Data Engineer Bootcamp

Capstone Project 2

Presented by Alanoud Alsuwailem, Bador Alsharif, and Sami Alfifi

Agenda

Capstone Project 2

WeCloudData

Project Objectives

- Data will be ingested from 2 data sources and ingested into the Data Lake in Azure.
- In the Data Lake, after a couple of data transformation and Machine Learning Model running, we make the Machine Learning result back to the Data Lake.
- Then we will use Azure Synapse to connect to the Data Lake, and we will generate a report from the Synapse.

Data Overview

- Dataset is from StackOverflow, a famous online IT developer community.
- The dataset records daily online posts, including post types and users' information.

Project Steps

01 Azure Data Factory

02 Databricks

03 Machine Learning Training

04 NLP prediction

05 Data Visualization

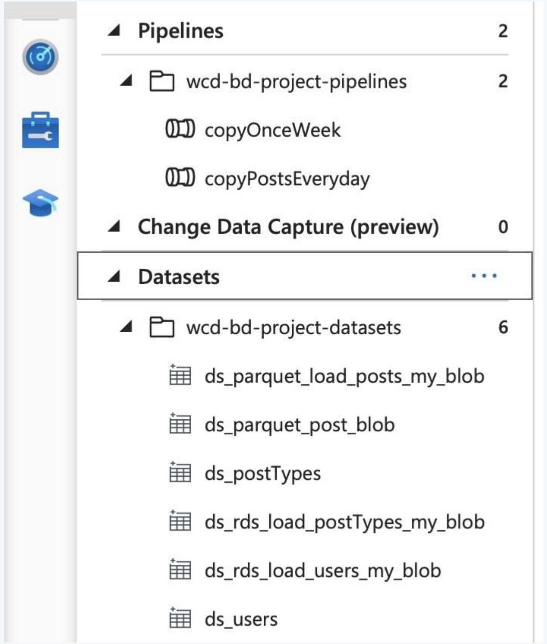
WeCloudData

Data Ingestion

WeCloudData

Azure Data Factory

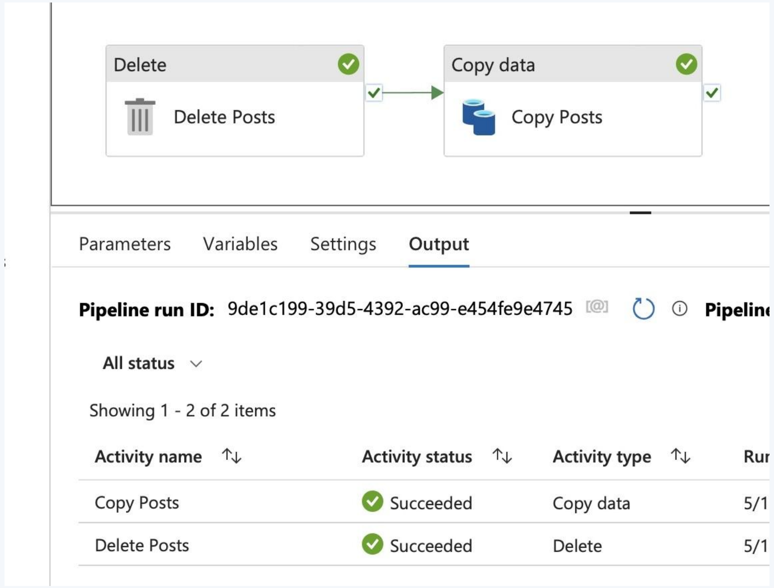
1. use Azure data factory to ingest data from 2 data sources, a postgres database on RDS and a WeCloudData's public Azure storage container.
2. build our data lake on Azure, and store the ingested data files in the data lake.



Pipelines 2
- wcd-bd-project-pipelines 2
  - copyOnceWeek
  - copyPostsEveryday

Change Data Capture (preview) 0

Datasets
- wcd-bd-project-datasets 6
  - ds_parquet_load_posts_my_blob
  - ds_parquet_post_blob
  - ds_postTypes
  - ds_rds_load_postTypes_my_blob
  - ds_rds_load_users_my_blob
  - ds_users



Pipeline run ID: 9de1c199-39d5-4392-ac99-e454fe9e4745

Showing 1 - 2 of 2 items

Activity name	Activity status	Activity type	Rur
Copy Posts	✓ Succeeded	Copy data	5/1
Delete Posts	✓ Succeeded	Delete	5/1

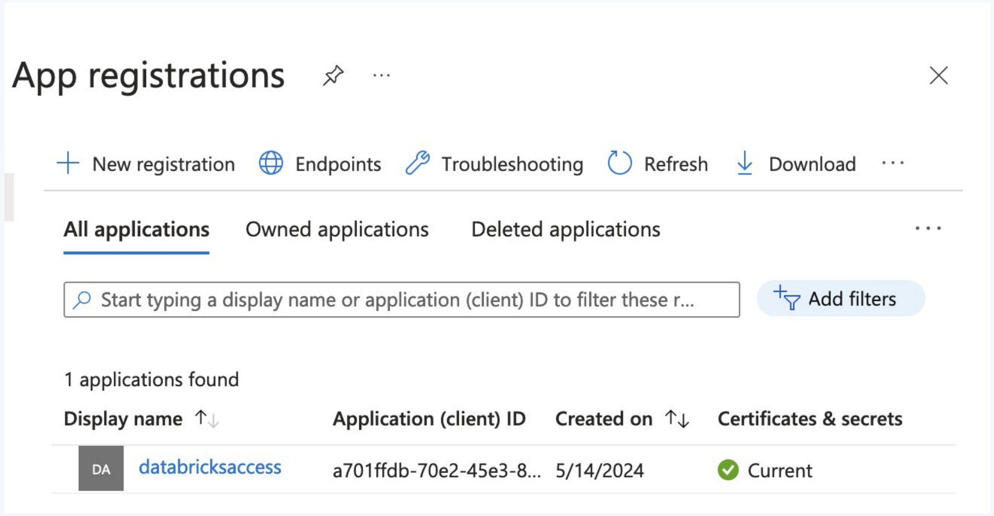
WeCloudData

Data Transformation

WeCloudData

Databricks

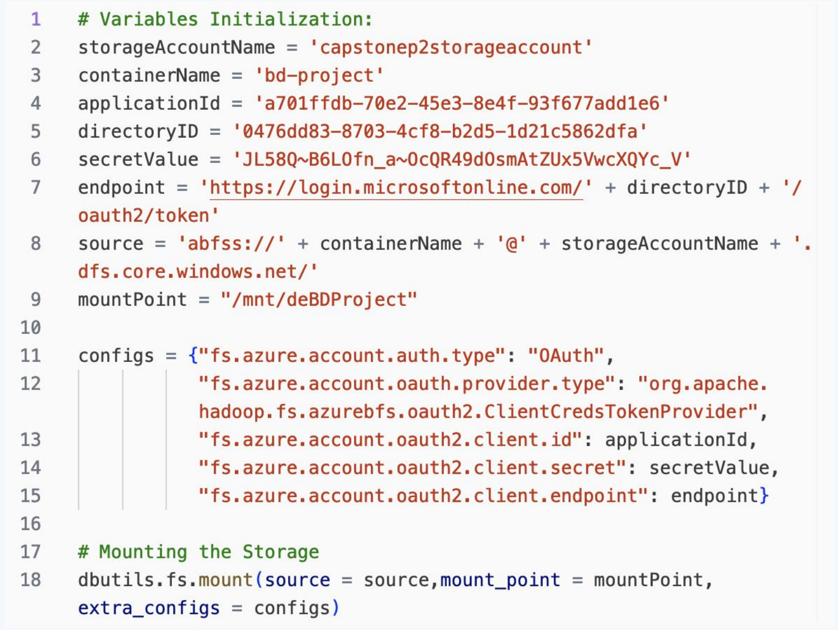
1. Create an Azure Databricks workspace, a compute cluster, and a notebook.
2. Allow the Azure Databricks to access the Storage container.
3. Mount Storage container to the Azure Databricks directory.



App registrations

1 applications found

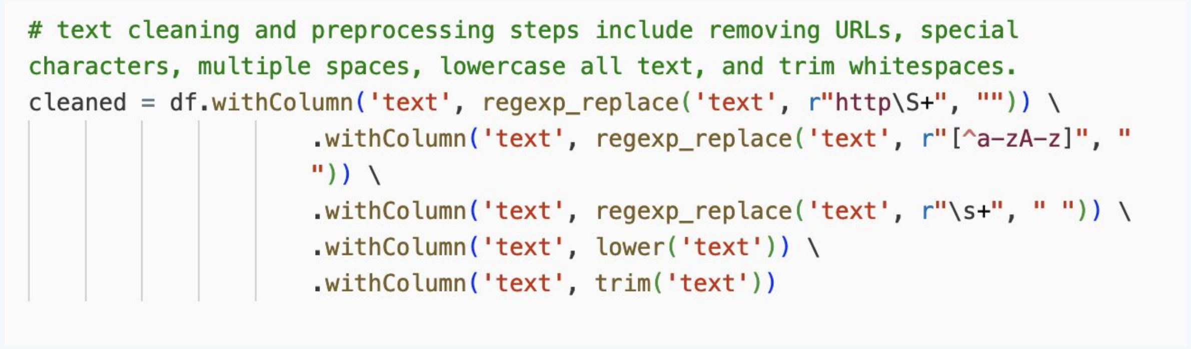
Display name	Application (client) ID	Created on	Certificates & secrets
DA databricksaccess	a701ffdb-70e2-45e3-8...	5/14/2024	✓ Current



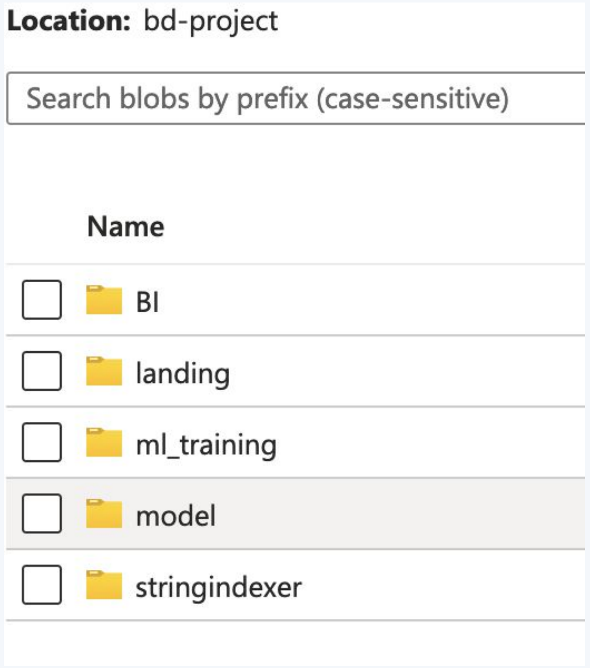
```python
# Variables Initialization:
storageAccountName = 'capstonep2storageaccount'
containerName = 'bd-project'
applicationId = 'a701ffdb-70e2-45e3-8e4f-93f677add1e6'
directoryID = '0476dd83-8703-4cf8-b2d5-1d21c5862dfa'
secretValue = 'JL58Q~B6LOfn_a~OcQR49dOsmAtZUx5VwcXQYc_V'
endpoint = 'https://login.microsoftonline.com/' + directoryID + '/oauth2/token'
source = 'abfss://' + containerName + '@' + storageAccountName + '.dfs.core.windows.net/'
mountPoint = "/mnt/deBDProject"

configs = {"fs.azure.account.auth.type": "OAuth",
           "fs.azure.account.oauth.provider.type": "org.apache.hadoop.fs.azurebfs.oauth2.ClientCredsTokenProvider",
           "fs.azure.account.oauth2.client.id": applicationId,
           "fs.azure.account.oauth2.client.secret": secretValue,
           "fs.azure.account.oauth2.client.endpoint": endpoint}

# Mounting the Storage
dbutils.fs.mount(source = source,mount_point = mountPoint, extra_configs = configs)
```

ML Model Training

1. Join the Posts and posttypes tables
2. Prepare the training data for the machine learning training.
3. Train the machine learning model.
4. Save the model to a Azure storage folder.



```
# text cleaning and preprocessing steps include removing URLs, special
characters, multiple spaces, lowercase all text, and trim whitespaces.
cleaned = df.withColumn('text', regexp_replace('text', r"http\S+", "")) \
                    .withColumn('text', regexp_replace('text', r"[^a-zA-z]", "
                    ")) \
                    .withColumn('text', regexp_replace('text', r"\s+", " ")) \
                    .withColumn('text', lower('text')) \
                    .withColumn('text', trim('text'))
```



Location: bd-project

Search blobs by prefix (case-sensitive)

Name
BI
landing
ml_training
model
stringindexer

WeCloud**Data**

ML Model Training Steps

01 Feature Transformation

02 Label Encoding

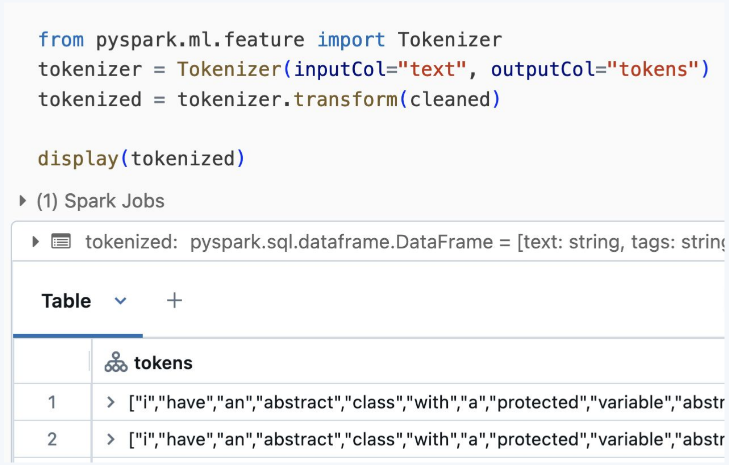
03 Model Training

04 Model Evaluation

WeCloudData

1. Feature Transformation

Tokenize the text, remove stopwords, and perform term frequency (TF), and inverse document frequency (IDF) vectorization.

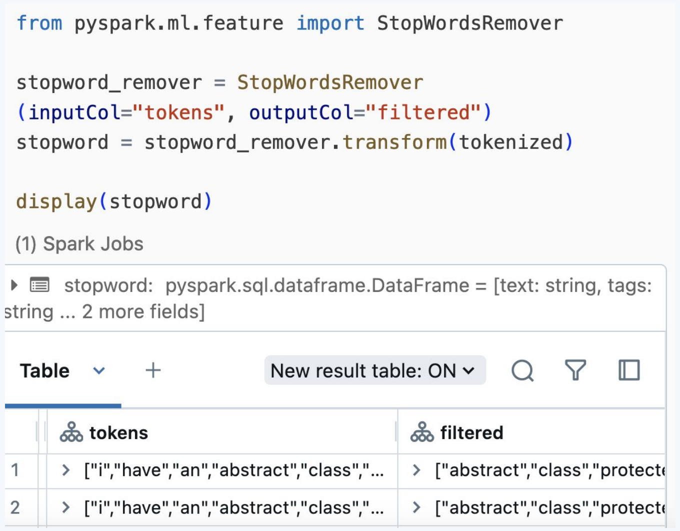


```python
from pyspark.ml.feature import Tokenizer
tokenizer = Tokenizer(inputCol="text", outputCol="tokens")
tokenized = tokenizer.transform(cleaned)

display(tokenized)
```

▸ (1) Spark Jobs

▸ tokenized: pyspark.sql.dataframe.DataFrame = [text: string, tags: strin

	tokens
1	› ["i","have","an","abstract","class","with","a","protected","variable","abstr
2	› ["i","have","an","abstract","class","with","a","protected","variable","abstr

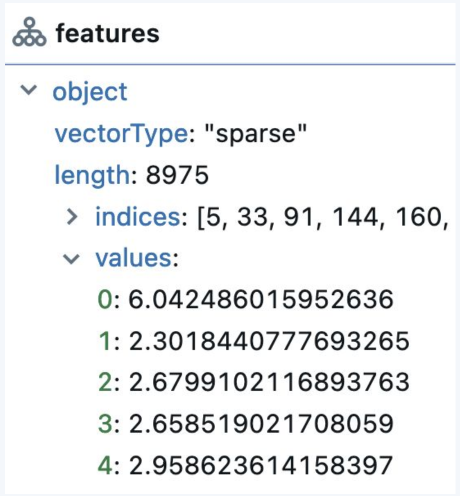


```python
from pyspark.ml.feature import StopWordsRemover

stopword_remover = StopWordsRemover
(inputCol="tokens", outputCol="filtered")
stopword = stopword_remover.transform(tokenized)

display(stopword)
```

(1) Spark Jobs

▸ stopword: pyspark.sql.dataframe.DataFrame = [text: string, tags: string ... 2 more fields]

	tokens	filtered
1	› ["i","have","an","abstract","class","...	› ["abstract","class","protecte
2	› ["i","have","an","abstract","class","...	› ["abstract","class","protecte



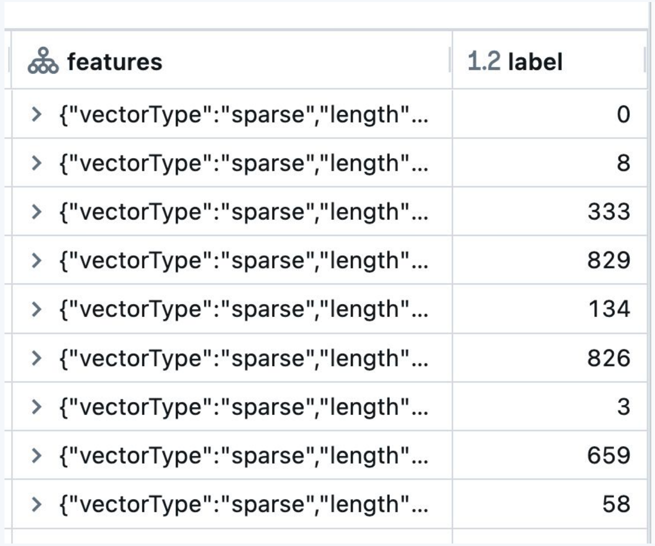
```
features
⌄ object
  vectorType: "sparse"
  length: 8975
  › indices: [5, 33, 91, 144, 160,
  ⌄ values:
      0: 6.042486015952636
      1: 2.3018440777693265
      2: 2.6799102116893763
      3: 2.658519021708059
      4: 2.958623614158397
```

WeCloudData

2. Label Encoding

Simplifies the handling of categorical data by assigning unique numbers to each category. By encoding categories into numerical values, StringIndexer enables easier analysis and processing of data in PySpark.

```python
from pyspark.ml.feature import StringIndexer

label_encoder = StringIndexer(inputCol = "tags", outputCol = "label")
le_model = label_encoder.fit(text_idf)
final = le_model.transform(text_idf)

display(final)
```

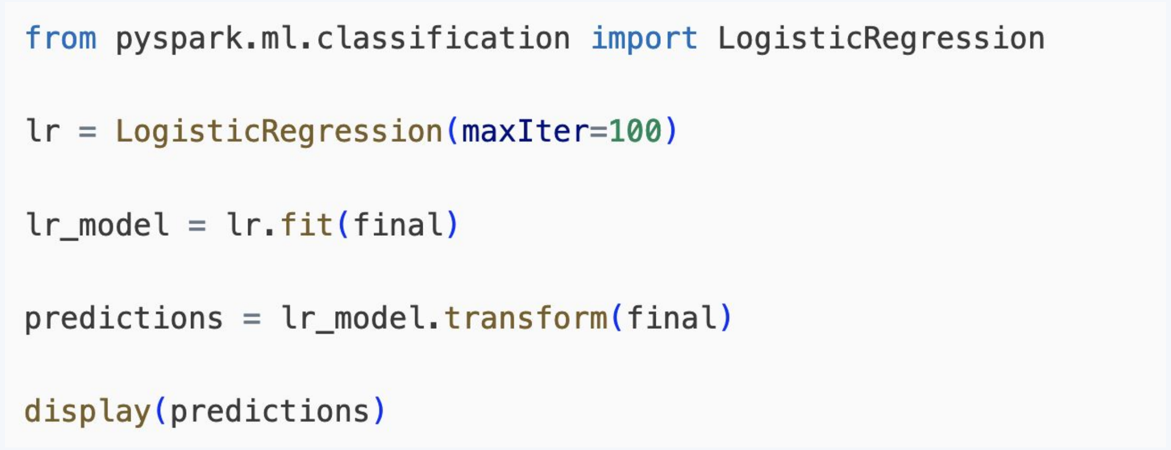


features	1.2 label
> {"vectorType":"sparse","length"...	0
> {"vectorType":"sparse","length"...	8
> {"vectorType":"sparse","length"...	333
> {"vectorType":"sparse","length"...	829
> {"vectorType":"sparse","length"...	134
> {"vectorType":"sparse","length"...	826
> {"vectorType":"sparse","length"...	3
> {"vectorType":"sparse","length"...	659
> {"vectorType":"sparse","length"...	58

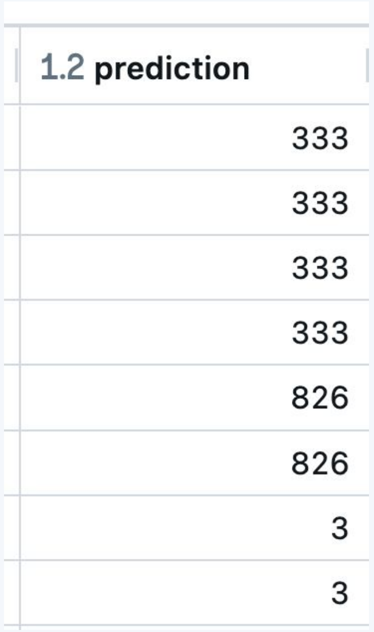
WeCloudData

3. Model Training

Train a Logistic Regression model using the preprocessed features and encoded labels to make predictions. The model learns patterns from the input features and their corresponding labels during training.



```python
from pyspark.ml.classification import LogisticRegression

lr = LogisticRegression(maxIter=100)

lr_model = lr.fit(final)

predictions = lr_model.transform(final)

display(predictions)
```

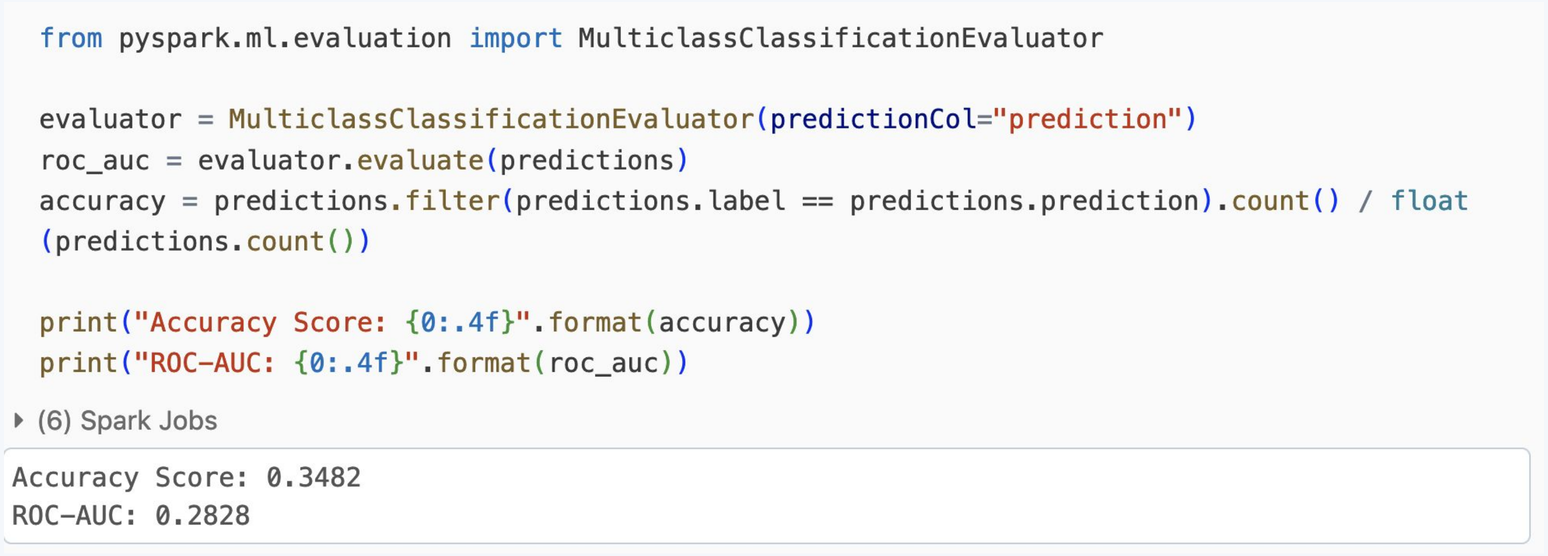


1.2 prediction
333
333
333
333
826
826
3
3

WeCloudData

4. Model Evaluation

calculate the accuracy of the model. The accuracy is computed by comparing the predicted labels with the actual labels in the DataFrame predictions. We count the number of correct predictions and divide it by the total number of predictions to obtain the accuracy score.



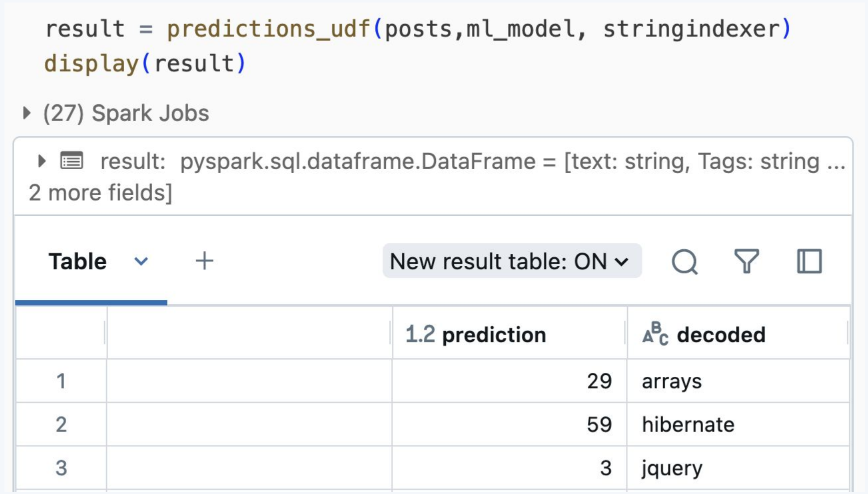
```
from pyspark.ml.evaluation import MulticlassClassificationEvaluator

evaluator = MulticlassClassificationEvaluator(predictionCol="prediction")
roc_auc = evaluator.evaluate(predictions)
accuracy = predictions.filter(predictions.label == predictions.prediction).count() / float
(predictions.count())

print("Accuracy Score: {0:.4f}".format(accuracy))
print("ROC-AUC: {0:.4f}".format(roc_auc))
```

```
Accuracy Score: 0.3482
ROC-AUC: 0.2828
```

WeCloudData

NLP prediction

1. Utilize the trained model to generate predictions on unseen data.
2. Calculate the topics predicted by the model to summarize the quantity of each topic.
3. Save the summary table to the Azure storage folder which will be used for BI.
4. Add the Databricks activity into the everyday running pipeline.



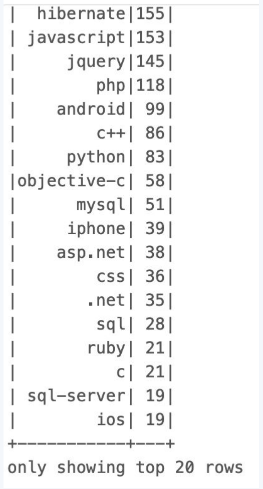
```
result = predictions_udf(posts,ml_model, stringindexer)
display(result)
```

▸ (27) Spark Jobs

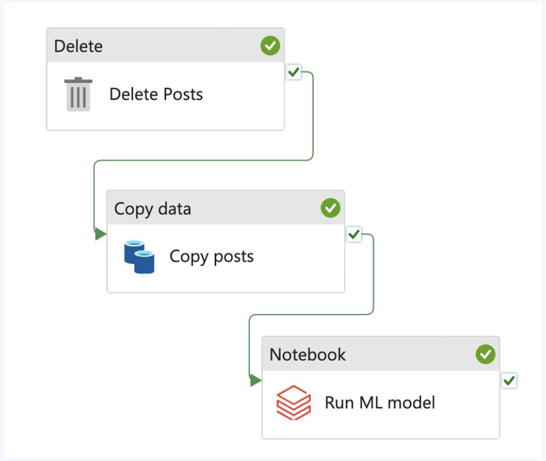
▸ result: pyspark.sql.dataframe.DataFrame = [text: string, Tags: string ... 2 more fields]

Table | New result table: ON

		1.2 prediction	decoded
1		29	arrays
2		59	hibernate
3		3	jquery



```
|   hibernate|155|
|  javascript|153|
|      jquery|145|
|         php|118|
|     android| 99|
|         c++| 86|
|      python| 83|
|objective-c| 58|
|       mysql| 51|
|      iphone| 39|
|     asp.net| 38|
|         css| 36|
|        .net| 35|
|         sql| 28|
|        ruby| 21|
|           c| 21|
|  sql-server| 19|
|         ios| 19|
+------------+---+
only showing top 20 rows
```

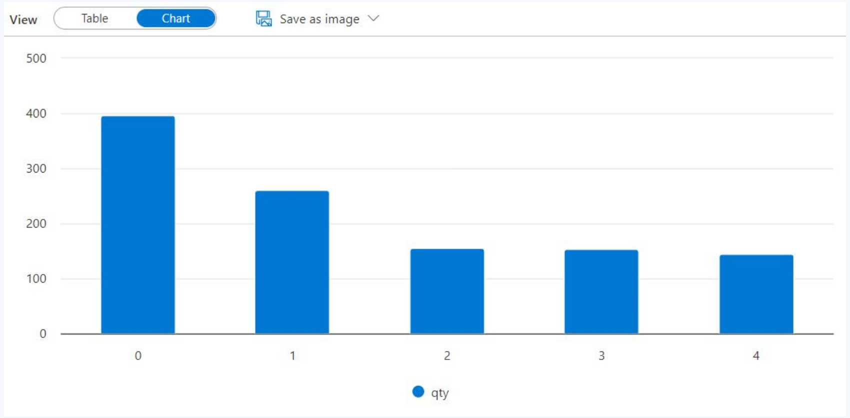


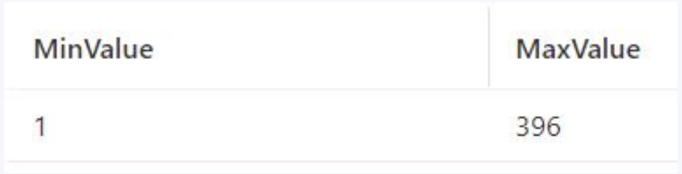
Data Visualization

WeCloudData

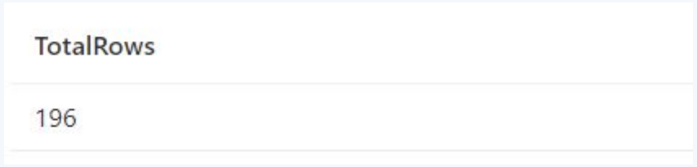
Synapse

1. Creating Synapse Workspace
2. Writing T-SQL Queries





MinValue	MaxValue
1	396

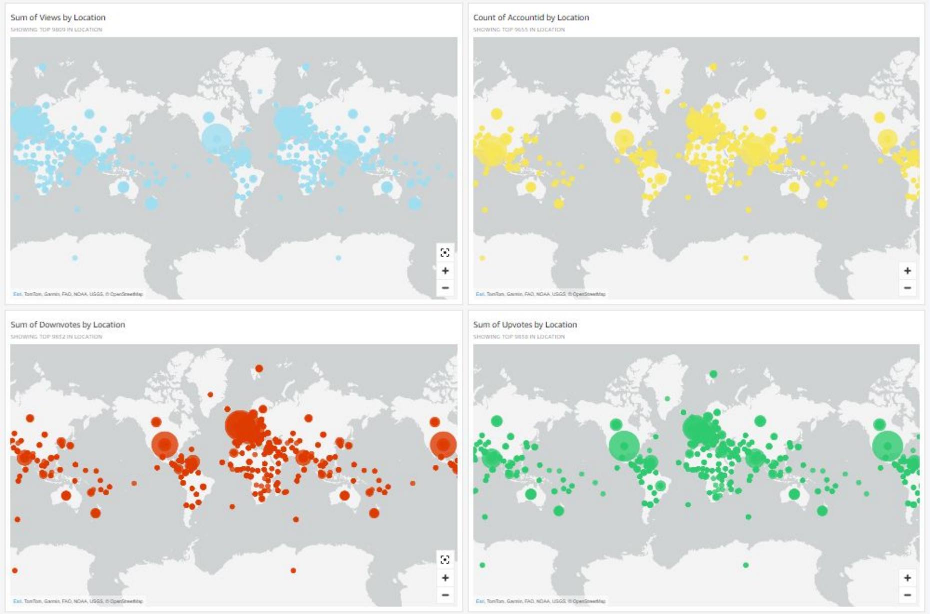


TotalRows
196

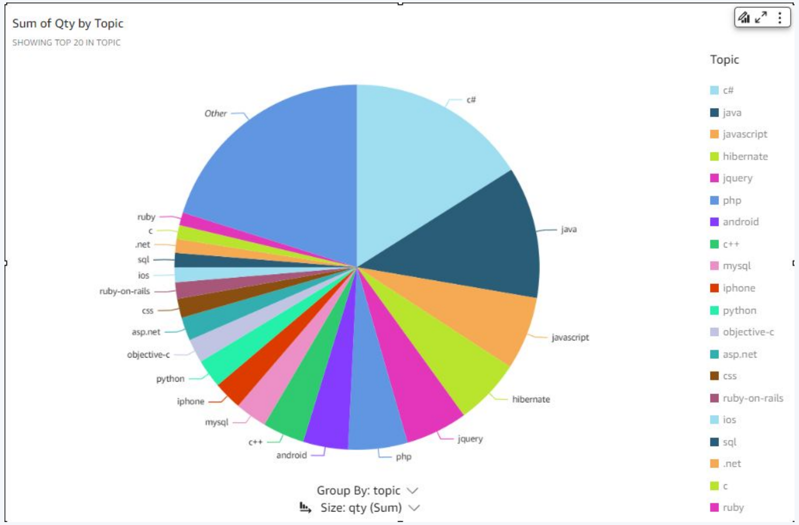
WeCloud**Data**

AWS Quicksight

1. Loading Data to S3.
2. Creating a dataset connected to S3 bucket.
3. Creating dashboards.



Sum of Views by Location
Count of Accountid by Location
Sum of Downvotes by Location
Sum of Upvotes by Location



WeCloudData

Future

1. Enhanced Data Quality Measures:

   Implementing robust data cleansing and validation techniques to ensure high data quality and accuracy.

2. Advanced Analytics Capabilities:

   Incorporating advanced analytics techniques such as predictive modeling or machine learning to derive deeper insights from the data.

3. Automation Enhancements:

   Implementing additional security measures to protect sensitive data throughout the entire data lifecycle.

Thank you!

WeCloud**Data**