or to align with corporate branding guidelines.

For updates to the model—such as incorporating new domain-specific data or data from an entirely different domain—the process is straightforward. The existing pipelines, which have been pre-configured and tested, enable retraining of the model with the new dataset. Post-training, the application can be redeployed on the App Engine with ease, ensuring that enhancements are integrated without disrupting the service.

12. Security Considerations

We have integrated robust security measures to protect the application and its data. By utilising GCP's Identity Aware Proxy (IAP), we ensure that the UI is not publicly accessible but rather available only to authenticated and authorised users. This layer of security is crucial for safeguarding sensitive information and maintaining user privacy.
Moreover, all customer content stored within GCP services is encrypted at rest. This encryption is a standard feature, provided without additional cost. Data stored on persistent disks, for example, is secured using the 256-bit Advanced Encryption Standard (AES), with encryption keys managed and rotated regularly. Such security measures are intrinsic to GCP services, offering peace of mind and demonstrating our commitment to data protection.

13. References

1. https://cloud.google.com/automl?hl=en
2. https://medium.com/google-cloud-platform-by-cloud-ace/how-to-build-a-custom-sentiment-analysis-model-with-google-automl-natural-language-fb23f668ce88
3. https://cloud.google.com/vertex-ai/docs/text-data/sentiment-analysis/train-model