Islamophobia Detector

Our free topic is to create a tool that detects islamophobia. Islamophobia is prejudice against Islam or Muslims. Such prejudice can manifest in speech, comments, news articles, and other sources. Creating a text classifier with the ability to identify islamophobic rhetoric allows us to efficiently classify various news articles, subreddits, and other web sources as islamophobic. Also, we plan to collect metadata for each analyzed item, which allows us to create visualizations about islamophobia. Simply put, our tool's main functionality is to detect any islamophobic rhetoric, allow users to get an islamophobic classification for text collected from any source, keep track of all classified results, and present such data in understandable visualizations.

This topic piqued our interest because islamophobia is an ongoing and relevant issue. The current protests in Iran are proof that islamophobia is well and alive. Despite the protests being targeted against the government, many are using the situation to justify their hateful messages against Islam and Muslims. The same pattern of hate and prejudice repeats itself throughout history. By creating this tool, we plan on identifying and labeling islamophobic posts to allow users to detect any hateful rhetoric against Islam or Muslims. Also, our tool can be used to collect data and detect patterns and spikes of islamophobia.

We plan to use Python as the main programming language for our tool. Python has a wide selection of libraries and fits our use case perfectly. Also, we need a database to keep track of the collected stats about the various text items collected. We are still researching and looking into various different database options. Depending on our needs, we might go with a traditional database (MySQL, Postgres, MongoDB, etc.), a memory mapped file (loaded at runtime), or a text file dump. We are also going to manually collect a decent number of text snippets from various web sources and manually tag the text as Islamophobic. We will use Google Forms for that part. We decided to follow this approach to create training and testing datasets for our models and choose the model with the highest score. Finally, for our visualization part, we are leaning towards using Prometheus and Grafana. Both tools are perfect for creating numerous visualizations.

For our team of three, we aim to spend 60 hours in total for the project. We expect 20 hours to collect data to build the dataset needed to test and train the classifier. This includes selecting and scraping subreddits and articles into a data frame. Exploring data to decide which labels need to be used for classification. Manually labeling posts/articles using google forms to build training data. For training and testing the model we expect around 10 hours, to test various different models and identify the model with the highest accuracy score. We also expect around 10 hours of work for storing the collected data in our database and developing a python program that uses the collected data and model to classify newly entered text from different sources and store such data in our database. Finally an additional 20 hours of work is expected for setting up

Prometheus and Grafana, creating a metadata exporter for Prometheus, and creating graphs and visualizations for Grafana.

Members:

Nelofer	Arjumand	narjum2Illinois.edu	leader
Emily	Sallenback	emilygs2@illinois.edu	
Mohammed	Alabdullatif	maa27@illinois.edu	