black hat
USA 2023

AUGUST 9-10, 2023

BRIEFINGS

AI Assisted Decision Making of Security Review Needs

Mrityunjay Gautam
Pavan Kolachoor

$ whoami

Mrityunjay Gautam
Sr. Director, Product Security
Databricks.

@xdead10cc

Pavan Kolachoor
Sr. Manager, Product Security
Databricks.

@kolachoor

Disclaimer

The views expressed in this presentation are strictly those of the speaker(s). Any comments made or views expressed during this presentation are not endorsed by Databricks, and hence would not be a legal liability of Databricks.

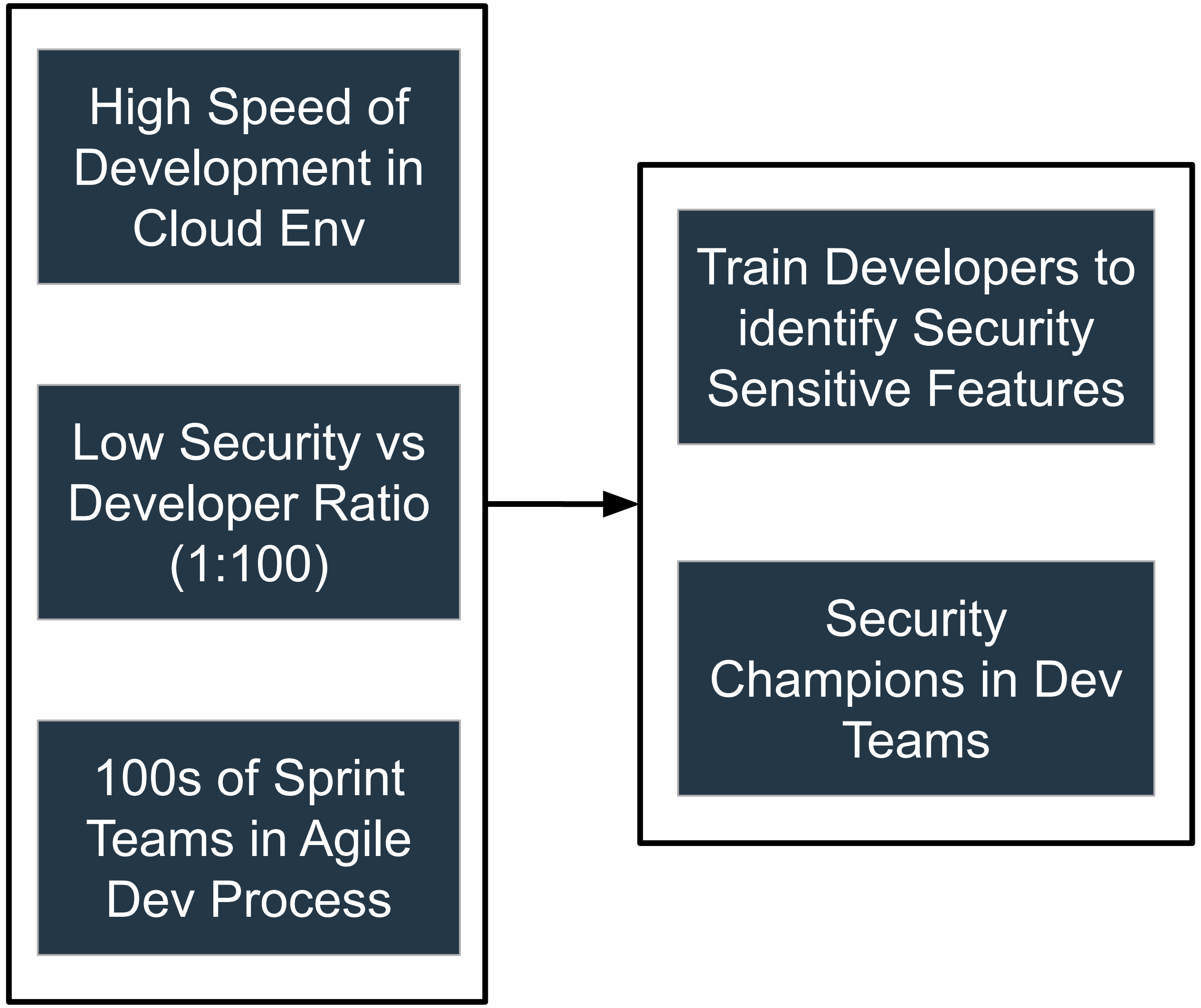
Problem Statement

High Speed of Development in Cloud Env

Low Security vs Developer Ratio (1:100)

100s of Sprint Teams in Agile Dev Process

Problem Statement



High Speed of Development in Cloud Env

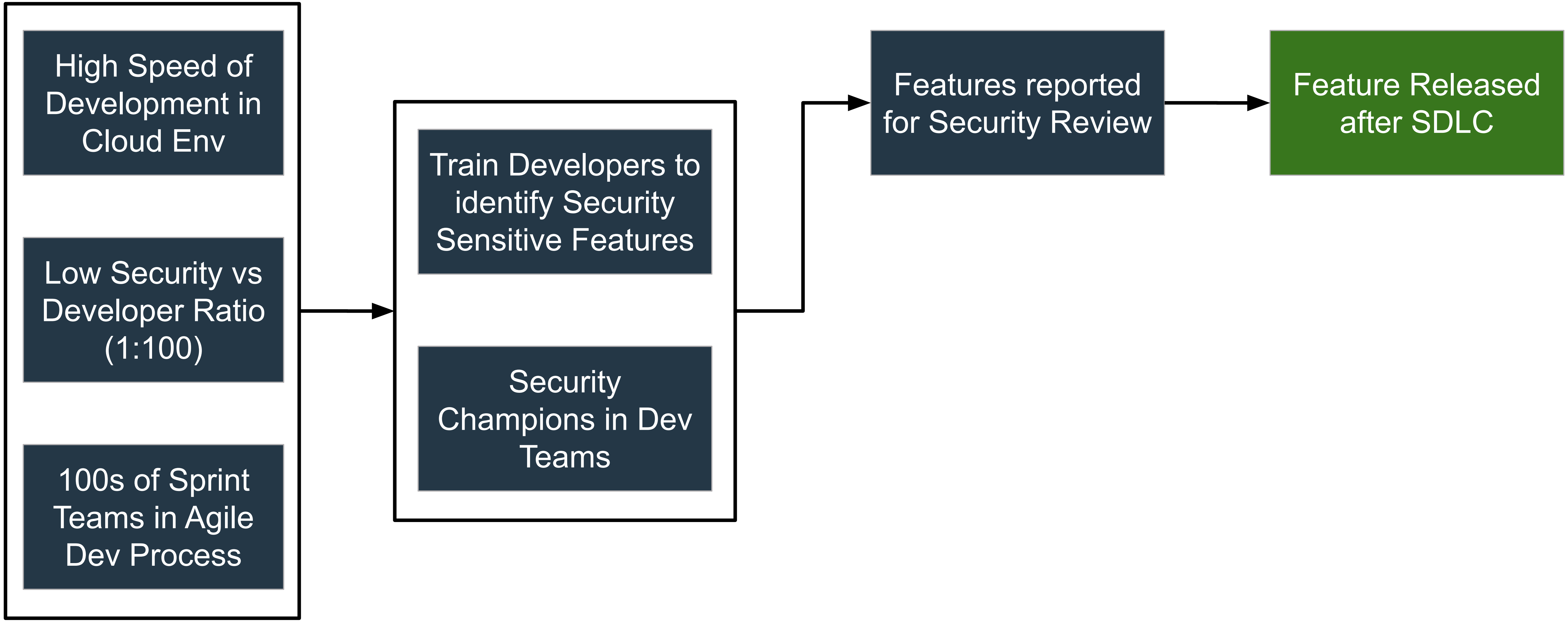
Low Security vs Developer Ratio (1:100)

100s of Sprint Teams in Agile Dev Process

Train Developers to identify Security Sensitive Features

Security Champions in Dev Teams

Problem Statement



High Speed of Development in Cloud Env

Low Security vs Developer Ratio (1:100)

100s of Sprint Teams in Agile Dev Process

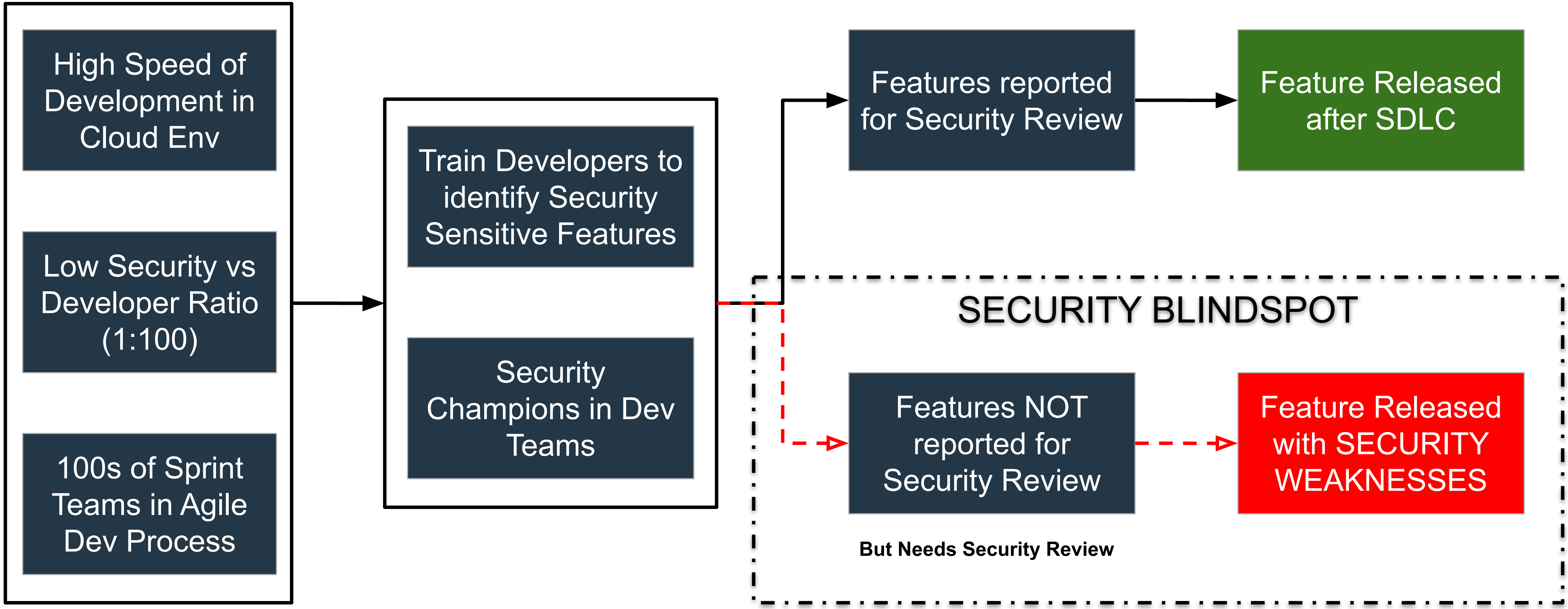
Train Developers to identify Security Sensitive Features

Security Champions in Dev Teams

Features reported for Security Review

Feature Released after SDLC

Problem Statement



High Speed of Development in Cloud Env

Low Security vs Developer Ratio (1:100)

100s of Sprint Teams in Agile Dev Process

Train Developers to identify Security Sensitive Features

Security Champions in Dev Teams

Features reported for Security Review

Feature Released after SDLC

SECURITY BLINDSPOT

Features NOT reported for Security Review

Feature Released with SECURITY WEAKNESSES

But Needs Security Review

Security Review Decision Making

- Is this a good candidate for Automation?
- Additionally needs a base human intelligence and some domain expertise in security and product knowledge

Hypothesis

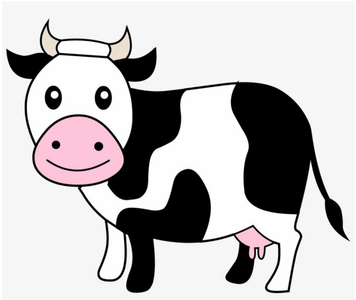
- **Can Deep Learning & NLP meet these requirements ?**

“Engineering” English vs “Spoken” English

- Engineering language is unique for any Organization
  - Product Names, Code Names, Abbreviations, etc..

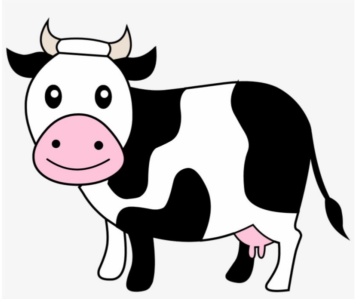
“Engineering” English vs “Spoken” English

- Engineering language is unique for any Organization
  - Product Names, Code Names, Abbreviations, etc..

	COW
Engineering	Copy on Write
Spoken English	

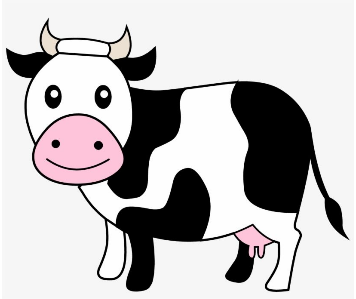
“Engineering” English vs “Spoken” English

- Engineering language is unique for any Organization
  - Product Names, Code Names, Abbreviations, etc..

	COW	PoC
Engineering	Copy on Write	Proof of Concept
Spoken English		

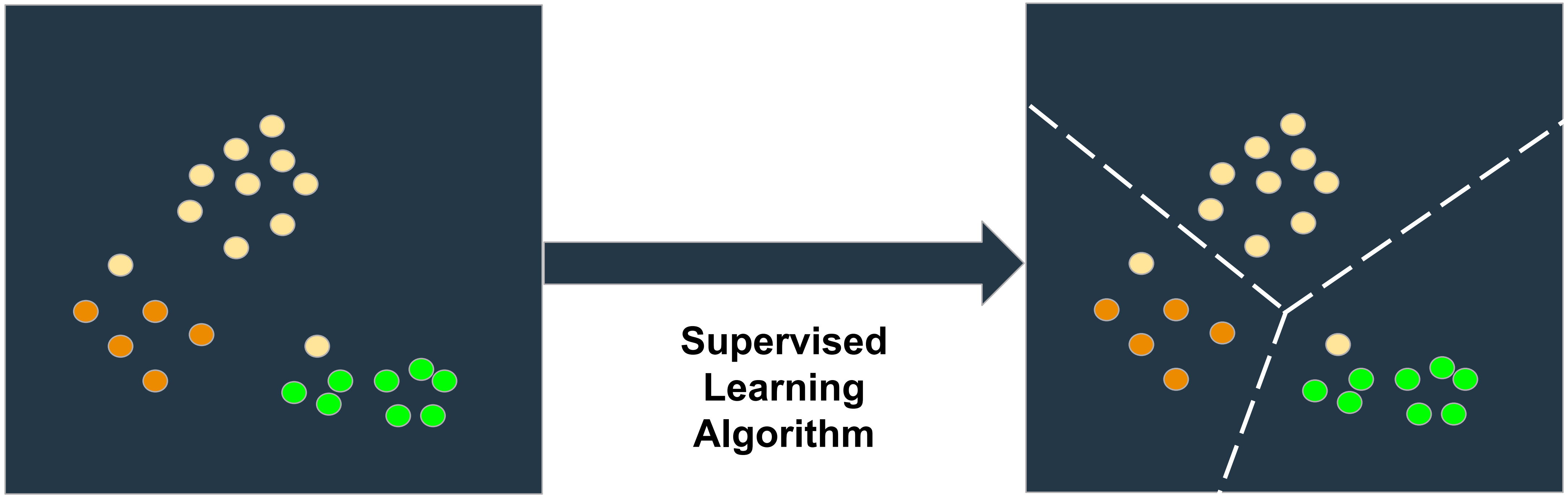
“Engineering” English vs “Spoken” English

- Engineering language is unique for any Organization
  - Product Names, Code Names, Abbreviations, etc..

	COW	PoC	Spark
Engineering	Copy on Write	Proof of Concept	Apache Spark
Spoken English			

Machine Learning Basics

Supervised Learning: Classifiers

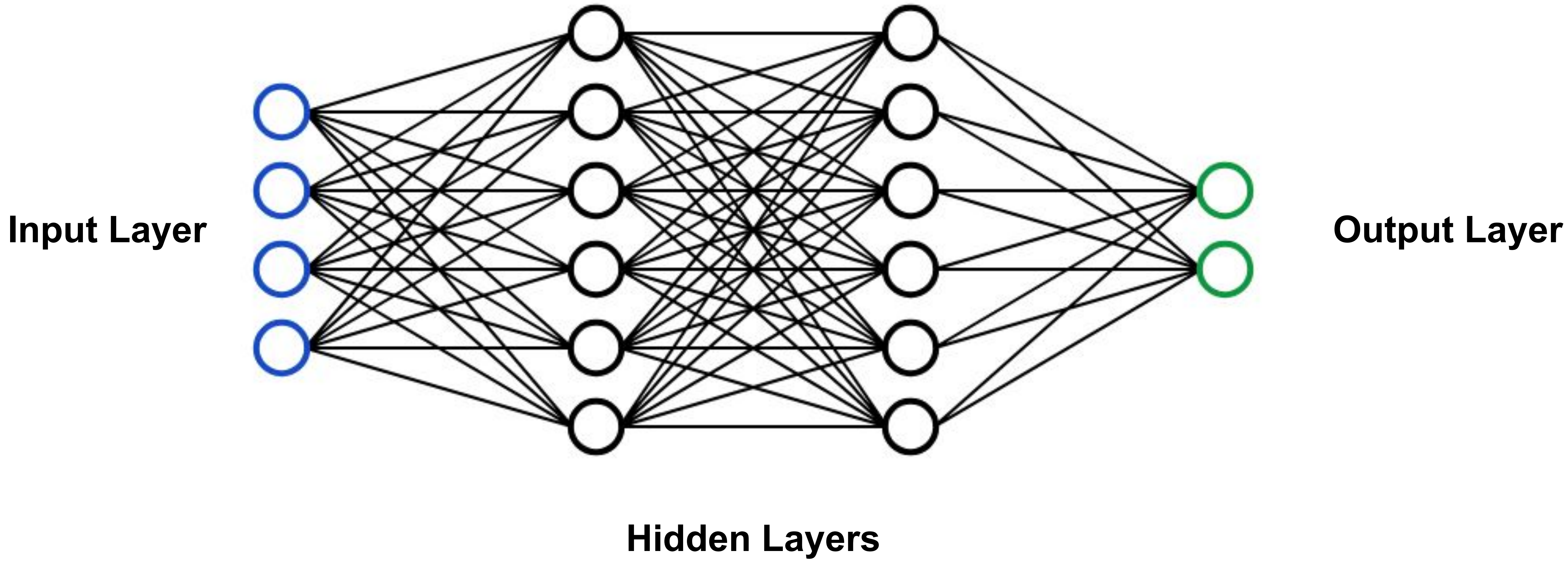


Supervised Learning Algorithm

Labelled Input

Taxonomy

Deep Learning: Multi Layer Perceptron



Training Data Collection

Because no training is possible without the right kind of data…

Engineering Text Sources

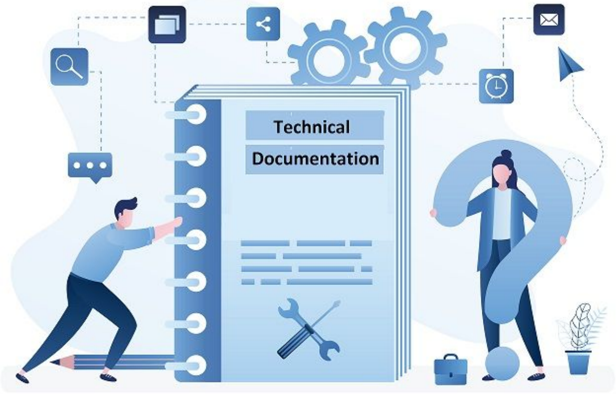
PDF

JIRA

Aha!

Confluence

OneDrive



Office 365

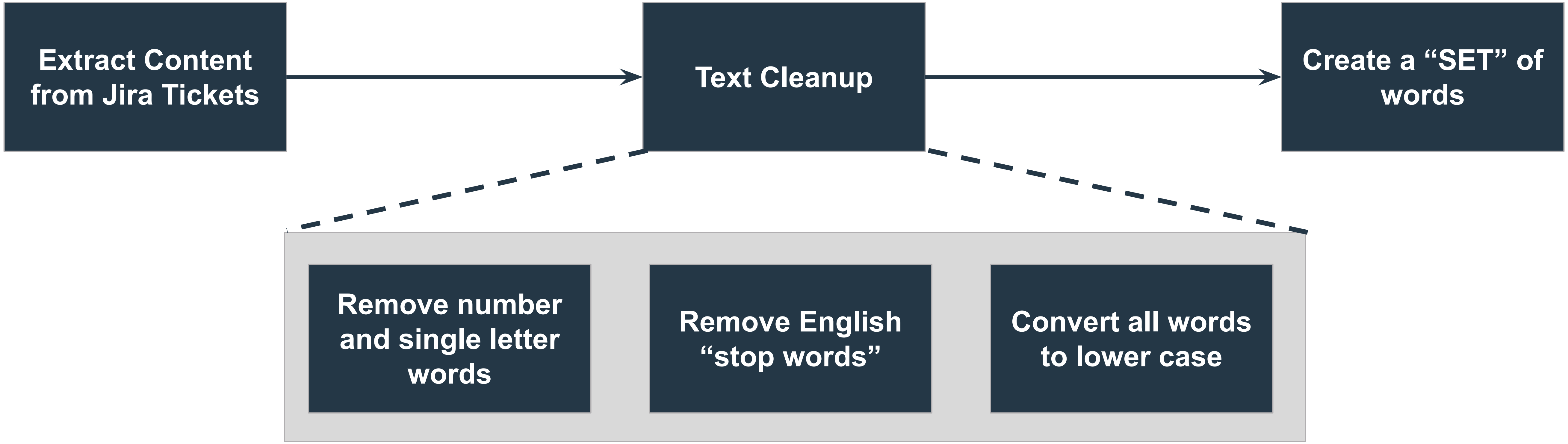
Data Download Strategy

Data Source	Authentication	Text Extraction Strategy
Jira	PAT Token	JSON Extraction => Linked Tickets
Confluence	PAT Token	Requests API => HTML Parsing => HTML Tag Cleanup
Google Doc	OAuth2 Token	Use Google Docs APIs => Extract Text
Aha!	API Key	Python Aha Package => Extract Feature Content
Public Webpages	NONE	Requests API => HTML Parsing => HTML Tag Cleanup
Local Files (Pdf, Docx, Xlsx, etc)	NONE	PDF: Standard Python packages and Text extraction techniques Office: Unzip and Parse using standard reg-ex

First Attempt: Model v1.0

Now that we have the data…

Building the Vocabulary



Extract Content from Jira Tickets

Text Cleanup

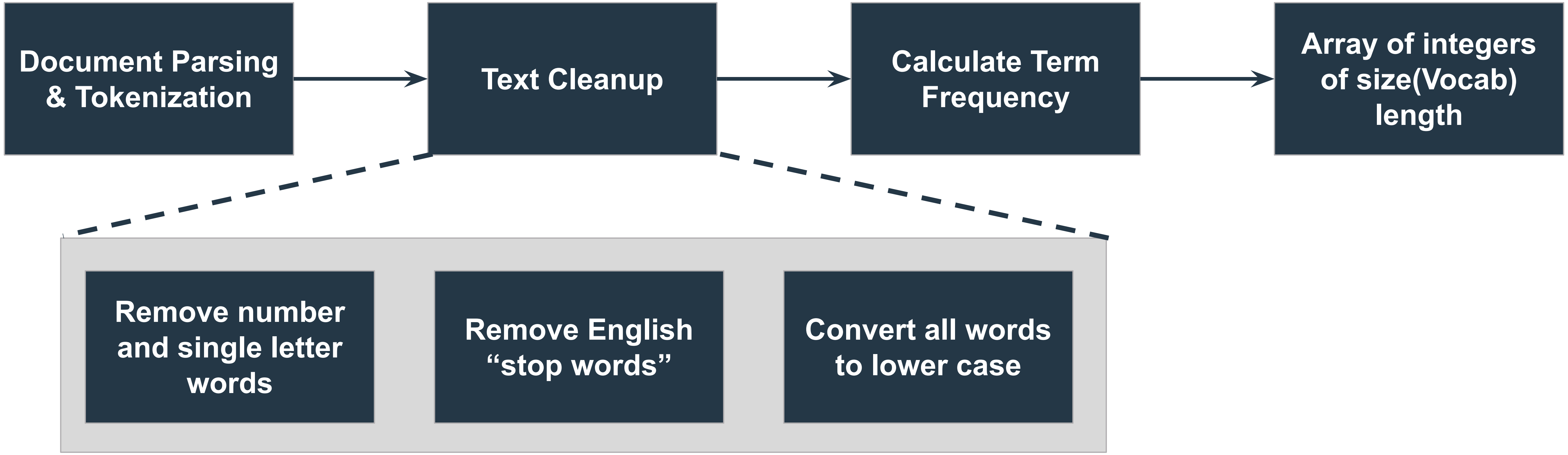
Create a “SET” of words

Remove number and single letter words

Remove English “stop words”

Convert all words to lower case

Vectorization of Documents


Document Parsing & Tokenization

Text Cleanup

Calculate Term Frequency

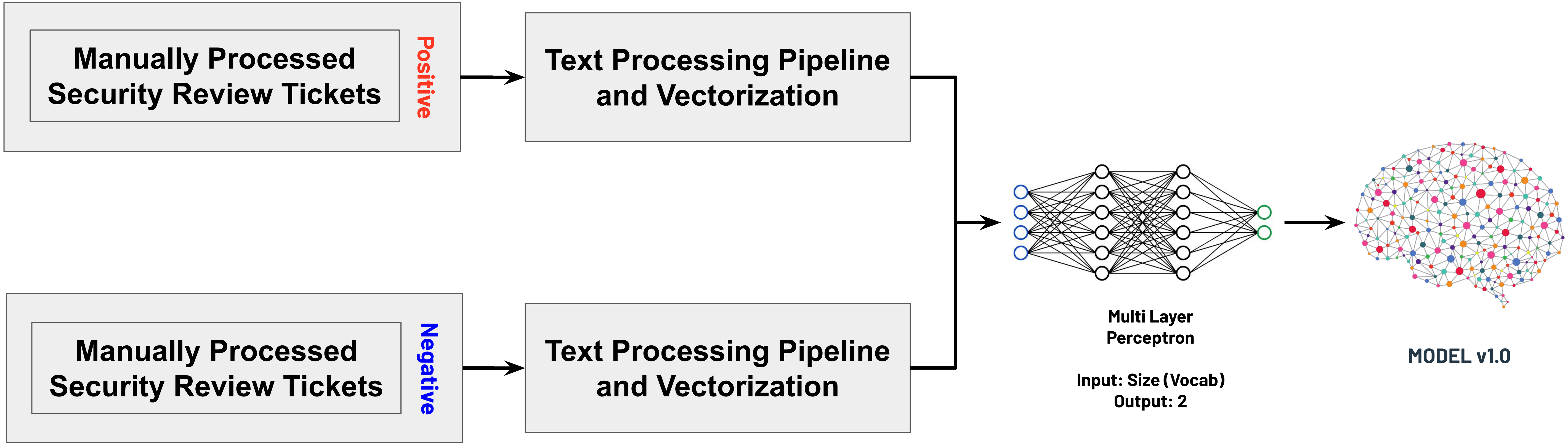
Array of integers of size(Vocab) length

Remove number and single letter words

Remove English “stop words”

Convert all words to lower case

Training Pipeline



Manually Processed Security Review Tickets

Positive

Text Processing Pipeline and Vectorization

Manually Processed Security Review Tickets

Negative

Text Processing Pipeline and Vectorization

Multi Layer Perceptron

Input: Size (Vocab)
Output: 2

MODEL v1.0

Results & Observations

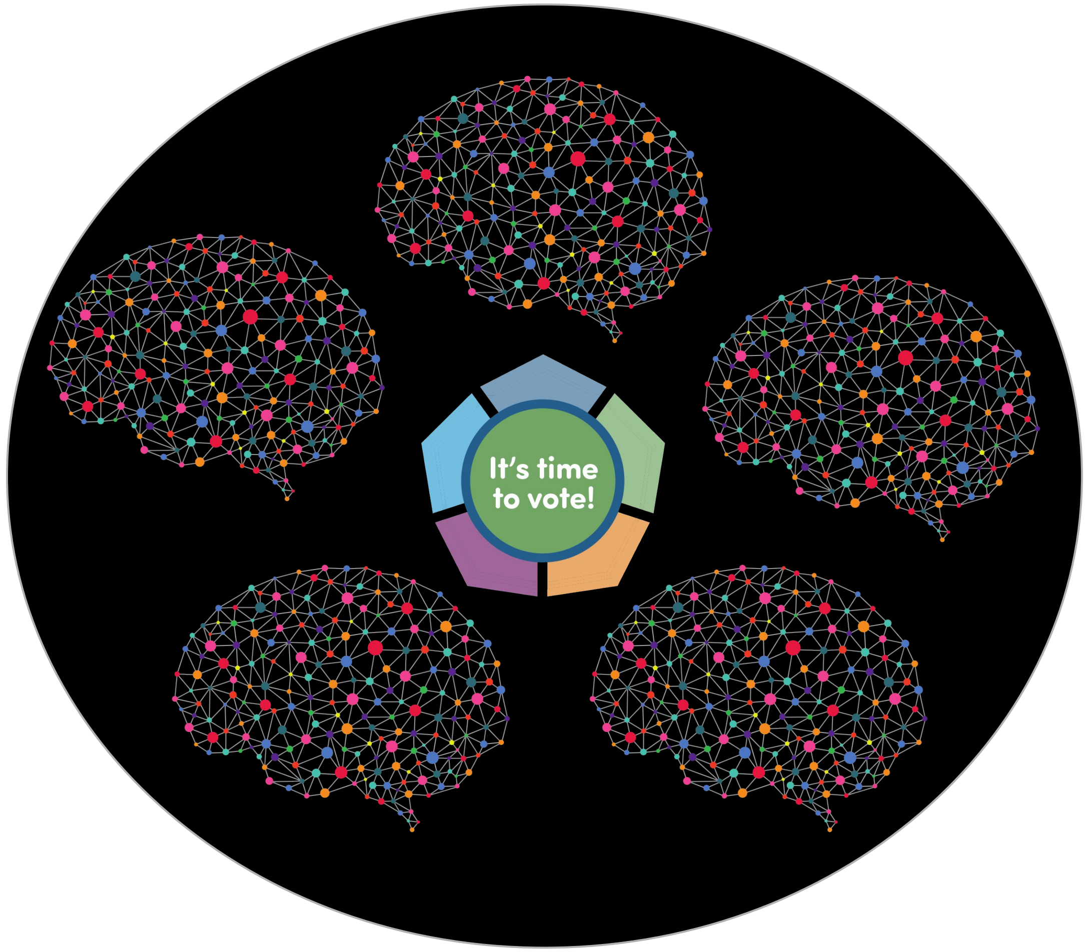
Model v1.0

Notes:

- Multiple configurations with different hidden layers
- Individual Model Accuracy = 63% to 71%

Can we try Ensemble Classifier ?

Results & Observations



Ensemble Model v1.5

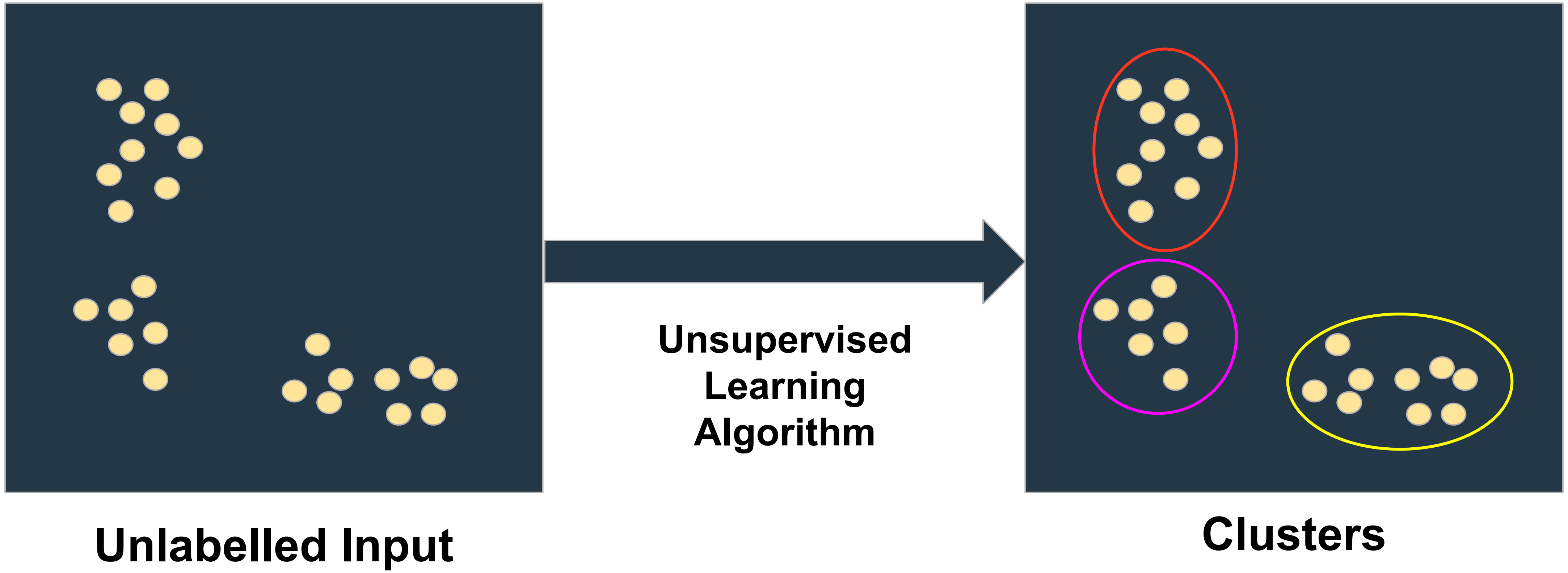
Accuracy: 78%

VERDICT

Not Acceptable for Use without Human Interference

Deeper into Machine Learning

Unsupervised Learning

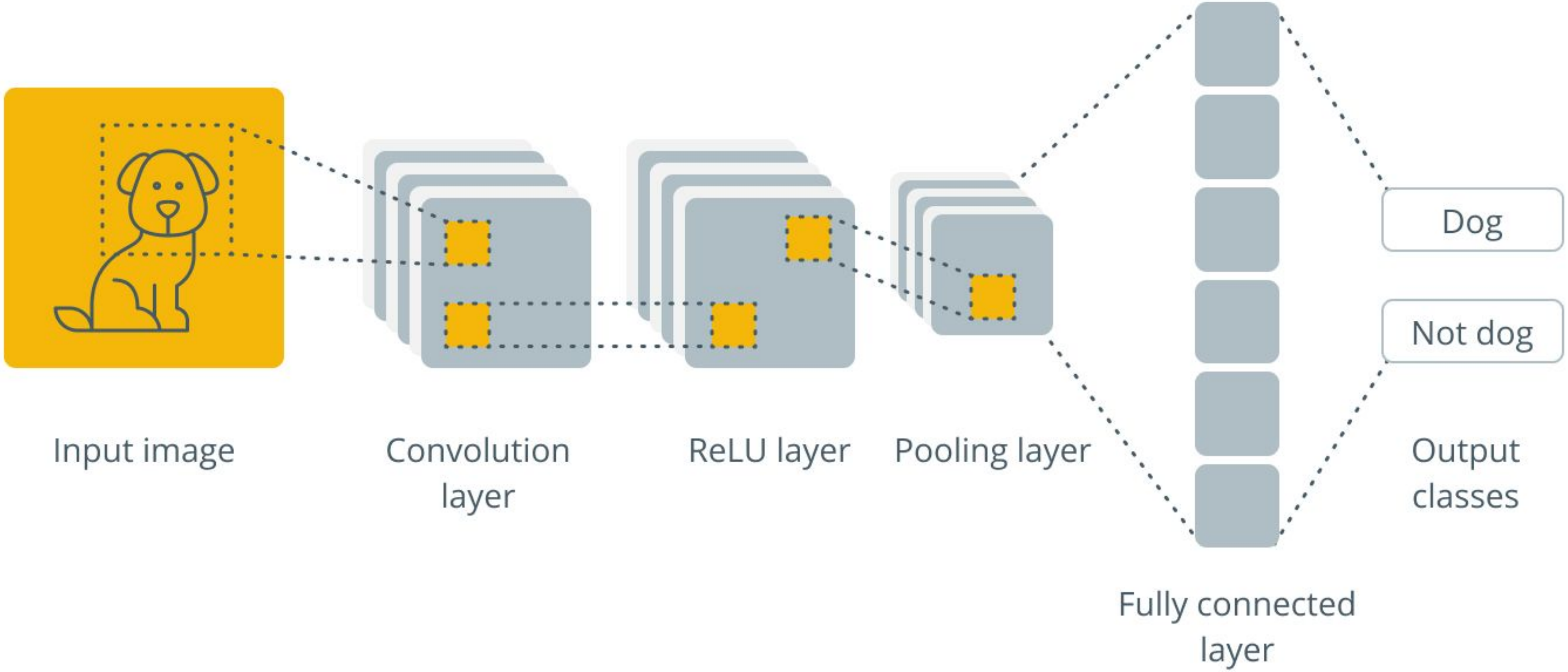


Unsupervised Learning Algorithm

Unlabelled Input

Clusters

Convolutional Neural Network



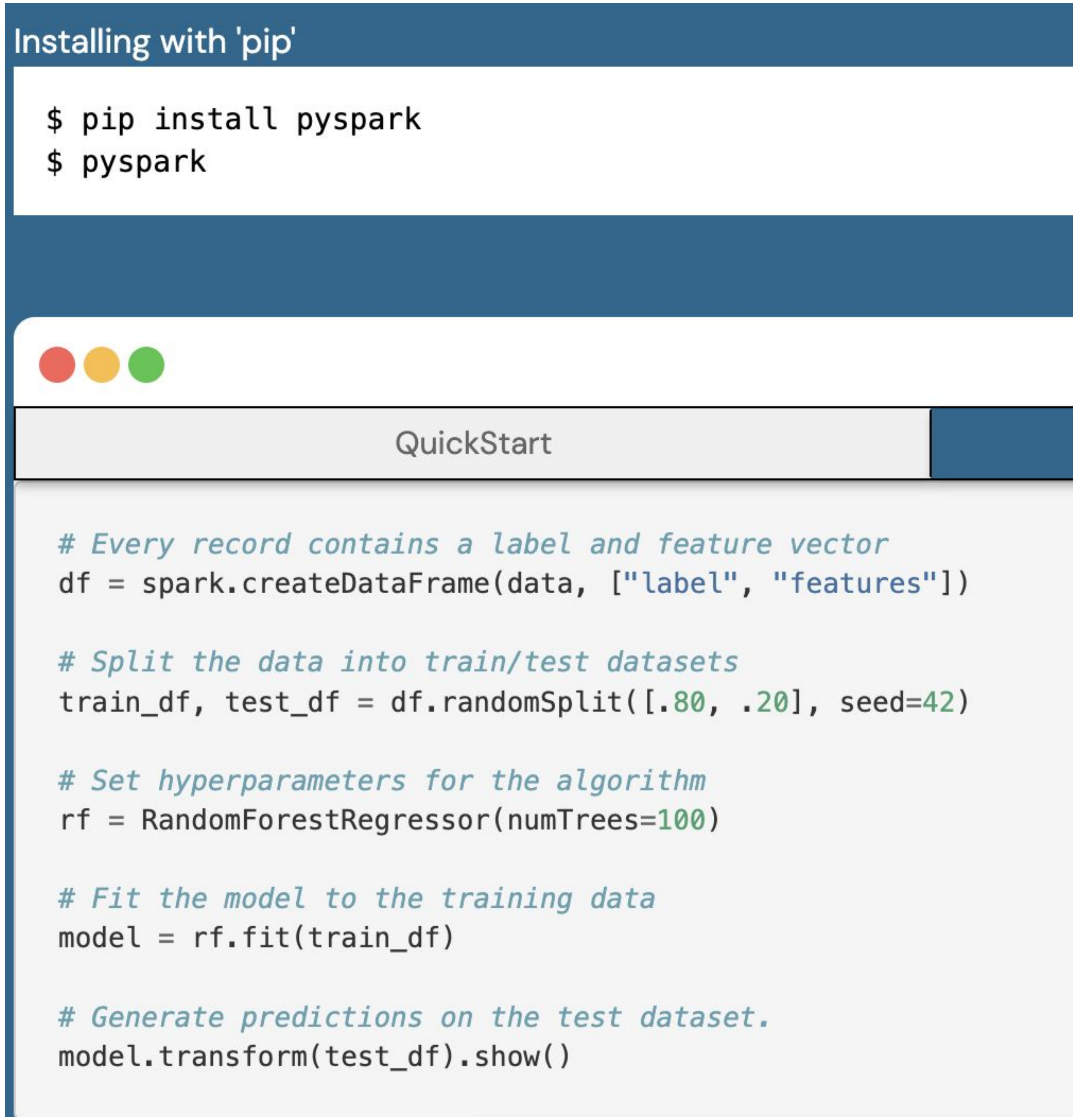
Final Implementation: Clairvoyant

The one that works…

Sample Use Case: Apache Spark

What is Apache Spark™?

Apache Spark™ is a multi-language engine for executing data engineering, data science, and machine learning on single-node machines or clusters.

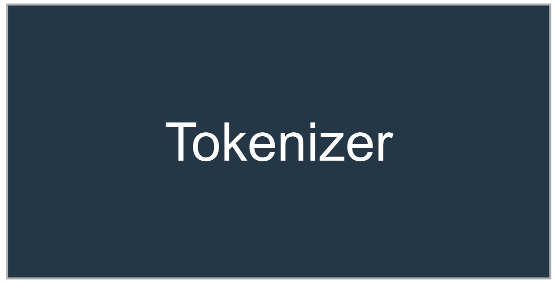


Installing with 'pip'

```
$ pip install pyspark
$ pyspark
```

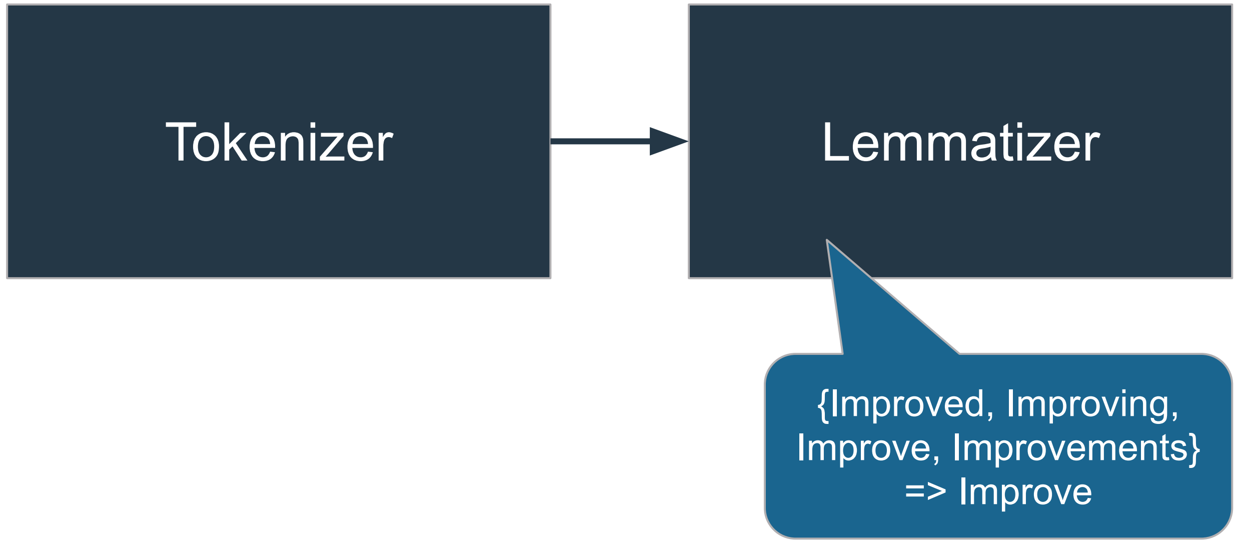
```
# Every record contains a label and feature vector
df = spark.createDataFrame(data, ["label", "features"])

# Split the data into train/test datasets
train_df, test_df = df.randomSplit([.80, .20], seed=42)

# Set hyperparameters for the algorithm
rf = RandomForestRegressor(numTrees=100)

# Fit the model to the training data
model = rf.fit(train_df)

# Generate predictions on the test dataset.
model.transform(test_df).show()
```

Text Cleanup: SparkNLP Pipeline



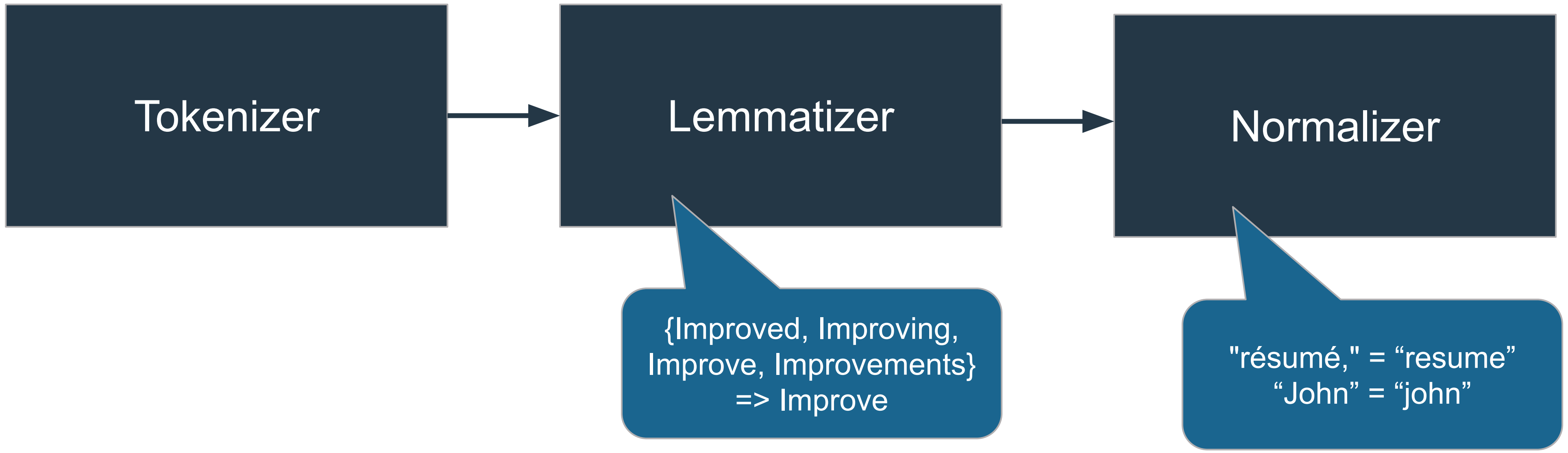
Tokenizer

Text Cleanup: SparkNLP Pipeline



{Improved, Improving, Improve, Improvements} => Improve

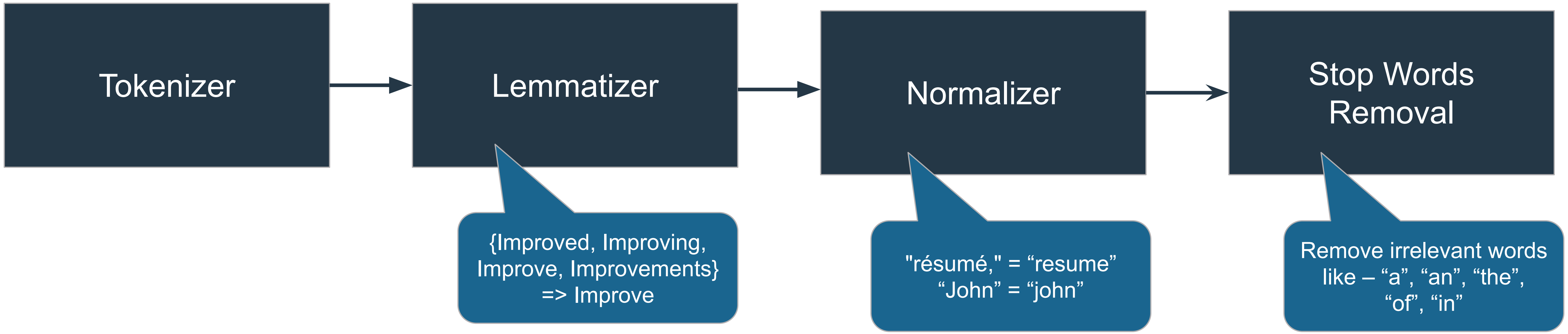
Text Cleanup: SparkNLP Pipeline



{Improved, Improving, Improve, Improvements} => Improve

"résumé," = “resume”
“John” = “john”

Text Cleanup: SparkNLP Pipeline



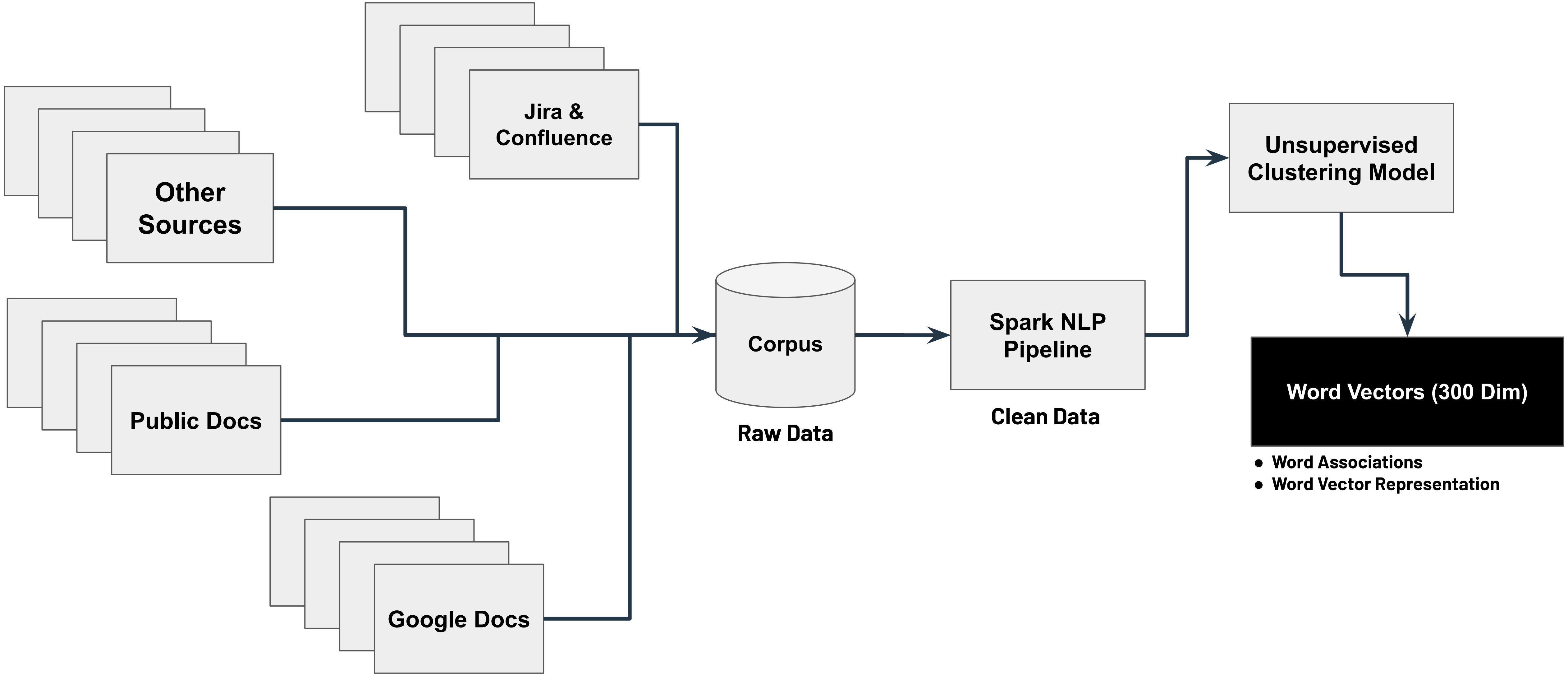
Tokenizer → Lemmatizer → Normalizer → Stop Words Removal

{Improved, Improving, Improve, Improvements} => Improve

"résumé," = "resume"
"John" = "john"

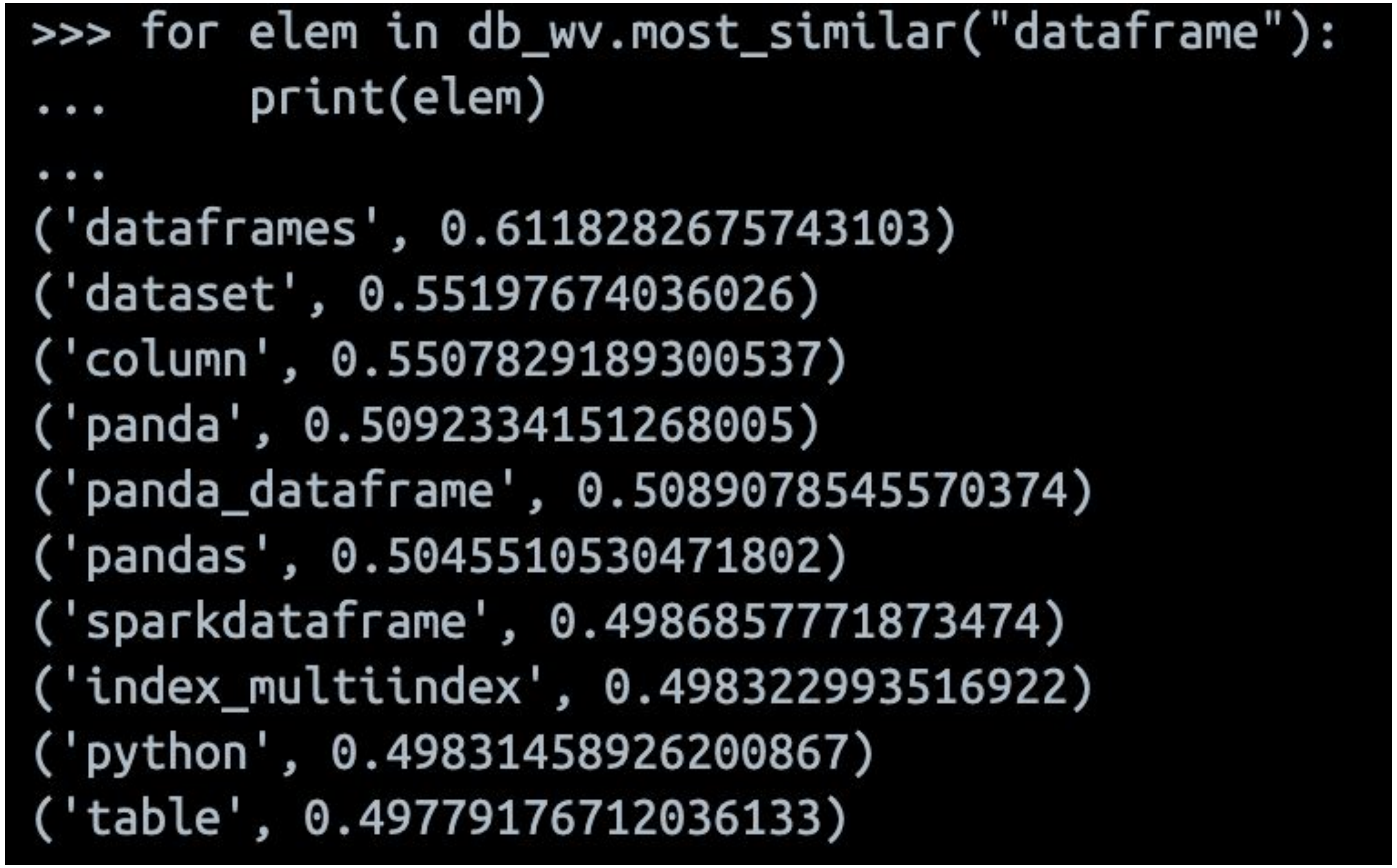
Remove irrelevant words like – "a", "an", "the", "of", "in"

Engineering Language Model Creation

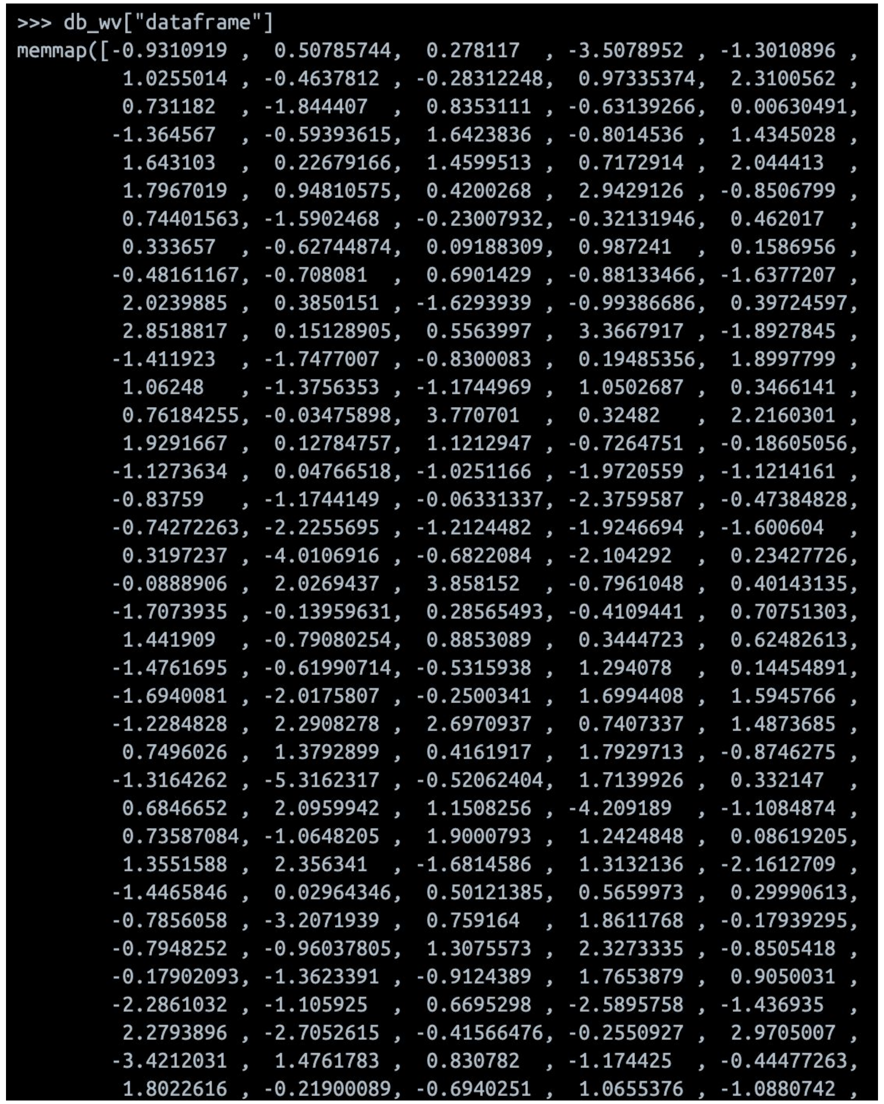


- Word Associations
- Word Vector Representation

Sample Word Vectors over Apache Spark

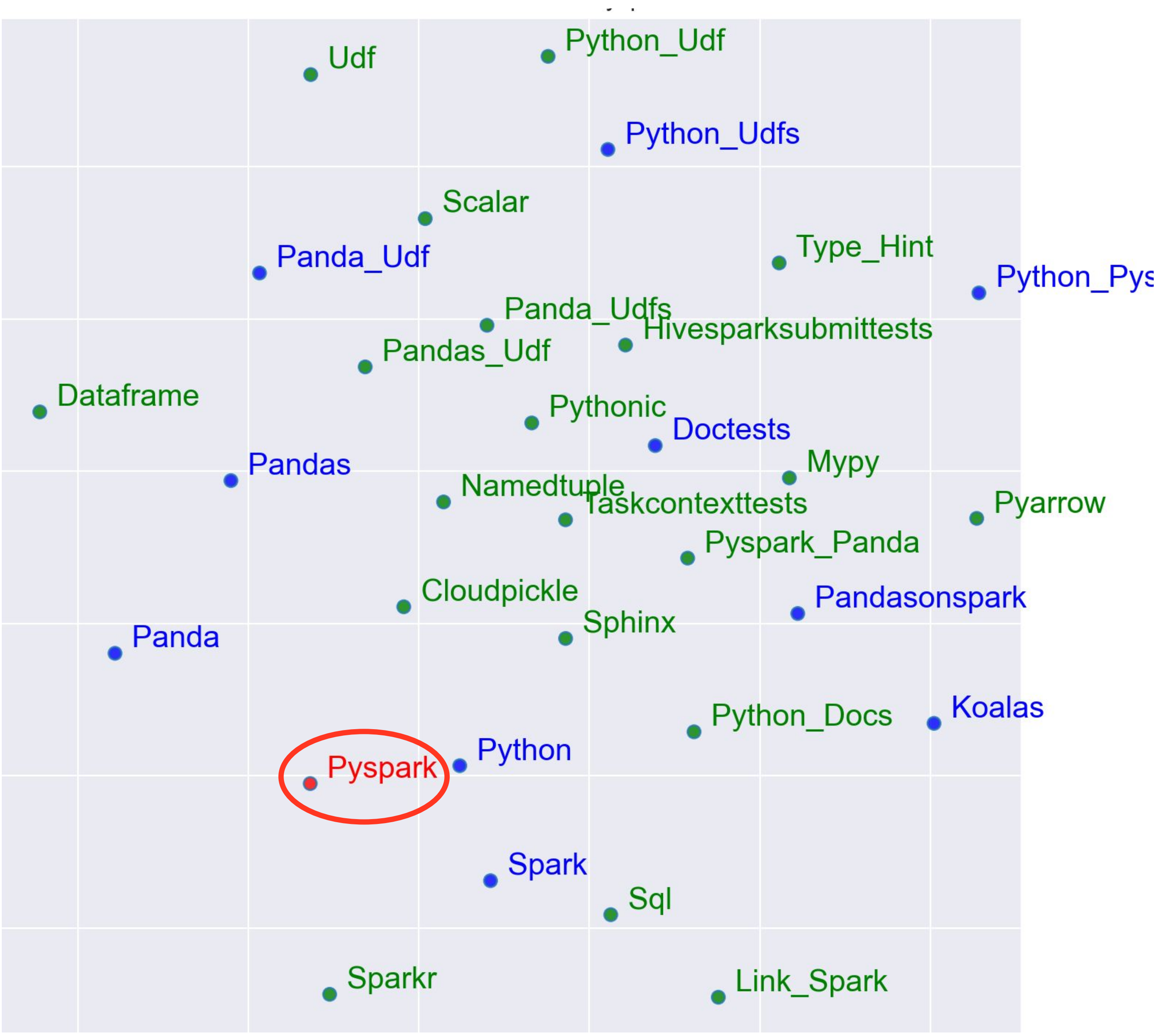


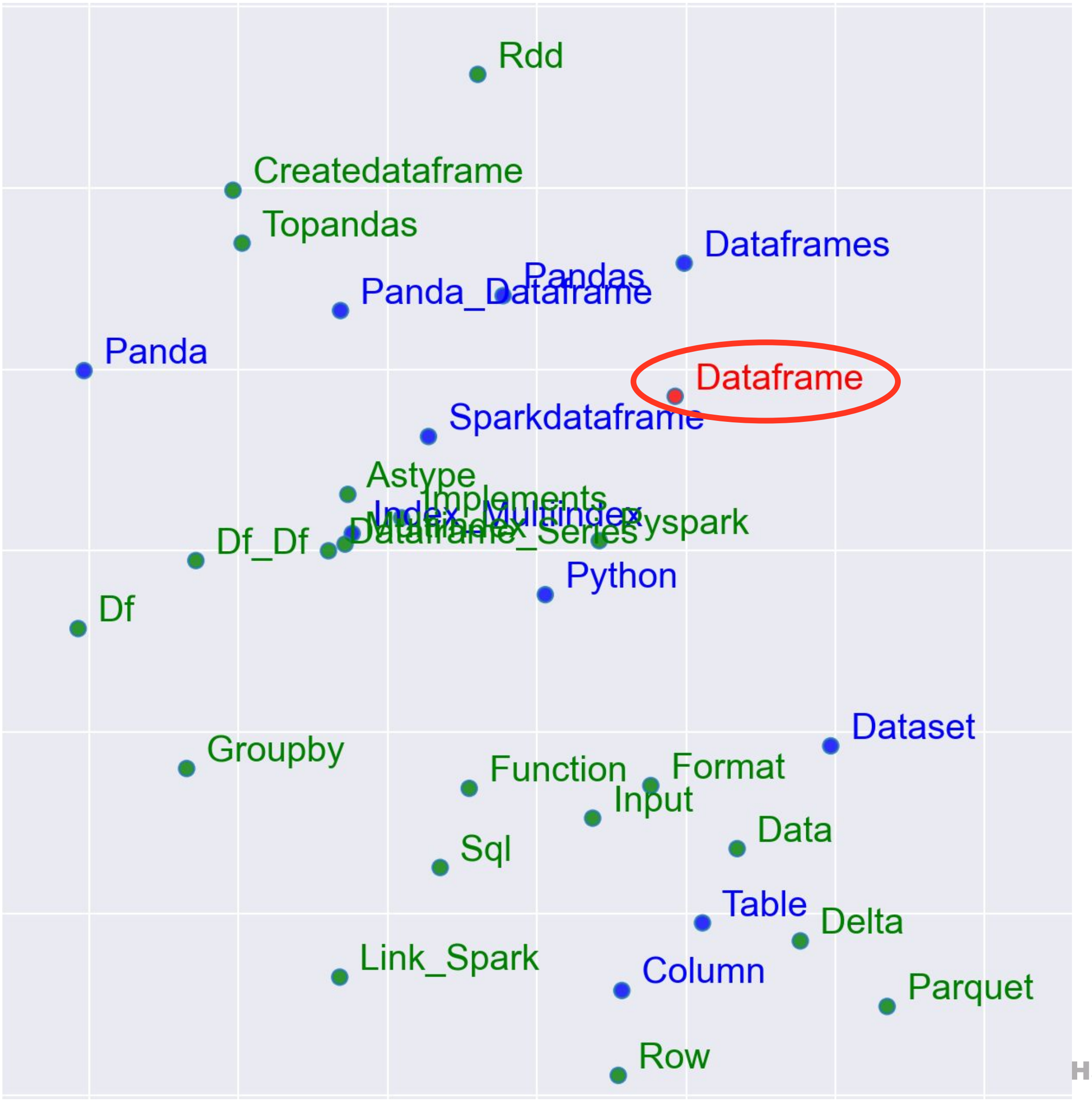
```
>>> for elem in db_wv.most_similar("dataframe"):
...     print(elem)
...
('dataframes', 0.611828267574310 3)
('dataset', 0.55197674036026)
('column', 0.5507829189300537)
('panda', 0.5092334151268005)
('panda_dataframe', 0.5089078545570374)
('pandas', 0.5045510530471802)
('sparkdataframe', 0.4986857771873474)
('index_multiindex', 0.49832293516922)
('python', 0.49831458926200867)
('table', 0.49779176712036133)
```



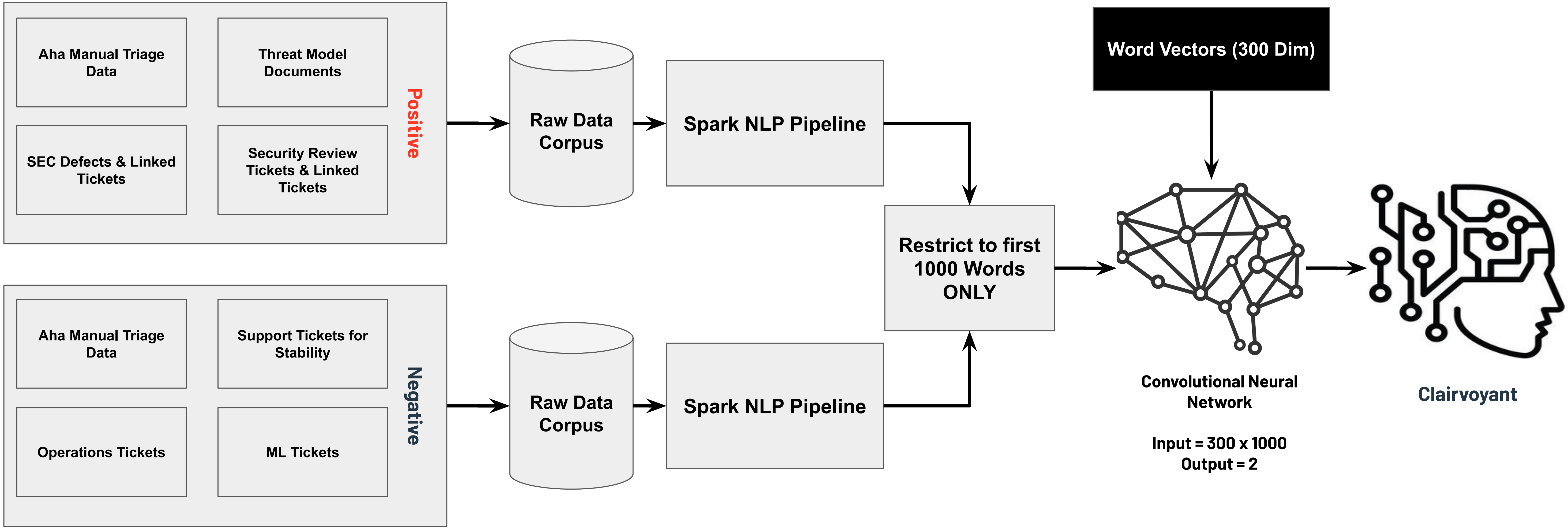
```
>>> db_wv["dataframe"]
memmap([-0.9310919 ,  0.50785744,  0.278117  , -3.5078952 , -1.3010896 ,
         1.0255014 , -0.4637812 , -0.28312248,  0.97335374,  2.3100562 ,
         0.731182  , -1.844407  ,  0.8353111 , -0.63139266,  0.00630491,
        -1.364567  , -0.59393615,  1.6423836 , -0.8014536 ,  1.4345028 ,
         1.643103  ,  0.22679166,  1.4599513 ,  0.7172914 ,  2.044413  ,
         1.7967019 ,  0.94810575,  0.4200268 ,  2.9429126 , -0.8506799 ,
         0.74401563, -1.5902468 , -0.23007932, -0.32131946,  0.462017  ,
         0.333657  , -0.62744874,  0.09188309,  0.987241  ,  0.1586956 ,
        -0.48161167, -0.708081  ,  0.6901429 , -0.88133466, -1.6377207 ,
         2.0239885 ,  0.3850151 , -1.6293939 , -0.99386686,  0.39724597,
         2.8518817 ,  0.15128905,  0.5563997 ,  3.3667917 , -1.8927845 ,
        -1.411923  , -1.7477007 , -0.8300083 ,  0.19485356,  1.8997799 ,
         1.06248   , -1.3756353 , -1.1744969 ,  1.0502687 ,  0.3466141 ,
         0.76184255, -0.03475898,  3.770701  ,  0.32482   ,  2.2160301 ,
         1.9291667 ,  0.12784757,  1.1212947 , -0.7264751 , -0.18605056,
        -1.1273634 ,  0.04766518, -1.0251166 , -1.9720559 , -1.1214161 ,
        -0.83759   , -1.1744149 , -0.06331337, -2.3759587 , -0.47384828,
        -0.74272263, -2.2255695 , -1.2124482 , -1.9246694 , -1.600604  ,
         0.3197237 , -4.0106916 , -0.6822084 , -2.104292  ,  0.23427726,
        -0.0888906 ,  2.0269437 ,  3.858152  , -0.7961048 ,  0.40143135,
        -1.7073935 , -0.13959631,  0.28565493, -0.4109441 ,  0.70751303,
         1.441909  , -0.79080254,  0.8853089 ,  0.3444723 ,  0.62482613,
        -1.4761695 , -0.61990714, -0.5315938 ,  1.294078  ,  0.14454891,
        -1.6940081 , -2.0175807 , -0.2500341 ,  1.6994408 ,  1.5945766 ,
        -1.2284828 ,  2.2908278 ,  2.6970937 ,  0.7407337 ,  1.4873685 ,
         0.7496026 ,  1.3792899 ,  0.4161917 ,  1.7929713 , -0.8746275 ,
        -1.3164262 , -5.3162317 , -0.52062404,  1.7139926 ,  0.332147  ,
         0.6846652 ,  2.0959942 ,  1.1508256 , -4.209189  , -1.1084874 ,
         0.73587084, -1.0648205 ,  1.9000793 ,  1.2424848 ,  0.08619205,
         1.3551588 ,  2.356341  , -1.6814586 ,  1.3132136 , -2.1612709 ,
        -1.4465846 ,  0.02964346,  0.50121385,  0.5659973 ,  0.29990613,
        -0.7856058 , -3.2071939 ,  0.759164  ,  1.8611768 , -0.17939295,
        -0.7948252 , -0.96037805,  1.3075573 ,  2.3273335 , -0.8505418 ,
        -0.17902093, -1.3623391 , -0.9124389 ,  1.7653879 ,  0.9050031 ,
        -2.2861032 , -1.105925  ,  0.6695298 , -2.5895758 , -1.436935  ,
         2.2793896 , -2.7052615 , -0.41566476, -0.2550927 ,  2.9705007 ,
        -3.4212031 ,  1.4761783 ,  0.830782  , -1.174425  , -0.44477263,
         1.8022616 , -0.21900089, -0.6940251 ,  1.0655376 , -1.0880742 ,
```

Visualizing the Word Vectors in TWO Dimensions





Training Deep Learning Classifier



Aha Manual Triage Data

Threat Model Documents

SEC Defects & Linked Tickets

Security Review Tickets & Linked Tickets

Positive

Aha Manual Triage Data

Support Tickets for Stability

Operations Tickets

ML Tickets

Negative

Raw Data Corpus

Spark NLP Pipeline

Raw Data Corpus

Spark NLP Pipeline

Restrict to first 1000 Words ONLY

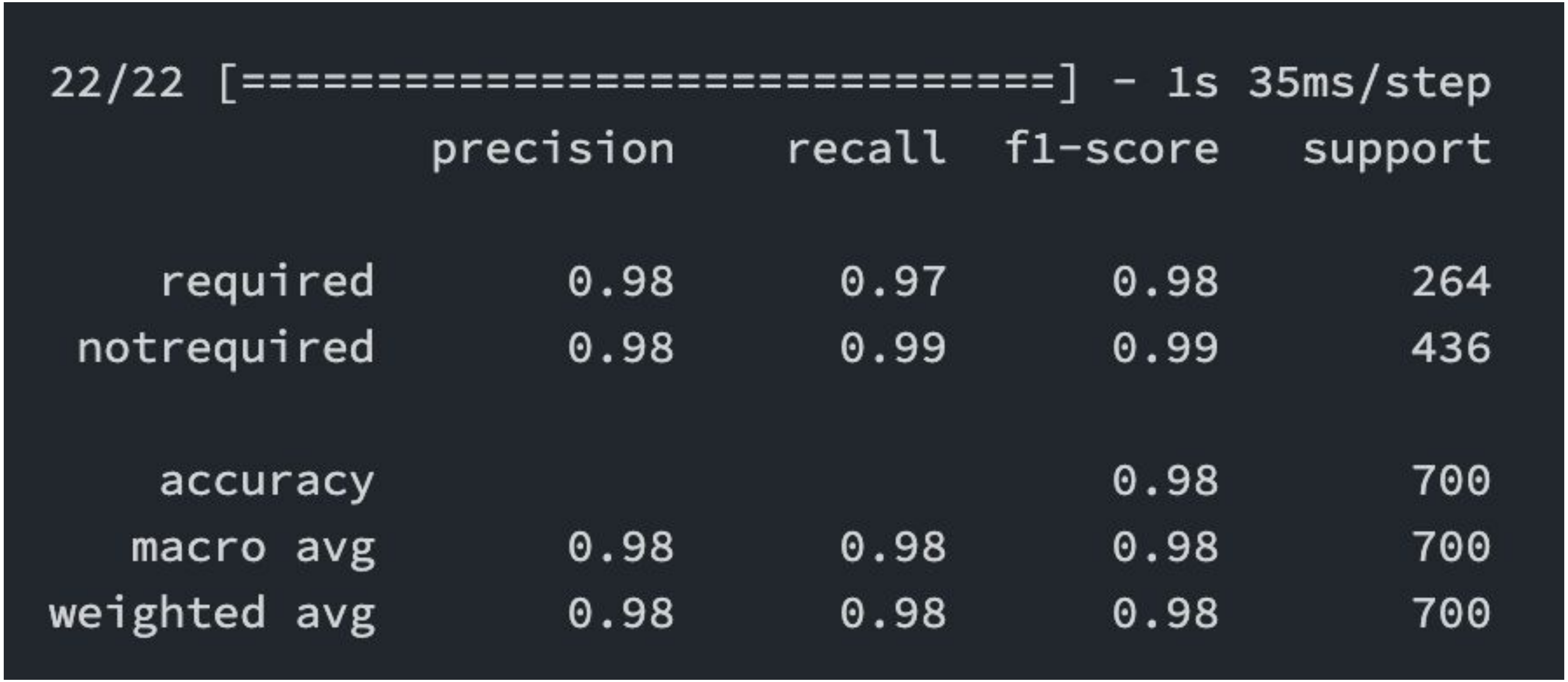
Word Vectors (300 Dim)

Convolutional Neural Network

Input = 300 x 1000
Output = 2

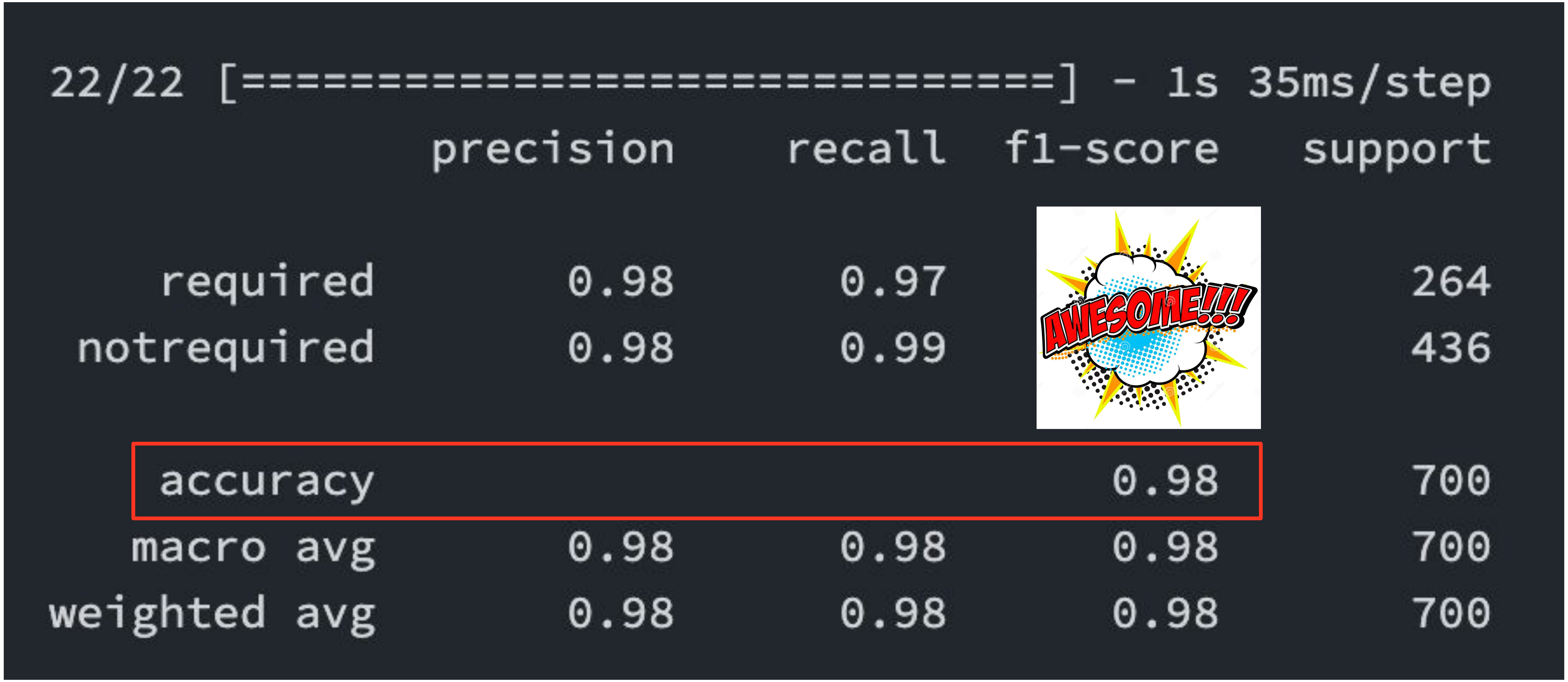
Clairvoyant

Confusion Matrix



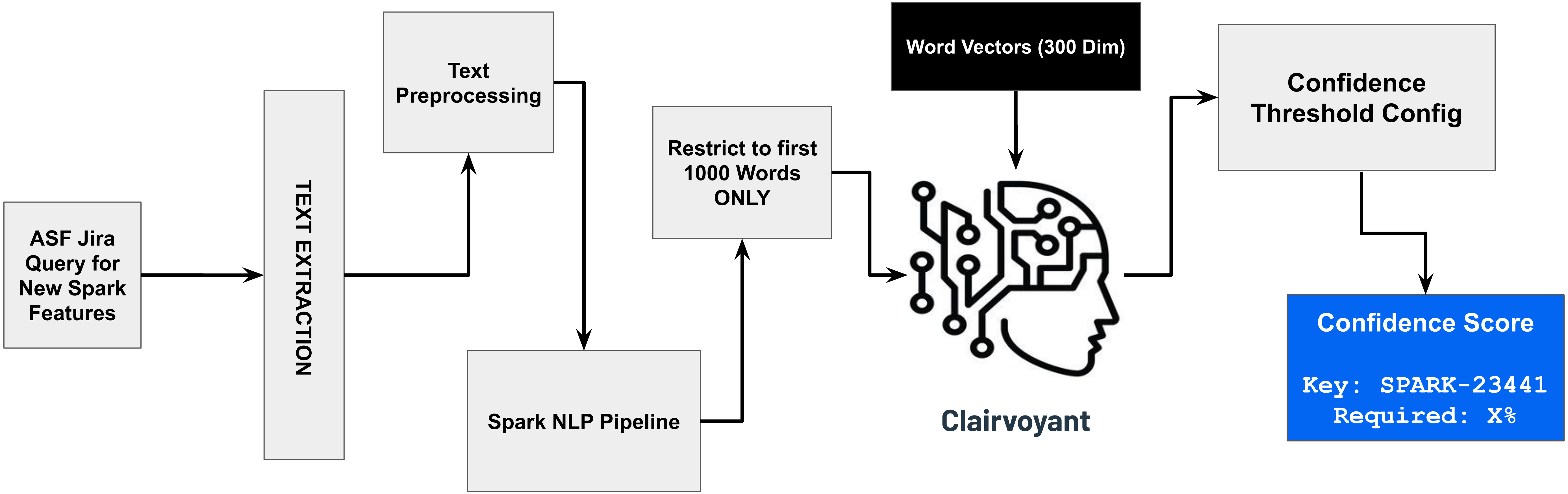
```
22/22 [==============================] - 1s 35ms/step
              precision    recall  f1-score   support

    required       0.98      0.97      0.98       264
 notrequired       0.98      0.99      0.99       436

    accuracy                           0.98       700
   macro avg       0.98      0.98      0.98       700
weighted avg       0.98      0.98      0.98       700
```

Confusion Matrix



```
22/22 [==============================] - 1s 35ms/step
               precision    recall  f1-score   support

    required        0.98      0.97                 264
 notrequired        0.98      0.99                 436

    accuracy                            0.98       700
   macro avg        0.98      0.98      0.98       700
weighted avg        0.98      0.98      0.98       700
```

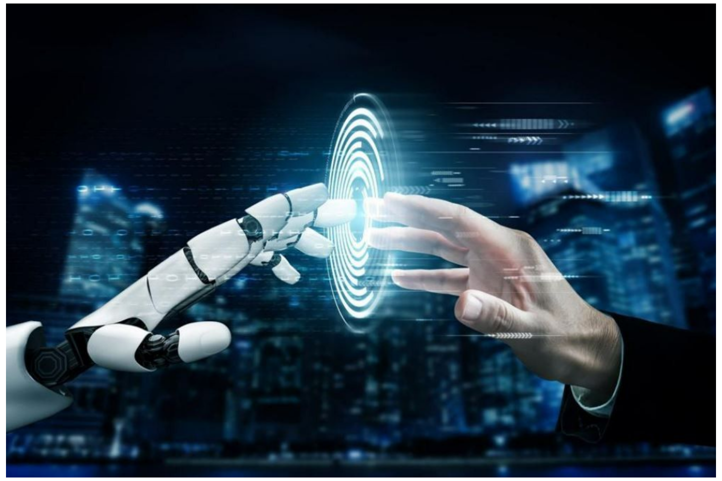
Using the Trained Model for Prediction



DEMO

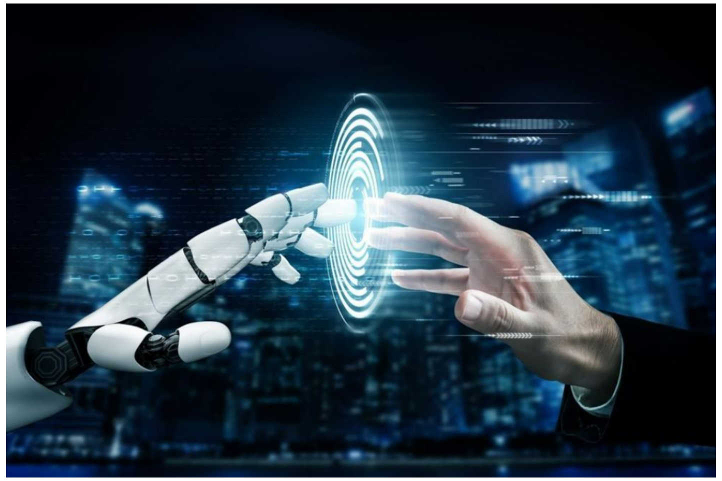
Key Takeaways

Time to move to the next stage of automation power by AI



Key Takeaways

Time to move to the next stage of automation power by AI



Engineering English is NOT same as Spoken English

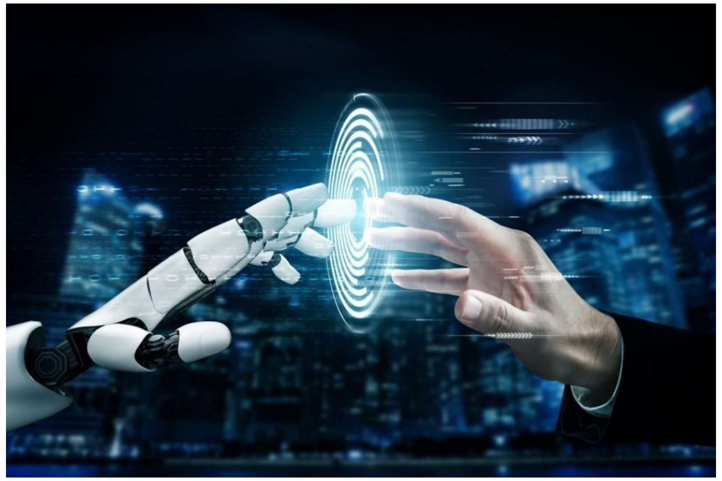
CNN = “Convolutional Neural Network”

OR

CNN

Key Takeaways

Time to move to the next stage of automation power by AI



Engineering English is NOT same as Spoken English

CNN = “Convolutional Neural Network”

OR

CNN

AI can “nitro boost” Security Development Lifecycle and DevSecOps

Questions?