Machine Learning Pipeline with PySpark

Date : 09-08-2020 | Speaker : Ayon Roy | Event : Kaggle Days Meetup Surat

Visit - AYONROY.ML

Hello **Buddy!**

I am **Ayon Roy**

B.Tech CSE (2017-2021)

Data Science Intern @ **Lulu International Exchange**, Abu Dhabi

(**World's Leading Financial Services Company**)

Brought **Kaggle Days Meetup** Community in India for the 1st time

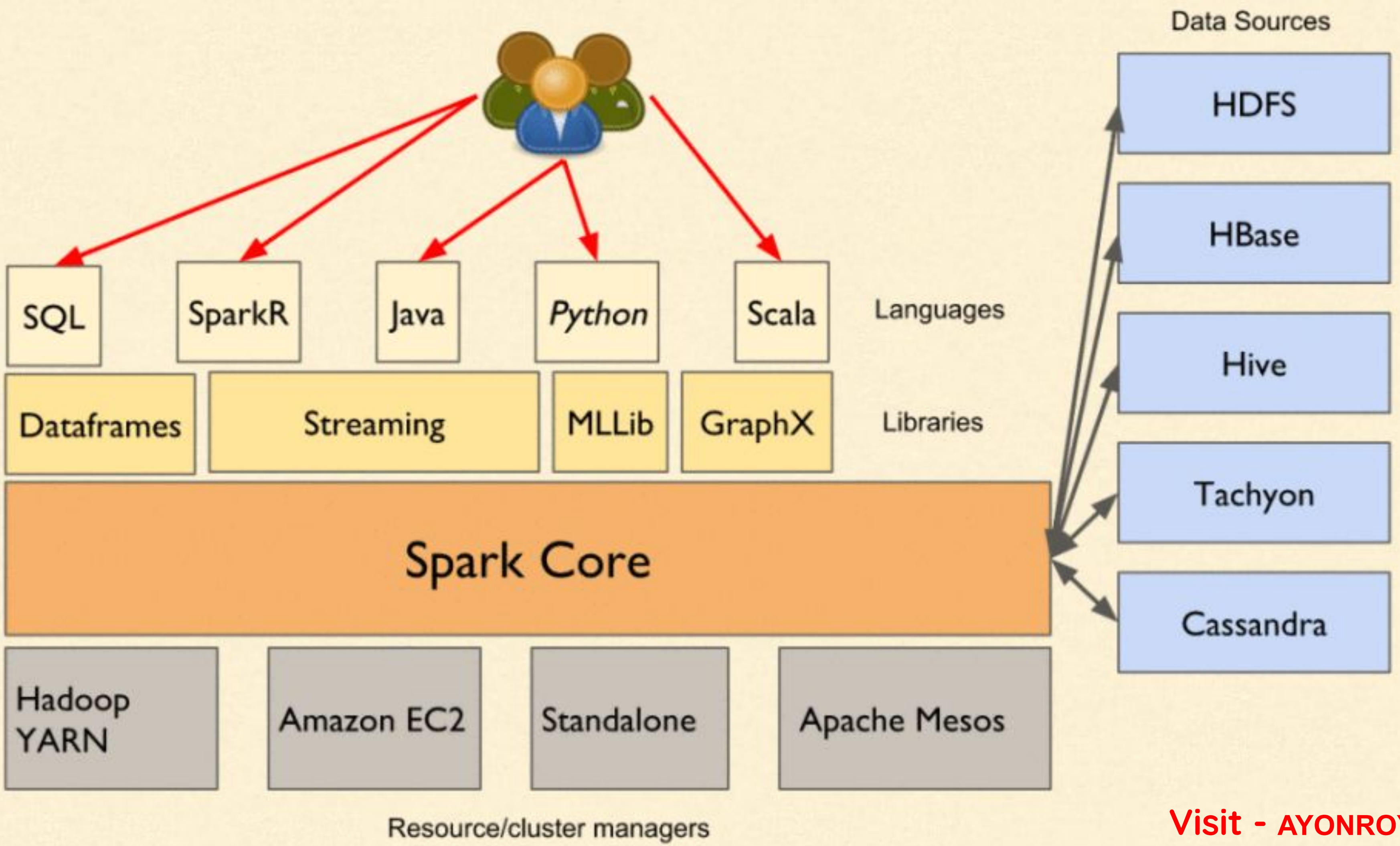
Visit - AYONROY.ML

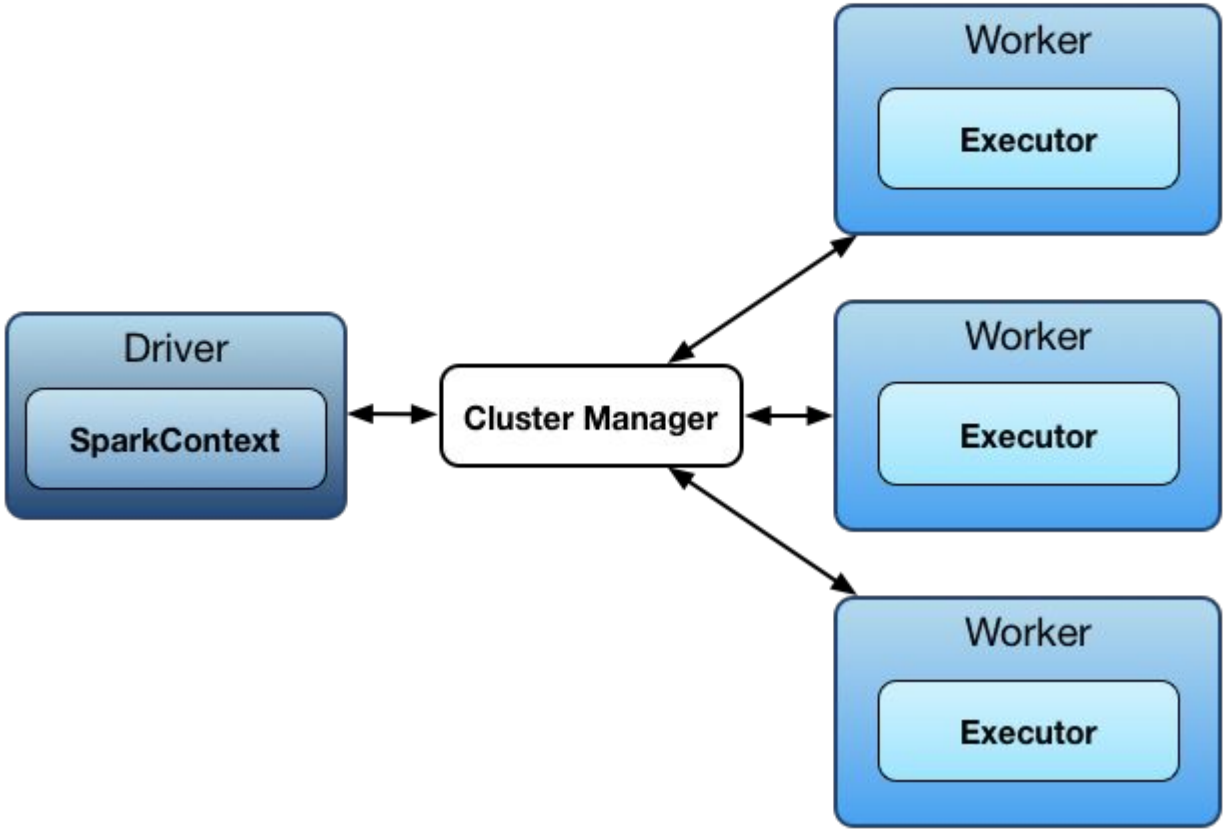
Agenda (09-08-2020)

- How Spark's architecture will help us in doing ML ?
- How to make a ML Pipeline using the already existing functionalities in PySpark?
- How to make a custom ML Pipeline by building own functionalities in PySpark ?

How Spark's Architecture will help us in doing Machine Learning ?

Visit - AYONROY.ML


Data Sources
HDFS
HBase
Hive
Tachyon
Cassandra
SQL
SparkR
Java
Python
Scala
Languages
Dataframes
Streaming
MLLib
GraphX
Libraries
Spark Core
Hadoop YARN
Amazon EC2
Standalone
Apache Mesos
Resource/cluster managers


Driver
SparkContext
Cluster Manager
Worker
Executor
Worker
Executor
Worker
Executor

- **Spark Context:** It holds a connection with Spark cluster manager. All Spark applications run as independent set of processes, coordinated by a SparkContext in a program.

- **Driver :** A driver is incharge of the process of running the main() function of an application and creating the SparkContext.

- **Executor** : Executors are worker nodes' processes in charge of running individual tasks in a given Spark job. They are launched at the beginning of a Spark application and typically run for the entire lifetime of an application.

- **Worker** : A worker, on the other hand, is any node that can run program in the cluster. If a process is launched for an application, then this application acquires executors at worker node.

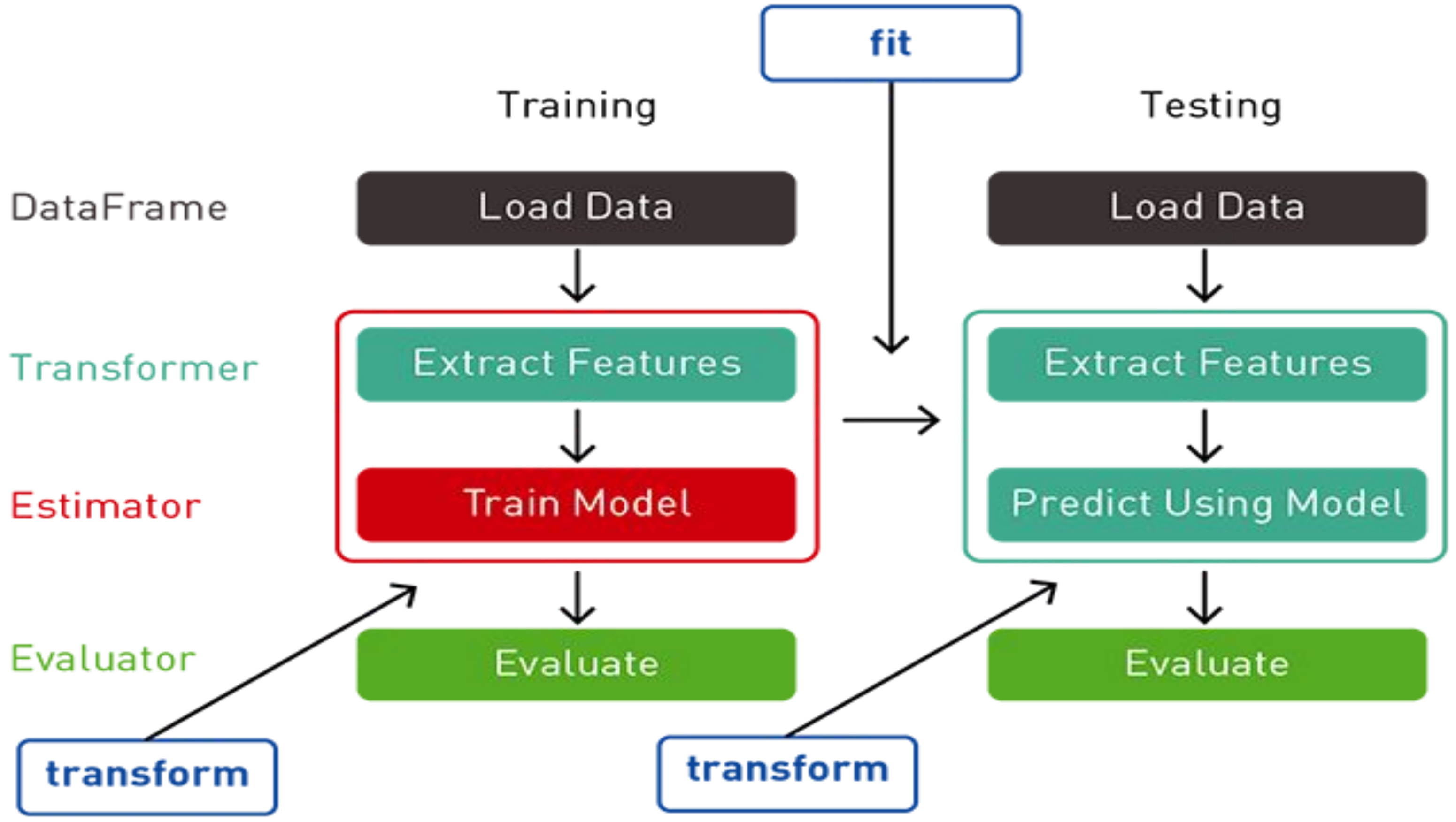
- **Cluster Manager:** Cluster manager allocates resources to each application in driver program. There are three types of cluster managers supported by Apache Spark – Standalone, Mesos and YARN.

How to make a Machine Learning Pipeline using PySpark ?

(With Existing Functionalities)

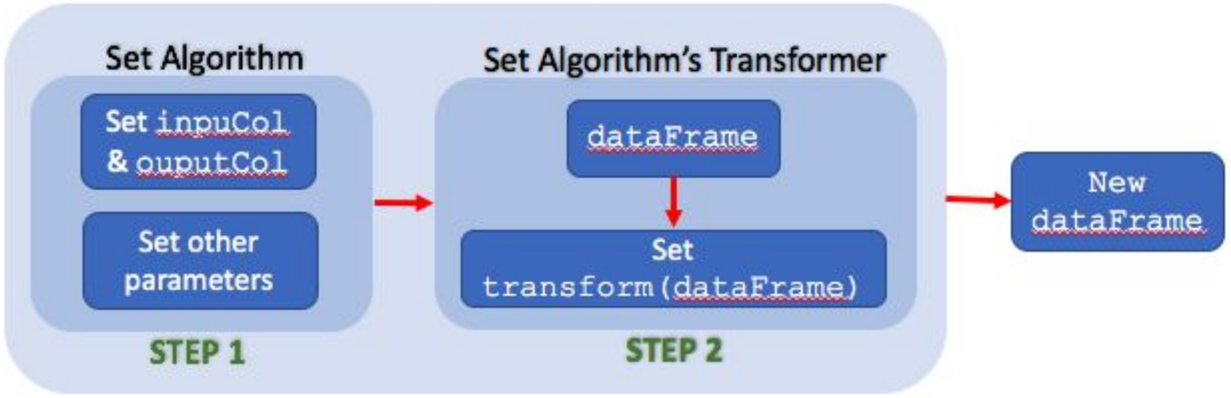
Visit - AYONROY.ML

Spark ML Workflow

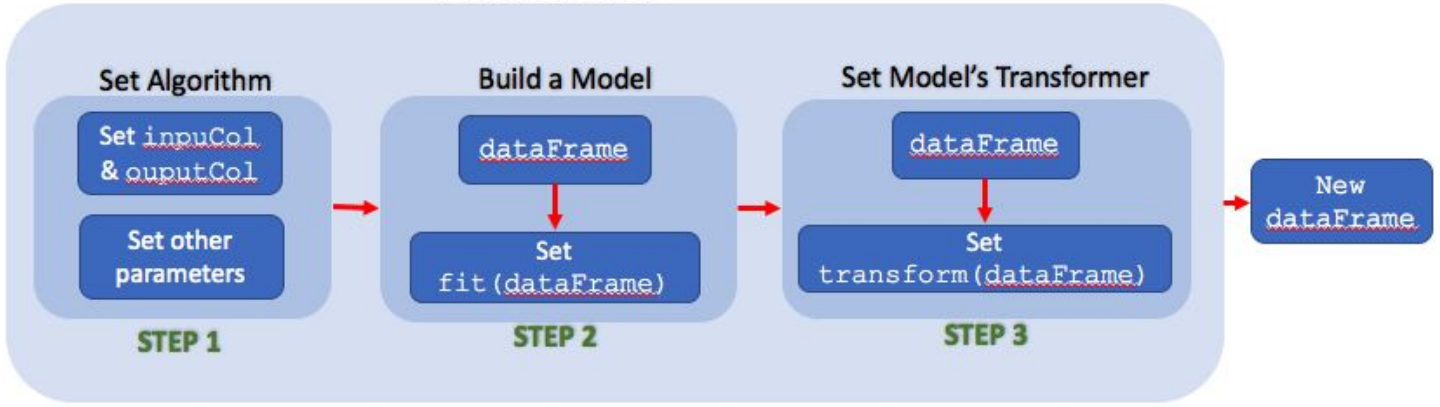


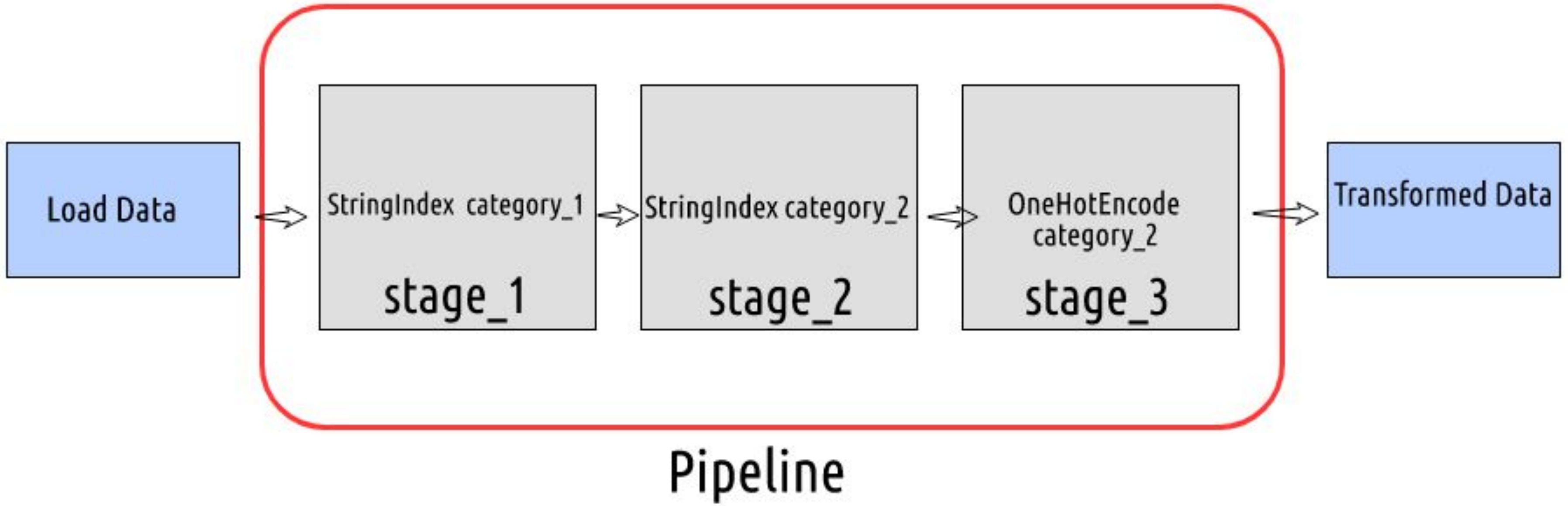
- **DataFrame**: This ML API uses DataFrame from Spark SQL as an ML dataset, which can hold a variety of data types. E.g., a DataFrame could have different columns storing text, feature vectors, true labels, and predictions.
- **Transformer**: A Transformer is an algorithm which can transform one DataFrame into another DataFrame. E.g., an ML model is a Transformer which transforms a DataFrame with features into a DataFrame with predictions.
- **Estimator**: An Estimator is an algorithm which can be fit on a DataFrame to produce a Transformer. E.g., a learning algorithm is an Estimator which trains on a DataFrame and produces a model.
- **Pipeline**: A Pipeline chains multiple Transformers and Estimators together to specify an ML workflow.

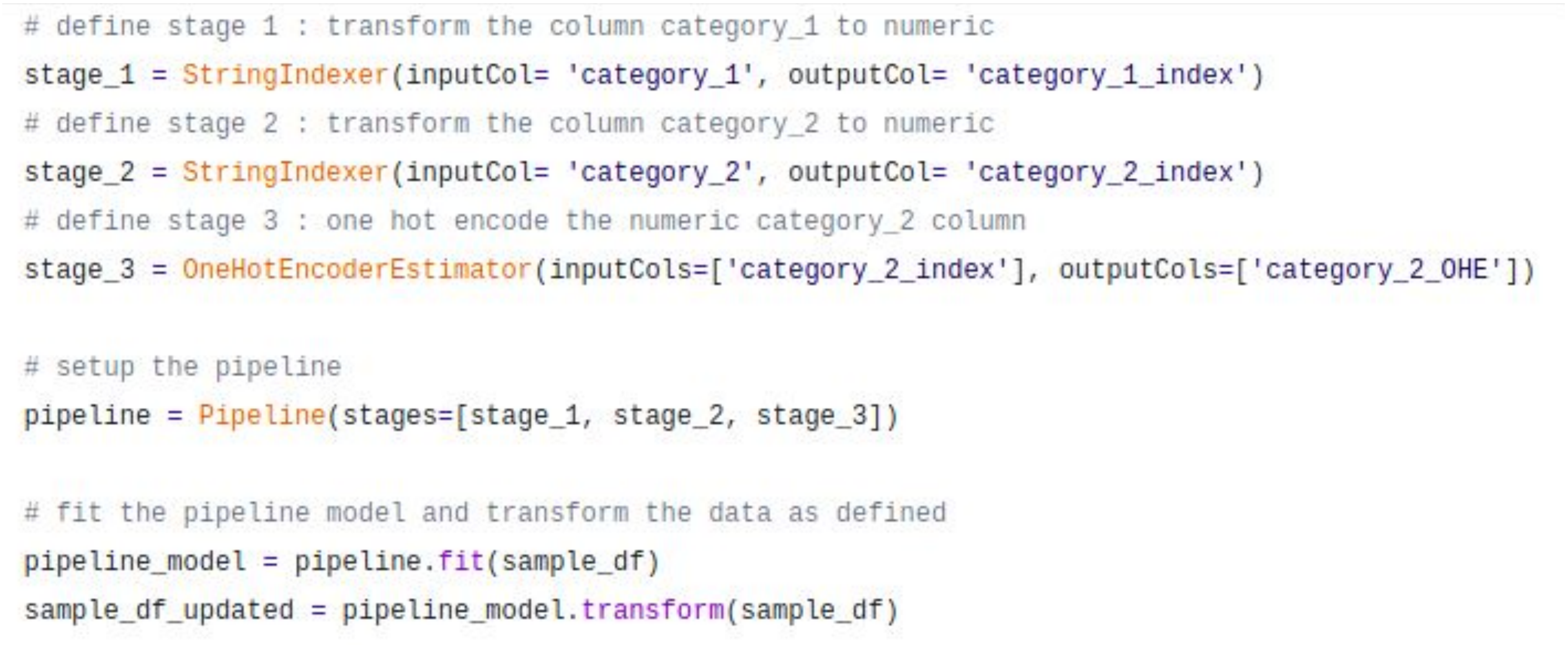
Transformer



Estimator




Load Data
StringIndex category_1
stage_1
StringIndex category_2
stage_2
OneHotEncode category_2
stage_3
Transformed Data
Pipeline



```python
# define stage 1 : transform the column category_1 to numeric
stage_1 = StringIndexer(inputCol= 'category_1', outputCol= 'category_1_index')
# define stage 2 : transform the column category_2 to numeric
stage_2 = StringIndexer(inputCol= 'category_2', outputCol= 'category_2_index')
# define stage 3 : one hot encode the numeric category_2 column
stage_3 = OneHotEncoderEstimator(inputCols=['category_2_index'], outputCols=['category_2_OHE'])

# setup the pipeline
pipeline = Pipeline(stages=[stage_1, stage_2, stage_3])

# fit the pipeline model and transform the data as defined
pipeline_model = pipeline.fit(sample_df)
sample_df_updated = pipeline_model.transform(sample_df)
```

How to make a Machine Learning Pipeline using PySpark ?

(With Custom Made Functionalities)

Visit - AYONROY.ML

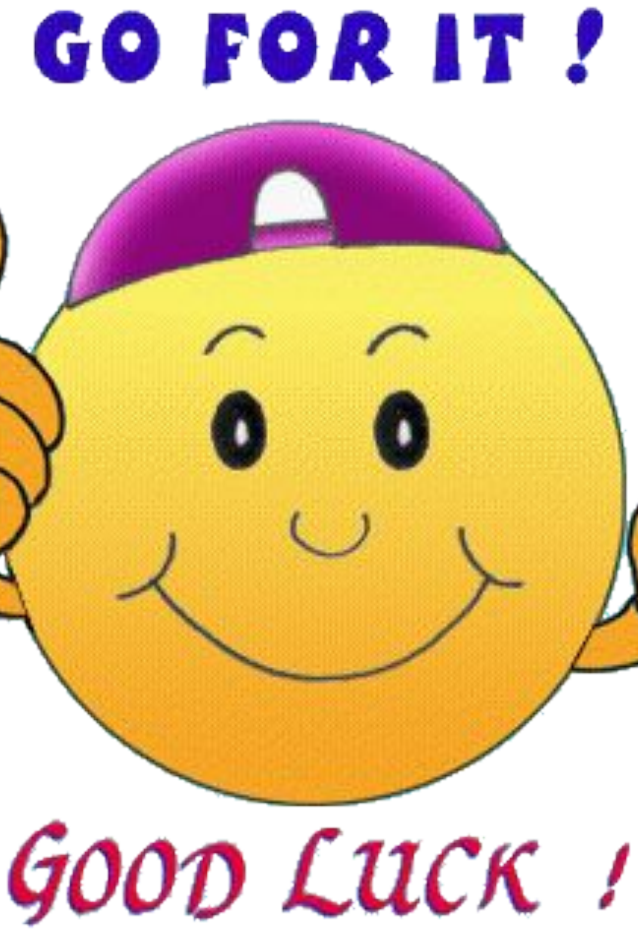
The basic rules to follow are that a `Transformer` needs to:

1. implement the `transform` method
2. specify an `inputCol` and `outputCol`
3. accept a `DataFrame` as input and return a `DataFrame` as output

```
from pyspark.ml.util import keyword_only
from pyspark.ml.pipeline import Transformer
from pyspark.ml.param.shared import HasInputCol, HasOutputCol


# Create a custom word count transformer class
class MyWordCounter(Transformer, HasInputCol, HasOutputCol):
    @keyword_only
    def __init__(self, inputCol=None, outputCol=None):
        super(WordCounter, self).__init__()
        kwargs = self.__init__._input_kwargs
        self.setParams(**kwargs)


    @keyword_only
    def setParams(self, inputCol=None, outputCol=None):
        kwargs = self.setParams._input_kwargs
        return self._set(**kwargs)


    def _transform(self, dataset):
        out_col = self.getOutputCol()
        in_col = dataset[self.getInputCol()]


    # Define transformer
    logic def f(s):
        return len(s.split(' '))
    t = LongType()
    return dataset.withColumn(out_col, udf(f, t)(in_col))


# Instantiate the new word count transformer
wc = MyWordCounter(inputCol="review", outputCol="wc")
```

A few useful resources

1. https://spark.apache.org/
2. https://www.analyticsvidhya.com/blog/2019/11/build-machine-learning-pipelines-pyspark/
3. https://pysparktutorial.blogspot.com/2018/02/transformer-vs-estimator.html
4. https://www.semicolonworld.com/question/55650/create-a-custom-transformer-in-pyspark-ml
5. https://danvatterott.com/blog/2019/07/12/limiting-cardinality-with-a-pyspark-custom-transformer/
6. https://blog.insightdatascience.com/spark-pipelines-elegant-yet-powerful-7be93afcdd42



Visit - AYONROY.ML

Let me answer your Questions now.

Finally, it's your time to speak.

Danke Scheon

Questions ? Any Feedbacks ? Did you like the talk?
Tell me about it.

If you think I can help you,
connect with me via

Email : ayonroy2000@gmail.com

LinkedIn / Github / Telegram Username : ayonroy2000

Website : https://AYONROY.ML/