Capstone Project-5

Automated Question Answering System

By-Prabir Debnath

Points of Discussion

1. The Problem Statement
2. Summary of the Experience Set
3. Exploration and Pre-processing of Data
4. Building of Topic Model
5. Checking the general answering ability of the 'question-answering' model
6. Building of Q&A Model-1
7. Building of Q&A Model-2
8. Building of Q&A Model-3
9. Building of Q&A Model-4
10. Building of Q&A Model-5
11. Model Deployment
12. Learn from “drona” within 2 seconds, inside Your Coding

The Problem Statement

In a system based on Competency-Based Learning (“CBL”), the group is accepted as a set of diverse individuals who may come with some common traits but also recognizes their different educational backgrounds, life & work experiences, and learning styles.

We believe that the model of allowing all students to learn at the same pace with the same level of interest is quite detrimental to the quality of education.

In this light, to be able to cater to a large pool of students to enable seamless learning, we envision an AI-based automated question answering model which can understand the context of the question and can provide relevant answers to the questions.

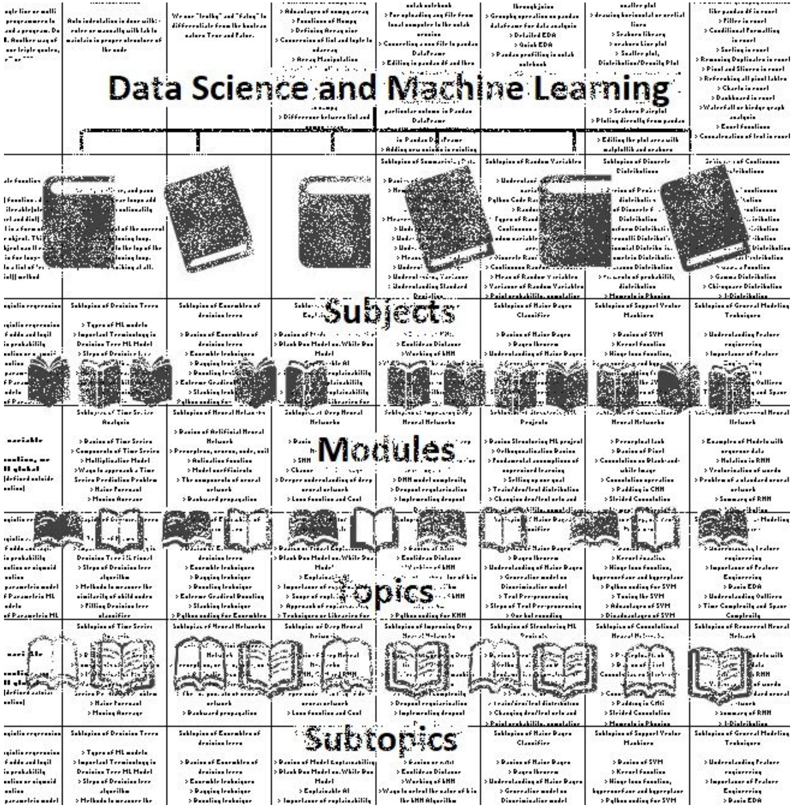
In this project, at first, our task is to create a dataset of data science documents and then extract the topics of the documents and finally build an automated question answering model which will retrieve the relevant document and generate the answer for the question.

Summary of the Experience Set

Here, the dataset is my entire set of experiences regarding different concepts of data science, machine learning, deep learning, etc. from day 1 at Almabetter.

It has 1052 rows and 1 ‘documents’ column, which means we have one experience with 1052 different features, dimensions, or concepts.

Let’s Decode the Experiences !



Exploration and Pre-processing of Data

We have done the exploration and pre-processing in five steps to transform raw data into quality data for our nlp model.

1. Connection with the Data
2. First Feelings of the Data
3. Deeper Understanding of the Data
4. Cleaning the Data and
5. Preparation of Input Data

Cleaning the Data

> We have created a single ‘documents’ column with all individual cells of excel file

> We have dropped 855 null values in the ‘documents’ column

> There were only new line characters in few of the documents. We have removed those rows.

> There were documents that mentioned ‘No data’ in the ‘documents’ column. We have removed those experiences.

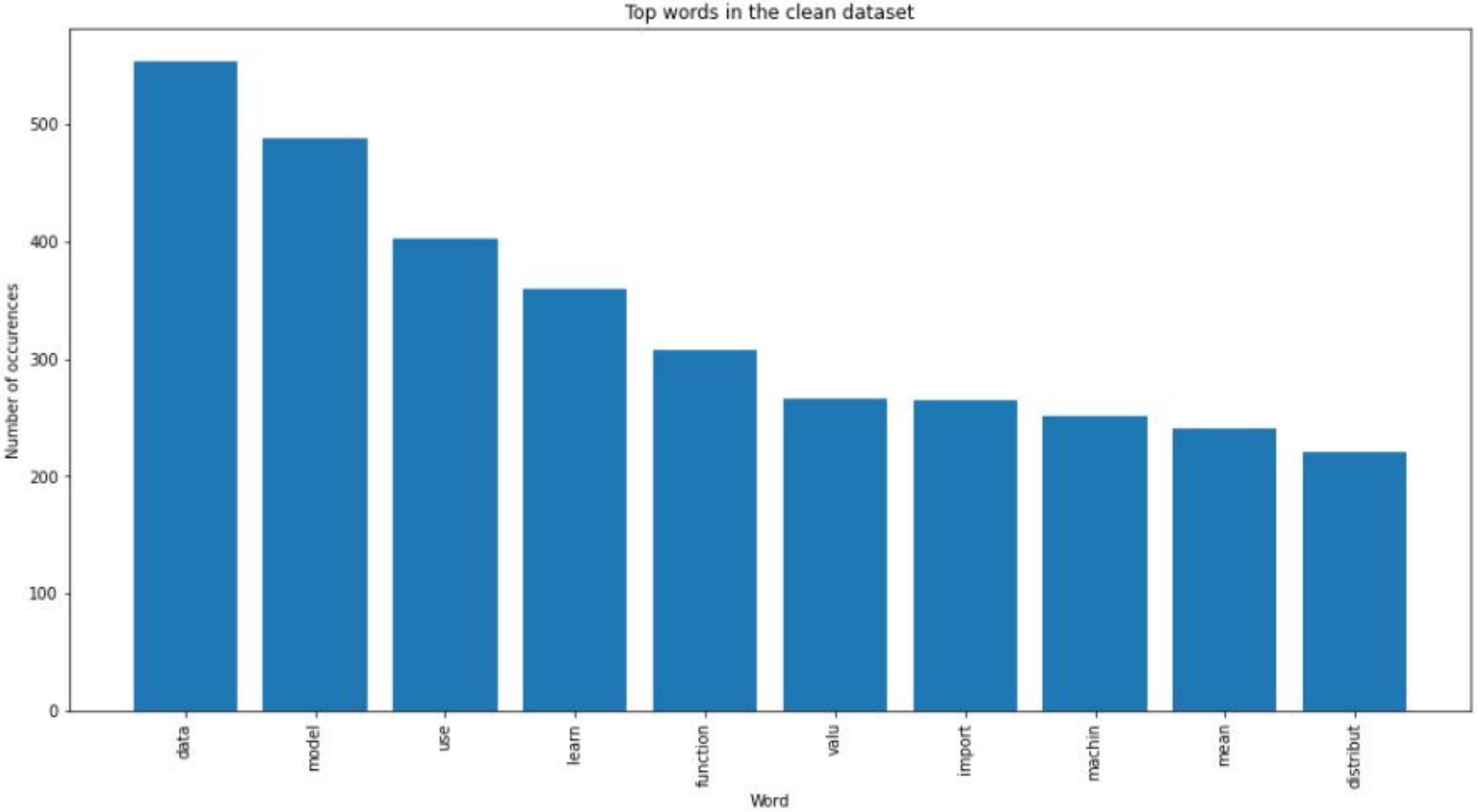
Preparation of Input Data

> We have done text pre-processing through two different functions: ‘abbreviation_process’ and ‘text_process’

> Inside the ‘abbreviation_process’ function we have converted all the data science and ml related abbreviations to their long-form before performing vectorization for uniform understanding by our NLP model.

> Inside the ‘text_process’ function, we have removed punctuations, stopwords and carried out stemming operation.

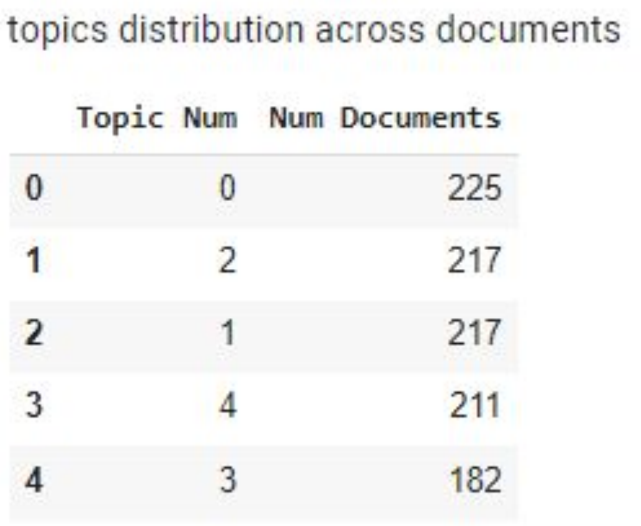
Building of Topic Model



Building of Topic Model (Continued)

After GridSearchCV,

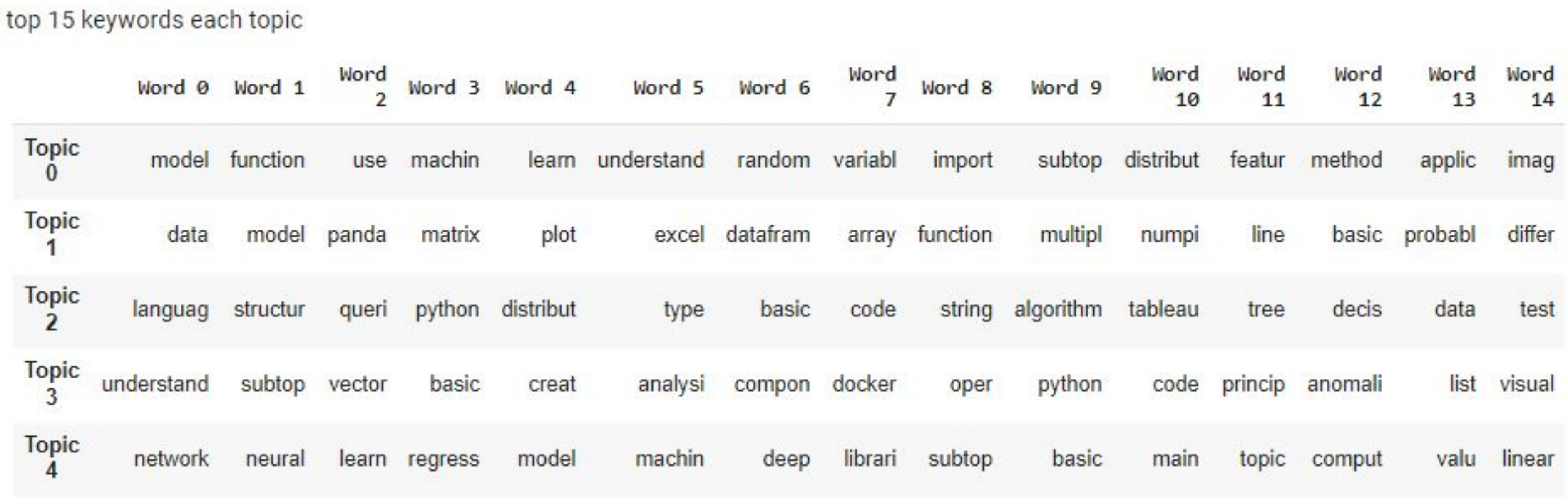
Best Model's Params: {'learning_decay': 0.9, 'n_components': 5}
Model Perplexity: 918.55



topics distribution across documents

	Topic Num	Num Documents
0	0	225
1	2	217
2	1	217
3	4	211
4	3	182

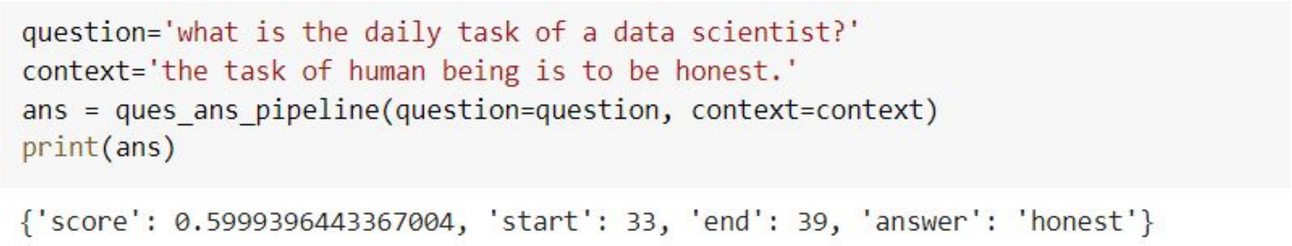
Building of Topic Model (Continued)



top 15 keywords each topic

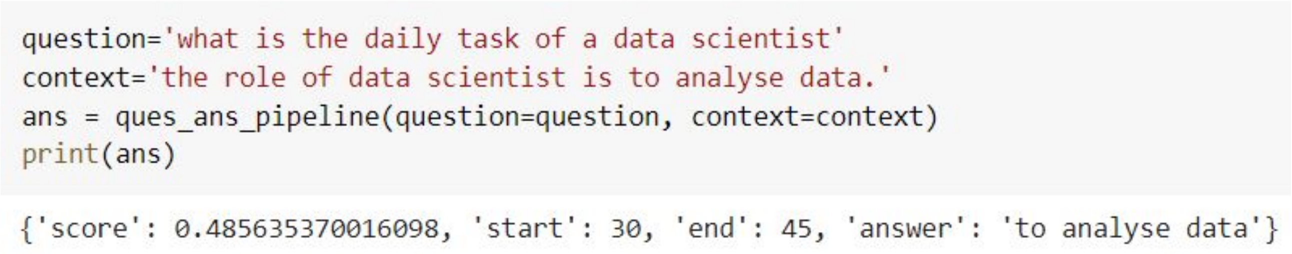
	Word 0	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7	Word 8	Word 9	Word 10	Word 11	Word 12	Word 13	Word 14
Topic 0	model	function	use	machin	learn	understand	random	variabl	import	subtop	distribut	featur	method	applic	imag
Topic 1	data	model	panda	matrix	plot	excel	datafram	array	function	multipl	numpi	line	basic	probabl	differ
Topic 2	languag	structur	queri	python	distribut	type	basic	code	string	algorithm	tableau	tree	decis	data	test
Topic 3	understand	subtop	vector	basic	creat	analysi	compon	docker	oper	python	code	princip	anomali	list	visual
Topic 4	network	neural	learn	regress	model	machin	deep	librari	subtop	basic	main	topic	comput	valu	linear

Checking the general answering ability of the 'question-answering' model



```
question='what is the daily task of a data scientist?'
context='the task of human being is to be honest.'
ans = ques_ans_pipeline(question=question, context=context)
print(ans)
```

```
{'score': 0.5999396443367004, 'start': 33, 'end': 39, 'answer': 'honest'}
```



```
question='what is the daily task of a data scientist'
context='the role of data scientist is to analyse data.'
ans = ques_ans_pipeline(question=question, context=context)
print(ans)
```

```
{'score': 0.485635370016098, 'start': 30, 'end': 45, 'answer': 'to analyse data'}
```

As the Hugging Face "question-answering" pipeline is giving very short answers and is very much dependent on the context, we are only using the "question-answering" ‘score’ in our model. (In actual deployment, we have not used the pipeline as it is not contributing to serving our purpose)

Building of Q&A Model-1

> For vectorization of natural language, CountVec unigram is used in this Question-Answering Model

> Hyperparameters used: max_features=4000

> Here, CountVec unigram model is giving correct answer in most of the cases.

Building of Q&A Model-2

> For vectorization of natural language, TFIDF is used in this Question- Answering Model

> Hyperparameters used: max_df = 0.9 and min_df = 10

> Here, TFIDF model is giving incorrect answer in most of the cases.

> Thus, countvec unigram question-answering model (Model-1) is better than Model-2

Building of Q&A Model-3

> For vectorization of natural language, LDA is used in this Question- Answering Model

> Hyperparameters used: 'learning_decay': 0.9, 'n_components': 5

> Here, LDA model is giving incorrect answer in most of the cases.

> Thus, countvec unigram question-answering model (Model-1) is better than Model-2 and Model-3

Building of Q&A Model-4

> For vectorization of natural language, Word2Vec is used in this Question-Answering Model

> Hyperparameters used: workers=4, min_count=1, window=5

> Here, Word2Vec model is giving correct answer in most of the cases.

> Thus, Word2Vec model is equivalent to countvec unigram question-answering model

Building of Q&A Model-5

> For vectorization of natural language, Countvec ngram is used in this Question -Answering Model

> Hyperparameters used: ngram_range=(1,3)

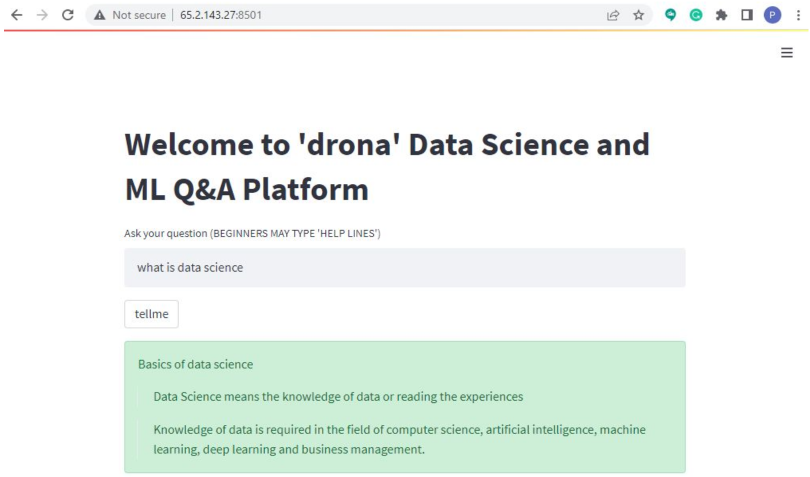
> Here, Countvec ngram model is giving correct answer in all of the cases.

> Thus, Countvec ngram question-answering model is the best among all five models, and the same is deployed for actual use.

Model Deployment

We can view the streamlit demo via the following link https://share.streamlit.io/prabirdeb/automated-question-answering-system/main/app.py

http://65.2.143.27:8501/ (Alternate)



Welcome to 'drona' Data Science and ML Q&A Platform

Ask your question (BEGINNERS MAY TYPE 'HELP LINES')

what is data science

tellme

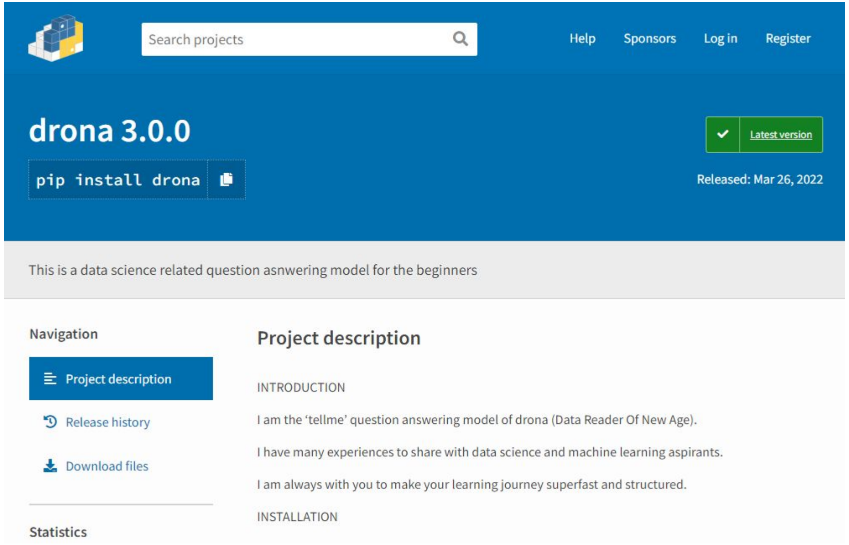
Basics of data science

Data Science means the knowledge of data or reading the experiences

Knowledge of data is required in the field of computer science, artificial intelligence, machine learning, deep learning and business management.

The project is published as python library 'drona' for the benefit of data science and machine learning aspirants.

https://pypi.org/project/drona/



drona 3.0.0

```
pip install drona
```

Released: Mar 26, 2022

This is a data science related question asnwering model for the beginners

Project description

INTRODUCTION

I am the 'tellme' question answering model of drona (Data Reader Of New Age).

I have many experiences to share with data science and machine learning aspirants.

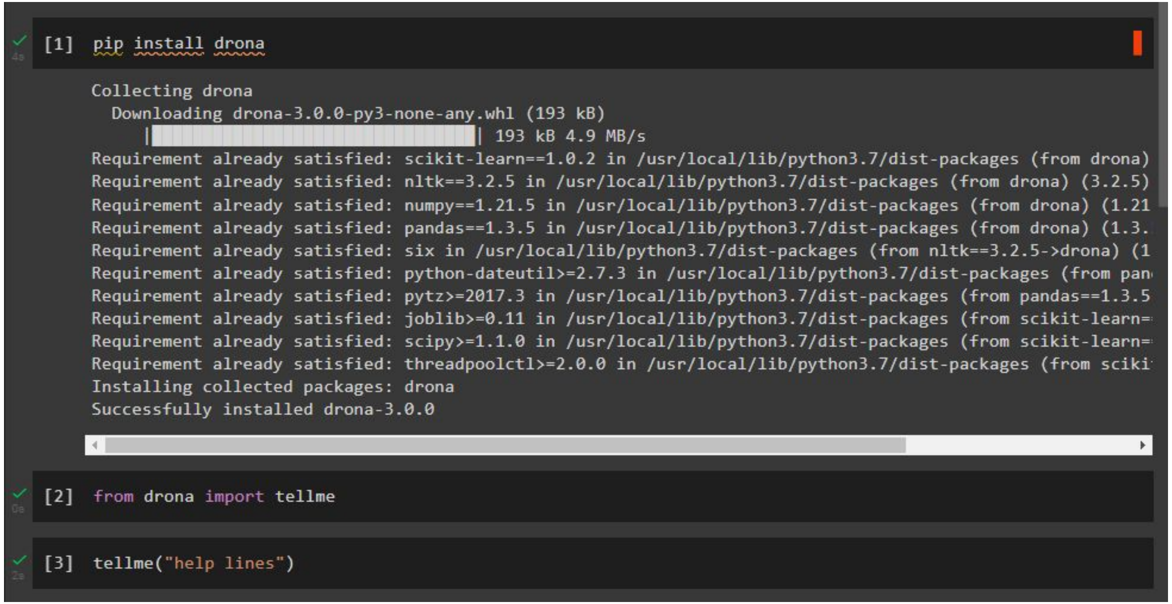
I am always with you to make your learning journey superfast and structured.

INSTALLATION

Learn from “drona” within 2 seconds, inside Your Coding

The ‘tellme’ model of ‘drona’ can be called within the python environment during coding to make the learning journey of data science and ml aspirants super fast and structured.

“Data Reader Of New Age”



```
[1] pip install drona

    Collecting drona
      Downloading drona-3.0.0-py3-none-any.whl (193 kB)
         |████████████████████████████████| 193 kB 4.9 MB/s
    Requirement already satisfied: scikit-learn==1.0.2 in /usr/local/lib/python3.7/dist-packages (from drona)
    Requirement already satisfied: nltk==3.2.5 in /usr/local/lib/python3.7/dist-packages (from drona) (3.2.5)
    Requirement already satisfied: numpy==1.21.5 in /usr/local/lib/python3.7/dist-packages (from drona) (1.21
    Requirement already satisfied: pandas==1.3.5 in /usr/local/lib/python3.7/dist-packages (from drona) (1.3.
    Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from nltk==3.2.5->drona) (1
    Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.7/dist-packages (from pan
    Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packages (from pandas==1.3.5
    Requirement already satisfied: joblib>=0.11 in /usr/local/lib/python3.7/dist-packages (from scikit-learn=
    Requirement already satisfied: scipy>=1.1.0 in /usr/local/lib/python3.7/dist-packages (from scikit-learn=
    Requirement already satisfied: threadpoolctl>=2.0.0 in /usr/local/lib/python3.7/dist-packages (from sciki
    Installing collected packages: drona
    Successfully installed drona-3.0.0

[2] from drona import tellme

[3] tellme("help lines")
```

Thank you !