CMPT 733 - Big Data Programming 2
Project Report

Duplicate Questions across multiple Question-Answering Forums

Group Name: Data Divers

Neda Zolaktaf
Vaishnavi Malhotra

Content

1. Motivation and Background

Question Answering is a widely studied task in natural language processing where a machine learning model is trained on large corpora and 'taught' to answer questions that are asked in spoken language. While these techniques yield reasonable performances when the domain of the questions is constrained, such systems often fall apart in an open-domain setting. The conventional approaches to remedy this issue involve adding giant knowledge bases (KB) into the pipeline to help the model learn better and more.

Although the results show that such techniques work well, an evaluation of systems that involve a static knowledge base of question-answers from a subset of the internet forums with an independent model trained just to connect a user-posed question to this knowledge base remains unexplored.

Let's say we have a question from stack overflow which is unanswered. Further, consider a question on ask ubuntu having similar intent which has been answered by many users. We intend to build a system that searches for similar questions across multiple question-answering platforms and helps in fast retrieval of answers for the intended question based on similarity.

This poses a challenging task as we do not have a common dataset having question pairs from different platforms with labels distinguishing duplicate and non-duplicate question pairs. We use different NLP and machine learning techniques and harness the prowess of big data to achieve this objective.

2. Problem Statement

The project aims at identifying duplicate questions across various question-answering platforms using NLP and machine learning. This helps in finding answers to various unanswered questions on one question-answering platform by referring to questions having similar intent across other platforms.

3. Data Science Pipeline

3.1. Data collection

We have used 6 individual datasets for detecting duplicate questions pairs, namely, Quora, AskUbuntu, Android, Sprint, Superuser, and Apple. We downloaded the individual datasets from this link. The integrated Quora question pairs dataset was downloaded from kaggle while we got the integrated AskUbuntu dataset from the Stack exchange data dump.

3.2. Data Cleaning and Preprocessing

Data cleaning and preprocessing involved removing HTML tags, punctuations, symbols, stopwords and non-ASCII characters from question pairs. Further, the text in questions was lower-cased and lemmatized. Finally, all datasets had columns `question1`, `question2`, and `is_duplicate` label.

All datasets except AskUbuntu were structured. The `posts.xml` in AskUbuntu dataset was parsed to extract questions and `postlinks.xml` file was parsed to extract duplicate question pair ids. Then we retrieved questions based on the duplicate question pair ids extracted from the `postlinks.xml` file and labeled them duplicates.

The original datasets that we obtained were compressed such that the corpora of the questions was kept distinctly in a file with their ids while the question pair ids were kept in a separate file. After parsing these files, each row was converted into the following fields `qid1`, `qid2`, and `is_duplicate` to `id`, `qid1`, `qid2`, `question1`, `question2`, and `is_duplicate`.

3.3. Exploratory Data Analysis (EDA)

We did some general analysis of the datasets like understanding the structure of the datasets, comparing the number of duplicates and non-duplicates samples present in the dataset, identifying unique questions in each dataset. Table 1 gives an overview of the Quora and AskUbuntu dataset. Table 2 shows details the individual datasets that we use.

We used matplotlib and seaborn to plot visualizations for summarizing the characteristics of data in each forum like length of sentences, distribution of sentence length, analysis of word overlap and word count in questions etc. Fig. 1 to Fig. 7 show such insights.

	Quora	**AskUbuntu**
Data structure	Structured data consisting of question pairs, their ids, and label `is_duplicate`	XML files - `posts.xml` with question-answers - `postlinks.xml` with questions
Total question pairs	404,290	36,947
% of uniq question pairs	66.52 %	72.43 %

Table 1: Overview of the combined Quora and AskUbuntu dataset

	Quora	Apple	Android	Sprint	Superuser	AskUbuntu
Total question pairs	404,350	228,969	239,473	735,280	919,102	2,153,726
% of unique question pairs	63%	99%	99%	99%	99%	99%

Table 2: Overview of the individual datasets.

Plots for Quora Dataset:

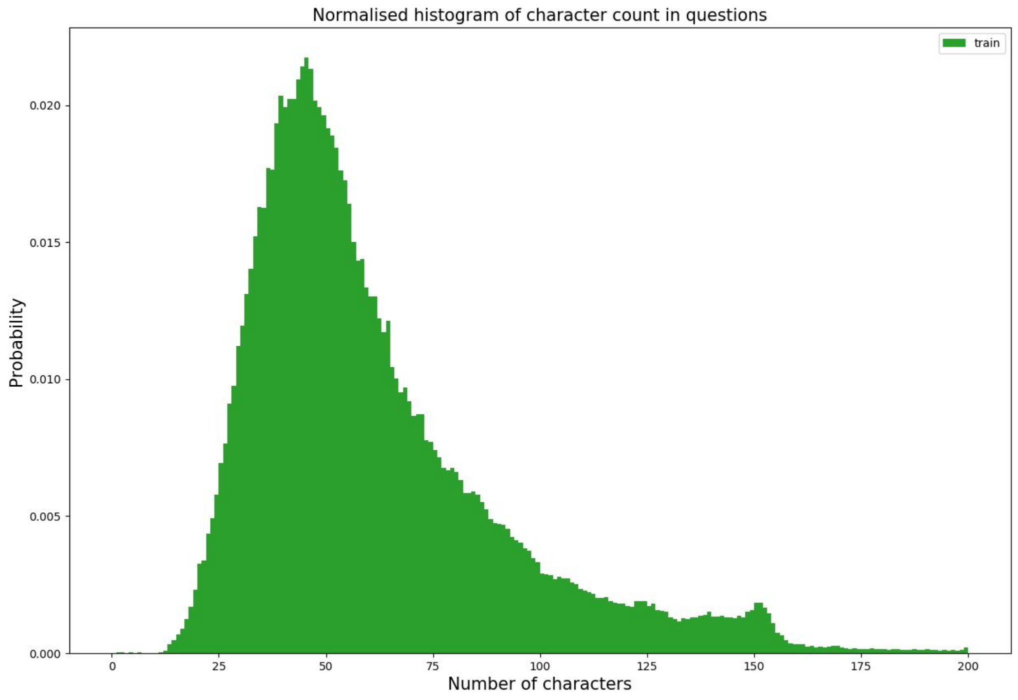


Fig. 1: Histogram showing character count per question in Quora dataset

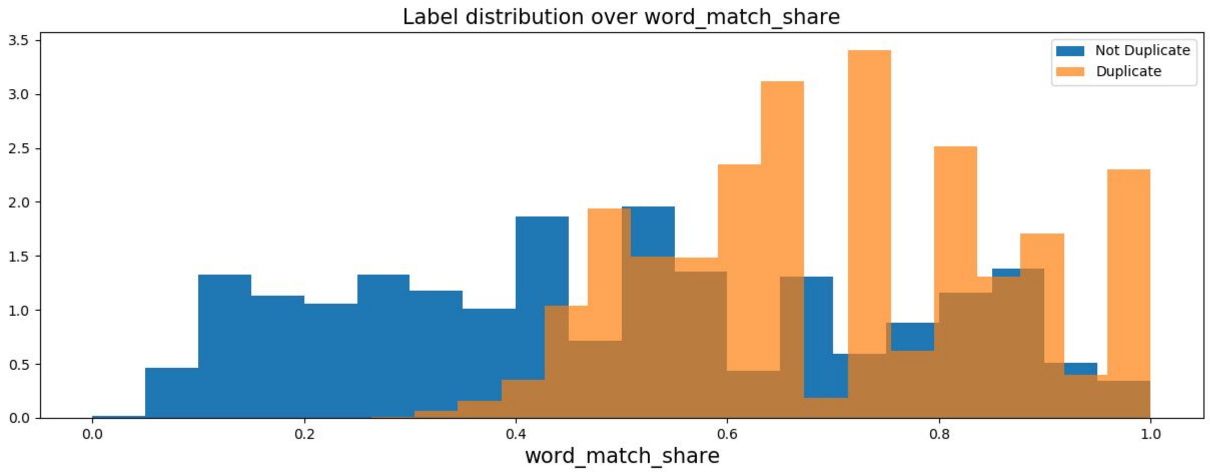


Fig. 2: Histogram showing distribution of label over shared words in Quora dataset

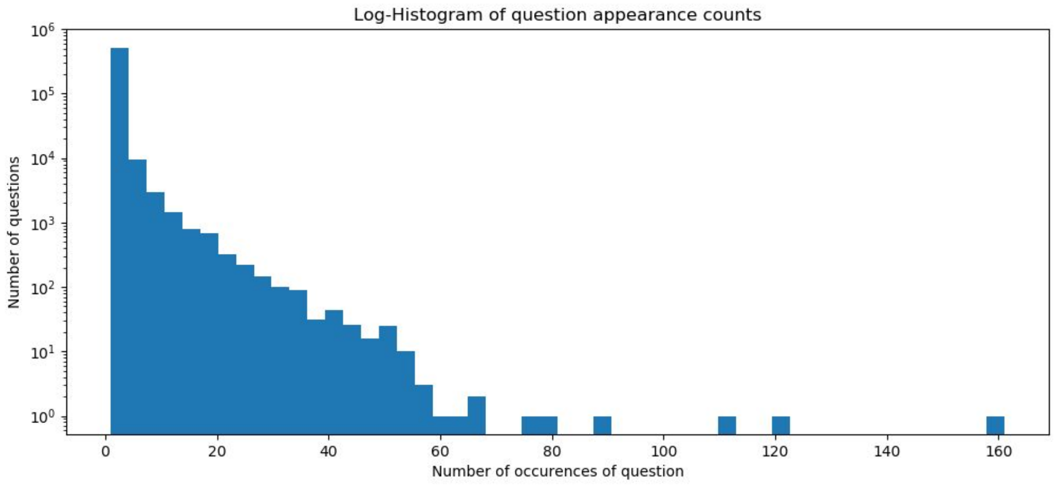


Fig. 3: Log-histogram displaying occurrence of questions and their count in Quora dataset

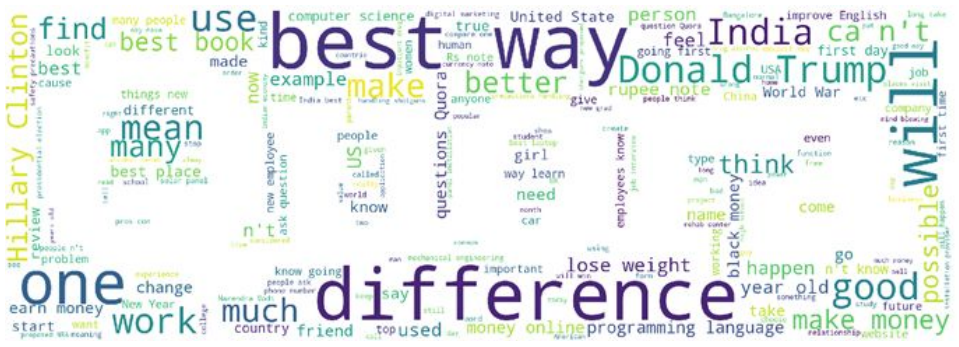


Fig. 4: Word cloud for Quora dataset

Plots for AskUbuntu Dataset:

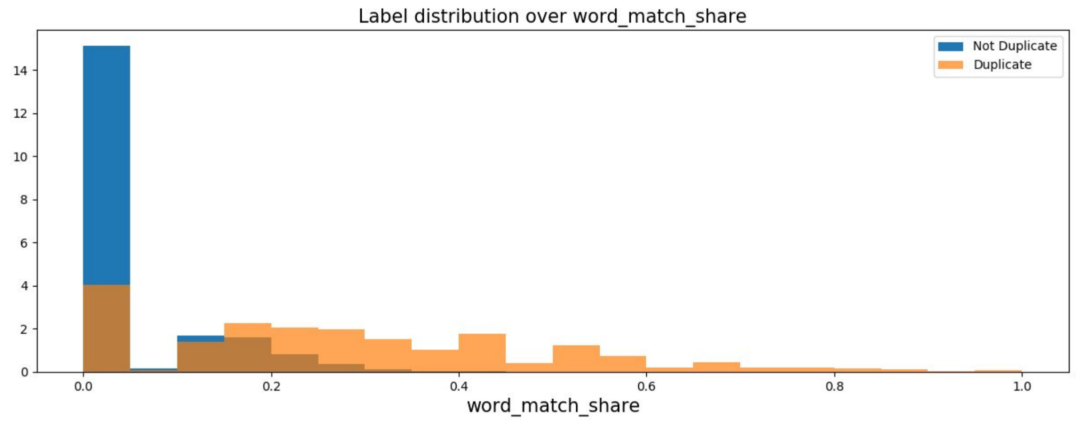


Fig. 5: Histogram showing word count per question in AskUbuntu dataset.

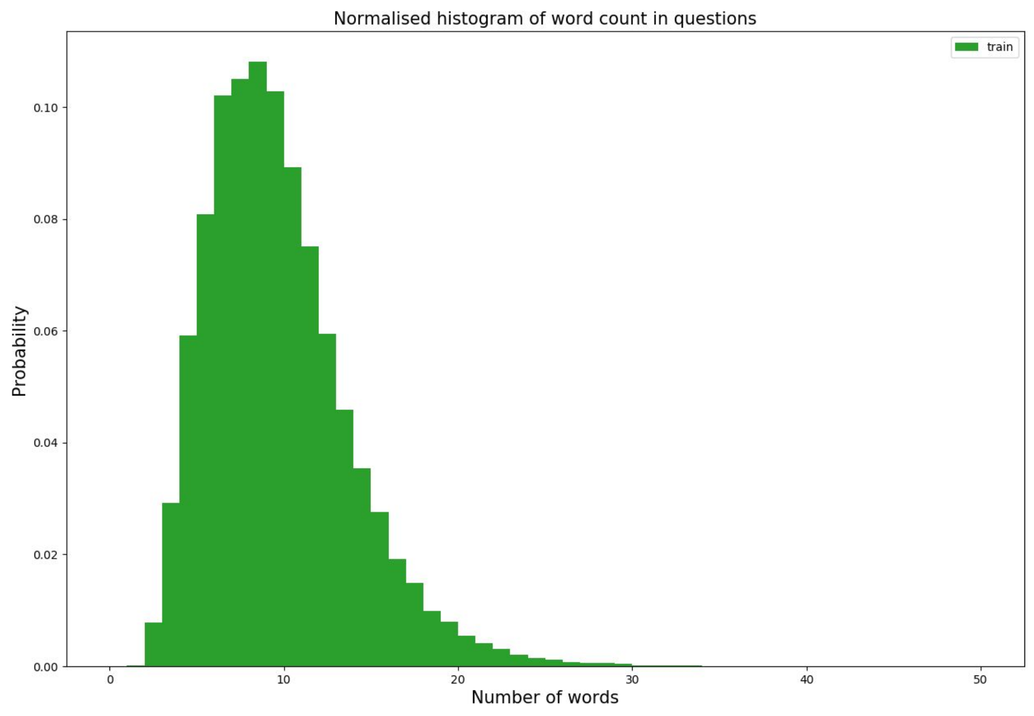


Fig. 6: Histogram showing word count per question in AskUbuntu dataset.

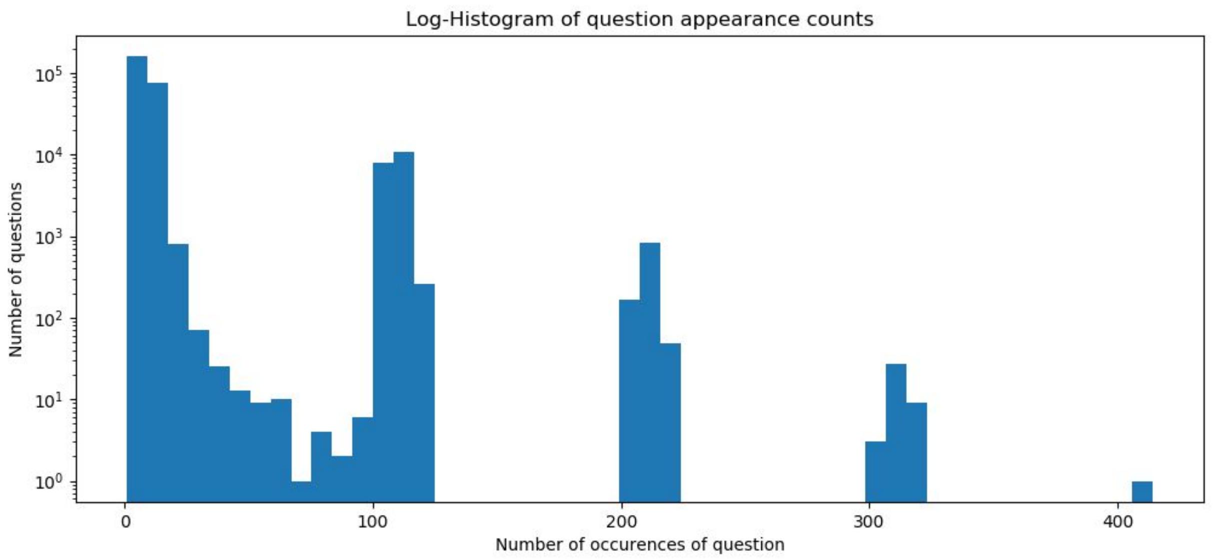


Fig. 7: Histogram showing the distribution of label over shared words in AskUbuntu dataset.

3.4. Data Integration

The Quora and AskUbuntu datasets were integrated into one dataset and different algorithms were implemented on the integrated dataset (Section 3.5 describes the algorithms). Both these datasets had class imbalance problem as highlighted in Fig. 2 and Fig. 5 above. We used the SMOTE [9] technique to perform oversampling of individual datasets and resolve class imbalance issue. The combination of these datasets has proven to be good as the model learns about both general and technical question pairs, which helps the model to differentiate between a wider range of questions.

3.5. Methodology (Solution)

We address the problem of duplicate question detection using a supervised binary classification approach, where we predict whether the two questions are duplicate or not. From the proliferation of binary classification algorithms, we use Logistic Regression [3], Random Forest [1], Extreme Gradient Boosting (XGBoost) [2], and we also use the Manhattan long short term memory (MaLSTM) Neural Network [4].

Logistic Regression is a classification algorithm that uses the logistic sigmoid function to map outputs to two discrete classes. Random Forest is an ensemble learning algorithm that trains multiple decision trees independently to get an accurate prediction. XgBoost is a boosting method, which uses weak learners sequentially to improve the predictions. Logistic Regression, Random Forest, and XgBoost require a vector of features for every question pair example. For these algorithms, we manually extract feature vectors for every question pair example. We consider using n-gram TFIDF, and character level TFIDF, and Bag of Words (BOW) as features for question pair examples, where we represent each pair of questions by concatenating the feature vectors of the two questions.

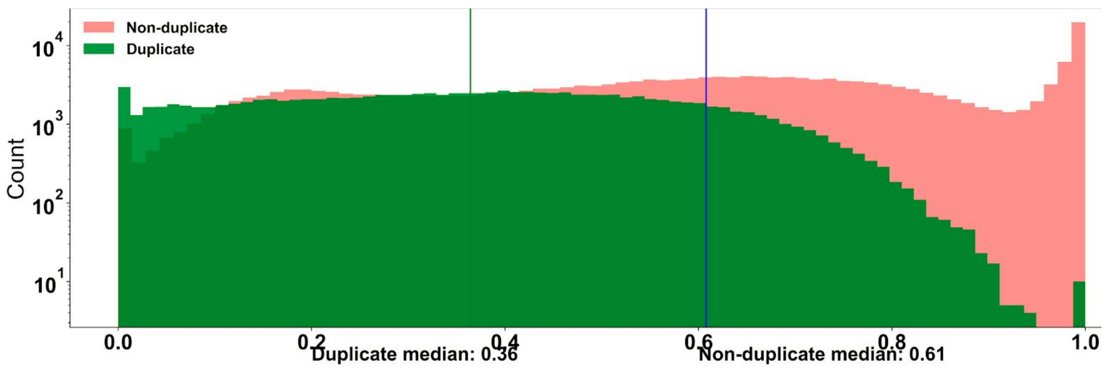


Fig 8: Cosine distance for duplicate and non-duplicate question pairs

In Fig. 8, we plot a histogram of the cosine distance between the 1-gram TFIDF vectors of duplicate questions and non-duplicate questions. The TF-IDF vectors are extracted from the training set. The duplication questions have a median of 0.36 for their cosine distance, whereas the non-duplicate questions have a median of 0.61 for their cosine distance.

In addition to the statistical algorithms described above, we also use the MaLSTM network. An overview of the MaLSTM network is given in Fig. 9. The MaLSTM network is a siamese adaptation of the long short term memory network. Siamese neural networks have two or more identical sub-networks in them and are commonly used for labeled data that have pairs of variable-length sequences, such as sentence similarity and recognizing forged signatures. The MaLSTM network uses word embeddings of the sentences, word2vec vectors, as input and computes sentence similarity using Manhattan distance between the output of sub-networks. The network uses the Mean Squared Error (MSE) between the predictions and the dataset as the loss function to train the network.

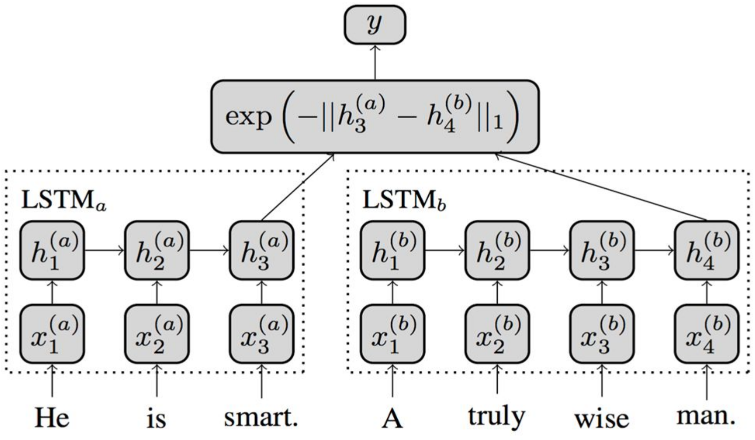

y
$\exp\left(-\|h_3^{(a)} - h_4^{(b)}\|_1\right)$
LSTM$_a$
LSTM$_b$
$h_1^{(a)}$ $h_2^{(a)}$ $h_3^{(a)}$
$h_1^{(b)}$ $h_2^{(b)}$ $h_3^{(b)}$ $h_4^{(b)}$
$x_1^{(a)}$ $x_2^{(a)}$ $x_3^{(a)}$
$x_1^{(b)}$ $x_2^{(b)}$ $x_3^{(b)}$ $x_4^{(b)}$
He is smart.
A truly wise man.

Fig 9: Overview of the MaLSTM network [4]. The network reads word embedding vectors and uses the final hidden states as a vector representation for each sentence. Then it uses the Manhattan similarity as the semantic similarity of the questions.

4. Technologies

The pipeline shown in Fig. 10 highlights the tools and analysis methods used in our project.

Apache Spark is a unified analytics engine for large-scale data processing. It is based on Hadoop MapReduce and it extends the MapReduce model to efficiently use it for more types of computations, which includes interactive queries and stream processing. The main feature of Spark is its in-memory cluster computing that increases the processing speed of an application.

Spark is designed to cover a wide range of workloads such as batch applications, iterative algorithms, interactive queries, and streaming. Since we worked on multiple datasets having textual information, using Spark for data-preprocessing helped to speed up the work. Further we also used the Spark-Cassandra connector to connect to Cassandra for the demo.

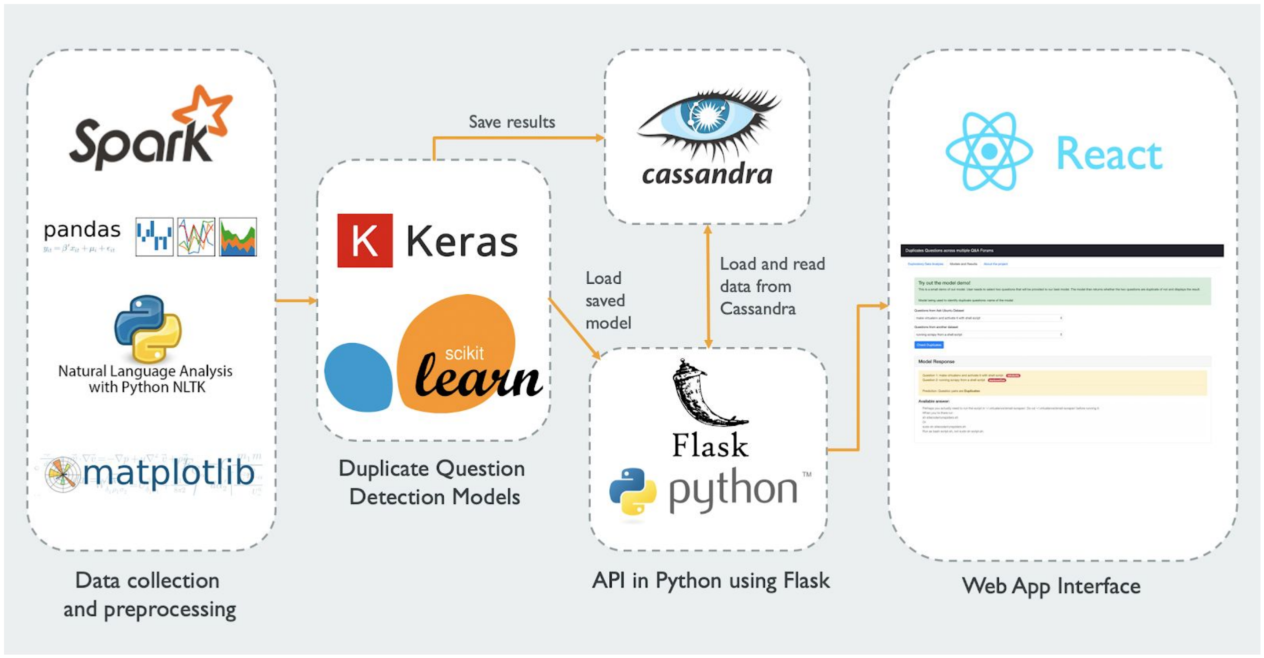

Spark
pandas
Natural Language Analysis with Python NLTK
matplotlib
Data collection and preprocessing
Save results
Keras
scikit learn
Duplicate Question Detection Models
cassandra
Load saved model
Load and read data from Cassandra
Flask
python
API in Python using Flask
React
Web App Interface

Fig. 10: Technologies used and how they interact

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries. We have used NLTK to perform stemming and lemmatization of text in the question pairs for each dataset.

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hard copy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter notebook, web application servers, and four graphical user interface toolkits. We have used Matplotlib in the python scripts to plot graphs in exploratory data analysis of our datasets.

Keras is an open-source neural-network library written in Python which is designed to enable fast experimentation with deep neural networks**.** We have used Keras for simplifying the implementation of models predicting duplicate question pairs.

Scikit-Learn is an open-source machine learning library in python built on top of NumPy, SciPy, and matplotlib. It provides simple and efficient tools for data mining and analysis. We have used scikit-learn to implement and evaluate models for detecting duplicate question pairs.

Apache Cassandra is a free and open-source, distributed, wide column store, NoSQL database management system designed to handle large amounts of data across many commodity servers, providing high availability with no single point of failure. We have worked with Cassandra to store dataset used for our demo.

Flask is a microframework for Python which simplifies the process of designing a web application. Flask helps to handle user's requests as well as also takes care of the type of response that should be given back. We have used Flask for building API in python as well as for running react app and connecting it to python.

React is a JavaScript library for building interactive user interfaces by building and composing encapsulated components that manage their own state. We have created an interactive web application using react and flask which demonstrates the working of our project by interacting with python APIs using Flask.

5. Evaluation and Results

We conduct two sets of experiments. First, we conduct experiments on individual datasets (Quora, Apple, Android, Sprint, Superuser, and AskUbuntu). Second, we conduct experiments on our integrated dataset (Quora and AskUbuntu). We use 80% of the datasets as training set (and also validation set) and 20% as testing set.

Table 3 shows the performance of Logistic Regression with TF-IDF vectors, Random Forest with TF-IDF vectors, and the MaLSTM network on the individual testing datasets. Logistic Regression with TF-IDF performs the best on the Quora testing dataset in terms of the F-Score measure.

Logistic Regression and Random Forest are trained based on the accuracy score. We use cross-validation to tune the hyper-parameter, including the hyperparameters of the TF-IDF vectors and hyperparameters of the learning algorithms. The hyperparameters of the TF-IDF vectors that were tuned were the n-gram length (1-gram, 2-gram, and 3-gram) and a cut-off threshold for discarding n-grams that have a low frequency (0, 100, and 200 frequency threshold). For Logistic Regression we tuned the regularization strength (1 and 1/100) and we also considered both weighted and non-weighted classes. For Random Forest, we tune the number of trees (100 and 300), the maximum depth of the trees (10 and 100), and the split criterion (entropy and gini). For the MaLSTM network, we use word2vec vectors as input (word embedding dimension is set to 300). We use a one-layer LSTM, were the hidden dimension is set to 50 for all datasets and for Quora we additionally set it to 100. The number of epochs is set to 5 for all datasets and for Quora we additionally used 30 epochs but did not obtain improvements. We used a batch size of 1024 and we used the Adam optimizer to minimize the loss function.

The MaLSTM network is trained based on the MSE loss function. We also used the cross-entropy loss but did not obtain improvements. Fig. 11 shows how the train and validation error changed during training on the Quora dataset.

Table 4 shows the performance of XGBoost with BOW, word level TFIDF, n-gram TFIDF and character-level TFIDF on our integrated testing dataset. XgBoost with character-level TFIDF performs the best in terms of Precision, Recall, and F-Score on the integrated Quora and Ask Ubuntu testing dataset.

Model	Quora		Apple		Android		Sprint		Superuser		AskUbuntu	
	Accuracy	Fscore	Accuracy	Fscore	Accuracy	Fscore	Accuracy	Fscore	Accuracy	Fscore	Accuracy	Fscore
TFIDF + Logistic Regression	0.7554	0.6336	0.9906	0.1161	0.9901	0.1960	0.9901	0.	0.9902	0.0319	0.9899	0.0091
TFIDF + Random Forest	0.7783	0.6248	0.9911	0.1948	0.9922	0.4335	0.9900	0.0010	0.9909	0.1639	0.9917	0.3370
MaLSTM	0.6756	0.2814	0.991	0.002	0.9899	0.001	0.9917	0.05	0.9904	0.006	0.9900	0.003

Table 3: Accuracy and F-score of different models on individual testing datasets.

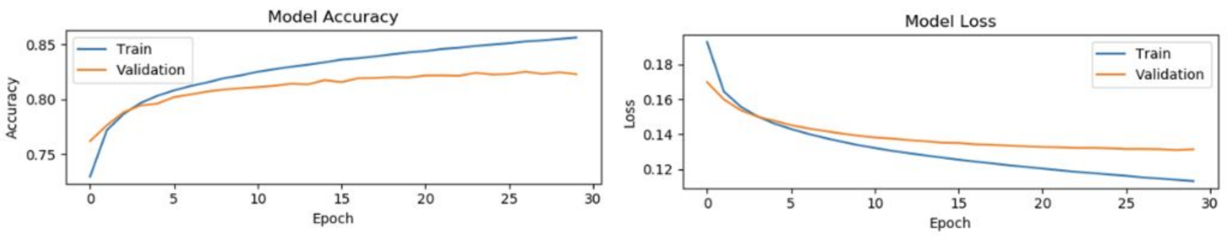


Fig. 11: MaLSTM model loss and accuracy on the Quora training and validation dataset.

Model	Precision	Recall	Fscore
BOW + XGBoost	0.82	0.61	0.70
Word level TFIDF + XGBoost	0.82	0.62	0.71
n-gramTFIDF+ XGBoost	0.81	0.41	0.54
Character-level TFIDF + XGBoost	0.84	0.72	0.78

Table 4: Precision, Recall, and F-score of XGBoost with different features on our integrated testing dataset (Quora + AskUbuntu).

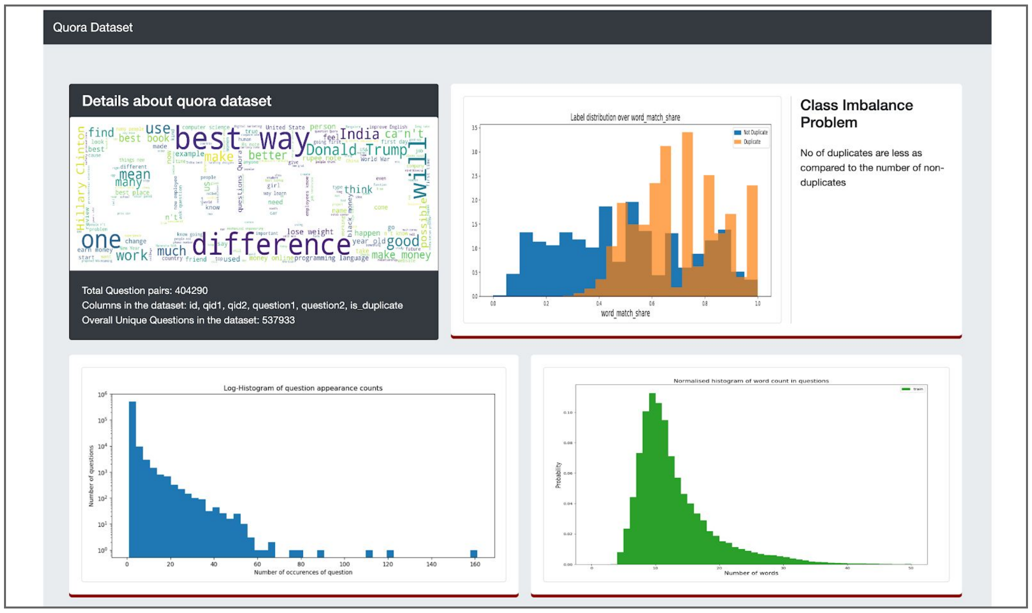
6. Data Product

The final product is an interactive web app interface that allows the user to select questions pairs from different datasets and determines similarity, the answers to questions if they are duplicate and suggestions to similar questions. This is explained in detail in section 6.2 below.

The interface is divided into three parts as follows:

1. Exploratory Data Analysis (EDA)
2. Models and Results - The Demo
3. Project Information.

6.1. Exploratory Data Analysis (EDA)


Quora Dataset
Details about quora dataset
Label distribution over word_match_share
Not Duplicate
Duplicate
word_match_share
Log-Histogram of question appearance counts
Number of questions
Number of occurences of question
Normalised histogram of word count in questions
Probability
Number of words

Total Question pairs: 404290
Columns in the dataset: id, qid1, qid2, question1, question2, is_duplicate
Overall Unique Questions in the dataset: 537933

Class Imbalance Problem

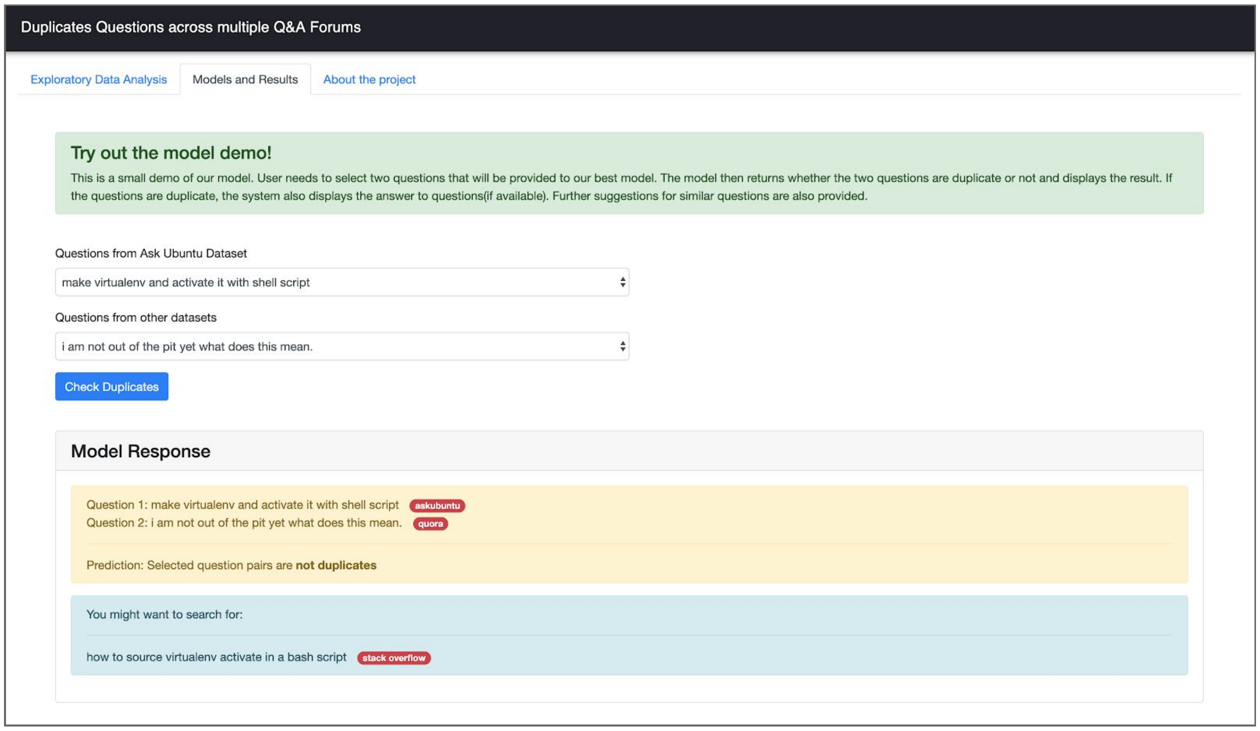
No of duplicates are less as compared to the number of non-duplicates

Fig. 12: Plots for Quora dataset in the Exploratory Data Analysis tab of the interface

The EDA page consists of visualizations for summarizing the characteristics of data like the number of duplicate and non-duplicate question pairs, length of sentences, word overlaps in question pairs. For Example, Fig. 12 shows various visualization plots for the Quora dataset. Similarly, we have plots for other datasets used in the project.

6.2. Models and Results - Demo

This page provides a small scale working demo of the project. The demo uses Character-level TFIDF + XGBoost model which gave best results on our integrated Quora and AskUbuntu datasets as mentioned in Section 5.


Duplicates Questions across multiple Q&A Forums

Exploratory Data Analysis | Models and Results | About the project

Try out the model demo!

This is a small demo of our model. User needs to select two questions that will be provided to our best model. The model then returns whether the two questions are duplicate or not and displays the result. If the questions are duplicate, the system also displays the answer to questions(if available). Further suggestions for similar questions are also provided.

Questions from Ask Ubuntu Dataset

make virtualenv and activate it with shell script

Questions from other datasets

i am not out of the pit yet what does this mean.

Check Duplicates

Model Response

Question 1: make virtualenv and activate it with shell script askubuntu
Question 2: i am not out of the pit yet what does this mean. quora

Prediction: Selected question pairs are **not duplicates**

You might want to search for:

how to source virtualenv activate in a bash script stack overflow

Fig. 13: The model and results section in interface displaying prediction and similar questions for non-duplicate question pair

Prerequisite for demo

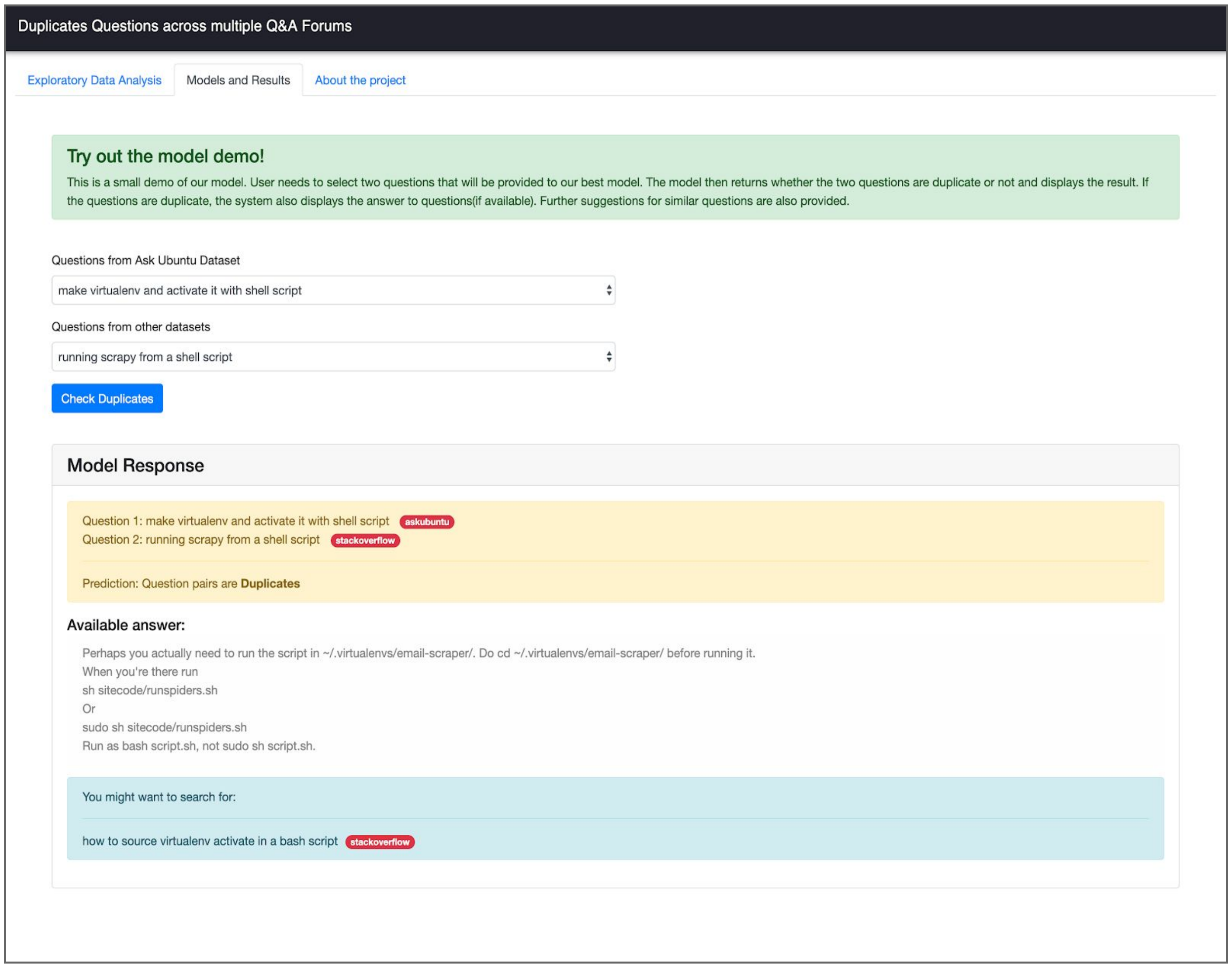
- **Data:** The demo provides two drop downs, first contains a set of questions from AskUbuntu datasets and the second dropdown contains sample questions from other datasets like Stack Overflow, Quora, English dataset (Stack Exchange).

- **Data Storage**: Cassandra is used for storing datasets. Information stored in tables in Cassandra is as follows: Question id, Question text, Answer, Tag(forum/ platform from where the question has been taken), Similar Question(based on labels predicted by the model, question pairs with label duplicate are stored as similar questions in cassandra)

- The similar question suggestions shown in the demo are related to question selected in dropdown 1.

Working of demo

As mentioned in section 3.4, we have integrated quora and AskUbuntu datasets, so that the model learns about both general and technical questions. Then we used different models to detect duplicate question pairs in the integrated dataset. Since character level TF-IDF with XGBoost gave the best performance for this integrated dataset, hence we have used it for the demo.



Duplicates Questions across multiple Q&A Forums

Exploratory Data Analysis | Models and Results | About the project

Try out the model demo!

This is a small demo of our model. User needs to select two questions that will be provided to our best model. The model then returns whether the two questions are duplicate or not and displays the result. If the questions are duplicate, the system also displays the answer to questions(if available). Further suggestions for similar questions are also provided.

Questions from Ask Ubuntu Dataset

make virtualenv and activate it with shell script

Questions from other datasets

running scrapy from a shell script

Check Duplicates

Model Response

Question 1: make virtualenv and activate it with shell script askubuntu
Question 2: running scrapy from a shell script stackoverflow

Prediction: Question pairs are **Duplicates**

Available answer:

Perhaps you actually need to run the script in ~/.virtualenvs/email-scraper/. Do cd ~/.virtualenvs/email-scraper/ before running it.
When you're there run
sh sitecode/runspiders.sh
Or
sudo sh sitecode/runspiders.sh
Run as bash script.sh, not sudo sh script.sh.

You might want to search for:

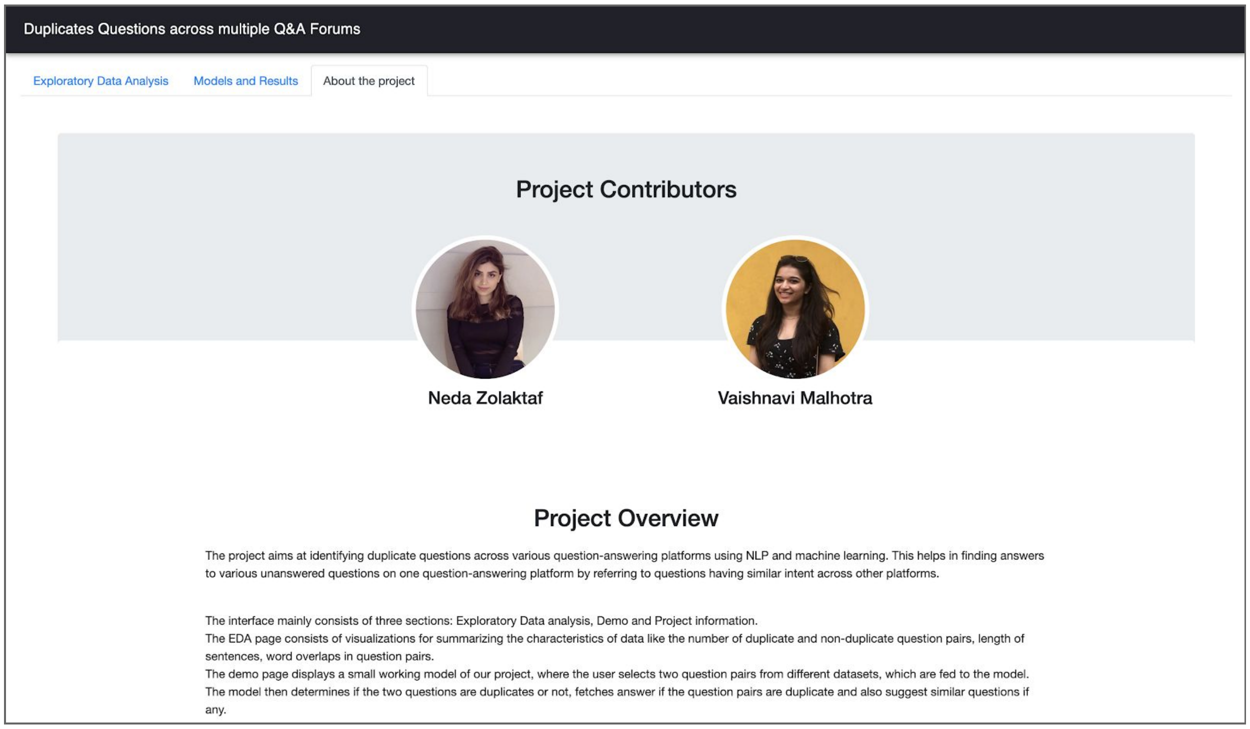
how to source virtualenv activate in a bash script stackoverflow

Fig. 14: The model and results section displaying prediction, answer and similar question suggestion for duplicate question pair.

Once the user selects questions from the dropdown, these questions are preprocessed i.e. converted to feature vectors which are then provided as inputs to the stored model (character level TF-IDF). The model then predicts whether the question pair is duplicate or not. Once the prediction is made, the API returns the prediction, along with answers to the questions if they are duplicate and suggestions for similar questions if any.

Fig. 13 displays API's response when the selected question pair is non-duplicate. Model's prediction and similar question suggestions are displayed if the selected question pair is predicted non-duplicate. While, in case the selected question pair is labeled duplicate by the model, the API also fetches answers to selected questions(if they are available) along with prediction and suggestions for similar questions. Fig. 14 shows response in case model predicts the selected question pair as duplicate.

6.3. Project Information



The project aims at identifying duplicate questions across various question-answering platforms using NLP and machine learning. This helps in finding answers to various unanswered questions on one question-answering platform by referring to questions having similar intent across other platforms.

The interface mainly consists of three sections: Exploratory Data analysis, Demo and Project information.
The EDA page consists of visualizations for summarizing the characteristics of data like the number of duplicate and non-duplicate question pairs, length of sentences, word overlaps in question pairs.
The demo page displays a small working model of our project, where the user selects two question pairs from different datasets, which are fed to the model.
The model then determines if the two questions are duplicates or not, fetches answer if the question pairs are duplicate and also suggest similar questions if any.

Fig. 15: The project information section in the interface.

This section gives an overview of the project. It also provides information about the working of the demo and the technologies used in the project. All information mentioned in this section has already been covered in detail in the report. Screenshot of the project information section is displayed in Fig. 15.

7. Lessons Learned

- **Stemming and Lemmatization are not always helpful.** Using Porter Stemmer and lemmatizer from Natural Language Toolkit did not improve the results, in fact, it lead to information loss in our case. For instance, after lemmatization pronouns were replaced by the word -PRON- in all questions. In some cases, words used in similar content were lemmatized differently like, friends was lemmatized to friend while friendship was lemmatized to friendship.

8. Summary and Future Work

In this project, we worked on 6 individual (Quora, Apple, Android, Sprint, Superuser, and AskUbuntu) and an integrated dataset (Quora and AskUbuntu) to tackle the problem of duplicate question detection across multiple question-answering platforms. Firstly, in our integrated dataset, we used oversampling to handle the class imbalance problems highlighted during the EDA of our datasets. After the cleaning and preprocessing of data, we implemented and compared different models on individual and integrated datasets and recorded the results. For the models, we used Logistic Regression, Random Forest, XGBoost with different levels of TF-IDF and BOW vectors and also a MaLSTM neural network with word2vec input vectors.

Further, we used the best performing model to build a small scale working system (web app interface) which detects duplicate questions across multiple forums and retrieves answers and similar questions based on the questions selected in the demo. The system performed well on the small dataset containing questions from different sources.

With further improvements in the model, and integrating datasets containing question pairs from different topics to increase knowledge base, we can build a more robust system which tackles the problem of duplicate question detection on a large scale and helps in easy and fast answer retrieval for duplicate questions.

As part of future work, we would:

1. try to finetune the siamese neural network that we use and use a different objective function.
2. consider using other neural network models, such as convolutional neural networks to improve the quality of our predictions.
3. consider integrating records from more datasets so the model learns about a wider set of questions.

References

[1] Breiman L. Random forests. Machine learning. 2001 Oct 1;45(1):5-32.

[2] Friedman JH. Greedy function approximation: a gradient boosting machine. Annals of statistics. 2001 Oct 1:1189-232.

[3] Hosmer Jr DW, Lemeshow S, Sturdivant RX. Applied logistic regression. John Wiley & Sons; 2013.

[4] Mueller J, Thyagarajan A. Siamese recurrent architectures for learning sentence similarity. In Thirtieth AAAI Conference on Artificial Intelligence 2016 Mar 5.

[5] Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J. Scikit-learn: Machine learning in Python. Journal of machine learning research. 2011;12(Oct):2825-30

[6] https://github.com/aspk/Quora_question_pairs_NLP_Kaggle/blob/master/bow_logistic_xgb_with_plots.ipynb

[7] https://medium.com/mlreview/implementing-malstm-onkaggles-quora-question-pairs-competition-8b31b0b16a07

[8] train.py for [5] based on code from: https://github.com/likejazz/Siamese-LSTM

[9] https://www.cs.cmu.edu/afs/cs/project/jair/pub/volume16/chawla02a-html/chawla2002.html