Unlocking the Art of Writing

(Analyzing Behavioral Patterns for Enhanced Writing Quality Assessment)

CONTENT

1. ABSTRACT

This study investigates the dynamic relationship between the process of writing and the resultant quality of written content. Through an analysis of various writing methodologies, strategies, and stages, it aims to establish correlations between the processes employed by writers and the excellence of their output. By examining factors such as pre-writing techniques, drafting, revision, editing, and the utilization of feedback, this research seeks to identify the pivotal aspects of the writing process that significantly influence the overall quality of written works. The findings of this study are expected to contribute valuable insights into optimizing writing practices to enhance the quality and effectiveness of written communication across diverse contexts and audiences.

2. INTRODUCTION

The art of writing is a multifaceted process, encompassing various stages that writers navigate to produce high-quality content. Understanding the intricate connection between the writing process and the resulting writing quality is crucial for writers, educators, and researchers alike. This study delves into the correlation between the methodologies, strategies, and stages involved in the writing process and the ultimate caliber of the written work. By exploring these interconnections, it aims to shed light on how different writing processes impact the quality and efficacy of written compositions.

3. EXECUTIVE SUMMARY

The objective of this endeavor is to deploy a comprehensive framework that bridges the findings from the correlation between writing processes and writing quality into practical applications. This deployment aims to transition the identified correlations and methodologies, initially discovered through research, into accessible tools and educational resources. By leveraging technology and pedagogical strategies, the objective is to empower educators, writers, and learners with user-friendly platforms or guides derived from these insights. The deployment seeks to foster improved writing methodologies and enhanced writing quality among users, ultimately bridging the gap between research findings and practical implementation within educational and professional settings.

4. Methodology

4.1 BUSINESS UNDERSTANDING

Intelligent tutoring and grading systems aim to assist educational institutions in refining teaching methods and boosting students' writing skills and academic progress. These systems introduce innovative approaches, equipping educators with advanced tools tailored to individual student needs. By utilizing these systems, schools create more personalized learning experiences, pinpointing areas where students require additional support and offering specific guidance. This approach fosters enhanced writing abilities and overall academic achievements among students.

Skill assessment using effective tools offers great potential to empower individuals to maximize their capabilities. These assessment tools offer in-depth insights into a person's strengths and areas needing improvement across a wide range of skills. With this information, people can personalize their growth journey, honing specific skills

essential for personal and professional development. This empowerment through skill assessment allows individuals to proactively enhance their skills, leading to success in various life domains

4.1 DATA UNDERSTANDING

The table below represents the attributes of the data.

	id	event_id	down_time	up_time	action_time	activity	down_event	up_event	text_change	cursor_position	word_count
0	001519c8	1	4526	4557	31	Nonproduction	Leftclick	Leftclick	NoChange	0	0
1	001519c8	2	4558	4962	404	Nonproduction	Leftclick	Leftclick	NoChange	0	0
2	001519c8	3	106571	106571	0	Nonproduction	Shift	Shift	NoChange	0	0
3	001519c8	4	106686	106777	91	Input	q	q	q	1	1
4	001519c8	5	107196	107323	127	Input	q	q	q	2	1

Fig 4.1

- Id: Id column represents the user/writer.
- Event id: It is a unique column, which represents the number events that occurred.
- Down time: It is the press time of the key on the keyboard.
- Up time: It is the release time of the key on the keyboard.
- Action time: It shows the difference between up time and down time.
- Activity: It stores the values entered by user was an input or non-production.
- Down event: It represents the down-time event of the key as an input [q] or non-production key on the key board [shift key, enter key, right click, left click and so on].
- Up event: It represents the up-time event of the key as an input [q] or non-production key on the key board [shift key, enter key, right click, left click and so on].
- Text change: It shows whether if there is any change of the key from the keyboard.
- Cursor position: It holds the number of the inputs given and moves further as cursor from that position.
- Word count: It depicts the number of words.

4.2 DATA PREPARATION

I. **DATA TRANSFORMATION:** In the process of enhancing the predictive accuracy of our model, we implemented a critical data transformation step involving the creation and addition of new features to our dataset. Specifically, we introduced two four features named 'Total_Positive_wait_time', 'Total_Negative_wait_time' , 'Count_Positive_wait_time' and 'Count_Negative_wait_time' aimed at refining the

model's ability to forecast outcomes more accurately. Adding more features strengthens our model's forecasting ability by enabling it to extract more complex patterns from the data. The code to create new features are shown in figure 4.2.1 and 4.2.2.

```
# Creating new feature wait_time
train_logs['wait_time'] = train_logs['down_time'] - train_logs.groupby('id')['up_time'].shift()
train_logs['wait_time'].fillna(0, inplace=True)
train_logs.head()
```

	id	event_id	down_time	up_time	action_time	activity	down_event	up_event	text_change	cursor_position	word_count	wait_time
0	001519c8	1	4526	4557	31	Nonproduction	Leftclick	Leftclick	NoChange	0	0	0.0
1	001519c8	2	4558	4962	404	Nonproduction	Leftclick	Leftclick	NoChange	0	0	1.0
2	001519c8	3	106571	106571	0	Nonproduction	Shift	Shift	NoChange	0	0	101609.0
3	001519c8	4	106686	106777	91	Input	q	q	q	1	1	115.0
4	001519c8	5	107196	107323	127	Input	q	q	q	2	1	419.0

Fig 4.2.1

```
In [ ]: # Creating positive, negative wait time aggreagated by sum and count
        # Sum positive values, treat negative values as 0, and find counts
        wait_time_df = train_logs.groupby('id')['wait_time'].agg(
            Total_Positive_wait_time=lambda x: x[x > 0].sum(),
            Total_Negative_wait_time=lambda x: x[x < 0].sum(),
            Count_Positive_wait_time=lambda x: (x > 0).sum(),
            Count_Negative_wait_time=lambda x: (x < 0).sum()
        ).reset_index()

        wait_time_df.head()
```

Out[ ]:

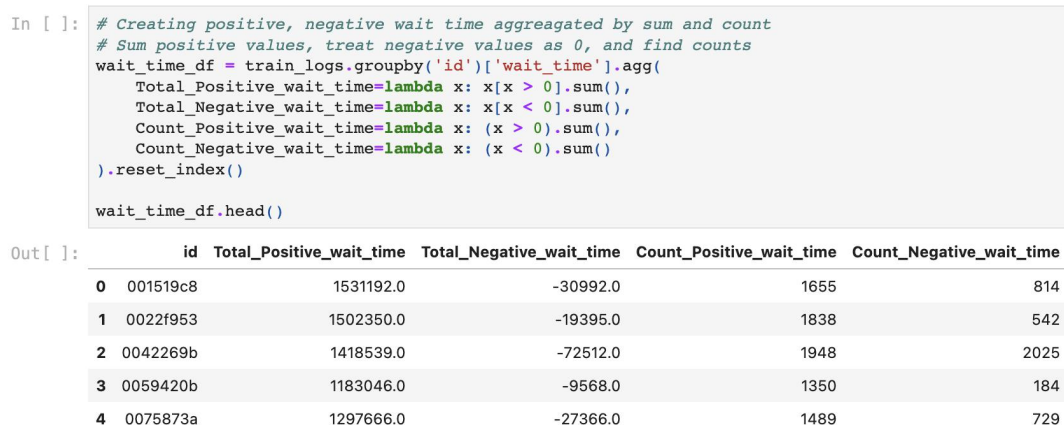|	id	Total_Positive_wait_time	Total_Negative_wait_time	Count_Positive_wait_time	Count_Negative_wait_time
0	001519c8	1531192.0	-30992.0	1655	814
1	0022f953	1502350.0	-19395.0	1838	542
2	0042269b	1418539.0	-72512.0	1948	2025
3	0059420b	1183046.0	-9568.0	1350	184
4	0075873a	1297666.0	-27366.0	1489	729

Fig 4.2.2

II. **FEATURE ENGINEERING:** We using essay constructor method to generate essays for different users as in below fig 4.2.3. Here the essay's generated are encoded as 'qqqq'.

```
def count_sentences(text):
    sentences = nltk.sent_tokenize(text)
    return len(sentences)

def count_paragraphs(text):
    paragraphs = [p for p in text.split('\n\n') if p.strip()]  # Split paragraphs and remove empty ones
    return len(paragraphs)

# Create new columns for number of sentences and paragraphs
train_essay_score['Num_Sentences'] = train_essay_score['essay'].apply(count_sentences)
train_essay_score['Num_Paragraphs'] = train_essay_score['essay'].apply(count_paragraphs)
train_essay_score.head()
```

	id	essay	word_count	score	All Move From	Input	Nonproduction	Paste	Remove/Cut	Replace	Total_Positive_wait_time	Total_Negative_wait_time	Count_Positive_wait_time
0	001519c8	qqqqqqqqqq qq qqqqq qq qqqq qqqq. qqqqqqq qqq q...	256	3.5	3	2010	120	0	417	7	1531192.0	-30992.0	1655
1	0022f953	qqqq qq qqqqqqqqqqq ? qq qq qqq qqq qqq. qqqqq...	324	3.5	0	1938	254	1	260	1	1502350.0	-19395.0	1838
2	0042269b	qqqqqqqqqqqq qq qqqqqq qqqqqqqqqq qq qqqqqqqqqqq ...	409	6.0	0	3515	175	0	439	7	1418539.0	-72512.0	1948
3	0059420b	qq qqqqqqqq qqqqqq qqqqqqqqqqqq qqqq q qqqq	207	2.0	0	1304	99	1	151	1	1183046.0	-9568.0	1350

Fig 4.2.3

We have also used 'nlkt' library in the feature engineering process for the count of number of sentences and paragraphs present in the respective user's essay. However, the number of the model features are enhanced such as word count, score, number of action input, number of Non production, total positive wait time, Total negative wait time, count positive wait time, count negative wait time, number of sentences and number of paragraphs for the precision model accuracy.
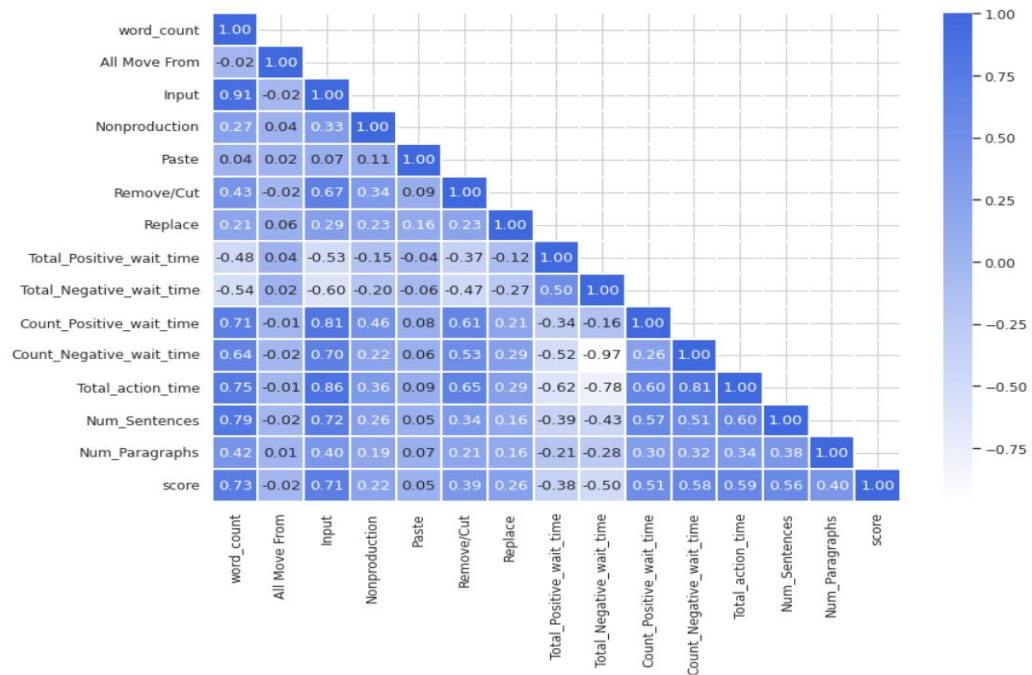

Fig 4.2.4

Fig 4.2.4 depicts that the word count is directly proportional to score and score is inversely proportional to Total negative wait time.

III. **DATA AUGMENTATION:** The below fig 4.2.5 depicts less data points of 1976 and there is class imbalance with lowest and highest data sets. Therefore, to overcome this class imbalance we are augmenting the data by extra noise factor as in fig 4.2.6.

```
train.shape
(1976, 15)

y_train.value_counts()
4.0    401
3.5    389
4.5    321
3.0    269
2.5    161
5.0    143
5.5    102
2.0     73
1.5     55
6.0     30
1.0     28
0.5      4
Name: score, dtype: int64
```

Fig 4.2.5

```
y_train.value_counts()
5.0    429
5.5    408
4.0    401
0.5    400
1.0    392
6.0    390
3.5    389
1.5    385
2.0    365
2.5    322
4.5    321
3.0    269
Name: score, dtype: int64
```

Fig 4.2.6

4.3 Modeling

```
Epoch 2/100
 6/70 [=>............................] - ETA: 1s - loss: 3.1076
/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3079: UserWarning: You are saving your mode
considered legacy. We recommend using instead the native Keras format, e.g. `model.save('my_model.keras')`.
  saving_api.save_model(
70/70 [==============================] - 2s 22ms/step - loss: 2.5480 - val_loss: 6.7932
Epoch 3/100
70/70 [==============================] - 2s 26ms/step - loss: 2.0351 - val_loss: 2.6139
Epoch 4/100
70/70 [==============================] - 2s 25ms/step - loss: 1.6336 - val_loss: 1.7025
Epoch 5/100
70/70 [==============================] - 2s 30ms/step - loss: 1.3215 - val_loss: 2.1414
Epoch 6/100
70/70 [==============================] - 1s 17ms/step - loss: 1.2010 - val_loss: 2.2354
Epoch 7/100
70/70 [==============================] - 1s 18ms/step - loss: 1.0775 - val_loss: 2.3753
Epoch 8/100
70/70 [==============================] - 1s 18ms/step - loss: 0.9195 - val_loss: 2.2649
Epoch 9/100
70/70 [==============================] - 1s 17ms/step - loss: 0.8622 - val_loss: 3.0921
Epoch 10/100
70/70 [==============================] - 1s 17ms/step - loss: 0.7645 - val_loss: 7.2220
Epoch 11/100
70/70 [==============================] - 1s 18ms/step - loss: 0.7045 - val_loss: 1.9068
Epoch 12/100
70/70 [==============================] - 2s 22ms/step - loss: 0.6311 - val_loss: 1.9907
Epoch 13/100
70/70 [==============================] - 2s 25ms/step - loss: 0.5947 - val_loss: 1.8560
Epoch 14/100
70/70 [==============================] - 2s 25ms/step - loss: 0.5804 - val_loss: 3.1803
16/16 [==============================] - 0s 3ms/step
True Value: 3.50, Predicted Value: 4.0
True Value: 3.50, Predicted Value: 3.0
True Value: 4.00, Predicted Value: 3.0
True Value: 4.00, Predicted Value: 3.0
True Value: 2.50, Predicted Value: 4.0
```

fig 4.3

It builds a neural network model with a sequential structure especially for regression

tasks using the Keras framework. A scaler is used to normalize the input data, and many

densely related layers with ReLU activation functions are used in the construction of the

model. Additionally, dropout layers are used to lessen overfitting. The mean squared error loss function is used in the model's creation, which occurs using the RMSprop optimizer. During the training process, callbacks for early stopping and model checkpointing are included to avoid overfitting and preserve the top-performing model. The best model is uploaded with the model checkpoint after training has ended, and it is then used to forecast the test data. These predictions are further adjusted by clipping them within a defined range and rounding them to the nearest half. Lastly, a sample of true values versus predicted values is printed to assess the performance of the model.

```
True Value: 3.50, Predicted Value: 3.81
True Value: 3.50, Predicted Value: 2.90
True Value: 4.00, Predicted Value: 3.09
True Value: 4.00, Predicted Value: 2.88
True Value: 2.50, Predicted Value: 4.15
```

Fig 3.3.1

The text that is provided shows the first five exact numbers in addition to the estimated values that come from a regression model's evaluation. It is an example of a comparison where the estimated values represent the model's predictions and the accurate values represent the actual results, like ratings or scores. The shown values show the extent to which the model's predictions and the real data match.

4.4 Evaluation
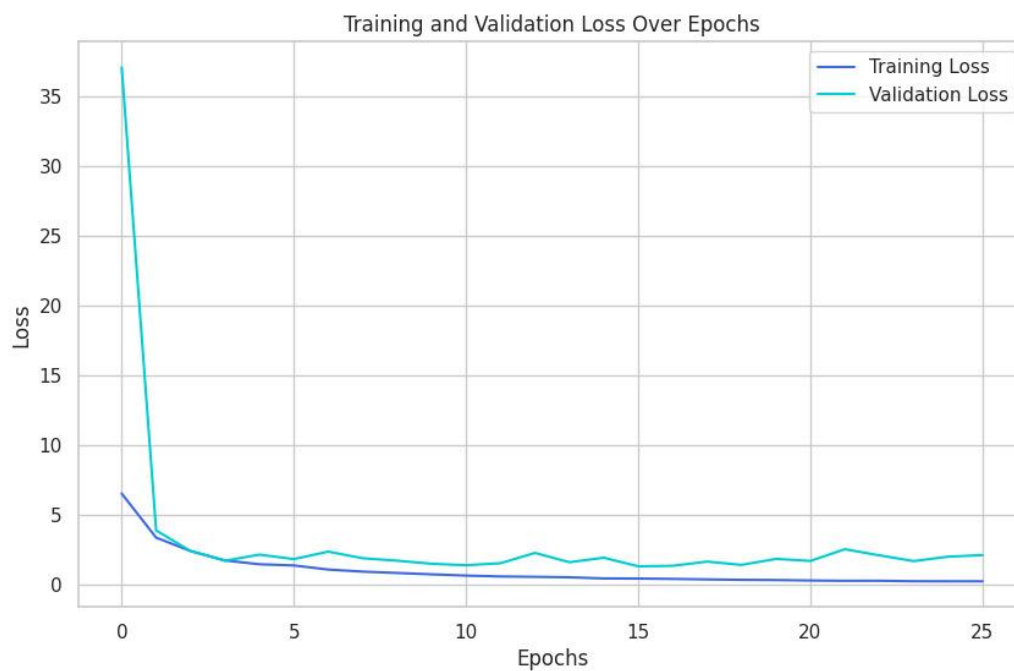
Fig 4.4-Training and validation loss over epochs

This interactive graph provides useful data about the model's performance during training. It clearly shows how the loss decreases with every epoch, showing a change in the model's functionality. The model looks to be having a good understanding of the data based on the training loss's small drop and the validation loss's somewhat bigger decline. Interactive feature makes it simple to look into the model's learning processes. This graph helps the process of choosing and improving the model by providing a thorough overview of the model's training progress.

4.5 Deployment

In the deployment phase of this project, the transition from Jupyter Notebook to a fully functional web application using Django on the Azure cloud platform was the focal point. Leveraging the insights and models derived from Jupyter Notebook's data analysis, the integration into Django framework facilitated the transformation into an interactive web-based tool. Azure's robust infrastructure, particularly the Azure Web Server, emerged as the ideal hosting environment for this application. Through Azure's App Service, the deployment process was streamlined, ensuring efficient utilization of resources and scalability. The culmination of this deployment journey marked the successful transition from development to a live environment accessible to end-users.

The demo application has been deployed at the url :

https://team-2-dsci-6002-f23-fp-unlocking-the-art-of-writing.azurewebsites.net/

The code is hosted on GitHub:

https://github.com/CRLannister/Unlocking-the-Art-of-Writing/
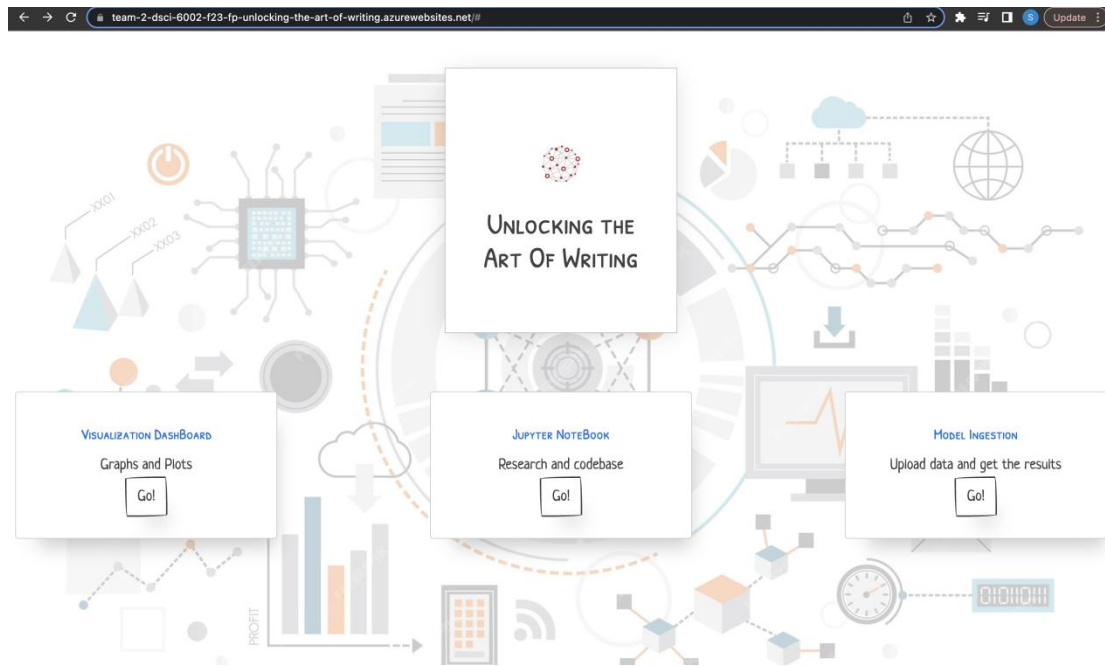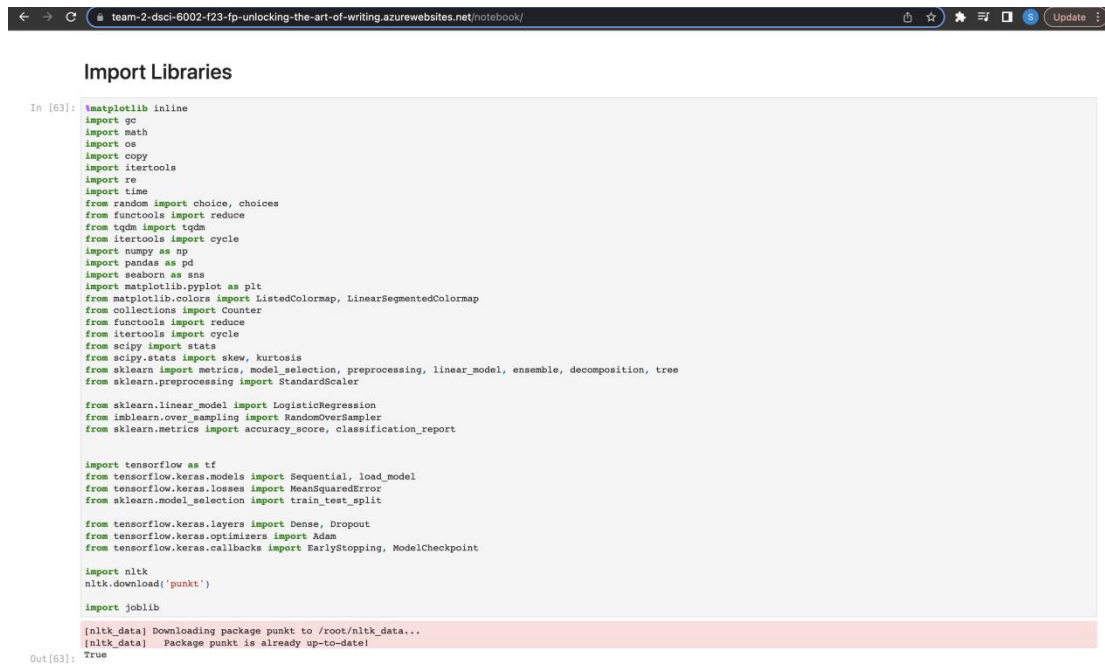
Fig 4.5 - Application main page
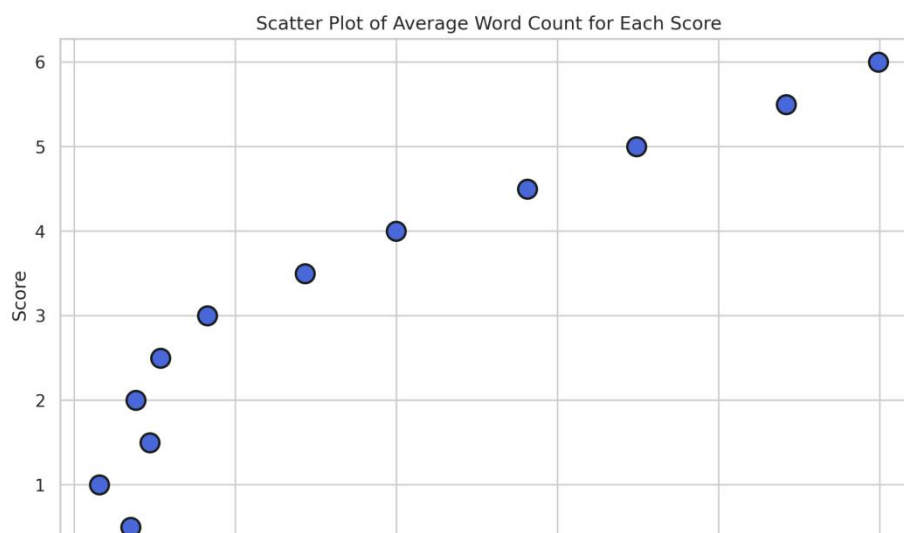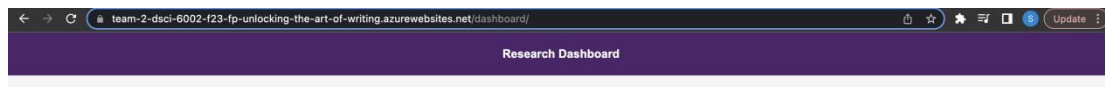
Fig 4.5.1 - Visualization Dashboard

Import Libraries

```
In [63]:  %matplotlib inline
          import gc
          import math
          import os
          import copy
          import itertools
          import re
          import time
          from random import choice, choices
          from functools import reduce
          from tqdm import tqdm
          from itertools import cycle
          import numpy as np
          import pandas as pd
          import seaborn as sns
          import matplotlib.pyplot as plt
          from matplotlib.colors import ListedColormap, LinearSegmentedColormap
          from collections import Counter
          from functools import reduce
          from itertools import cycle
          from scipy import stats
          from scipy.stats import skew, kurtosis
          from sklearn import metrics, model_selection, preprocessing, linear_model, ensemble, decomposition, tree
          from sklearn.preprocessing import StandardScaler

          from sklearn.linear_model import LogisticRegression
          from imblearn.over_sampling import RandomOverSampler
          from sklearn.metrics import accuracy_score, classification_report


          import tensorflow as tf
          from tensorflow.keras.models import Sequential, load_model
          from tensorflow.keras.losses import MeanSquaredError
          from sklearn.model_selection import train_test_split

          from tensorflow.keras.layers import Dense, Dropout
          from tensorflow.keras.optimizers import Adam
          from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint

          import nltk
          nltk.download('punkt')

          import joblib

          [nltk_data] Downloading package punkt to /root/nltk_data...
          [nltk_data]   Package punkt is already up-to-date!
Out[63]: True
```
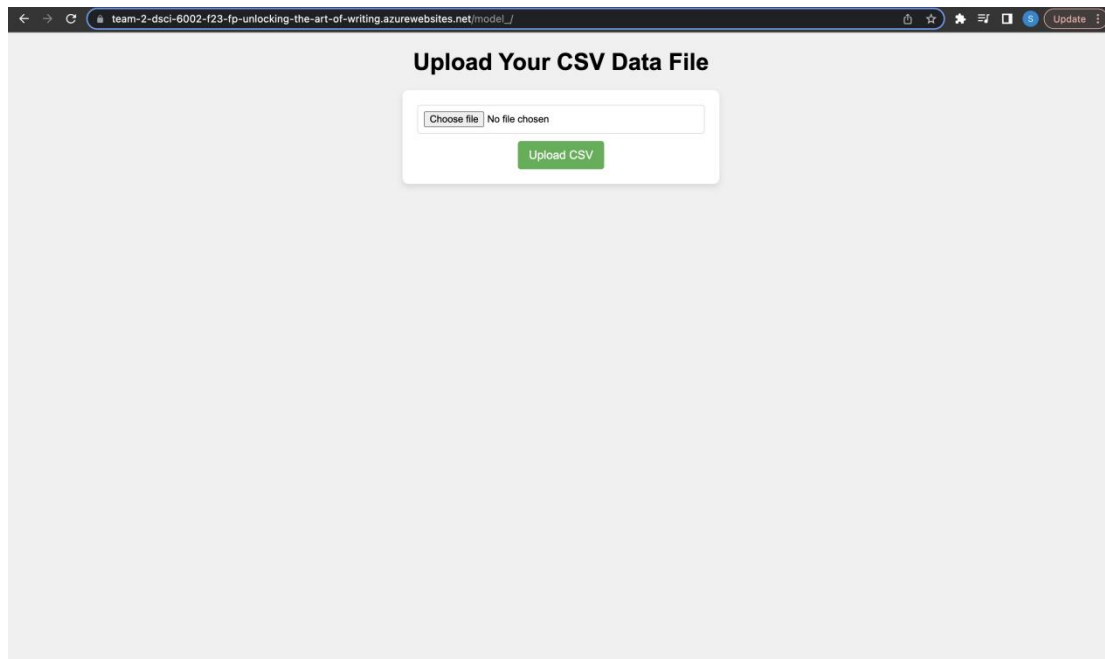
Fig 4.5.2 - Jupyter Notebook

Upload Your CSV Data File

Choose file No file chosen

Upload CSV

Fig 4.5.3 - Model Ingestion

5. CONCLUSION

In examining the intricate relationship between the writing process and the resulting quality of written work, this study has revealed significant insights into the interplay of methodologies and the caliber of written content. The findings underscore the pivotal role of each stage in the writing process, from initial ideation through revision and editing, in shaping the overall quality of the composition.

This study's insights offer valuable implications for educators, writers, and practitioners in fostering effective writing practices. By recognizing the symbiotic relationship between writing methodologies and writing quality, stakeholders can devise tailored strategies to nurture and enhance writing abilities, ultimately fostering a culture of excellence in written communication across various domains. As the landscape of communication evolves, continued research in this domain remains pivotal for refining approaches and empowering individuals to excel in their written expression.

6. FUTURE WORK

There are a number of interesting avenues for future research on the relationship between writing methods and writing quality. Cutting-edge analytical methods, such as the combination of machine learning and natural language processing, provide a way to explore the subtle differences in writing processes among different genres and populations. Through the tracking of writing skills over time, longitudinal studies provide light on effective instructional techniques and developmental trajectories. Furthermore, research into how cultural diversity and contextual elements affect writing styles and quality is still necessary in order to comprehend how sociocultural backgrounds influence written communication. Examining how technology is integrated, especially with AI-powered writing aides and collaborative platforms, creates opportunities to evaluate how technology affects writing techniques.

7. REFERENCES

[1] https://www.kaggle.com/competitions/linking-writing-processes-to-writing-quality

[2] https://www.tensorflow.org/api_docs/python/tf/keras/Sequential

[3] https://analyticsindiamag.com/a-tutorial-on-sequential-machine-learning/