

News articles classifier

capstone project

Vatsala Vaddiraju and Shiuli Das | Software Engineering for Data Science | 31-11-2021

Table of Contents

[Purpose 2](#_Toc86608769)

[OBJECTIVE 2](#_Toc86608770)

[Overview 2](#_Toc86608771)

[System Architecture: 3](#_Toc86608772)

[Architecture Design: 3](#_Toc86608773)

[Architecture diagram: 3](#_Toc86608774)

[System overview: 4](#_Toc86608775)

[Other tools used 5](#_Toc86608776)

[Guidelines 5](#_Toc86608777)

[Machine learning model 5](#_Toc86608778)

[Retraining the model: 5](#_Toc86608779)

[Progress report 6](#_Toc86608780)

[Gantt chart: 7](#_Toc86608781)

[WEEK 1: Data Ingestion Report 7](#_Toc86608782)

[WEEK2: Data Preparation Report 11](#_Toc86608783)

[WEEK3: Model Training and Prediction Report 14](#_Toc86608784)

[Final Output: 19](#_Toc86608785)

[Reference Material 20](#_Toc86608786)

## Purpose

Currently, in digital world, we are overloaded with various kinds of news information coming from different sources. The purpose of this project is to classify entered text into various news articles. In this project as a sample approach, we are letting user enter any news article data and using machine learning model, we are predicting the news category. But this project can easily be extended to show users their choice of interest in news articles and also can be used to retrain the data with new data as needed.

This document shows more details about our approach.

## **OBJECTIVE**

* Design complete solution to demonstrate end-to-end pipeline development and manage a machine/ deep learning project
* Develop a understanding of all stages of a machine learning project lifecycle
* Demonstrate understanding of challenges encountered during the project development and provide ways to tackle them
* Showcase understanding of software engineering best practices while developing the project.

## Overview

Classify News Articles into categories - With information overload today users are inundated with news articles of all topics, even the ones which may not be relevant to users. Design a system which can classify incoming news articles and appropriately tag the corresponding category. Develop a data pipeline which includes the all the following stages of Machine Learning Project Life Cycle.

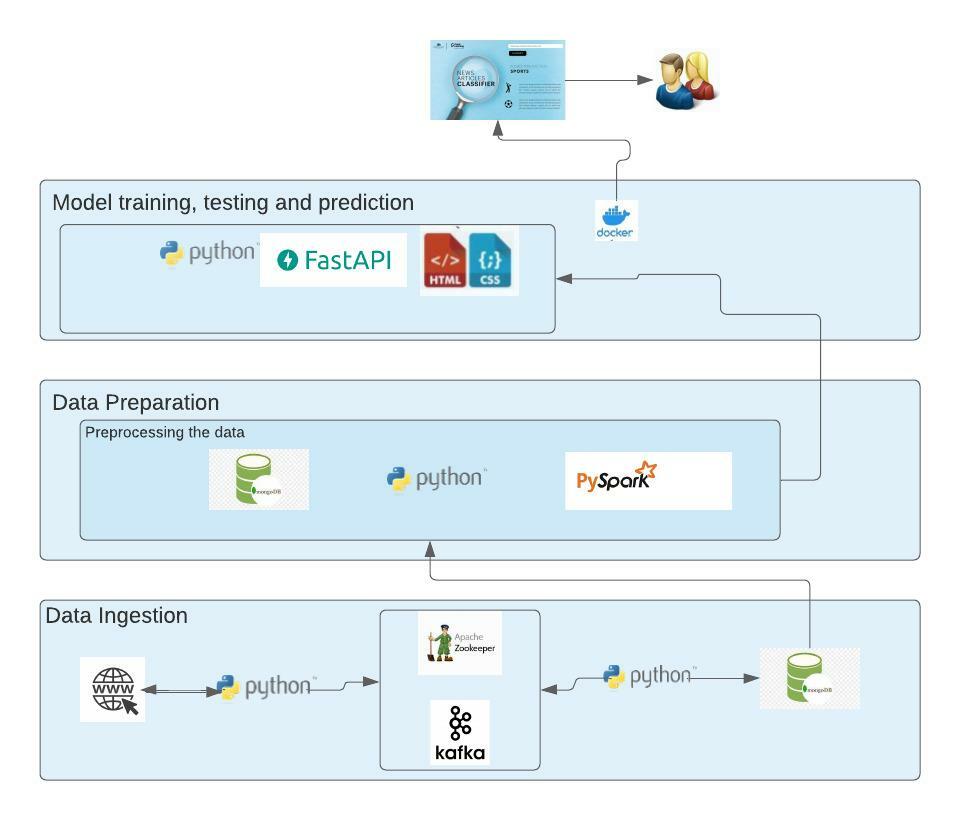
* Data Ingestion
* Data Preparation
* Data segregation & Model Training
* Model Deployment
* Model Prediction

## System Architecture:

### Architecture Design:

We have used loosely coupled architecture, where Data Ingestion, Data Processing, model training and prediction are done as individual components. At each stage modifications and improvements can be done at any stage without impacting other components.

### Architecture diagram:



## 

## System overview:

For each data pipeline, we have used various technologies to fulfill the requirement.

Data Ingestion:

We have used rapid API to collect the data and sent it to kafka queue. Used pymongo to sink kafka topic into MongoDB. Along with the data from our custom ingestion, we have to used Kaggle dataset for better model training.

Following environment details are used for data ingestion:

* Python IDE
* Kafka & Zoo Keeper (using Vagrant)
* MongoDB
* pymongo

Data Preparation and modeling:

We have used raw and preprocessed the data using pyspark. Preprocessed the raw data, trained and tested the model. Save the model as “\*.pkl” file so that it can be used from Front End. Used Fast API to test the prediction using Swagger UI.

Following environment details are used for data preparation and modeling:

* Python IDE
* pyspark
* MongoDB
* Main python packages used are pandas and sklearn
* Fast API

#### User Interface:

We have used a simple user interface, where user can enter any text and the prediction will be displayed using pretrained model.

Following environment details are used for User Interface:

* Html
* Java script
* CSS
* Jquery

## Other tools used

* Plantuml.com for Gantt chart generation
* Lucidchart.com for architectural diagram
* Mongo express
* Docker desktop
* Github
* Docker for dockerization of final project

## Guidelines

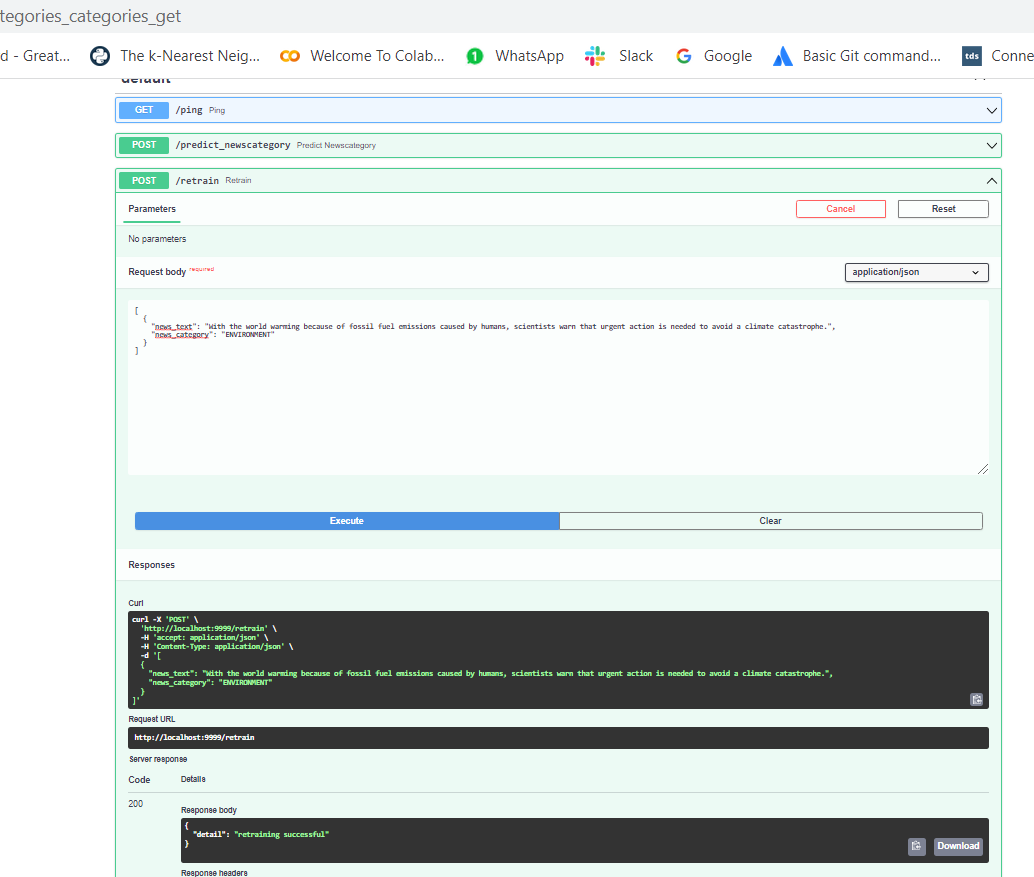
* Followed PEP8 guidelines for python code.
* Followed best practices of REST API development.

## Machine learning model

We have used **Multinomial Naïve Bayes** for our model for processing NLP as it is used for the analysis of the categorical text data. This model is highly scalable and easily handles large volume of data.

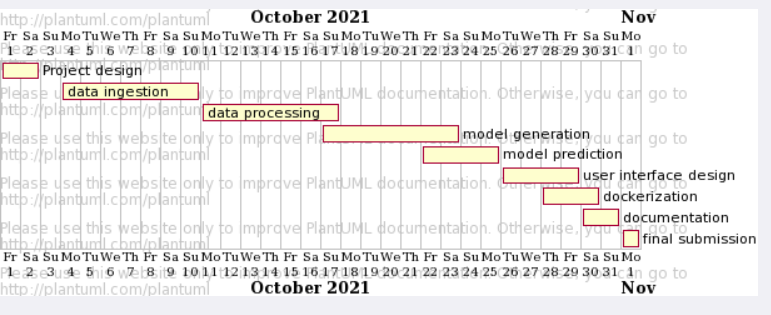
## Retraining the model:

We have provided the retraining capability of our model using feedback loop.



## Progress report

### Gantt chart:

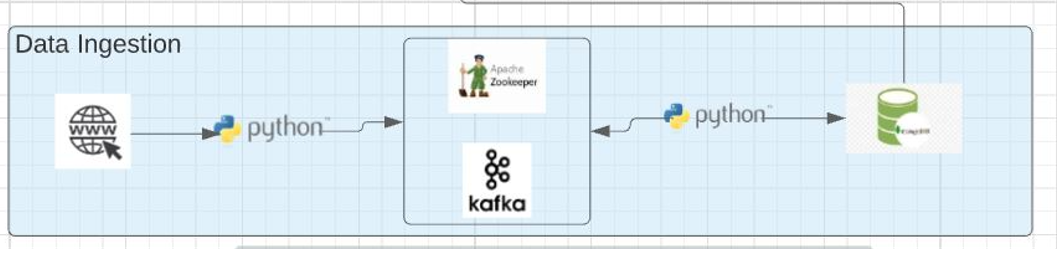


### WEEK 1: Data Ingestion Report

Objective: This week’s objective is to collect data from different sources and save it to a database through Kafka queue.

Approach: We have used rapid API to collect the data and sent it to kafka queue. Used pymongo to sink kafka topic into MongoDB.

High level architecture diagram for this week:



**1. Environment Details:**

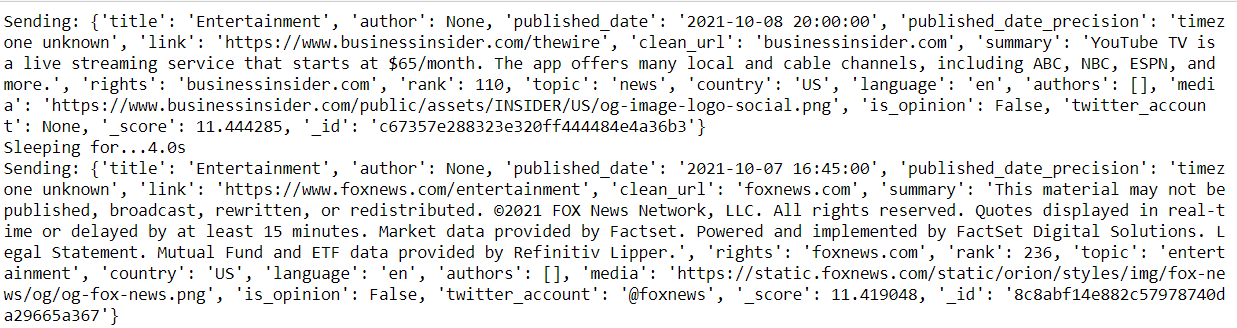
The Data Ingestion process for this project was performed using the following Environment details.

* Python IDE
* Kafka & Zoo Keeper (using Vagrant)
* MongoDB

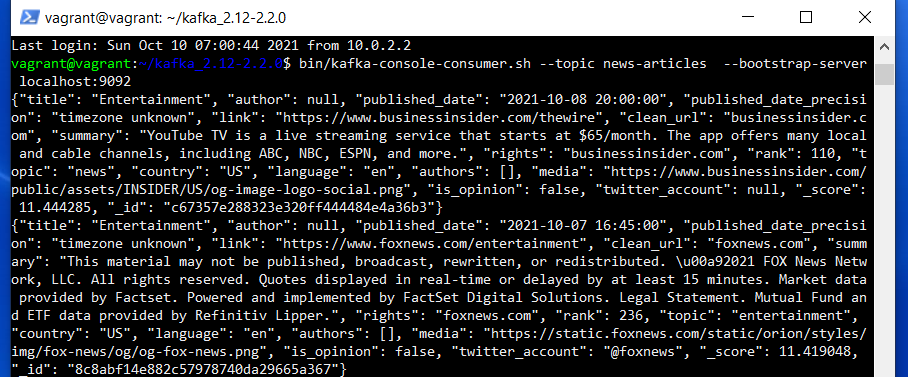
**2. Input:**

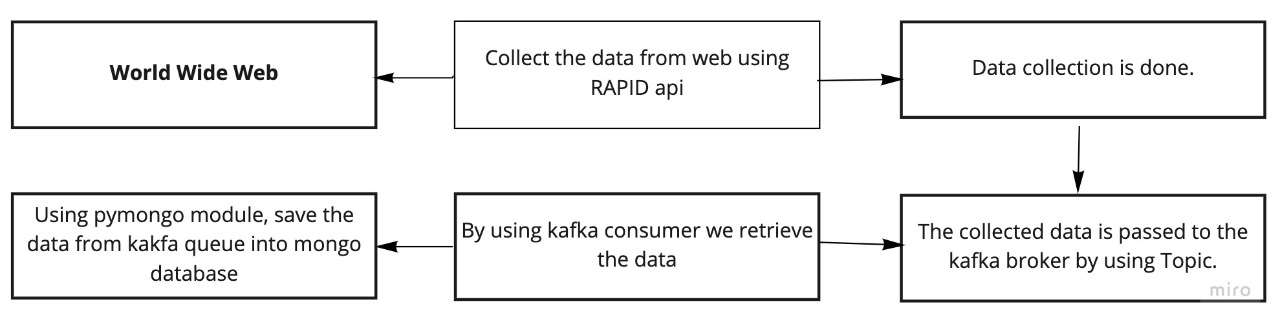
Input for this week’s data pipe line is data from different news sites.

Data collected from Rapid api, sending to kafka

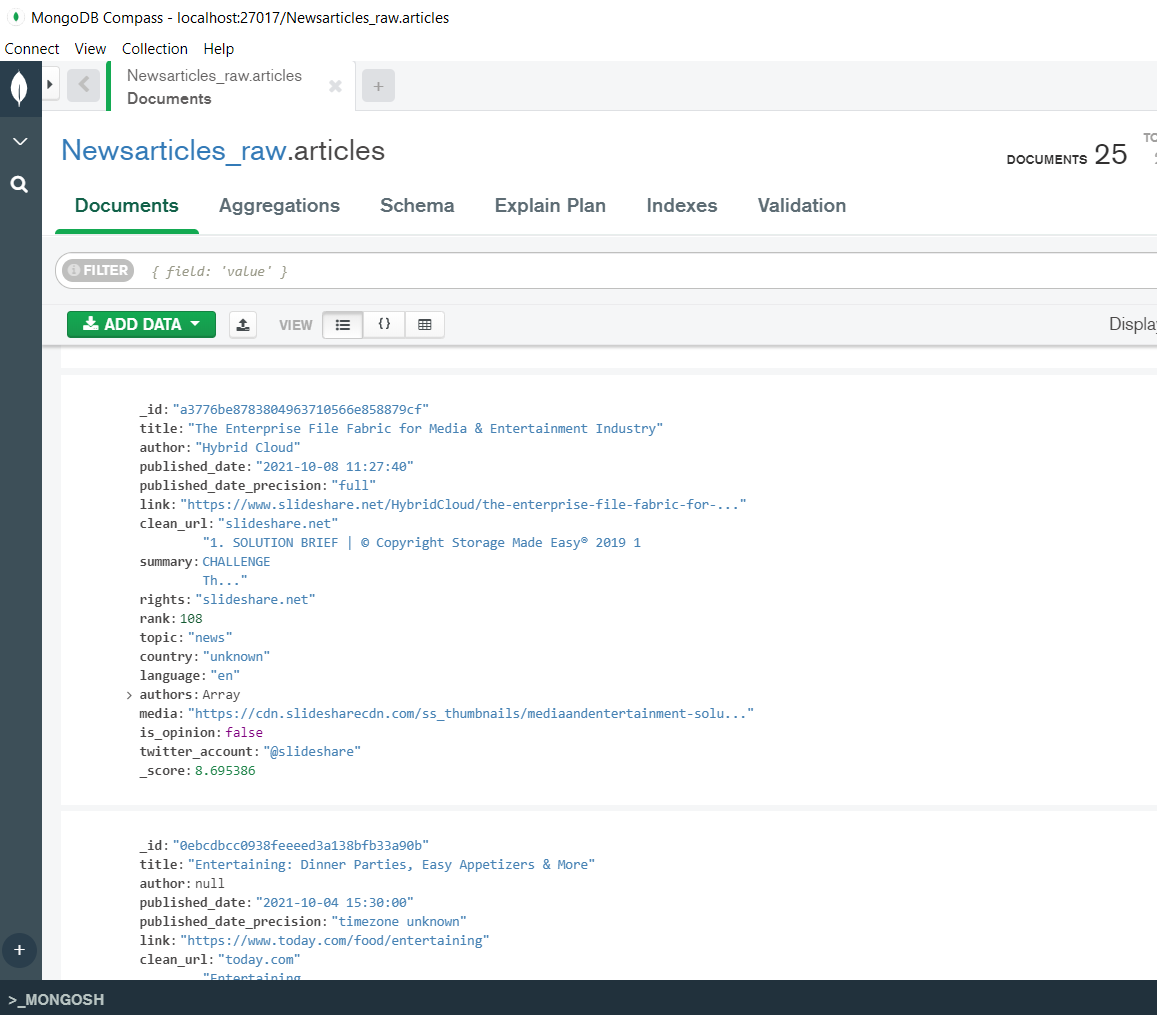


Kafka queue



**3.Processing of Input:**

**Output:** Output for this week is storing raw data into MongoDB from kafka.



**5.Challenges:**

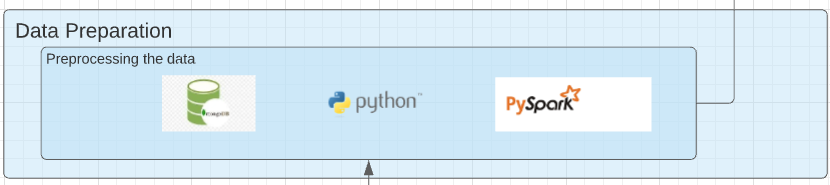
* Connecting from Windows to Vagrant & Vice-Versa.
* Started with Beautiful soup for web scrapping, but it was challenging to scrape some news website that I selected as it wasn’t giving enough information. We ended up using Rapid api for data collection.

### WEEK2: Data Preparation Report

Objective: This week’s objective is to take raw data from two sources, and preprocess the data using pyspark.

Approach: We have used raw data that we have ingested in week1 using rapidAPI and a static news classifier data set from Kaggle and preprocessed the data using pyspark.

High level architecture diagram for this week:



**1. Environment Details:**

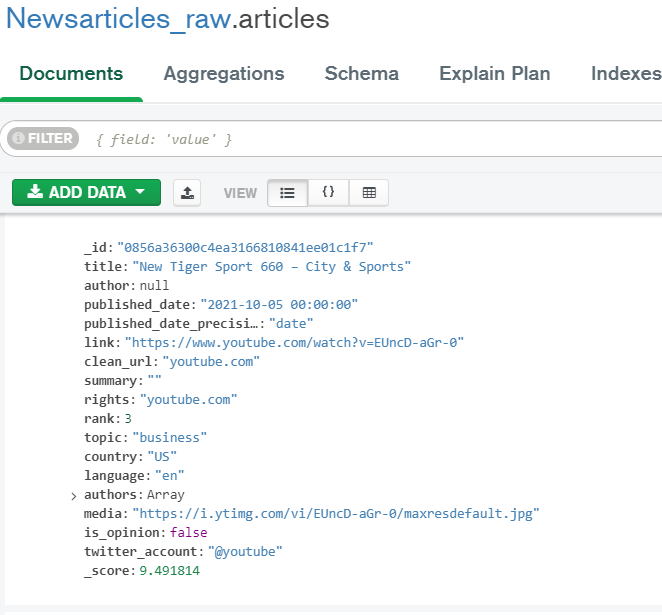
The Data preparation process for this project was performed using the following environment details.

* Python IDE
* pyspark
* MongoDB

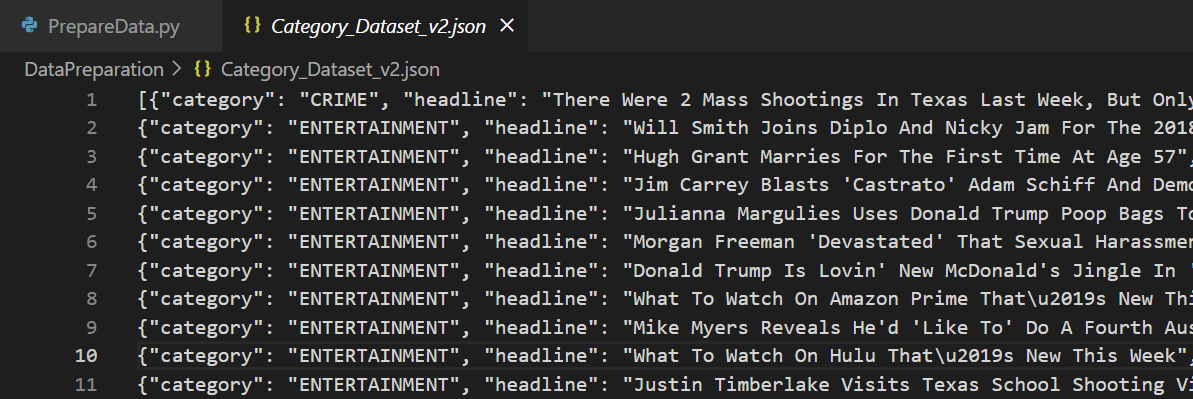
**2. Input:**

Input for this week’s data pipe line is the raw data from MongoDB and a static dataset.

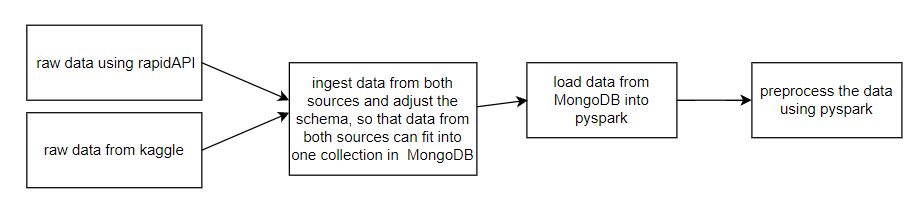
Raw data from rapidAPI



Raw data from kaggle:

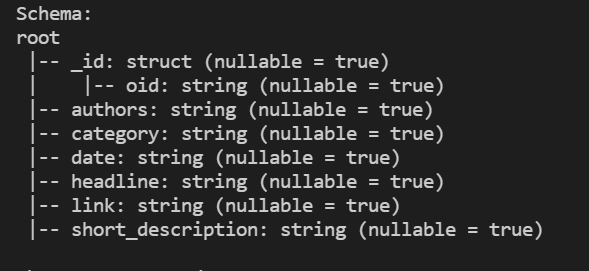


**3.Processing of Input:**



**Output:**

Output for this week is reading raw data and preprocess it using pyspark.





**5.Challenges:**

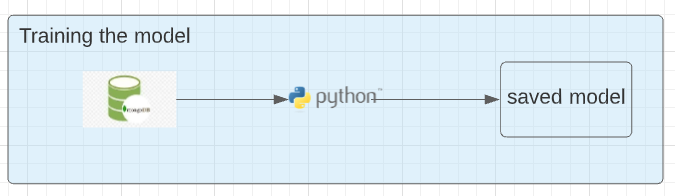
1. We had issues with connecting to spark after installing pyspark. After some research we have figured it out that it is looking for a jar file, which we have downloaded and make it available for the project.

### WEEK3: Model Training and Prediction Report

Objective: This week’s objective is to get the data from mongo database, preprocess the data, train and test the model. Save the model so that it can be used from FrontEnd.

Approach: Used spark session to load the database from mongo database. Used pandas and sklearn packages, model got fitted and trained using MultinomialNB.

High level architecture diagram for this week:

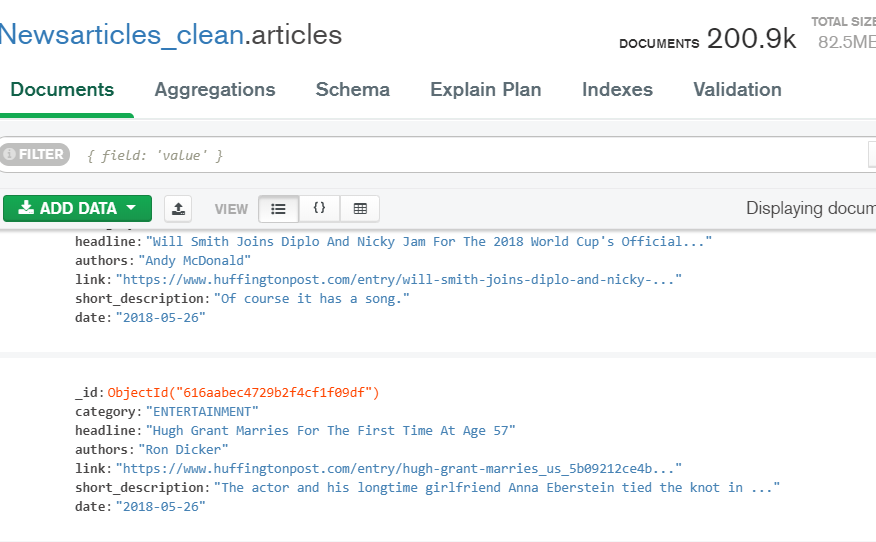


**1. Environment Details:**

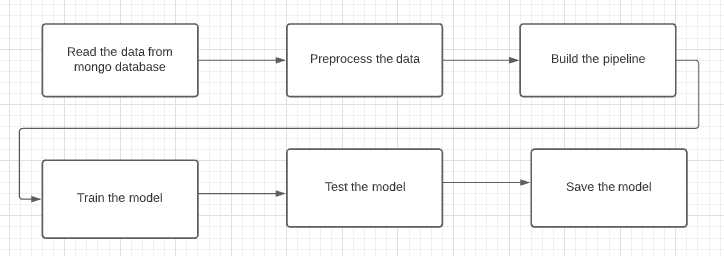
The Model preparation and training process for this project was performed using the following environment details.

* Python IDE
* pyspark
* MongoDB
* Main python packages used are pandas and sklearn

**2. Input:**

Input for this week’s data pipe line is the clean data from MongoDB.

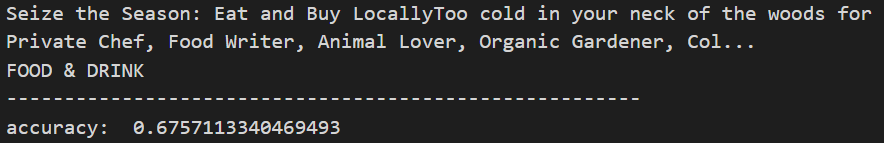
**3.Processing of Input:**

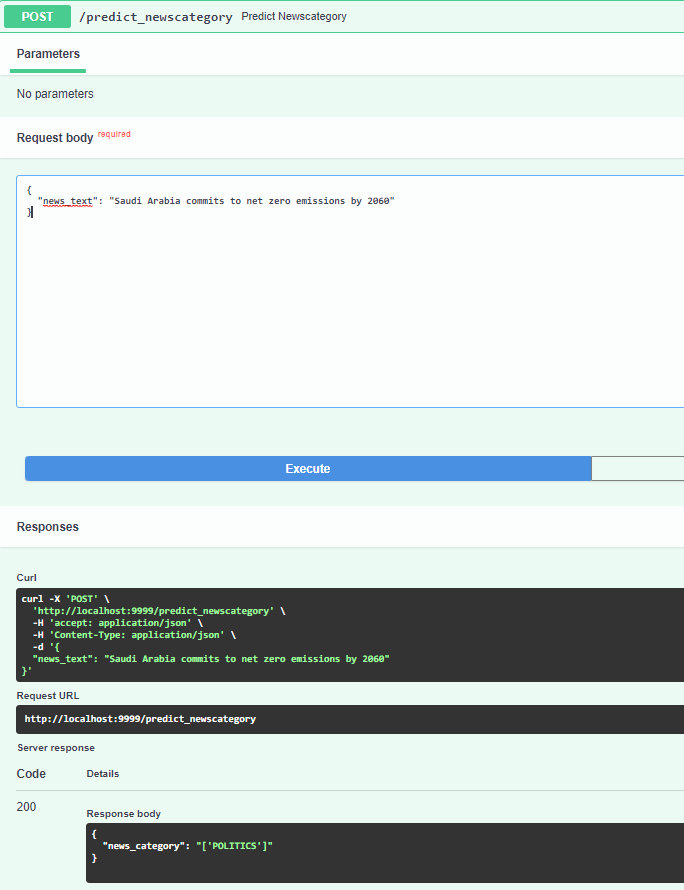


**Output:**

Output for this week is trained and tested model. Accuracy that we got is 0.6757. Tested the model using fast API. If time permits, we will work on improving the accuracy.

Training:



Testing:

**5.Challenges:**

1. we started with spark nlp pipeline, but the model has issues after saving it to local disk, since it doesn’t have HDFS setup. We have used sklearn pipeline and saved the model in .pkl format. This is model is much portable and more accurate.

### Final Output:





## Reference Material

Reference material we used as part of this project

* <https://repo1.maven.org/maven2/> for Jars
* News article dataset from Kaggle.com
* <https://towardsdatascience.com/your-first-apache-spark-ml-model-d2bb82b599dd>
* https://github.com/cdarlint/winutils