Plagiarism Detector – Batch Prediction and Cloud Deployment

Pranav Prajapati, Sonali Johari
Instructor: Prof. Edward Stohr

STEVENS INSTITUTE of TECHNOLOGY
THE INNOVATION UNIVERSITY®
1870
Business Intelligence & Analytics

Introduction

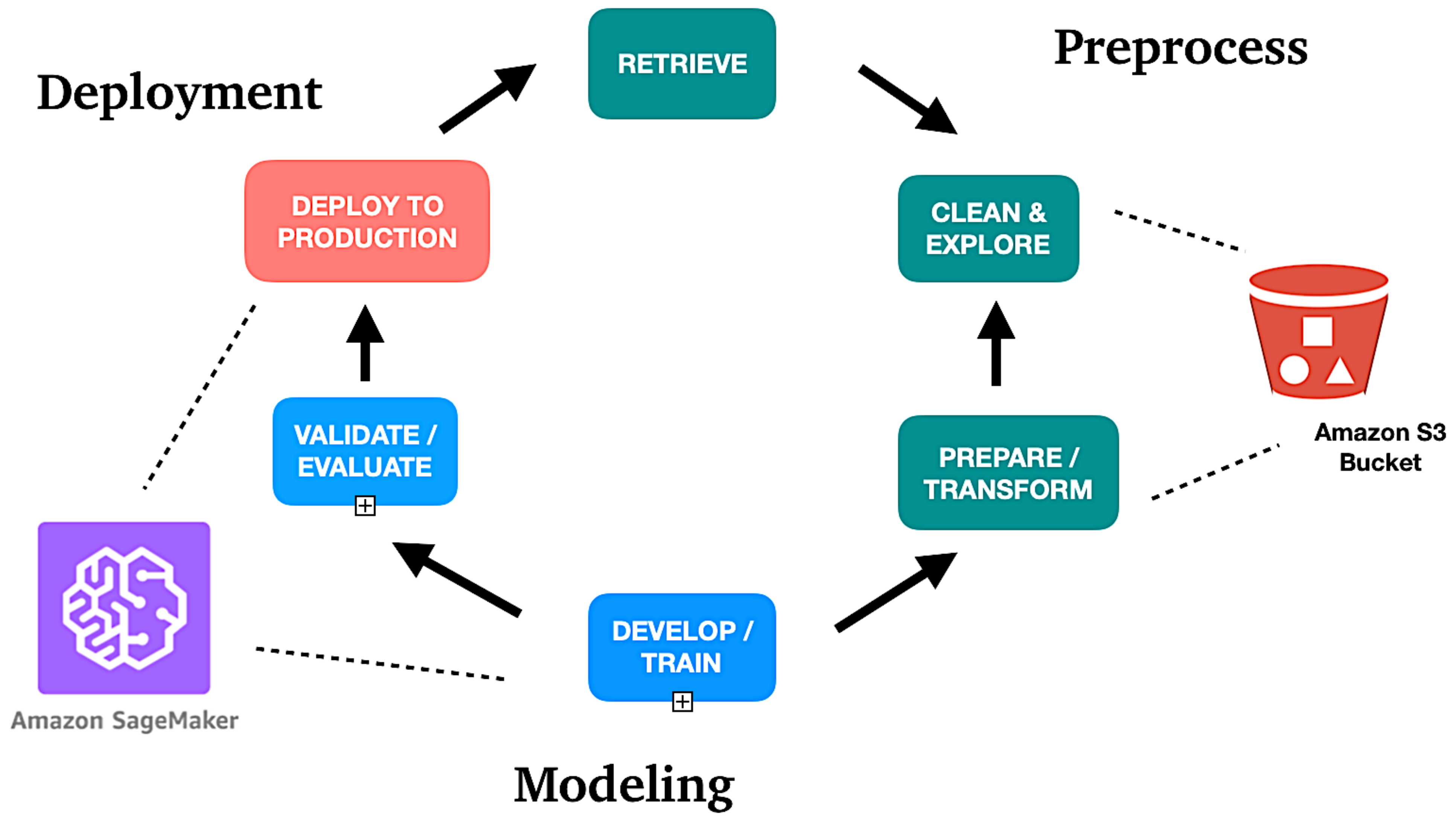
- Plagiarism detection is commonly used by universities and journals to maintain integrity of original content
- Detecting plagiarism is an area of active research – the task is non-trivial and the differences between content paraphrased and original content is not that obvious

Categories of Plagiarism

- Plagiarized : *cut, light, heavy*
- Non-Plagiarized: *non*
- Special, Source text : *org*

Data Preprocessing

- Implemented stratified random sampling to randomly split data by task and plagiarism amount
- Ensures an approximate 74% of training and 26% of testing data
- Text processing cleaned the data of punctuations and converted the entire text to lowercase



Numerical Columns

- According to set of rules, the categories were converted to numerical values

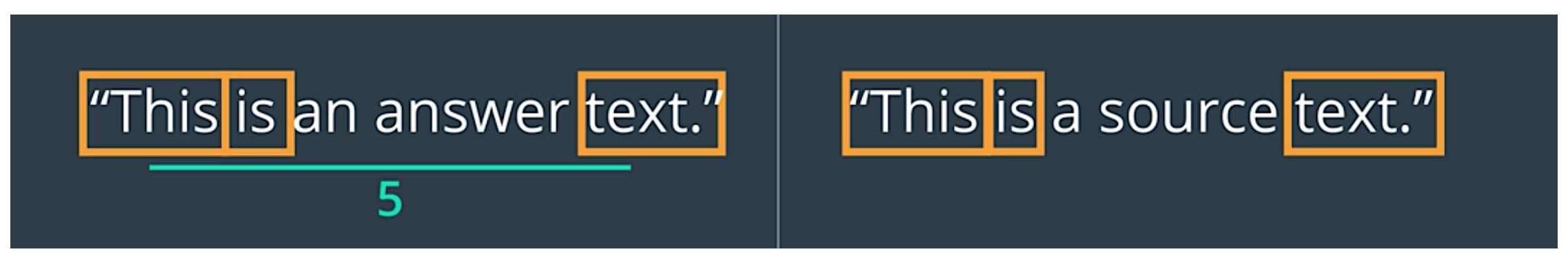
Class Labels

- The Goal is to create a binary classifier. Hence a binary class label 1 – plagiarized and 0 for not was made

Determining Plagiarism

- One of the ways to detect plagiarism is by computing similarity features - a measure of how similar a given text is to the original text.

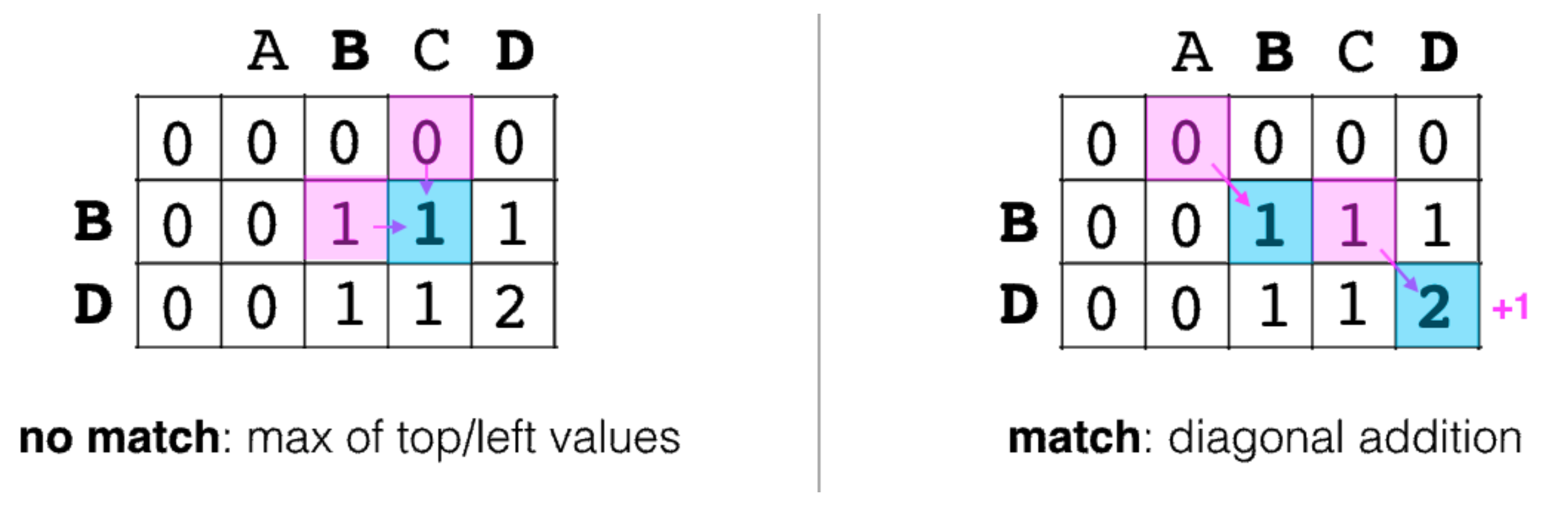
Containment


"This is an answer text."
5
"This is a source text."

- Containment is the ratio of the intersection of the n-gram word count of the original text with the n-gram word count of the paraphrased text to the n-gram word count of the paraphrased text, given by:

$$\frac{\Sigma\, count(ngram_P) \cap count(ngramO)}{\Sigma\, count(ngramP)}$$

Longest Common Subsequence



	A	B	C	D
	0	0	0	0	0
B	0	0	1	1	1
D	0	0	1	1	2

no match: max of top/left values

	A	B	C	D
	0	0	0	0	0
B	0	0	1	1	1
D	0	0	1	1	2 +1

match: diagonal addition

- LCS is the longest string of words (or letters) that are the same between the original text and the paraphrased text
- Using dynamic programming, an optimal method was implemented

Feature Selection

- Multiple containment features for different n-grams
- Removed highly correlated features and selected the best ones for our model

Loading Data

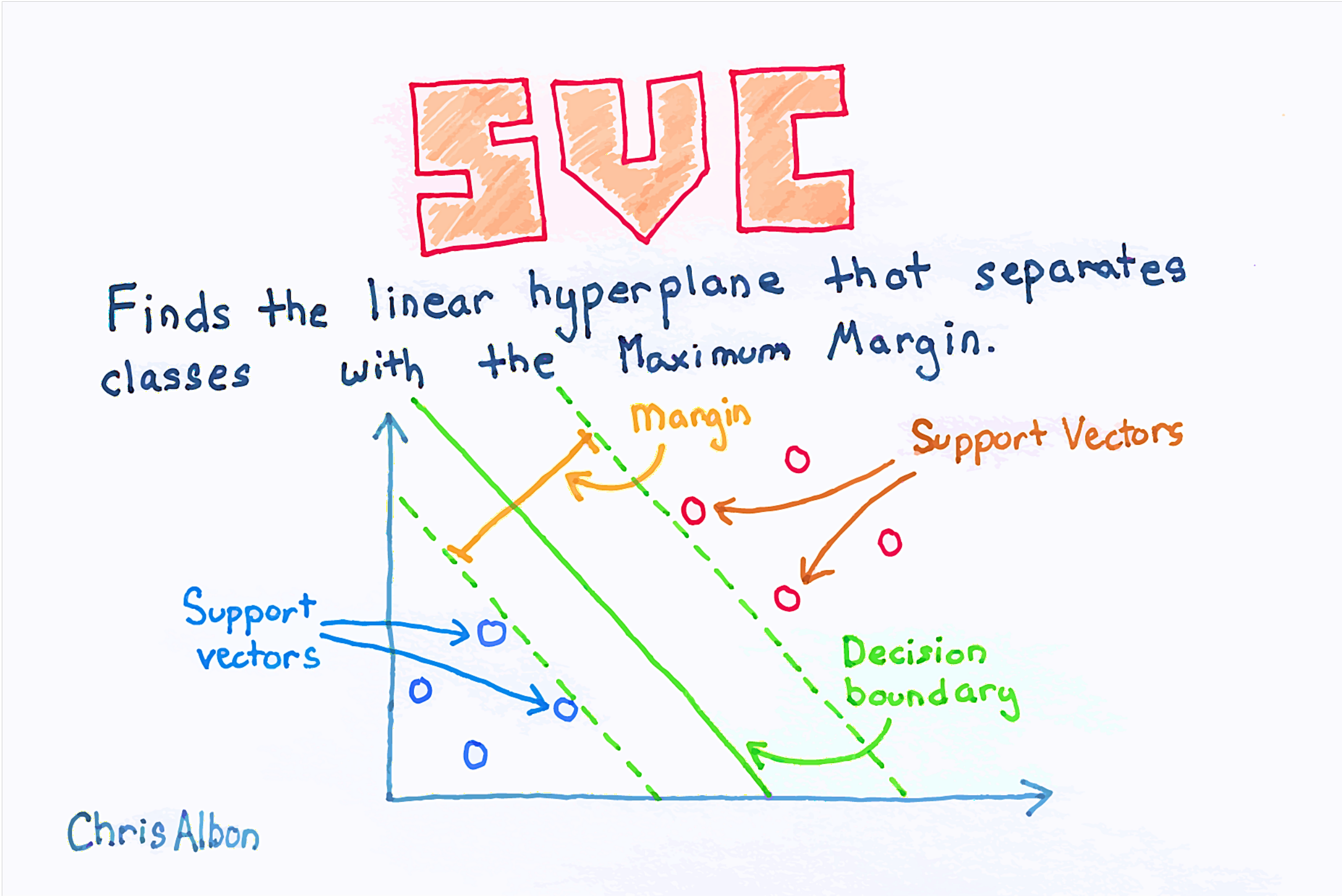
- Only Training data is loaded to Amazon S3 Buckets

Custom Scikit- learn Classifier in SageMaker

- With Amazon SageMaker, we can quickly and easily train our model and directly deploy them into production ready hosted environment
- It provides an integrated Jupyter authoring notebook instance for easy access without requiring to manage servers
- To run custom training script in SageMaker, an estimator was constructed with specific entry point and instance type

Support Vector Machines

- Once model is trained and deployed, we can create versions of different models and test in batches


SVC
Finds the linear hyperplane that separates classes with the Maximum Margin.
Margin
Support Vectors
Support vectors
Decision boundary
Chris Albon

Conclusion

- The model was successfully created with its endpoint being the actual function call that acts as a REST API
- The model was able to classify all the documents successfully with full accuracy
- Further scope – Use API Gateway and lambda function to deploy model to a web application