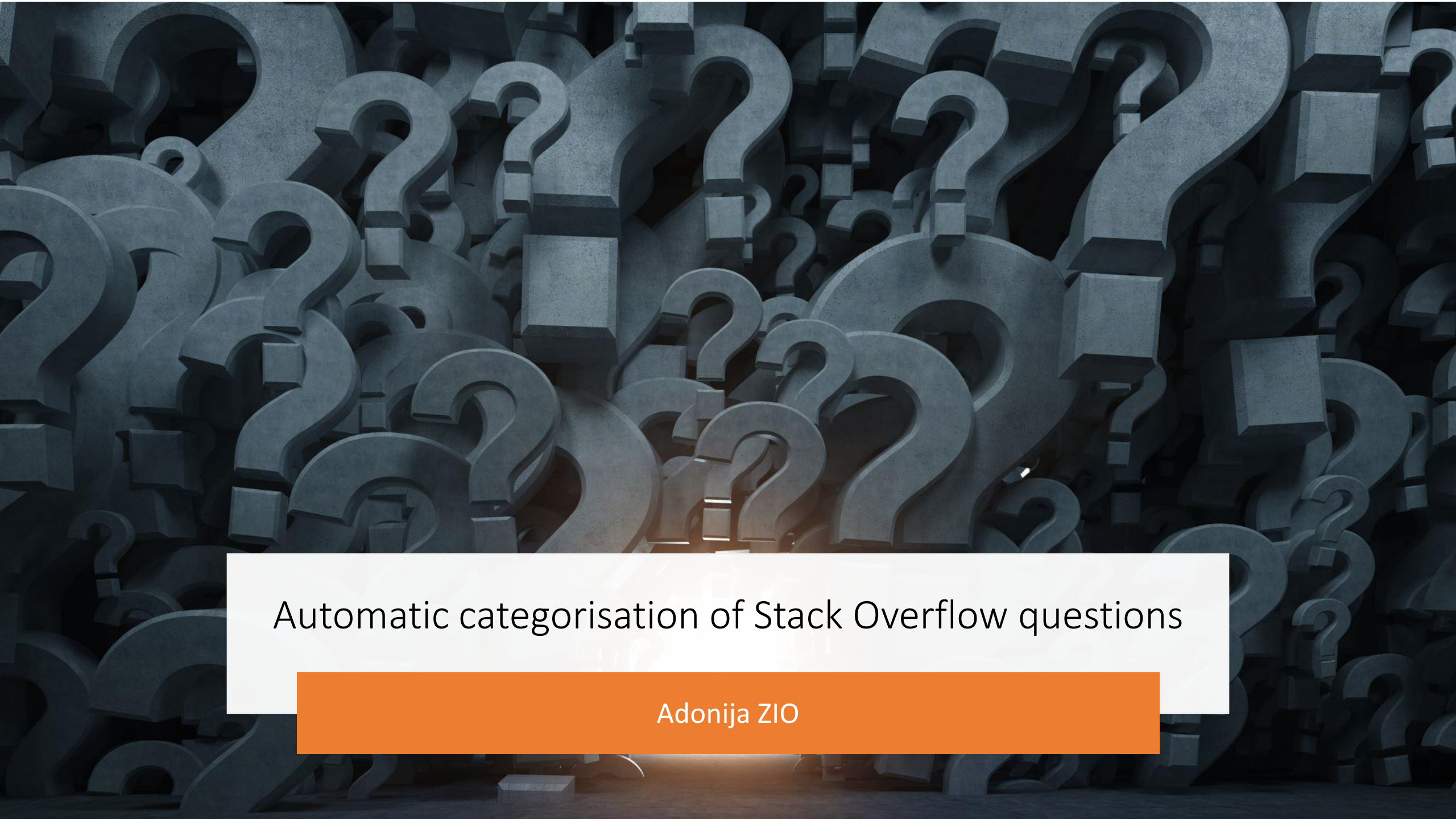


Automatic categorisation of Stack Overflow questions

Adonija ZIO

Outline

Executive Summary

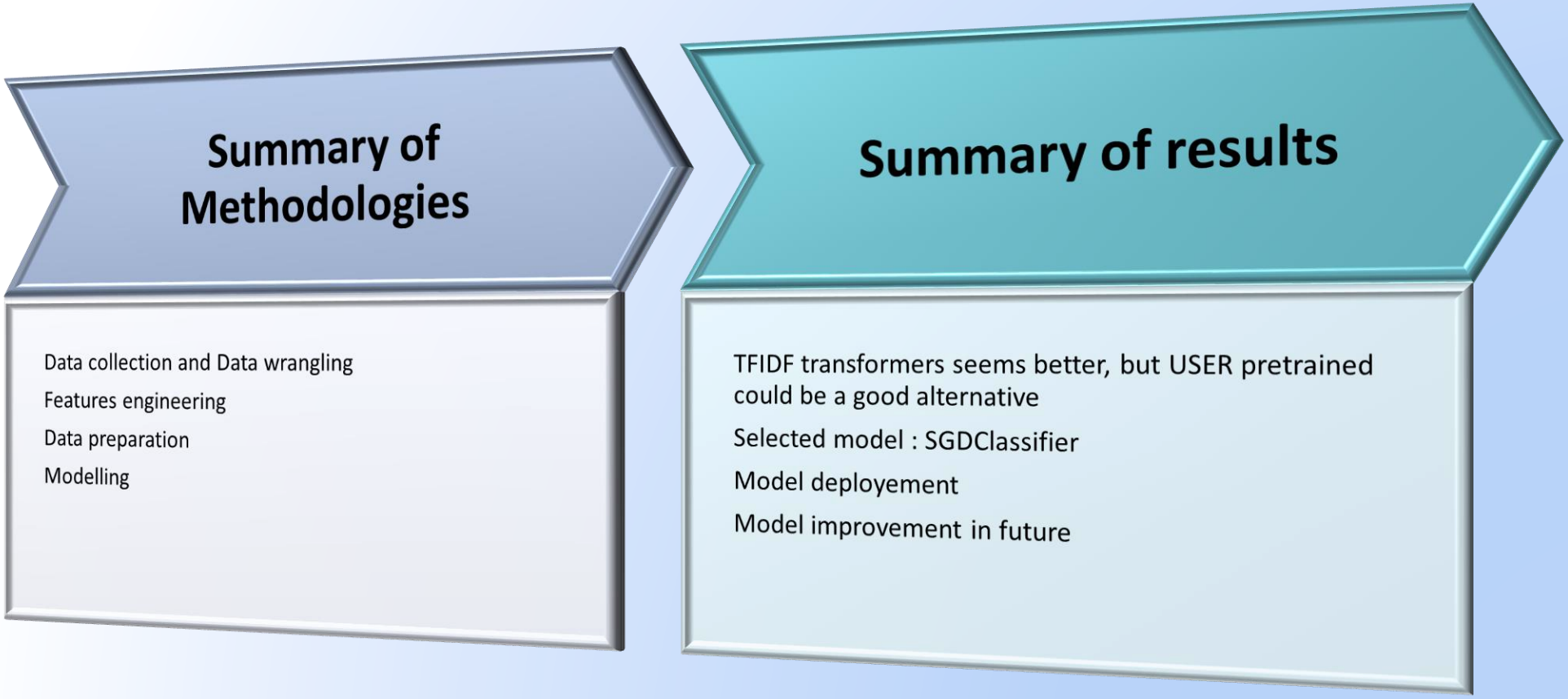
Introduction

Methodology

Results

Conclusion

Executive Summary



Introduction

Stak Overflow: part of Stack Exchange network

Website proposing questions and answers on a wide range of topics related to computer programming

Created in 2008 with 21+ million Questions asked to-date and 13.6 seconds Average time between new questions

Serving 100 million people every month and 50.6+ billion Times a developer got help, 10,000+ Customer companies for all products

The Objective of Stack Overflow, is to increase its audience and become (or keep himself among) the most popular site in the developer community

Keep his visitors

Attract a new visitor

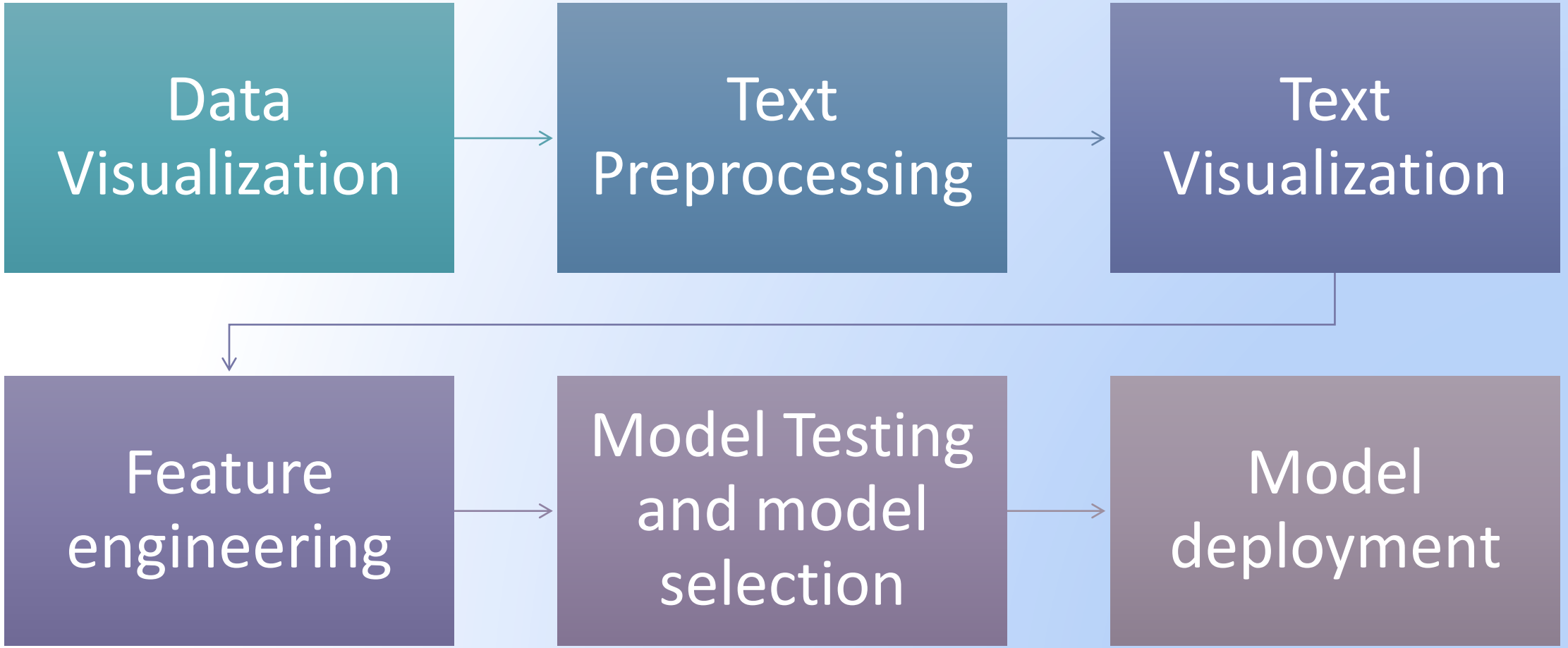
How to achieve this?

Tagging correctly the community question to help visitor to find quickly the answer of their problem

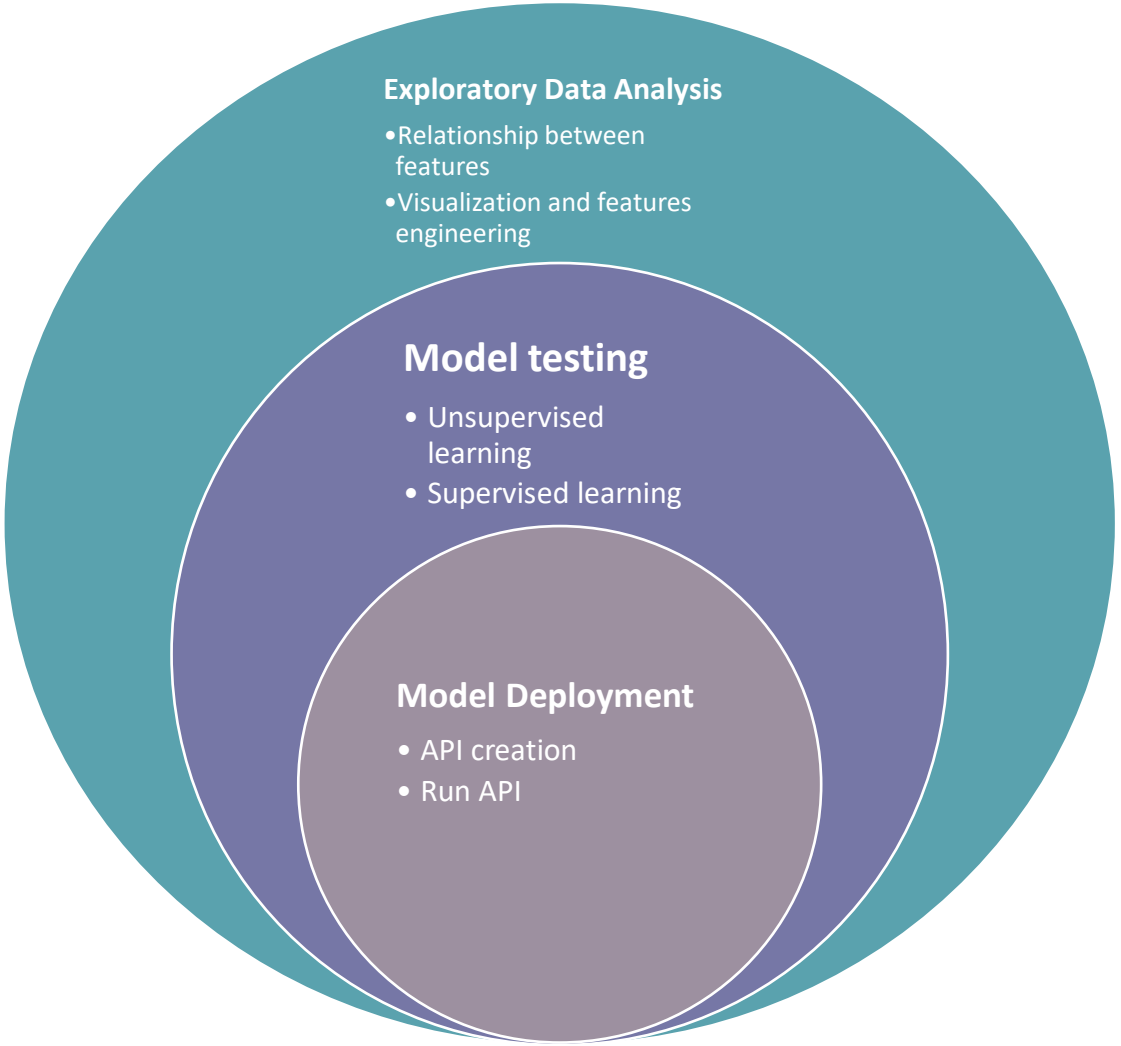
Business problem:

Classification in NLP

Methodology



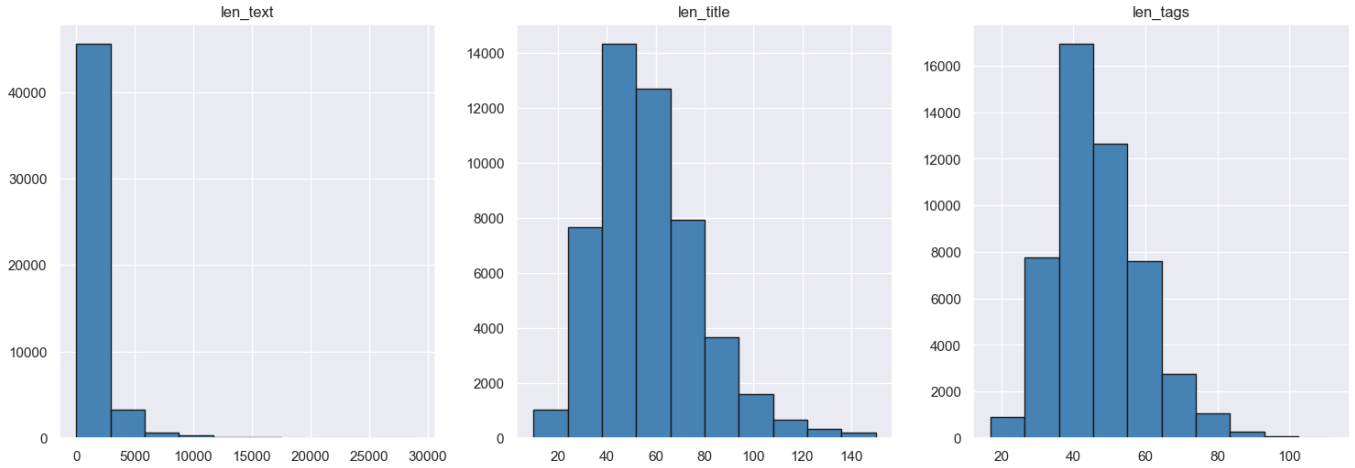
Results

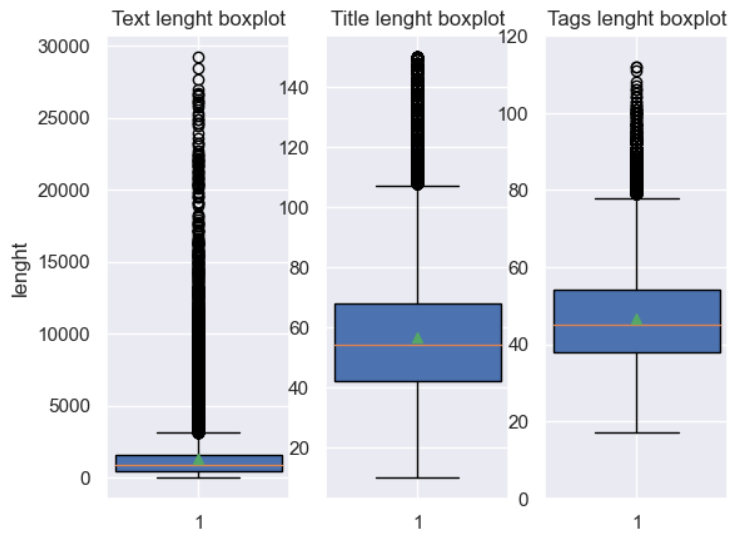

Exploratory Data Analysis
•Relationship between features
•Visualization and features engineering
Model testing
• Unsupervised learning
• Supervised learning
Model Deployment
• API creation
• Run API

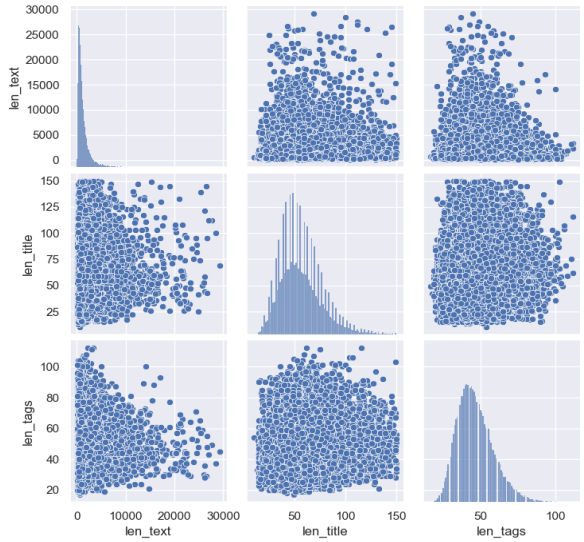
Exploratory Data Analysis

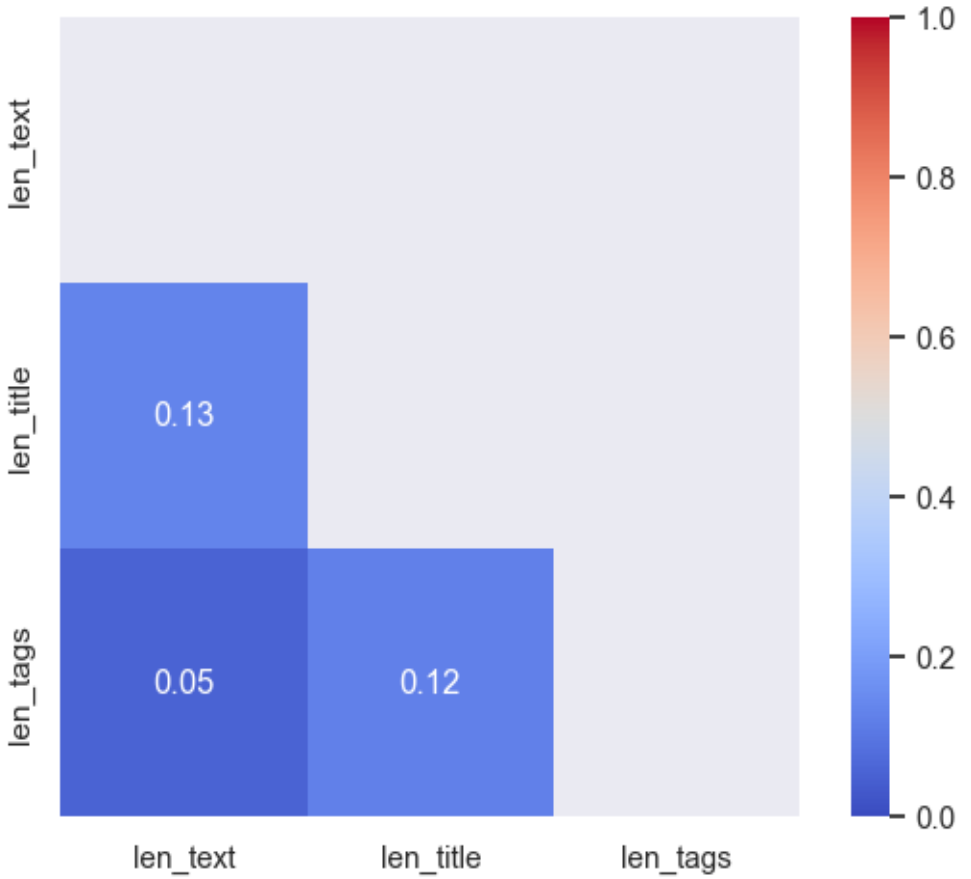
Features analysis	Univariate analysis Multivariate analysis
Text visualization and features engineering	Bag of word Word cloud Feature creation

Features analysis

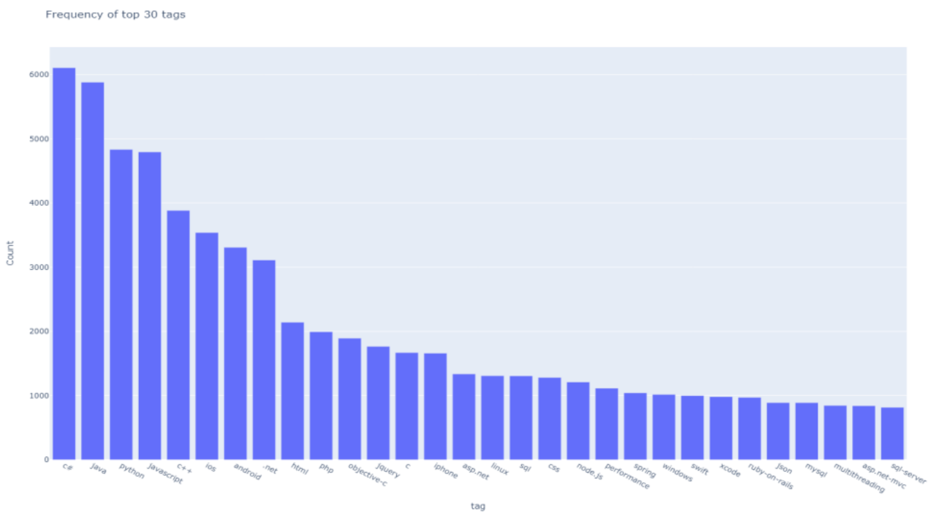


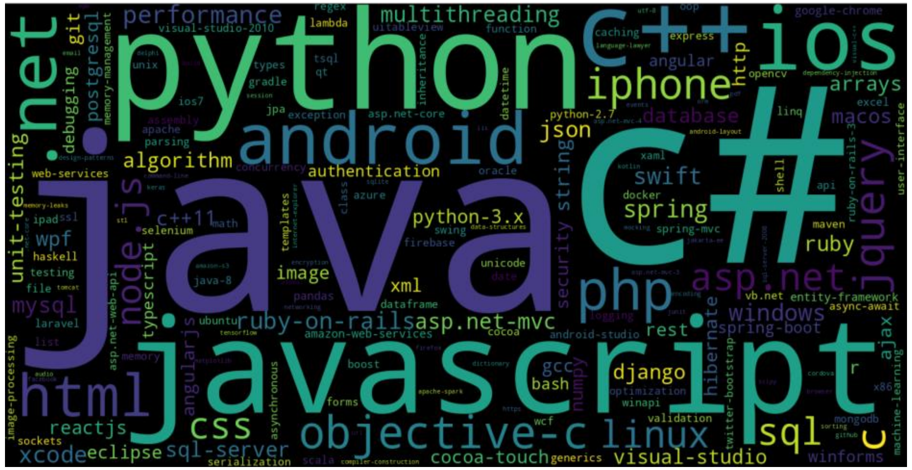


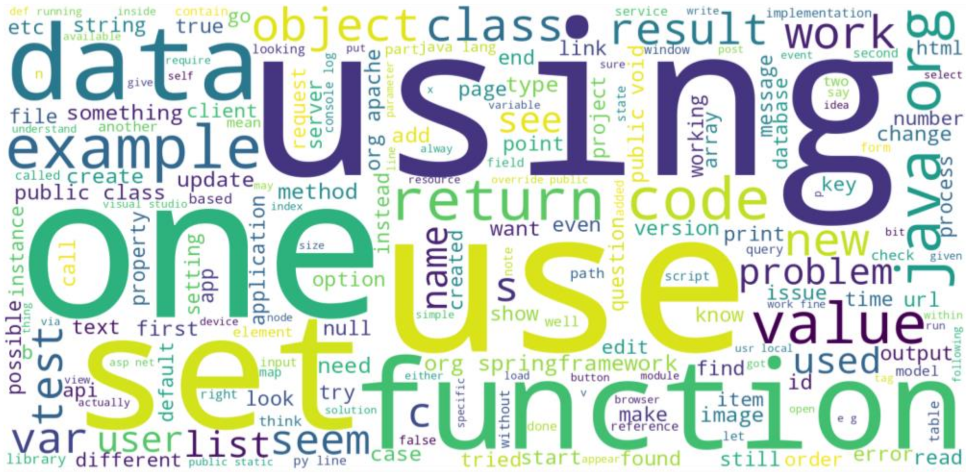




Text visualization and features engineering





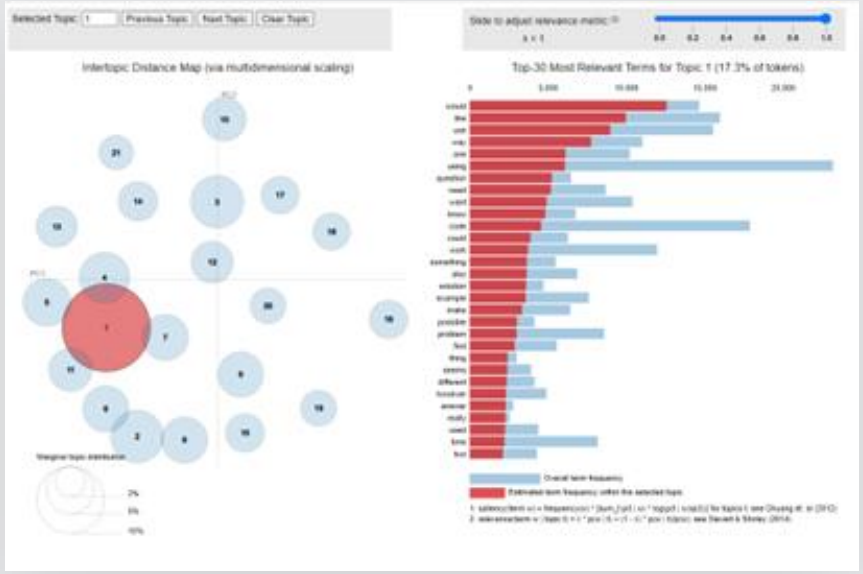


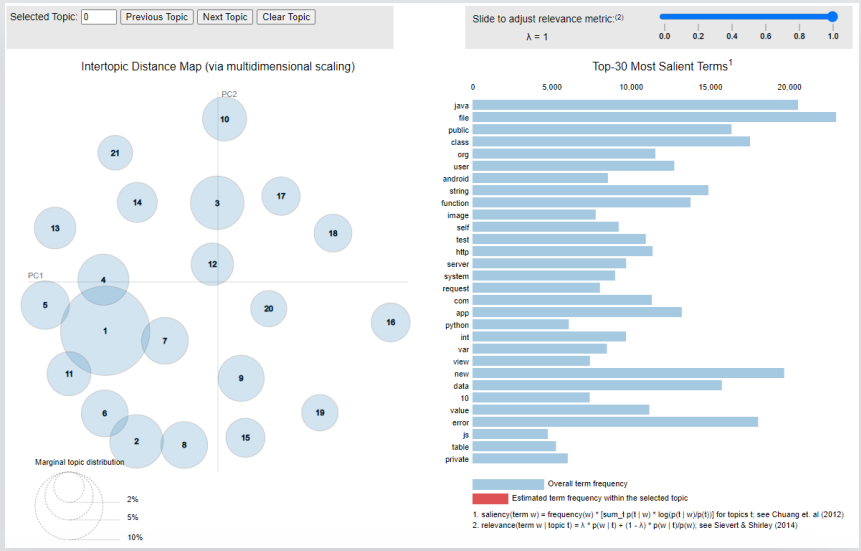
Modelling

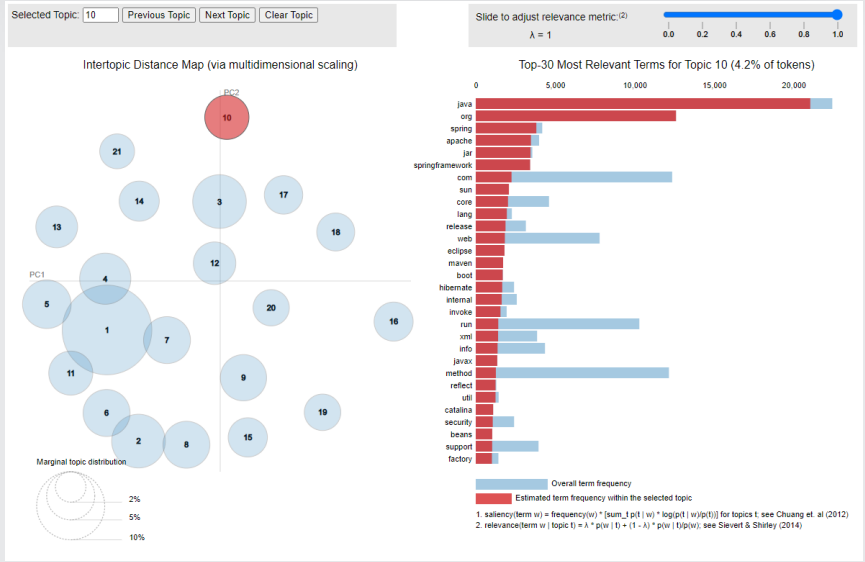
Unsupervised modelling test

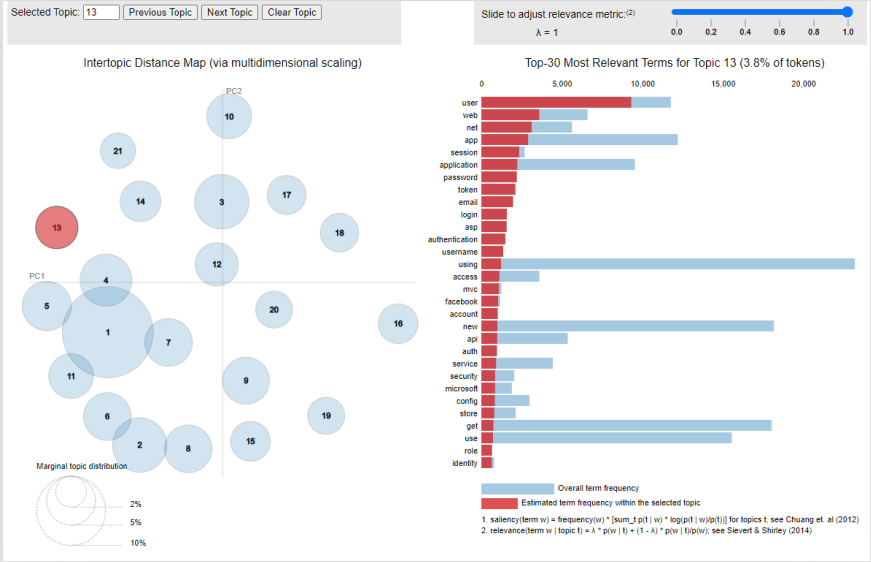
Supervised modelling test

LDA Visualization

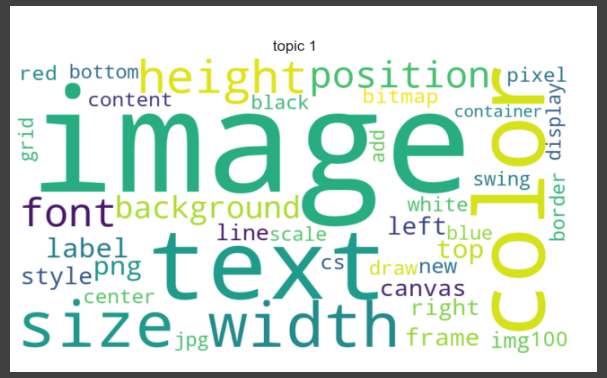


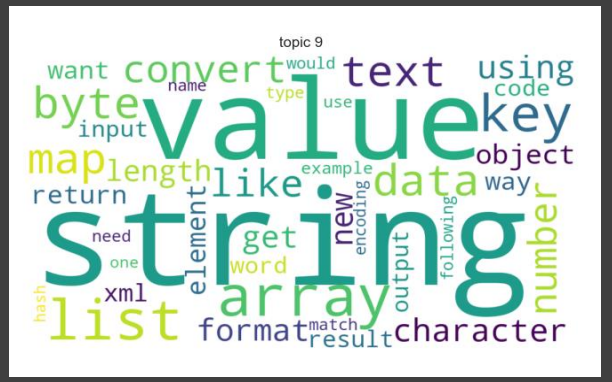


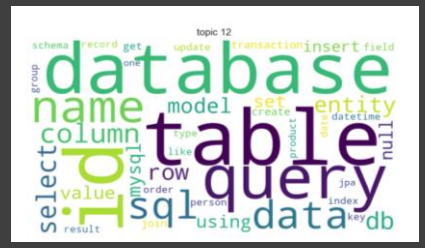


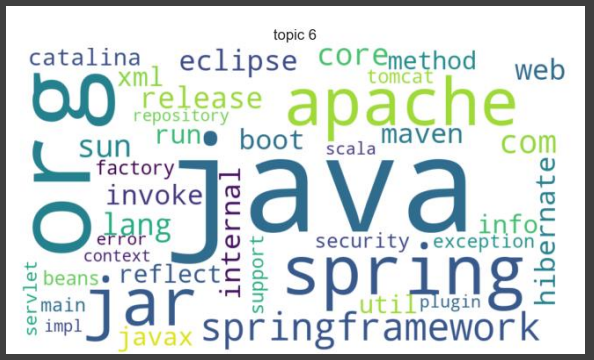


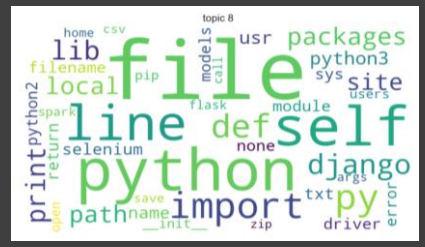
LDA Word cloud

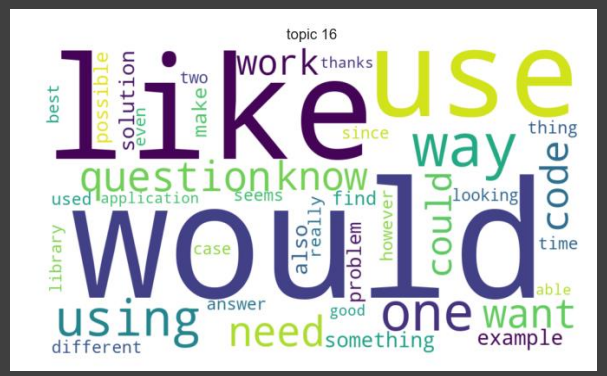


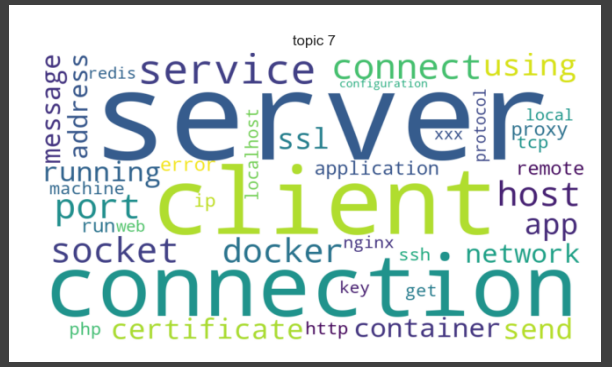


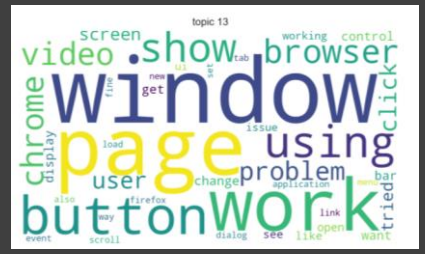


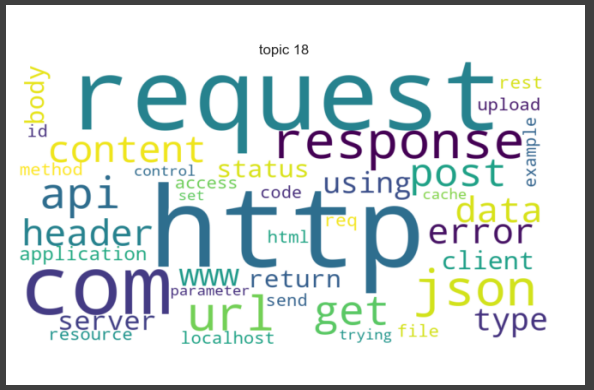












Topics	LDA : Cohérence = 0.642
13	0.024*"window" + 0.018*"page" + 0.015*"work" + 0.015*"button" + 0.013*"using" + 0.011*"show" + 0.010*"browser" + 0.009*"chrome" + 0.009*"video" + 0.009*"problem"
10	0.041*"int" + 0.030*"std" + 0.022*"function" + 0.020*"type" + 0.020*"include" + 0.016*"foo" + 0.016*"const" + 0.016*"return" + 0.014*"code" + 0.013*"template"
17	0.045*"js" + 0.032*"node" + 0.026*"module" + 0.022*"angular" + 0.021*"app" + 0.020*"react" + 0.019*"git" + 0.019*"component" + 0.018*"import" + 0.016*"node_modules"
9	0.052*"string" + 0.035*"value" + 0.028*"array" + 0.027*"list" + 0.023*"key" + 0.016*"byte" + 0.016*"data" + 0.015*"map" + 0.015*"text" + 0.012*"convert"
16	0.025*"would" + 0.020*"like" + 0.018*"use" + 0.015*"way" + 0.012*"one" + 0.012*"using" + 0.010*"question" + 0.010*"need" + 0.010*"want" + 0.010*"know"
14	0.049*"10" + 0.034*"00" + 0.030*"11" + 0.030*"12" + 0.020*"date" + 0.020*"15" + 0.017*"16" + 0.017*"13" + 0.016*"20" + 0.015*"01"
6	0.144*"java" + 0.086*"org" + 0.026*"spring" + 0.024*"apache" + 0.024*"jar" + 0.023*"springframework" + 0.015*"com" + 0.014*"sun" + 0.014*"core" + 0.013*"lang"
11	0.103*"android" + 0.040*"com" + 0.032*"app" + 0.028*"google" + 0.023*"view" + 0.018*"activity" + 0.015*"java" + 0.014*"firebase" + 0.014*"gradle" + 0.013*"fragment"
1	0.085*"image" + 0.030*"text" + 0.028*"color" + 0.028*"size" + 0.027*"width" + 0.025*"height" + 0.018*"font" + 0.015*"position" + 0.015*"background" + 0.012*"label"
2	0.020*"data" + 0.015*"point" + 0.014*"model" + 0.014*"np" + 0.012*"float" + 0.010*"value" + 0.010*"range" + 0.010*"matrix" + 0.009*"df" + 0.009*"import"
12	0.035*"id" + 0.035*"table" + 0.029*"database" + 0.027*"query" + 0.023*"name" + 0.022*"data" + 0.022*"sql" + 0.021*"column" + 0.019*"select" + 0.016*"db"
18	0.062*"http" + 0.049*"request" + 0.032*"com" + 0.031*"response" + 0.030*"json" + 0.027*"url" + 0.022*"api" + 0.020*"get" + 0.018*"post" + 0.016*"content"
5	0.031*"thread" + 0.026*"time" + 0.019*"memory" + 0.017*"process" + 0.013*"run" + 0.012*"data" + 0.011*"read" + 0.010*"second" + 0.009*"start" + 0.007*"performance"
20	0.049*"self" + 0.044*"view" + 0.027*"io" + 0.021*"app" + 0.018*"let" + 0.017*"nil" + 0.014*"cell" + 0.014*"error" + 0.012*"xcode" + 0.011*"iphone"
7	0.065*"server" + 0.033*"client" + 0.028*"connection" + 0.020*"service" + 0.014*"port" + 0.014*"connect" + 0.014*"socket" + 0.014*"host" + 0.013*"docker" + 0.010*"using"
19	0.084*"user" + 0.033*"web" + 0.028*"net" + 0.026*"app" + 0.021*"session" + 0.020*"application" + 0.020*"password" + 0.019*"token" + 0.018*"email" + 0.014*"login"
0	0.068*"function" + 0.046*"var" + 0.025*"event" + 0.022*"form" + 0.021*"html" + 0.018*"php" + 0.018*"document" + 0.017*"data" + 0.016*"jquery" + 0.016*"javascript"
4	0.083*"class" + 0.082*"public" + 0.046*"new" + 0.031*"string" + 0.030*"return" + 0.029*"private" + 0.026*"void" + 0.025*"object" + 0.024*"method" + 0.019*"get"
8	0.129*"file" + 0.057*"python" + 0.049*"self" + 0.039*"line" + 0.033*"import" + 0.032*"py" + 0.026*"def" + 0.019*"django" + 0.017*"lib" + 0.016*"print"
15	0.070*"test" + 0.057*"system" + 0.034*"message" + 0.032*"error" + 0.029*"exception" + 0.022*"code" + 0.019*"new" + 0.019*"log" + 0.016*"catch" + 0.016*"console"

Topics	NMF : Cohérence = 0.46
7	0.026*"python" + 0.015*"lib" + 0.015*"file" + 0.014*"library" + 0.012*"usr" + 0.011*"project" + 0.010*"py" + 0.010*"matrix" + 0.008*"window" + 0.007*"linux"
0	0.038*"function" + 0.011*"library" + 0.009*"include" + 0.009*"text" + 0.008*"reference" + 0.008*"bit" + 0.008*"number" + 0.007*"document" + 0.007*"var" + 0.007*"char"
4	0.018*"width" + 0.016*"java" + 0.013*"height" + 0.013*"text" + 0.012*"color" + 0.011*"android" + 0.011*"size" + 0.011*"xml" + 0.010*"memory" + 0.010*"button"
2	0.022*"array" + 0.017*"server" + 0.015*"list" + 0.014*"element" + 0.014*"video" + 0.009*"event" + 0.008*"input" + 0.008*"time" + 0.008*"process" + 0.007*"studio"
11	0.042*"image" + 0.025*"thread" + 0.019*"int" + 0.008*"main" + 0.007*"icon" + 0.007*"method" + 0.007*"size" + 0.006*"cell" + 0.006*"png" + 0.006*"void"
1	0.029*"user" + 0.014*"com" + 0.012*"http" + 0.011*"class" + 0.011*"python" + 0.011*"api" + 0.011*"foo" + 0.010*"django" + 0.010*"database" + 0.010*"method"
17	0.018*"name" + 0.017*"php" + 0.012*"bar" + 0.011*"would" + 0.008*"like" + 0.008*"way" + 0.007*"xcode" + 0.007*"question" + 0.006*"know" + 0.006*"example"
16	0.021*"framework" + 0.018*"net" + 0.016*"test" + 0.016*"application" + 0.015*"page" + 0.012*"build" + 0.010*"project" + 0.009*"access" + 0.009*"asp" + 0.007*"address"
6	0.027*"table" + 0.025*"database" + 0.022*"sql" + 0.019*"image" + 0.013*"session" + 0.012*"column" + 0.012*"memory" + 0.012*"db" + 0.010*"mysql" + 0.009*"event"
18	0.072*"self" + 0.018*"def" + 0.012*"import" + 0.009*"print" + 0.009*"np" + 0.008*"__init__" + 0.007*"view" + 0.007*"let" + 0.007*"nil" + 0.007*"frame"
14	0.029*"java" + 0.027*"org" + 0.015*"thread" + 0.014*"log" + 0.014*"message" + 0.013*"exception" + 0.012*"error" + 0.011*"eclipse" + 0.010*"function" + 0.009*"springframework"
10	0.050*"std" + 0.027*"int" + 0.025*"class" + 0.019*"public" + 0.017*"void" + 0.014*"const" + 0.013*"include" + 0.013*"static" + 0.013*"template" + 0.012*"return"
15	0.030*"data" + 0.023*"table" + 0.018*"query" + 0.018*"row" + 0.016*"id" + 0.015*"column" + 0.014*"select" + 0.011*"sql" + 0.008*"search" + 0.008*"type"
20	0.020*"git" + 0.020*"js" + 0.018*"http" + 0.016*"json" + 0.016*"com" + 0.014*"jquery" + 0.014*"var" + 0.012*"content" + 0.012*"html" + 0.012*"ajax"
19	0.061*"file" + 0.031*"java" + 0.015*"command" + 0.013*"stream" + 0.013*"directory" + 0.012*"audio" + 0.011*"path" + 0.010*"line" + 0.009*"error" + 0.009*"folder"
13	0.022*"request" + 0.021*"response" + 0.017*"message" + 0.016*"server" + 0.016*"error" + 0.015*"window" + 0.014*"client" + 0.014*"url" + 0.012*"http" + 0.011*"post"
9	0.032*"android" + 0.030*"test" + 0.018*"app" + 0.015*"new" + 0.015*"string" + 0.014*"react" + 0.013*"device" + 0.013*"system" + 0.011*"public" + 0.010*"void"
5	0.027*"object" + 0.026*"public" + 0.024*"key" + 0.022*"view" + 0.021*"set" + 0.019*"value" + 0.018*"property" + 0.015*"field" + 0.014*"class" + 0.013*"entity"
3	0.068*"string" + 0.022*"value" + 0.021*"array" + 0.021*"name" + 0.020*"list" + 0.019*"public" + 0.018*"model" + 0.016*"id" + 0.012*"xml" + 0.012*"object"
12	0.028*"web" + 0.020*"service" + 0.018*"system" + 0.018*"net" + 0.016*"user" + 0.014*"import" + 0.011*"spring" + 0.010*"asp" + 0.010*"package" + 0.009*"application"

LDA VS NMF

Model testing by vectorizer

TF-IDF	Logistic Regression	Linear SVC	lgbm Classifier	SGDClassifier	Passive Aggressive Classifier	Perceptron	Solves linear One-Class SVM	MultinomialNB
Precision	0.800122	0.757402	0.753002	0.832613	0.584346	0.527174	0.000000	0.754664
Recall	0.375694	0.496811	0.539862	0.381019	0.532336	0.532049	0.000000	0.198641
fscore	0.511306	0.600035	0.628863	0.522797	0.557130	0.529600	0.000000	0.314501
Jaccard	0.369368	0.468164	0.498904	0.388424	0.438135	0.413234	0.000000	0.202399
Accuracy	0.187245	0.237073	0.252449	0.202816	0.177253	0.148381	0.000000	0.094076
Hamming Loss	0.014611	0.013475	0.012964	0.014152	0.017219	0.019229	0.020345	0.017618

Countverizer	Logistic Regression	Linear SVC	SGDClassifier	Passive Aggressive Classifier	Perceptron	Solves linear One-Class SVM	MultinomialNB
Precision	0.669447	0.572433	0.585191	0.589528	0.492799	0.020595	0.471286
Recall	0.506856	0.540085	0.522450	0.532846	0.547739	0.324447	0.619969
fscore	0.576915	0.555789	0.552044	0.559756	0.518818	0.038732	0.535498
Jaccard	0.446224	0.435779	0.426390	0.441326	0.415007	0.006601	0.414167
Accuracy	0.207228	0.174463	0.173036	0.183027	0.150847	0.000000	0.127879
Hamming Loss	0.015125	0.017564	0.017250	0.017052	0.020671	0.327652	0.021882

LDA	Logistic Regression	Linear SVC	lgbm Classifier	SGDClassifier	Passive Aggressive Classifier	Perceptron	Solves linear One-Class SVM	MultinomialNB
Precision	0.617196	0.682333	0.479329	0.738959	0.510526	0.150232	0.000000	0.666667
Recall	0.143759	0.115282	0.282480	0.089642	0.181740	0.195803	0.000000	0.000064
fscore	0.233201	0.197239	0.355472	0.159889	0.268056	0.170017	0.000000	0.000128
Jaccard	0.137141	0.120324	0.247605	0.100956	0.169310	0.102352	0.000000	0.000097
Accuracy	0.050736	0.048011	0.084344	0.042886	0.060339	0.011289	0.000000	0.000065
Hamming Loss	0.019234	0.019092	0.020841	0.019166	0.020193	0.038894	0.020345	0.020345

Model testing by vectorizer

BERT	Logistic Regression	Linear SVC	lgbm Classifier	SGDClassifier	Passive Aggressive Classifier	Perceptron	Solves linear One-Class SVM
Precision	0.035262	0.048678	0.790669	0.050321	0.061990	0.049485	0.020345
Recall	0.008706	0.083264	0.101059	0.063014	0.080681	0.046304	1.000000
fscore	0.013964	0.061438	0.179212	0.055957	0.070111	0.047842	0.039879
Jaccard	0.004848	0.033980	0.109217	0.033745	0.042170	0.028135	0.020345
Accuracy	0.000389	0.000519	0.049828	0.003374	0.002855	0.003828	0.000000
Hamming Loss	0.025014	0.051758	0.018833	0.043258	0.043542	0.037498	0.979655

USE	Logistic Regression	Linear SVC	lgbm Classifier	SGDClassifier	Passive Aggressive Classifier	Perceptron	Solves linear One-Class SVM
Precision	0.779060	0.766430	0.719652	0.811808	0.637822	0.451047	0.000000
Recall	0.456534	0.506155	0.474711	0.406914	0.518528	0.529626	0.000000
fscore	0.575703	0.609676	0.572065	0.542102	0.572022	0.487188	0.000000
Jaccard	0.448653	0.488059	0.448927	0.424783	0.456982	0.365046	0.000000
Accuracy	0.227470	0.252839	0.215532	0.217868	0.201583	0.102251	0.000000
Hamming Loss	0.013691	0.013186	0.014449	0.013986	0.015786	0.022684	0.020345

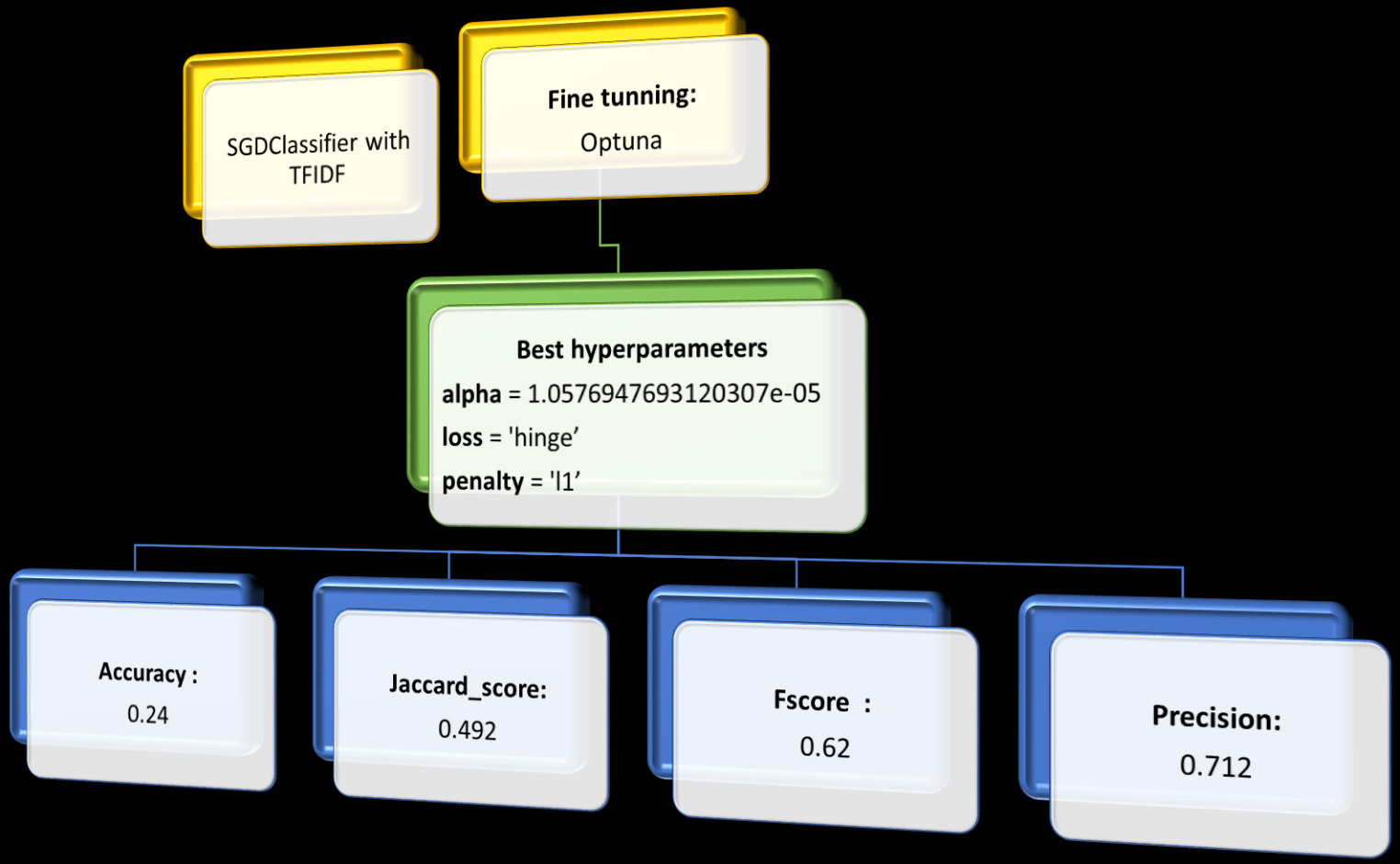
W2V	Logistic Regression	Linear SVC	lgbm Classifier	SGDClassifier	Passive Aggressive Classifier	Perceptron	Solves linear One-Class SVM
Precision	0.671335	0.751695	0.628383	0.665107	0.334457	0.323051	0.020462
Recall	0.305759	0.286338	0.297659	0.258339	0.316474	0.344984	0.928089
fscore	0.420158	0.414706	0.403964	0.372135	0.325217	0.333657	0.040041
Jaccard	0.284013	0.286153	0.278965	0.257924	0.222963	0.224797	0.018882
Accuracy	0.128593	0.136378	0.116330	0.113930	0.058717	0.040875	0.000000
Hamming Loss	0.017170	0.016444	0.017871	0.017736	0.026719	0.028034	0.905373

ML Production

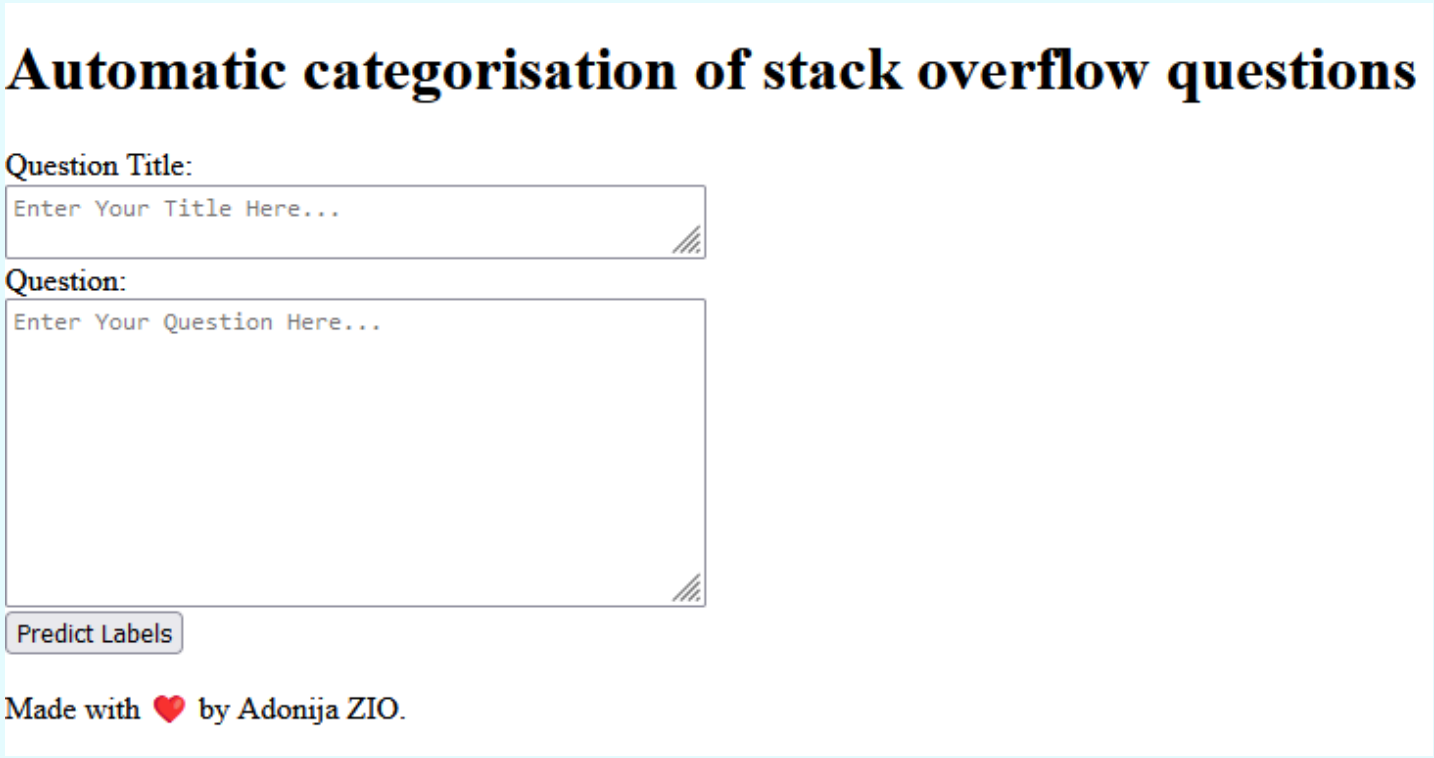
Model Selection

Model deployment

Model Selection



Model deployment



http://15.188.47.100:5000/

Conclusion

TF-IDF, Countvectorizer and USE are the best vectorizer

Ensemble model not perform well

SGDClassifier simple for fine tune

Improvement

- Fine tune for other model such logistic regression and LGBClassifier
- Running BERT with more memory
- Increase data and the number of tagging