ISSN 2278-3091

Volume 12, No.4, July - August 2023

International Journal of Advanced Trends in Computer Science and Engineering

Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse021242023.pdf

https://doi.org/10.30534/ijatcse/2023/021242023

An Analysis of Pairwise Question Matching with Machine Learning

Ashika Tarekar[1], Nitesh Hirulkar[2]

[1] Software Engineer at LnS-Infotech, India, tarekarashika1990@gmail.com

[2] Assistant Professor at Sinhgad Institute of Technology, India, nshirulkar.acem@gmail.com

Received Date June 14, 2023 Accepted Date: July 24, 2023 Published Date: August 06, 2023

ABSTRACT

In the realm of Natural Language Processing (NLP) and machine learning, lies the challenging quest to detect duplicate question pairs with semantic precision. Our research endeavors to craft a cutting-edge model capable of discerning whether two questions, despite their divergent phrasing, spelling, or grammatical variations, share a common intent on digital forums or search engines. A paramount facet of this study involves the creation and training of an exemplary model using a meticulously curated dataset of labeled question pairs, each annotated as either duplicates or distinct entities. By leveraging state-of-the-art NLP techniques, we aspire to build an exceptionally accurate model that will revolutionize the user search experience by facilitating the identification of duplicate questions. This pioneering research paves the way for a more refined and enhanced approach to tackle the challenges of semantic similarity in the context of question pairs.

Key words: Duplicate question, machine learning, natural language processing

1. INTRODUCTION

In the digital era, where the allure of the internet and social media is unparalleled, an astonishing triumph unfolds as a vast and active user base embraces these virtual landscapes. The online realm teems with a plethora of social media platforms, each carving its distinct niche in the digital fabric. Facebook, an epitome of user interaction; LinkedIn, a haven for professional networking; WhatsApp, the virtual sanctuary for seamless chat and video calling; Stack Overflow, the virtuoso addressing technical conundrums; and Instagram, the captivating realm of photo sharing, together embody the manifold facets of our interconnected world [1].

In the midst of this ever-evolving digital tapestry, a veritable jewel emerges, known as Quora. This ingeniously crafted Question & Answer platform thrives on a vibrant community of users who converge to exchange knowledge, unfurl opinions, and showcase their expertise across a myriad of captivating subjects. Enriched by this collective intellect, Quora stands as a shining testament to the boundless potential of human interaction and knowledge sharing in the digital age [3]. This research article presents an in-depth exploration of the captivating journey undertaken by diverse social media platforms and chronicles the ascendancy of Quora as a veritable nexus for the exchange of wisdom and ideas.

In the age of exponential knowledge expansion, the preservation of user trust and the assurance of content excellence have become paramount for Quora. To safeguard the sanctity of its knowledge base, the platform grapples with the pressing need to weed out undesirable elements such as junk, duplicates, and insincere information. Drawing upon the prowess of modern data science, Quora rises to this formidable challenge, orchestrating an effective and meticulous data organization strategy [4].

In this pioneering research article, we delve into Quora's ingenious journey of fortifying user confidence and curating a seamless knowledge-sharing experience. Through the adept application of advanced data science approaches, the platform eradicates question duplication, thus ensuring a rich and diverse repository of authentic content. This transformative endeavor not only elevates Quora's standing as a beacon of reliable information but also sets a precedent for other knowledge-driven platforms to follow. Embrace the revelations of our study as we illuminate the path towards a more trustworthy and impactful landscape of digital knowledge dissemination.

2. BACKGROUND

In the digital age, fake news has become a rampant problem, spreading misinformation and manipulating public opinion. Machine Learning (ML) and Natural Language Processing (NLP) offer promising solutions to detect fake news. ML algorithms, especially deep learning techniques, can analyze vast amounts of text to discern authenticity. Challenges include subtle fake news imitation and the need for real-time detection on social media [2], [5]. Researchers have explored various features and ensemble methods to improve accuracy. However, ongoing research aims to enhance resilience against adversarial attacks, ensuring the integrity of information and a more informed society.

3. APPROACH AND EXPERIMENT

The goal of this project is to develop a system that can automatically identify whether two given questions are duplicates or not. This can be useful in various applications; such as in question-answering systems or online forums, where it's important to identify similar questions that have already been asked.

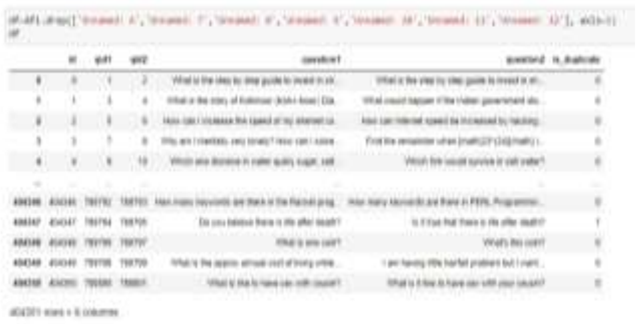
3.1. Data Collection

We need a large dataset of question pairs, where each pair consists of a question and its duplicate (or non-duplicate) counterpart. This dataset should cover a wide range of topics to ensure the model learns general patterns.

Data collection operates on diverse levels, with IT systems efficiently capturing customer, employee, and business data during transactions. Surveys and social media tracking enrich the landscape by gathering customer feedback. A formidable trio of data scientists, analysts, and business users then extract pertinent data from internal systems and external sources, laying the foundation for data preparation and subsequent utilization in business intelligence (BI) and analytics applications. This journey illuminates a realm of dynamic insights, culminating in a symphony of data that beckons us towards newfound understanding [6].

In various fields like science, medicine, and higher education, collecting data is a specialized process. Researchers carefully create and use measures to gather specific sets of data. Both in business and research, accurate data is crucial to ensure that the findings and results are trustworthy and valid. It is like a seal of truth that unlocks new knowledge and leads us to valuable insights. In this pursuit, accurate data collection stands as the guardian, safeguarding the essence of research with utmost care.

In the process of data collection, the esteemed clients entrust their valuable data to our proficient data analysis team. With meticulous attention to detail, the team diligently collects the data and adapts it to meet the specific project requirements, proceeding to conduct further preprocessing. For this particular research project, the dataset consists of a total of 404,351 rows and 13 columns, laying the foundation for in-depth analysis and exploration code as shown in Figure 1.



```
df=df1.drop(['Unnamed: 6','Unnamed: 7','Unnamed: 8','Unnamed: 9','Unnamed: 10','Unnamed: 11','Unnamed: 12'], axis=1)
df
```

	id	qid1	qid2	question1	question2	is_duplicate
0	0	1	2	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0
1	1	3	4	What is the story of Kohinoor (Koh-i-Noor) Dia...	What would happen if the Indian government sto...	0
2	2	5	6	How can I increase the speed of my internet co...	How can Internet speed be increased by hacking...	0
3	3	7	8	Why am I mentally very lonely? How can I solve...	Find the remainder when [math]23^{24}[/math] i...	0
4	4	9	10	Which one dissolve in water quikly sugar, salt...	Which fish would survive in salt water?	0
...	...	...	...	...	...	...
404346	404346	789792	789793	How many keywords are there in the Racket prog...	How many keywords are there in PERL Programmi...	0
404347	404347	789794	789795	Do you believe there is life after death?	Is it true that there is life after death?	1
404348	404348	789796	789797	What is one coin?	What's this coin?	0
404349	404349	789798	789799	What is the approx annual cost of living while...	I am having little hairfall problem but I want...	0
404350	404350	789800	789801	What is like to have sex with cousin?	What is it like to have sex with your cousin?	0

404351 rows × 6 columns

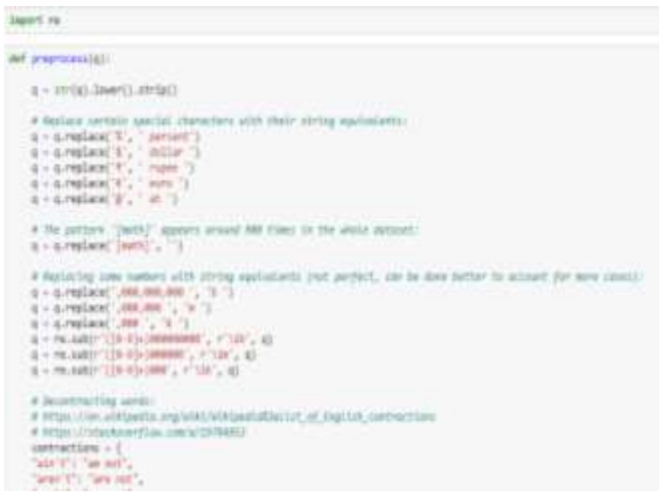
Figure 1: Data Collection

3.2. Data Preprocessing

The collected data needs to be cleaned and preprocessed to remove any noise or irrelevant information. This typically involves removing punctuation, converting text to lowercase, and handling special characters [7].

In the world of data analysis, there's a crucial step called Data Preprocessing, which works like magic! It takes messy, real-world data, like text, images, and videos, and transforms it into a neat and understandable format for computers and machine learning. You see, raw data can be quite messy, with errors, missing parts, and no regular structure [8]. Data preprocessing comes to the rescue, making the data clear and organized. It's like polishing a rough gem, making it shine brilliantly. This process allows us to uncover valuable insights from the data, leading us on a journey of discovery and knowledge. So, come along as we delve into the enchanting world of data preprocessing and witness the magic of turning chaos into clarity!

Presently, the data analysis team undertakes the critical task of preprocessing the acquired data, adeptly organizing it into coherent rows and columns code as shown in Figure 2. This meticulous process serves the dual purpose of reducing the data's size while distilling the most salient and meaningful information. By skillfully curating the dataset, the team unearths the crucial data points that will shape our research and lead us towards valuable insights and discoveries.



```
import re

def preprocess(q):

    q = str(q).lower().strip()

    # Replace certain special characters with their string equivalents
    q = q.replace('%', ' percent')
    q = q.replace('$', ' dollar ')
    q = q.replace('₹', ' rupee ')
    q = q.replace('€', ' euro ')
    q = q.replace('@', ' at ')

    # The pattern '[math]' appears around 900 times in the whole dataset.
    q = q.replace('[math]', '')

    # Replacing some numbers with string equivalents (not perfect, can be done better to account for more cases)
    q = q.replace(',000,000,000 ', 'b ')
    q = q.replace(',000,000 ', 'm ')
    q = q.replace(',000 ', 'k ')
    q = re.sub(r'([0-9]+)000000000', r'\1b', q)
    q = re.sub(r'([0-9]+)000000', r'\1m', q)
    q = re.sub(r'([0-9]+)000', r'\1k', q)

    # Decontracting words
    # https://en.wikipedia.org/wiki/Wikipedia%3aList_of_English_contractions
    # https://stackoverflow.com/a/19794953
    contractions = {
    "ain't": "am not",
    "aren't": "are not",
```

Figure 2: Data Preprocessing

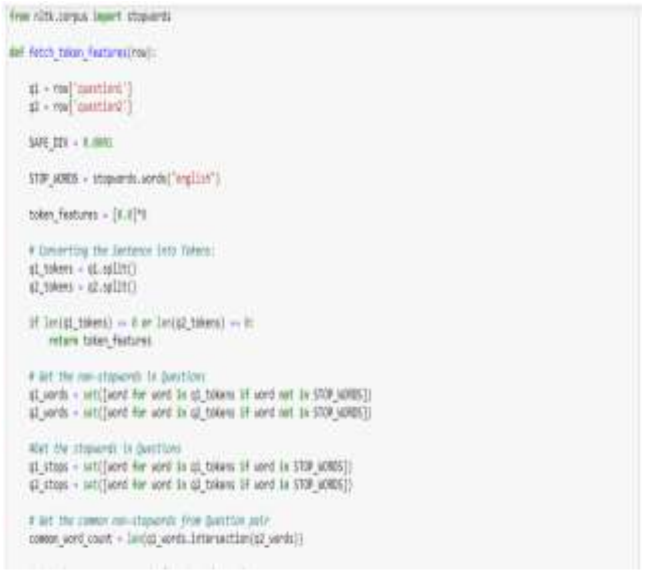
3.3. Feature Extraction

Next, we extract meaningful features from the preprocessed text. Commonly used features include word embedding's, which are vector representations of words that capture their semantic meaning. Other features could include n-grams (sequences of adjacent words) or syntactic patterns as shown in Figure 3.

In the world of machine learning, Feature Extraction is like a clever trick that helps to handle large datasets more easily.

By using feature extraction, we can reduce unnecessary information without losing anything important [9]. This means we can build models more quickly and efficiently, making the learning process faster and smarter. It's like a shortcut to get the best results without the extra effort [10]. Join us as we explore the wonders of feature extraction and how it makes machine learning simpler and more powerful than ever before!

In the process of feature extraction, we apply a sophisticated methodology that involves the elimination of STOP words, tokens, and other irrelevant elements. Through this meticulous curation, we are left with a refined set of words or data that is both meaningful and devoid of unnecessary noise. This preparatory step primes our dataset, readying it for further analysis and enabling us to unlock the essence of our research with enhanced accuracy and precision code as shown in Figure 3.



```
from nltk.corpus import stopwords

def fetch_token_features(row):

    q1 = row['question1']
    q2 = row['question2']

    SAFE_DIV = 0.0001

    STOP_WORDS = stopwords.words("english")

    token_features = [0.0]*8

    # Converting the Sentence into Tokens:
    q1_tokens = q1.split()
    q2_tokens = q2.split()

    if len(q1_tokens) == 0 or len(q2_tokens) == 0:
        return token_features

    # Get the non-stopwords in Questions
    q1_words = set([word for word in q1_tokens if word not in STOP_WORDS])
    q2_words = set([word for word in q2_tokens if word not in STOP_WORDS])

    #Get the stopwords in Questions
    q1_stops = set([word for word in q1_tokens if word in STOP_WORDS])
    q2_stops = set([word for word in q2_tokens if word in STOP_WORDS])

    # Get the common non-stopwords from Question pair
    common_word_count = len(q1_words.intersection(q2_words))
```

Figure 3: Feature Extraction

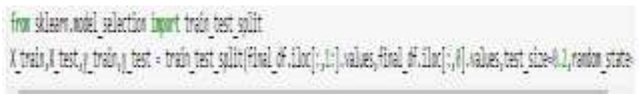
We split the dataset into two parts: a training set and a test set. The training set will be used to train the machine learning model, while the test set will be used to evaluate its performance.

In machine learning, the train-test split is a handy way to see how well prediction algorithms work. It helps us estimate the performance of our own machine learning model by comparing it to other models. The process is quick and simple: we divide our data into two sets. The test set contains 30% of the actual data, while the training set has the remaining 70%. This allows us to evaluate how accurate our model is in making predictions code as shown below in Figure 4.

To see how good our machine learning model is, we divide the data into two parts: the train set and the test set. The train set is used to fit the model, and we know its statistics [11]. The test set is used only for predictions to check how accurate our model is. It's like having a practice set and a test set in school to see how well we've learned.

- Dataset Splitting Process:

Scikit-learn (or sklearn for short) is super helpful library for machine learning in Python. Inside sklearn, there's a module called model_selection, and it has a very handy function called train_test_split(). This function helps us split our data into two parts, making it easy to train our machine learning models and test how well they work. It's like a useful tool that saves us lots of time and effort!



```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(final_df.iloc[:,1:].values,final_df.iloc[:,0].values,test_size=0.2,random_state
```

Figure 4: Train Test Split

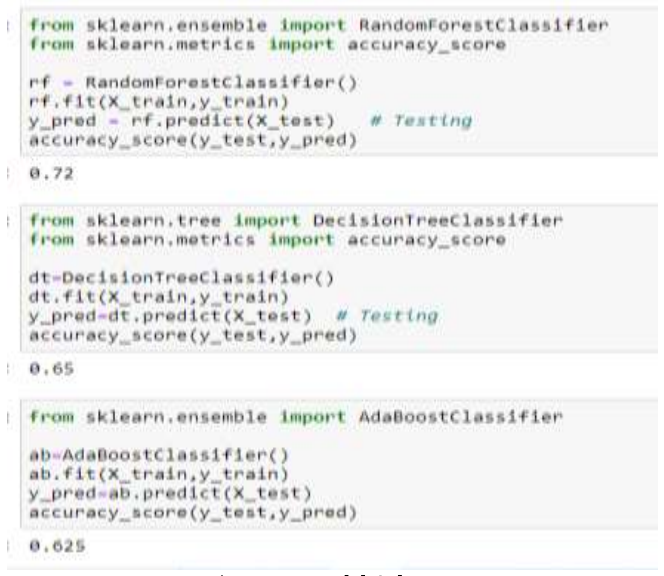
3.4. Model Selection

There are several machine learning algorithms we can choose from to build your model, such as logistic regression, random forests, or deep learning models like recurrent neural networks (RNNs) or transformers. The choice depends on the complexity of the problem and the size of the dataset.

The way projects are chosen can vary based on the organization's size and how it's set up. Sometimes, top-level management or an executive team decides which projects to go with. In corporate sector cases, project managers or team leaders take charge of picking the projects. It all depends on how the organization works and what makes the most sense for them [12].

- Model Selection Steps:
  - o Make sure the project fits the company's strategy
  - o Understand company environment
  - o Consider and analyze the historical data
  - o Decide who will be the project champion

During the pivotal phase of model selection, we deliberately employ three prominent algorithms: Random Forest, Decision Tree, and AdaBoost. This strategic choice arises from our endeavor to meticulously evaluate the accuracy of each algorithm for our specific project. Our research mandates a keen pursuit of the most accurate model, and thus, we meticulously scrutinize each algorithm's performance. Amongst the trio, Random Forest emerges as the frontrunner, surpassing the other two with its commendable accuracy. This discerning revelation becomes the compass that guides us towards adopting Random Forest as the preferred algorithm for our project, cementing our path towards optimal results and valuable insights code shown below in Figure 5.



```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

rf = RandomForestClassifier()
rf.fit(X_train,y_train)
y_pred = rf.predict(X_test)   # Testing
accuracy_score(y_test,y_pred)
```

```
0.72
```

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score

dt=DecisionTreeClassifier()
dt.fit(X_train,y_train)
y_pred=dt.predict(X_test)  # Testing
accuracy_score(y_test,y_pred)
```

```
0.65
```

```
from sklearn.ensemble import AdaBoostClassifier

ab=AdaBoostClassifier()
ab.fit(X_train,y_train)
y_pred=ab.predict(X_test)
accuracy_score(y_test,y_pred)
```

```
0.625
```

Figure 5: Model Selection

3.5. Training the Model

Using the training set, we train the chosen model by feeding it the question pairs along with their corresponding labels (duplicate or non-duplicate). The model learns to identify patterns and features that distinguish between duplicate and non-duplicate pairs.

In a dataset, we use a training set to build a model, and a test (or validation) set to check if the model works well. The data points in the training set are kept separate from the test (validation) set. Sometimes, we split the dataset into a training set and a validation set, and in other cases, we divide it into a training set, a validation set, and a test set. This helps us make sure the model is accurate and performs well on new data [13]. After we choose a model, we train it using the data we prepared earlier. Once the model is trained, we can use it to predict outcomes for new data that it hasn't seen before. To see how well our model performs, we create and it is called a "confusion matrix." This matrix has four parts: "True Positives," "True Negatives," "False Positives," and "False Negatives." We want more values in the "True Negatives" and "True Positives" to have a better and more accurate model. The size of the confusion matrix depends on the number of different categories we have in our data.

3.6. Model Evaluation

Once the model is trained, we evaluate its performance on the test set. Common evaluation metrics for this task include accuracy, precision, recall, and F1 score. These metrics help assess how well the model is able to correctly identify duplicate question pairs.

Model evaluation is like a report card for our model. It uses metrics to see how well the model is doing. Since model development has many steps, we want to know how well it will predict future outcomes. So, evaluating the model is important to check its performance. It also helps us identify any weaknesses in the model. It's like giving the model a grade to see how good it is at making predictions [14].

3.7. Model Deployment

If the model performs well on the test set, we can deploy it for real-world use. This could involve integrating it into a larger system or creating an API that allows users to input question pairs and receive predictions.

Deployment is like taking our trained ML model out of its training phase and putting it into action in a real application. It's a very important step because the model needs to work in the actual environment where it's meant to solve problems and be useful Only after deployment can the model truly serve its purpose and tackle the challenges it was created for.

The four steps to deploy a machine learning model are:

- Create the model in a training environment.
- Test and clean the code to get it ready for deployment.
- Prepare for deployment using containers.
- Plan for continuous monitoring and maintenance after deployment.

In the vast landscape of machine learning deployment tools, a plethora of options awaits, including Amazon SageMaker, Microsoft Azure, TensorFlow, Docker, and more. However, in the course of our research endeavor, our esteemed deployment team made a strategic choice to implement our project on the esteemed platform of Amazon SageMaker. This judicious selection aligned seamlessly with our project's unique requirements, paving the way for seamless integration and optimized performance.

3.8. Model Iteration

Machine learning models can be improved by iterating on the previous steps. This may include collecting more data, experimenting with different features or algorithms, or fine-tuning the model's hyper parameters.

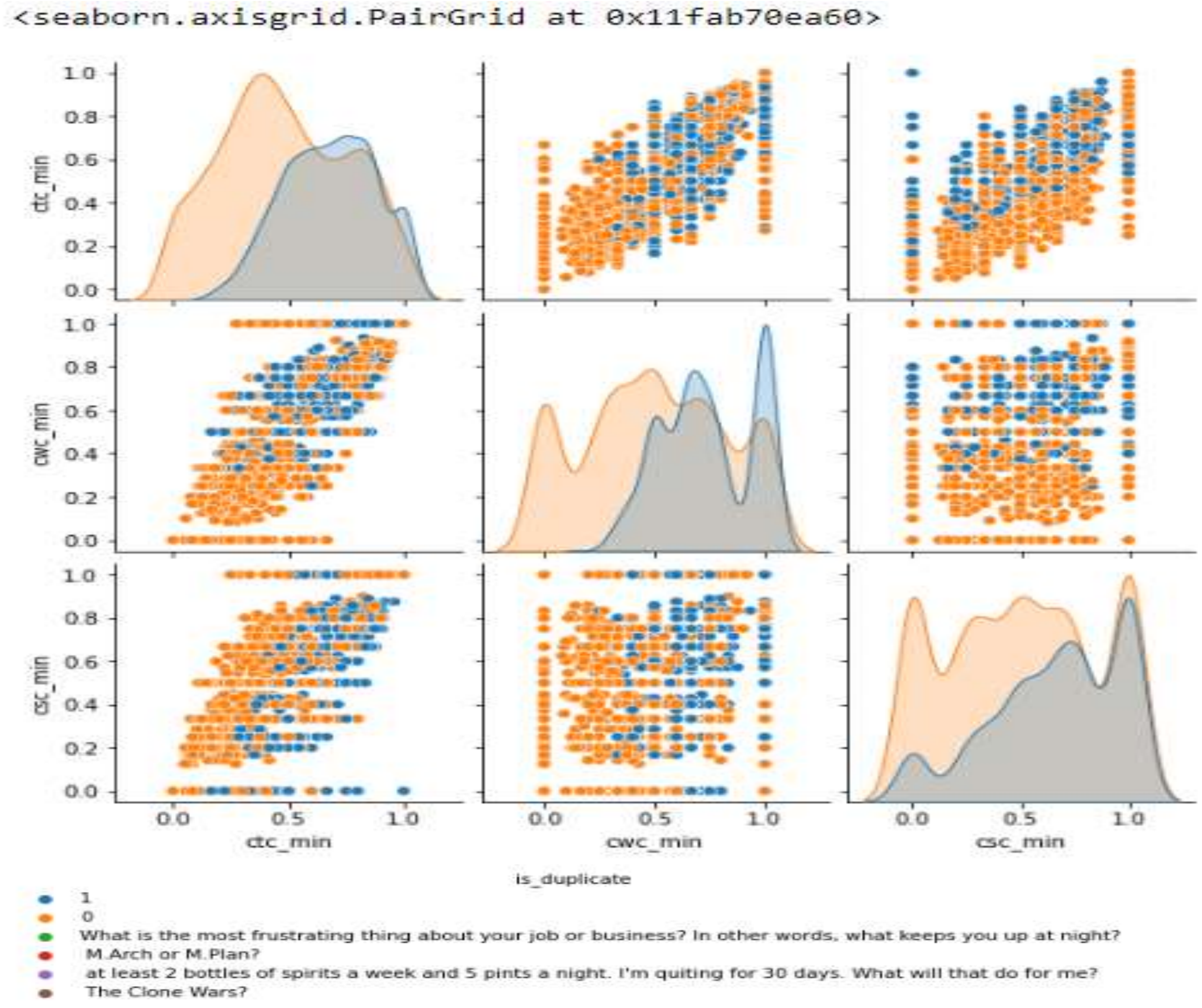
By following these steps, we can build a machine learning model that can automatically identify duplicate question pairs. Remember, the success of the project relies on having a good dataset, appropriate feature engineering, and selecting the right machine learning model for the task.

Through the proficient utilization of the seaborn library, a wealth of graphics calculations emerges, fueled by essential parameters such as the minimum length of common token count, minimum length of common word count, minimum length of common stop 0

counts, and the all-important "is duplicate" indicator that discerns duplicate questions. These meticulously crafted graphs illuminate the research landscape, bestowing upon us a lucid understanding of the intricacies of duplicate question pairs. As we traverse the graphed terrain, valuable insights and patterns surface, affording us a comprehensive view of the question similarity dynamics, poised to enhance the efficacy of our research endeavor shown below in Figure 6.

4. TF-IDF with ML Models

The esteemed Random Forest algorithm yielded an accuracy of 72.00%, a performance closely comparable to the 83.7% achieved in the same literature [15]. This compelling evidence underscores the effectiveness of ML models, such as Random Forest, in delivering impactful results akin to the achievements of Deep learning algorithms like LSTM. Our findings shine a light on the considerable potential of ML models as powerful contenders in solving complex problems, rivaling the prowess of their Deep learning counterparts.



```
<seaborn.axisgrid.PairGrid at 0x11fab70ea60>
```

Figure 6: Seaborn Pair Plot

Table 1: Performance of ML classifiers with its Accuracy

ML classifiers	Accuracy
Random Forest	72.00
Decision Tree	65.00
AdaBoost	62.50
SVM	64.16
Naïve Bayes	62.16

5. CONCLUSION AND FUTURE WORK

In adherence to best practices, we diligently divide our data into an 80% training set and a 20% test set, meticulously upholding this ratio across all experiments. It is of paramount importance to maintain a proportionate distribution of class labels within the test data set, reflecting the original dataset's characteristics. Our pursuit of optimal performance leads us to select hyperparameters through grid search, conducted on a representative 10% subset of the training set. This strategic approach ensures that our results remain safeguarded against overfitting, laying the groundwork for reliable and accurate outcomes.

The outcomes of our research, which encompassed TF-IDF and ML classifiers, unveiled varying levels of success in conjunction with TF-IDF character level. Among them, our top-performing model, Random Forest, demonstrated an accuracy of 72.00%. This compelling evidence showcases the efficiency of machine learning models in addressing the challenge of detecting semantically similar questions in natural language. In comparison to several deep learning methods, including LSTM and LSTM with Siamese, our machine learning approach using TF-IDF with Random Forest surpassed them, signifying its superiority in achieving commendable results.

Our achieved accuracy closely aligns with the state-of-the-art results reported by Quora [16], which stands at an impressive 87%. The disparity in outcomes arises due to Quora's utilization of their proprietary word embedding's, tailor-made for their unique question format and dataset. In contrast, our approach and results hold greater relevance to a broader range of general question and answering systems. This distinction emphasizes the versatility and applicability of our methods, paving the way for broader adoption in various contexts beyond Quora's specialized domain.

An alternative avenue for Quora to enhance their results lies in pre-processing their original question pair dataset. By understanding the context in which questions are posed, a judicious replacement of certain pronouns becomes feasible, potentially leading to improved accuracy. For instance, pronouns like "us," "we," and "they" could be replaced with their contextually relevant counterparts, such as "American," "Programmers," and "Prisoners," during the data preprocessing stage. This meticulous approach could yield superior outcomes. However, due to the lack of information regarding the specific context in which questions were asked, such pre-processing steps were not feasible on the original dataset.

The constraints outlined in the preceding paragraph hold the potential to pave the way for future research in the context of other social media platforms or Quora. Our research endeavors were conducted on a standard Intel Core i3 laptop with 4 GB of RAM, without additional GPU capacity. Training the TF-IDF+Random Forest model demanded close to 7 hours. Enhancing our GPU capacity could potentially yield slightly improved results, allowing for the exploration of constructing more deep learning models and conducting hyperparameters tuning experiments. These avenues hold promise for further enhancing the efficacy and scope of our research.

REFERENCES

1. G Wang, K Gill, M Mohanlal, H Zheng, and B Y Zhao**. Wisdom in the social crowd: An analysis of Quora,** Proceedings of the 22nd international conference on World Wide Web Pages 1341–1352, May 2013.
2. Yoon Kim. 2014. **Convolutional Neural Networks for Sentence Classification.** In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 1746–1751, Oct. 2014.
3. Steven Bird, Ewan Klein, and Edward Loper. [n.d.]. **Natural language processing with Python,** June 2009.
4. Dagna Bogdanova, C´ıcero dos Santos, Luciano Barbosa, and Bianca Zadrozny. **Detecting Semantically Equivalent Questions in Online User Forums**. Proceedings of the 19th Conference on Computational Language Learning, pages 123–131, July 30-31, 2015.
5. R. Raj, and A. Kos, "**Artificial Intelligence: Evolution, Developments, Applications, and Future Scope**", PRZEGLĄD ELEKTROTECHNICZNY, vol. 2, Feb. 2023, pp. 1-13, doi: 10.15199/48.2023.02.01.
6. Dadashov, Elkhan, Sukolsak Sakshuwong and KatherinYu, "**Quora Question Duplication**." web.stanford.edu, 2017.
7. R. Raj, and A. Kos, "**A Comprehensive Study of Optical Character Recognition**", 2022 29th International Conference on Mixed Design of Integrated Circuits and System (MIXDES), Wroclaw, Poland, June 2022, doi: 10.23919/MIXDES55591.2022.9837974.
8. Zhiguo Wang, Wael Hamza, Radu Florian. **Bilateral Multi-Perspective Matching for Natural Language Sentences.** Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, Pages 4144-4150, 2017.
9. R. Raj, and A. Kos, "**Different Techniques for Human Activity Recognition**", 2022 29th International Conference on Mixed Design of Integrated Circuits and System (MIXDES), Wroclaw, Poland, June 2022, doi: 10.23919/MIXDES55591.2022.9838050.
10. G Hinton. **Dropout: A Simple Way to Prevent Neural Networks from Overfitting**., The Journal of Machine Learning Research (JMLR) pages 1929-1958, 2014.
11. Jeffrey Pennington, Richard Socher, and Christopher Manning. **GloVe: Global Vectors for Word Representation**. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) Doha, Qatar. Association for Computational Linguistics., pages 1532–1543, 2014.
12. Y Homma, S Sy, and C Yeh. **Detecting Duplicate Questions with Deep Learning.** 30th Conf. Neural Inf. Process. Syst. NIPS 2016.
13. Samuel R. Bowman, Jon Gauthier, Abhinav Rastogi, Raghav Gupta, Christopher D. Manning, and Christopher Potts. **A Fast-Unified Model for Parsing and Sentence Understanding.** In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 1466–1477, 2016.
14. A Tung and E Xu. Determining **Entailment of Questions in the Quora Dataset**, web.stanford.edu, 2017.
15. Ansari, Navedanjum, and Rajesh Sharma. **"Identifying semantically duplicate questions using data science approach: A quora case study."** arXiv preprint arXiv:2004.11694, 2020.
16. Iyer, Shankar, Nikhil Dandekar, and Kornél Csernai. **"First quora dataset release: Question pairs."** data. quora. com, 2017.