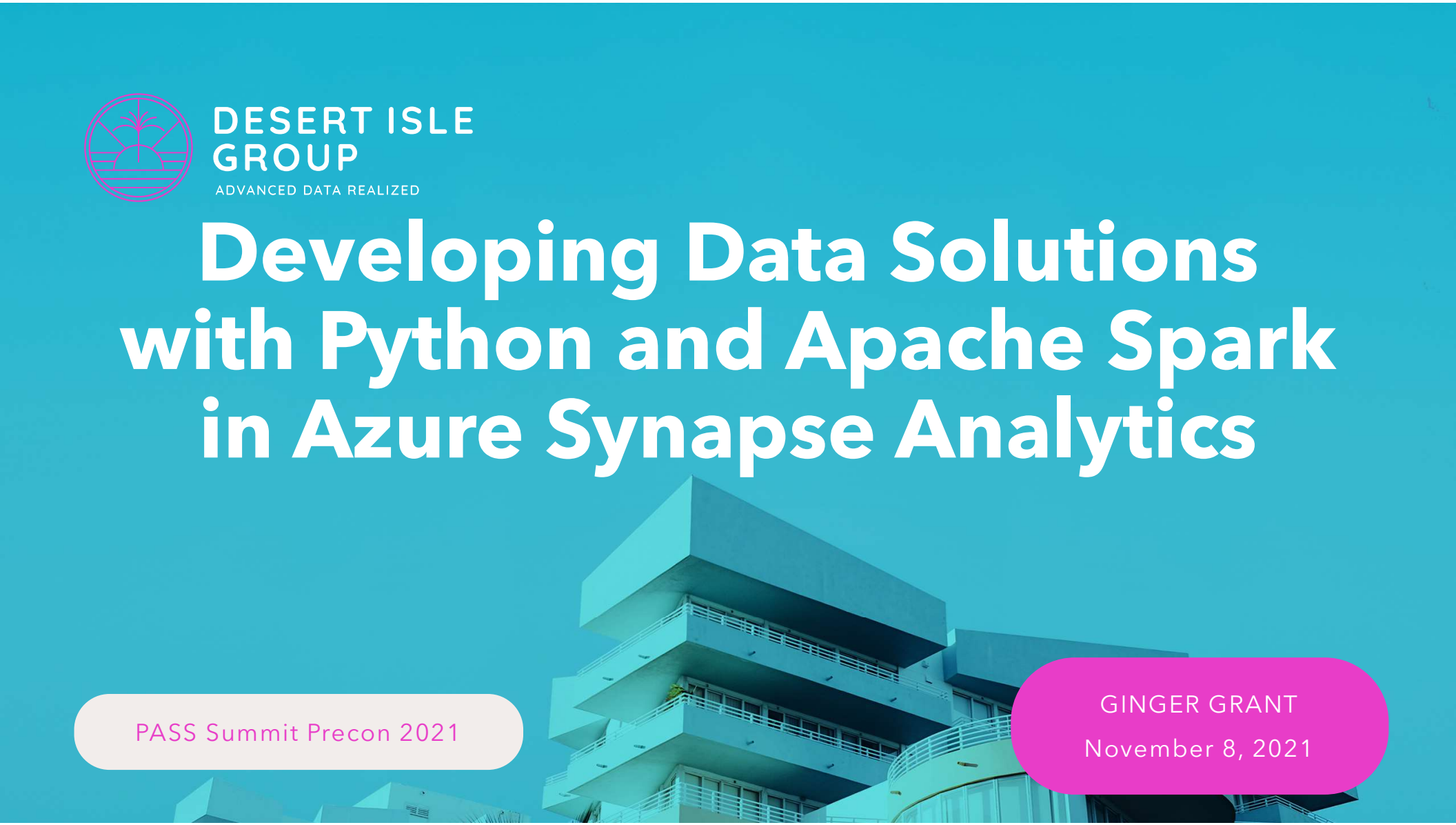
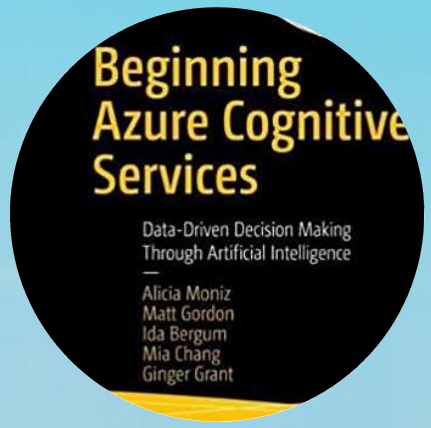

DESERT ISLE
GROUP
ADVANCED DATA REALIZED
Developing Data Solutions with Python and Apache Spark in Azure Synapse Analytics
PASS Summit Precon 2021
GINGER GRANT
November 8, 2021

About Me:
Ginger Grant



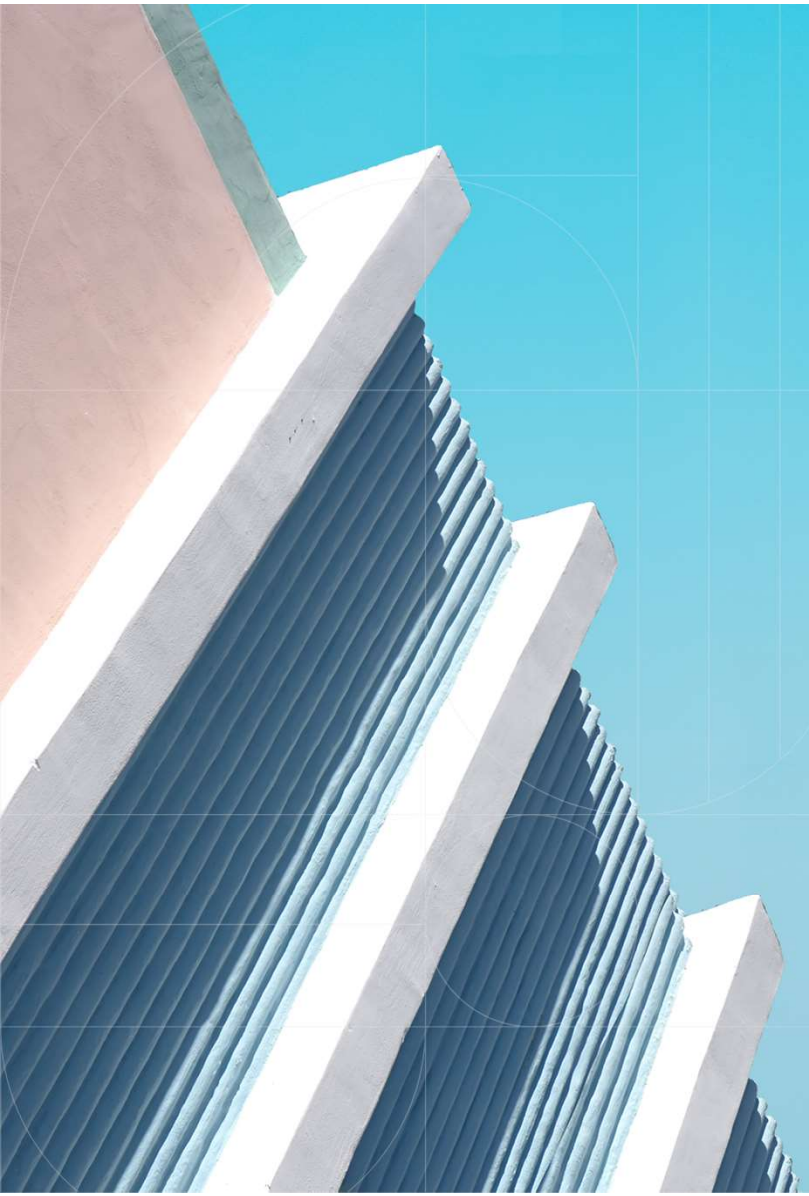
ginger.grant@desertislesql.com

desertislesql.com

desertislesql

Agenda

- Introduction to Python
- Introduction to Spark
- Using Spark for Data Exploration and Analysis
- Developing ETL Components with Python
- Organizing your data with Data Engineering
- Machine Learning Components with Spark ML Lib



Introduction to Python

Environments and Configuration

Introduction to Python

Released in 1994 by Guido Van Rossum

The current Python version is version 3

The Language was not named after a snake

Introduction to Python

Released in 1994 by Guido Van Rossum

The current Python version is version 3

The Language is named after Monty Python

Why Learn Python?

"Python is the most popular programming language in 2021."

Visualmodo, July 2021

"Learn Python. That's the biggest takeaway we can give you from its continued dominance of IEEE Spectrum's annual interactive rankings of the top programming languages." ieee, Aug 24, 2021

"Python overtakes R, becomes the leader in Data Science, Machine Learning platforms" KDNuggets Aug 2017

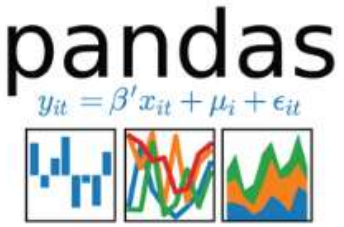
Python Add-ons

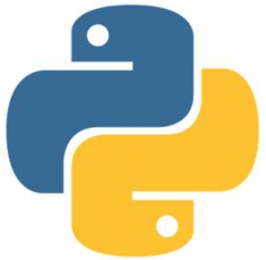
SciKit-Learn

Pandas

Spark

Anaconda



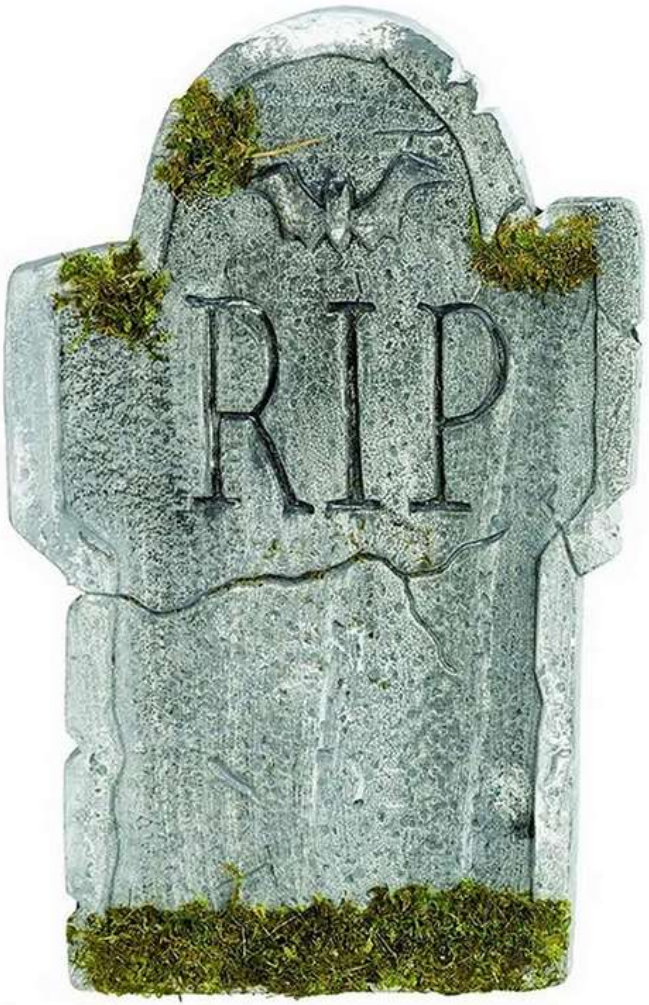


Azure Notebooks

Microsoft got rid of this application in October 2020

Embedded notebook functionality other places

You will see a page to tell you how to use other tools



Python Environment

- Jupyter
- Cloud Environments – Virtual Machines, Synapse, Azure ML, open source sites
- Local Environment

What is a Jupyter Notebook?

Originally known as iPython

Web Application

Open Source

Expanded to include an number of coding languages

Kernels

- **iPython**
- **iRKernel**
- **iRuby**
- **iGo**
- **Bash**
- **iJulia**
- **iJava**

Installing

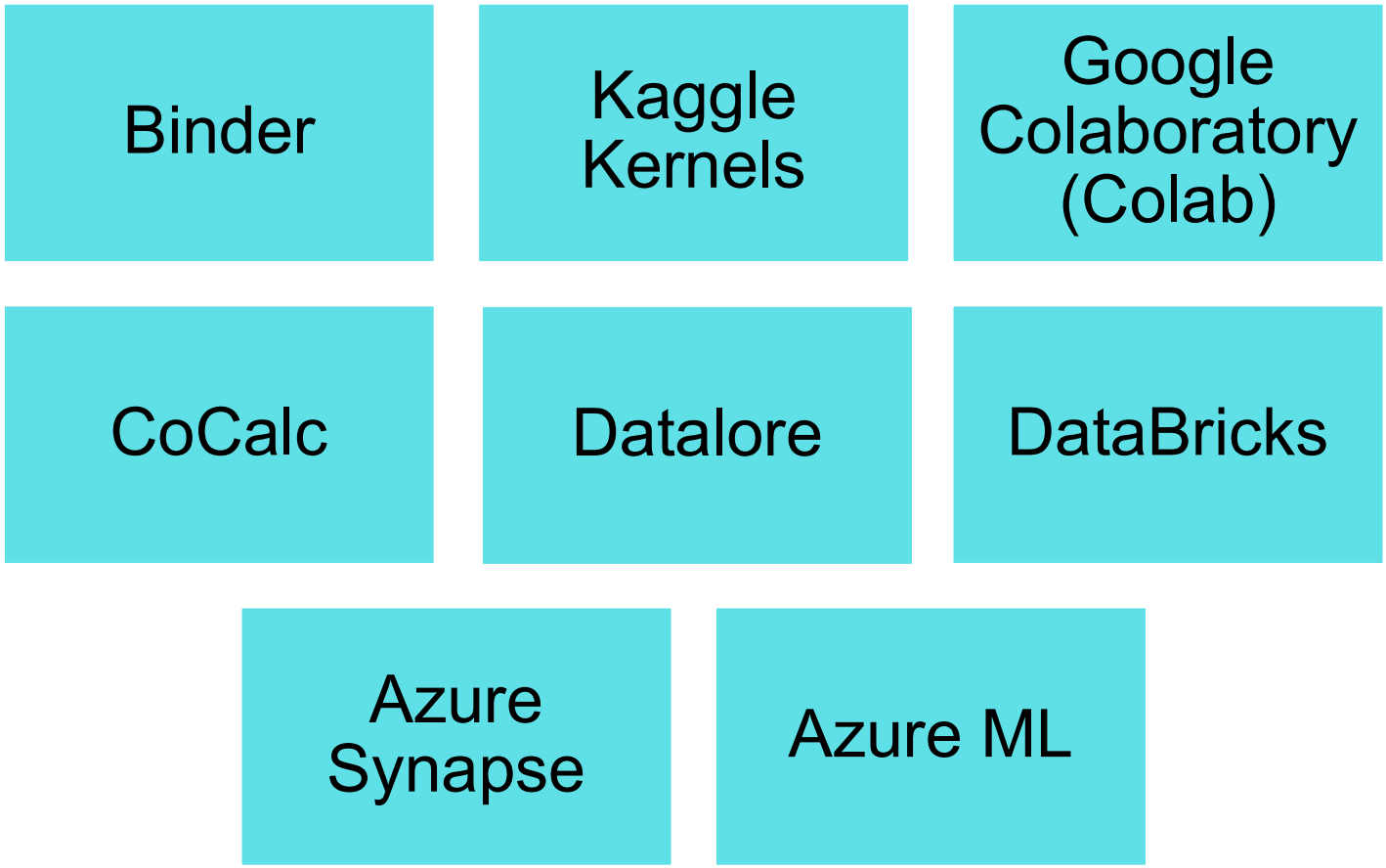
Must have Python installed Anaconda Distribution

installation is strongly recommended

Use pip with Python

```
python3 -m pip install jupyter
```

Cloud Jupyter Notebooks



Anaconda

Local install

Provides you with your own Jupyter Notebook

Also provides a local Jupyter environment

Provides the ability to run multiple environments

Prompts for optional Anaconda Nucleus account for cloud development (free)

Sypder IDE

Demonstration

Setting up a Python environment using Anaconda

Setting up VS Code as a Python Environment

Add Extensions	Python Jupyter Anaconda
Vscodecloud.com	Cloud based version of VS Code $10 a month 7 day free trial
Notebooks	(*Shift* + *Command* + *P*) to Create New Blank Notebook

Demonstration

Setting up a Python environment using VS code

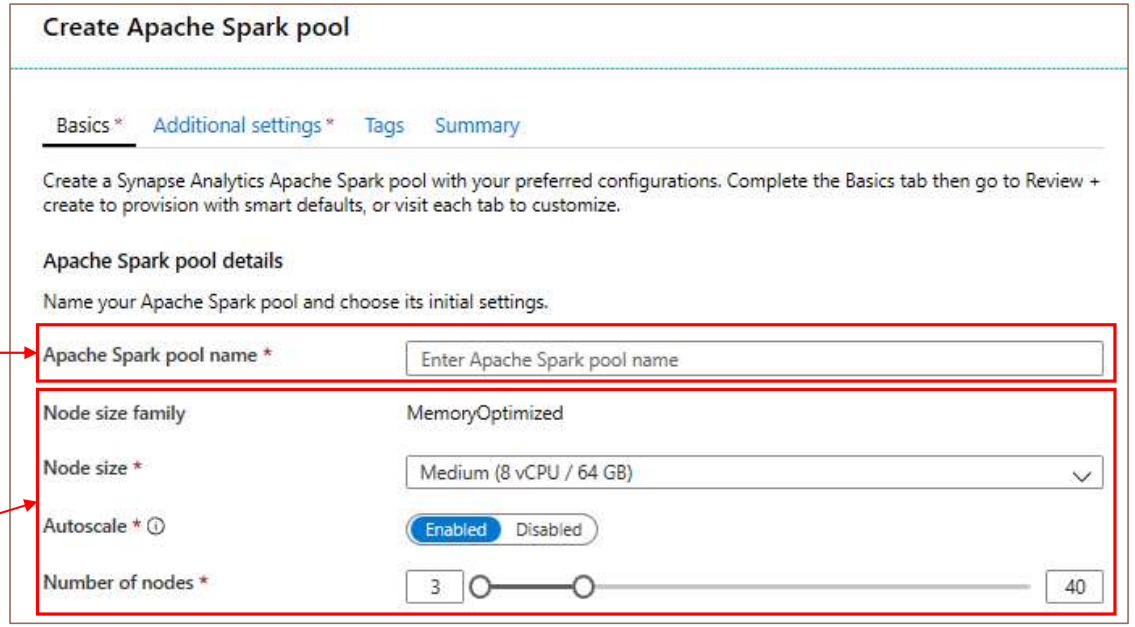
Azure Synapse Analytics Notebooks

- Part of the Analytics Environment
- Nteract not Jupyter
- Preconfigured for Data Analysis with Spark
- Must first configure a pool to run it
- Can import other notebooks *.ipynb

Creating a Spark pool (1 of 2)

Provision Spark Pool through Azure Portal with default settings or per requirements

Basic Settings – Minimum details required from user

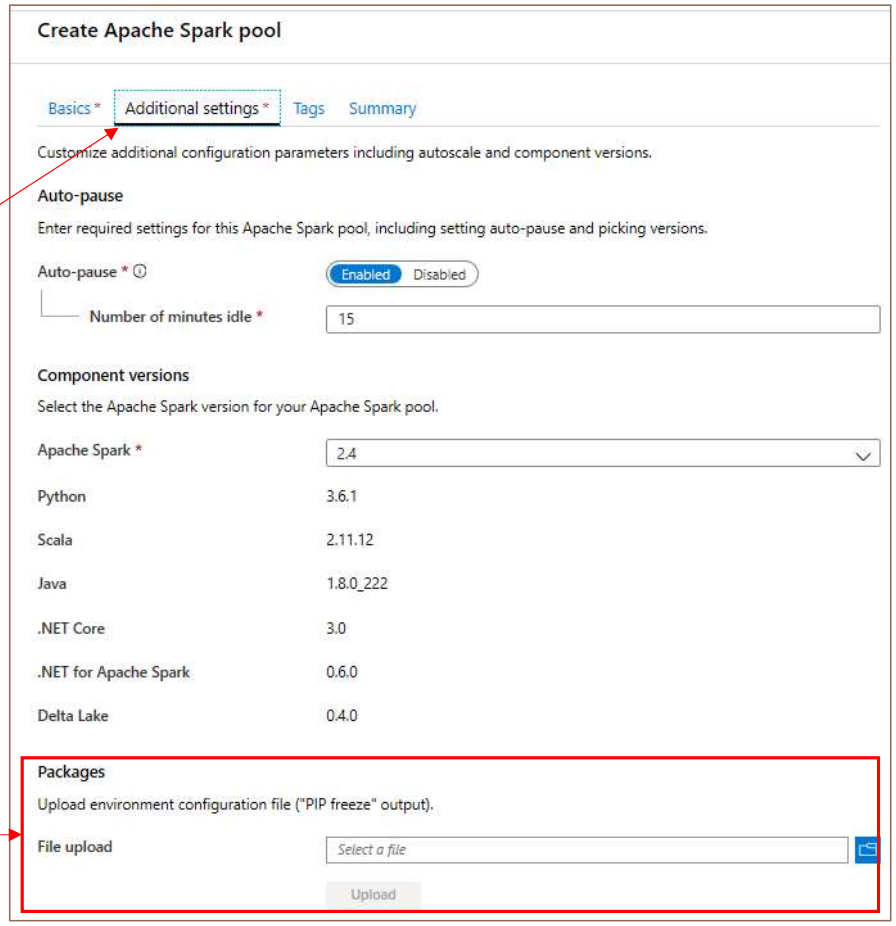


Creating a Spark pool (2 of 2) - optional

Additional Settings offer optional settings to customize Spark pool

Customize component versions, auto-pause

Import libraries by providing text file containing library name and version



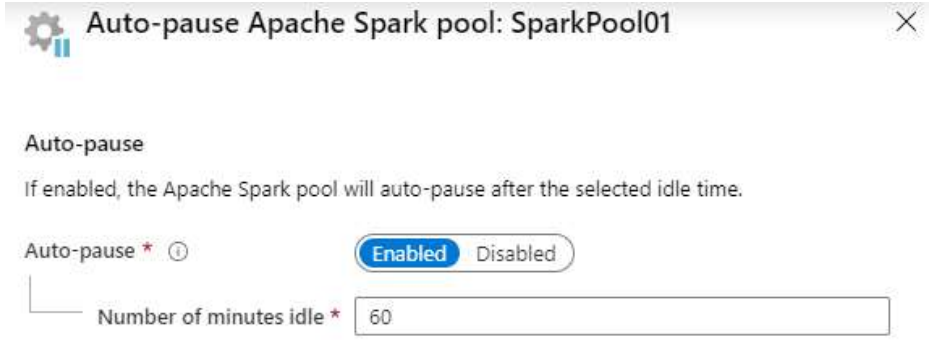
Spark Pools - Configuring

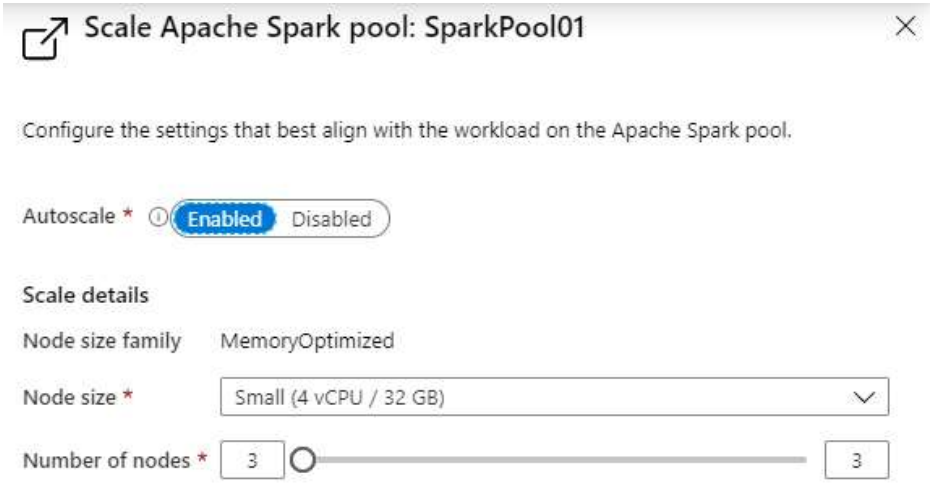
Auto-pause

Enables the pool to automatically stop running when

Autoscale

Enables the pool to automatically adjust the number of running nodes





Demonstration

Setting up a Python environment using Azure Synapse

Python Code Walkthough

Create your own Notebook environment

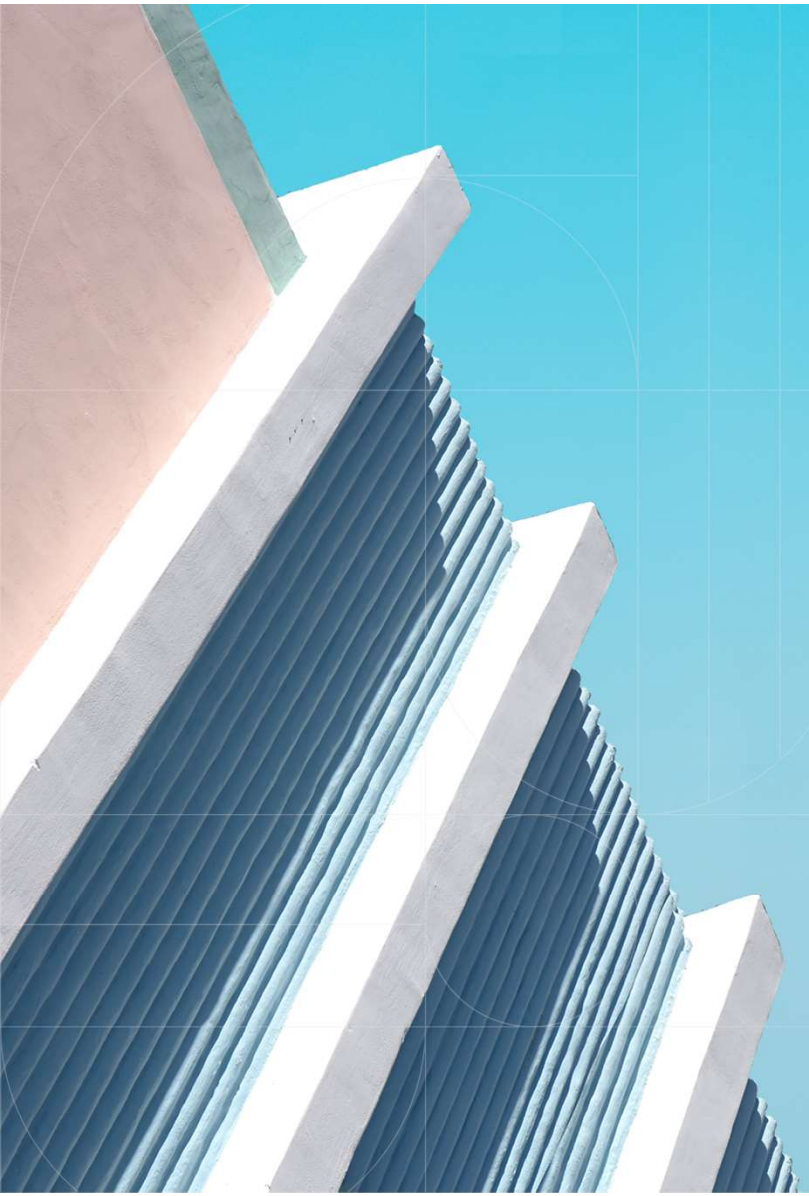
Open an Azure Synapse Analytics Notebook

or

Install Anaconda

or

Install and setup VS Code



Introduction to Python

Python 101

Python 101

Markdown	Packages	Data frames	Visualizations
Formatting and making it pretty	All about the Libraries	Data Container	Matplotlib Seaborn

Markdown

Time to learn it

Or just bring up a cheat sheet

https://www.markdownguide.org/cheat-sheet

Azure Synapse WYSiwig Markdown interface

Element	Markdown Syntax

Heading	`# H1`
	`## H2`
	`### H3`

Bold	`**bold text**`
Italic	`*italicized text*`
Blockquote	`> blockquote`

Ordered List	`1. First item`
	`2. Second item`
	`3. Third item`

Unordered List	`- First item`
	`- Second item`
	`- Third item`

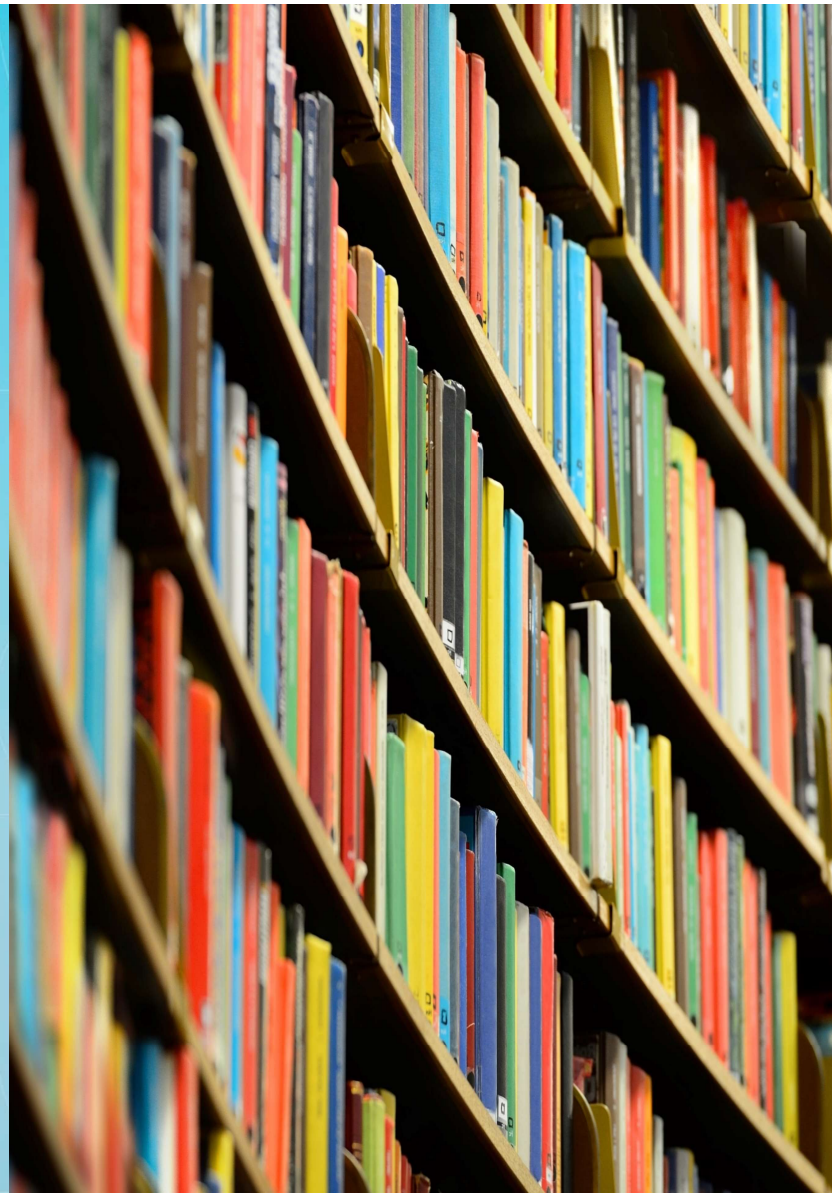
Code	`` `code` ``
Horizontal Rule	`---`
Link	`[title](https://www.example.com)`
Image	`![alt text](image.jpg)`

Python Libraries

Minimize the use of code

Provide instant functionality

Designed for different tasks like Machine learning



Common Libraries

Pandas	Koalas	Spark
Matplot Lib	Seaborne	Numpy

NumPy

Developed for Linear Algebra computations

Array structure where All values must be of one datatype

Allow calculations across entire arrays

Included as a dependency for a number of different libraries

Pandas

Most popular Python Library

Modeled after the R DataFrame

Designed for small to medium datasets, does not scale

Table-like structure of data

Pandas.pydata.org

Spark Dataframe - Koalas

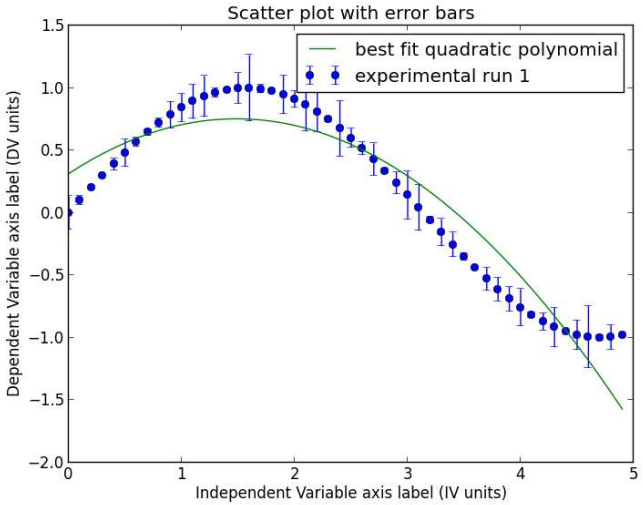
Designed to work like Pandas

Used in Spark

Uses Spark Dataframes architecture

Does not contain all of the features of Spark Dataframes

Matplotlib for Pictures of data



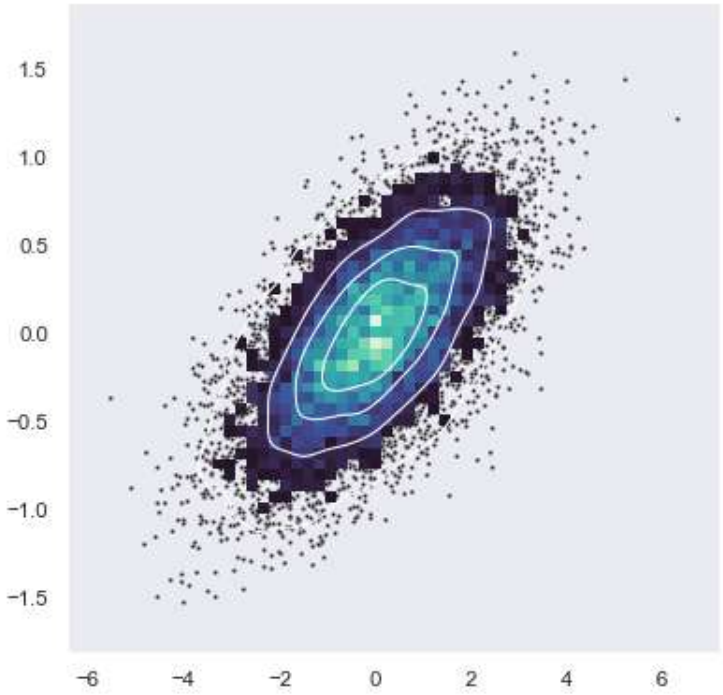
Primary method for visualizing data

Standard reference is plt

2D libraries

matplotlib

Seaborn Statistical Data Visulaliation



Based on Matplotlib

Standard reference is sns

More visualizations than Matplotlib

matplotlib

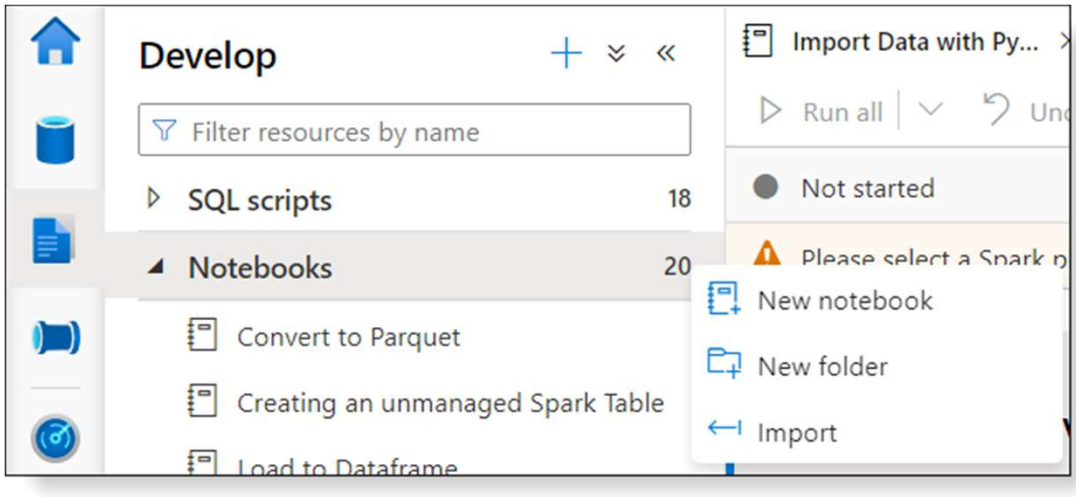
Importing code into Notebooks

Click on the ellipse(…) in Develop->Notebooks

Folder management

File format .ipnyb

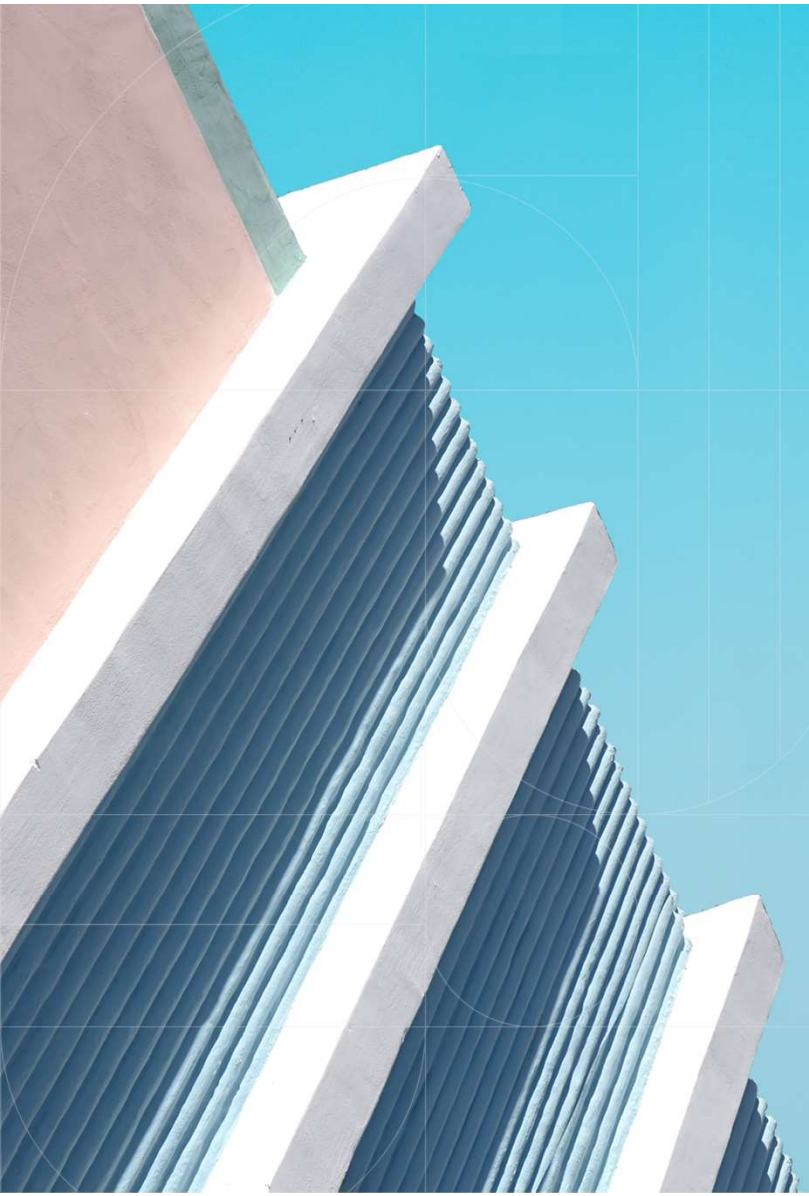
Start your pool



Demonstration

Python Code Walkthrough

Python Visuals.ipynb



Introduction to Spark

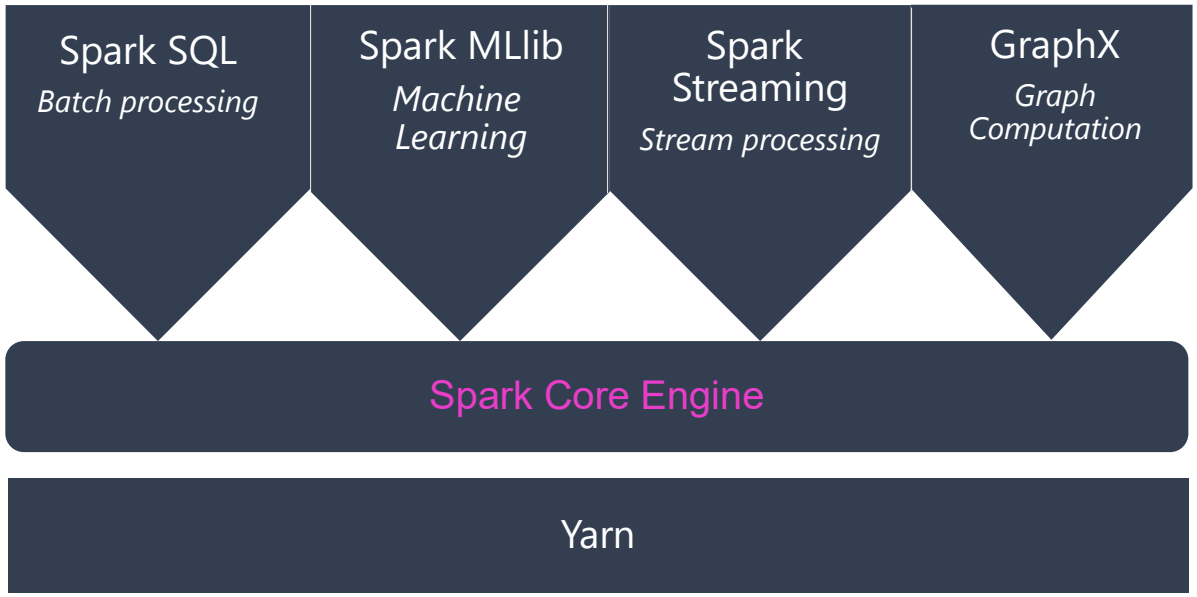
Internals

Apache Spark

An unified, open source, parallel, data processing framework for Big Data Analytics

Spark Unifies:

- Batch Processing
- Interactive SQL
- Real-time processing
- Machine Learning
- Deep Learning
- Graph Processing



http://spark.apache.org

Spark Component Features

Spark SQL

- Unified data access: Query structured data sets with SQL or DataFrame APIs
- Fast, familiar query language across all your enterprise data
- Use BI tools to connect and query via JDBC or ODBC drivers

MLlib & SparkML

- Predictive and prescriptive analytics
- Machine learning algorithms for:
  - Clustering
  - Classification
  - Regression
  - etc.
- Smart application design from pre-built, out-of-the-box statistical and algorithmic models

Spark Streaming

- Micro-batch event processing for near-real time analytics
- e.g. Internet of Things (IoT) devices, Twitter feeds, Kafka (event hub), etc.
- Spark's engine drives some action or outputs data in batches to various data stores

GraphX

- Represent and analyze systems represented by graph nodes
- Trace interconnections between graph nodes
- Applicable to use cases in transportation, telecommunications, road networks, modeling personal relationships, social media, etc.

How Spark Runs

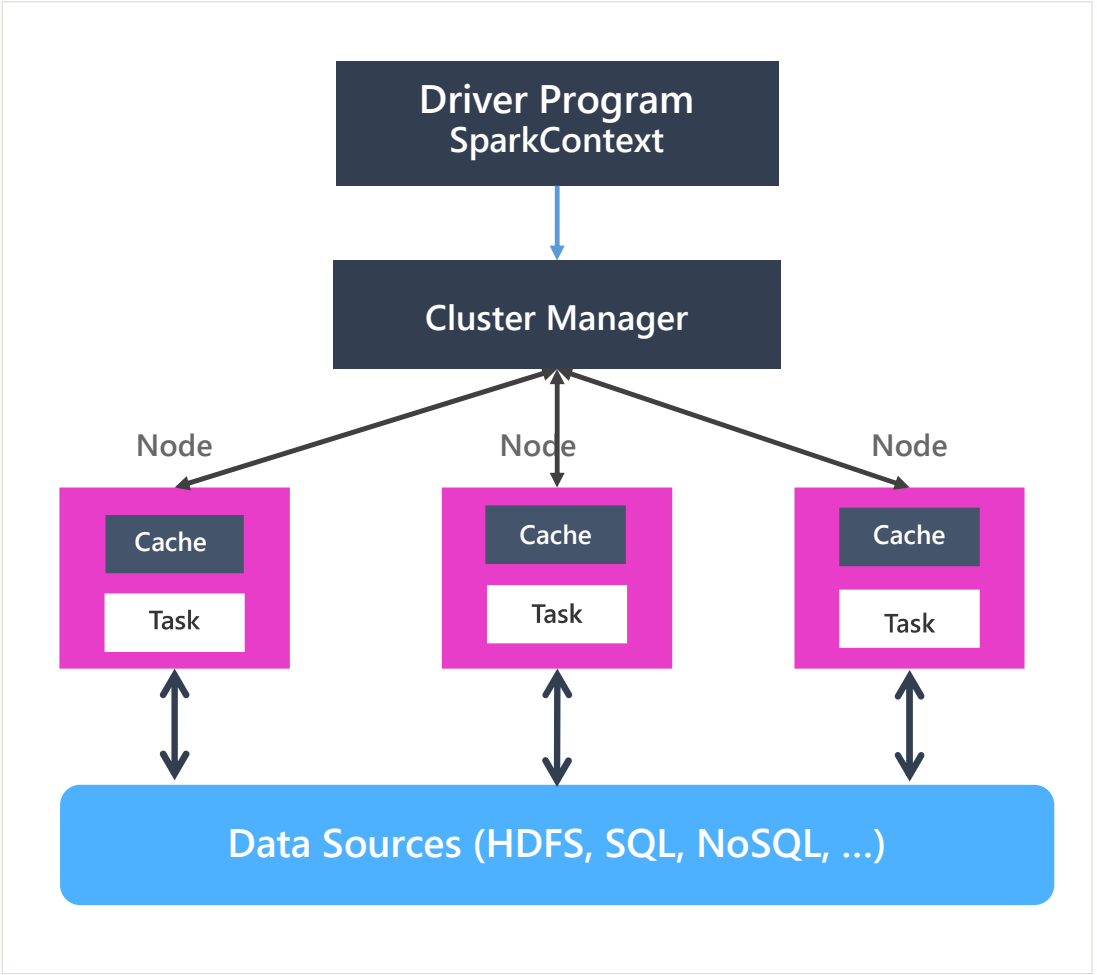
Take a logical operation

Breakup into a plan with dataset and clusters

Spark API makes that happen

General Spark Cluster Architecture

- ‘Driver’ runs the user’s ‘main’ function
- Worker nodes read and write data from/to Data Sources
- Worker node also cache transformed data in memory as Resilient Data Sets.



Using Spark Libraries when coding

If you do not use Spark Libraries, no multi-parallel results

No Pandas

Limits the use of Machine Learning libraries

Spark languages and Synapse supported languages

Spark natively supports

Python

Scala

ANSI SQL

R

Java

Synapse Supports

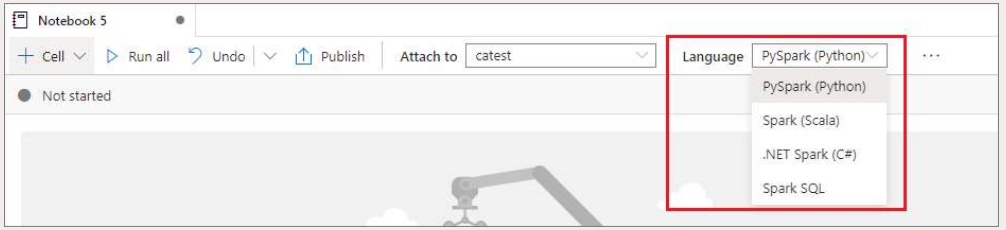
Python

Scala

ANSI SQL

.Net

ANSI SQL



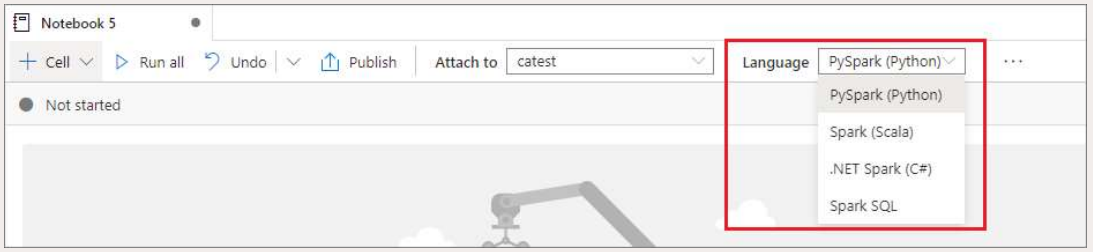
Case SensitiveMagic Comands

%%pyspark

%%spark

%%sql

%%csharp



Spark Dataframes

In memory table

Spark dataframe

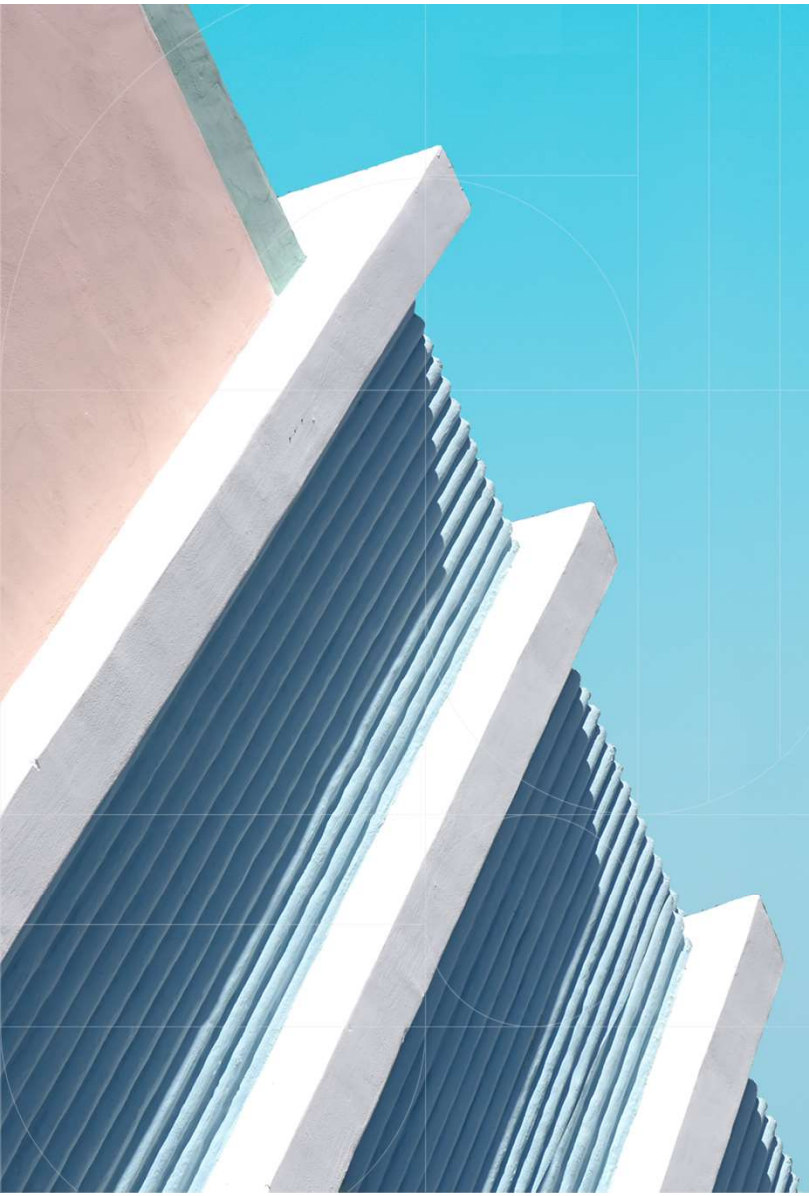
Pandas dataframe

Cannot be changed

Demonstration

Python Code Walkthrough

Import Data with Python.ipynb



Data Exploration with Spark

Data can come from anywhere

- Flatfiles
- CosmosDB
- Web
- Database
- Blob Storage

Azure Data Lake Gen 2

Hierarchical Name Spaces

File System Name

Hadoop Access

Logging

Reading CSV data from ADLS Blob Storage

```
%%pyspark

# Primary storage info

account_name = 'synapse11datalake' # primary ADLS account name

container_name = 'root' # Primary ADLS Gen2 file system from Synapse
relative_path = 'Raw/JSON AdventureWorks' #  relative folder path

filename =  'bank.csv'

data_path = 'abfss://%s@%s.dfs.core.windows.net/%s/%s' %
(container_name, account_name, relative_path,filename)

df = spark.read.csv(data_path, header = 'True', inferSchema = 'True')
```

File Formats for Dataframes

- CSV
- JSON
- TSV
- Delta Lake
- Parquet

Demonstration

Python Code Walkthrough

Load to Dataframe.ipynb

Delta Lake

Meant to address problems with Data Lakes

Open Source Based on Parquet

Contains ACID Transactions

https://docs.delta.io

Under the Delta under the Hood

Massive scale through decoupling Compute and Storage

Reliability by enforcing Schemas on write, rejected written separately

Performance by indexing and caching

Low-latency by through real time streaming ingest and write

Works on top of ADLS

Delta Lake Ensures Relability

Enforces the schema

Contains Unified Batch and Streaming as they can be the same table

Contains a transaction log

Data Snapshots

Unified Data Management

Delta improves Machine Learning performance

Faster access to the data used by MLSpark

Interactive analysis capabilities

Handles more data

Performance Tuning Delta Lake

Partitioning (filter part of the data)

```
OPTIMIZE <table> [WHERE <partition_filter>]
```

Compaction

```
OPTIMIZE <table> [WHERE <partition_filter>]
ZORDER BY (<column>[, …])
```

Z-Ordering (very Index like)

How Data Skipping Works internally

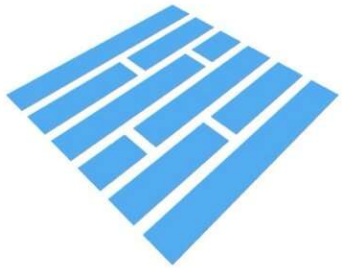
```
SELECT input_file_name() as "file_name",
       min(col) AS "col_min",
       max(col) AS "col_max"
FROM table
GROUP BY input_file_name()
```

file_name	col_min	col_max
data_file_1	6	8
data_file_2	3	10
data_file_3	1	4

```
Select * from table where column <= 5
```

Data Skipping means that Delta Lakes knows to only look at data_file_3 and data_file 2

Parquet Files



Open-source column oriented data store

Part of Apache Hadoop Collective

Supports nested data structures

Columnar Storage compression options

Compression can be huge resulting in decreased query time and cost to store compared to CSV

Demonstration

Python Code Walkthrough

Convert to Parquet.ipynb

Data Analysis

Determining if you have the data required to complete the task

Developing Appropriately Sized Data sets

Data Cleaning and Organizing

Summary Information

Use nympy to get stats on dataframes

```
summary = df.describe()
summary.head(20)
```

	mpg	cylinders	displacement	horsepower	weight	acceleration	model-year
count	398.000000	398.000000	398.000000	396.000000	398.000000	398.000000	398.000000
mean	23.514573	5.454774	193.425879	104.189394	2970.424623	15.568090	76.010050
std	7.815984	1.701004	104.269838	38.402030	846.841774	2.757689	3.697627
min	9.000000	3.000000	68.000000	46.000000	1613.000000	8.000000	70.000000
25%	17.500000	4.000000	104.250000	75.000000	2223.750000	13.825000	73.000000
50%	23.000000	4.000000	148.500000	92.000000	2803.500000	15.500000	76.000000
75%	29.000000	8.000000	262.000000	125.000000	3608.000000	17.175000	79.000000
max	46.600000	8.000000	455.000000	230.000000	5140.000000	24.800000	82.000000

Examining Data with SQL

SQL comprises a majority of the calls in Spark

As of Spark 3.0 ANSII SQL

Instantiate dataframes as tables

Two methods for writing SQL – PySpark and SQL

```
%%pyspark
person.createOrReplaceTempView("person")
graduateProgram.createOrReplaceTempView("graduateProgram")
sparkStatus.createOrReplaceTempView("sparkStatus")
```

Using Python to create tables

```
spark.read.json("/data/flight-
data/json/2015-summary.json")\

.createOrReplaceTempView("some_sql_view")
```

SQL in Dataframes

```
spark.sql("""
SELECT DEST_COUNTRY_NAME, sum(count)
FROM some_sql_view GROUP BY DEST_COUNTRY_NAME
""")\
  .where("DEST_COUNTRY_NAME like 'S%'").where("`sum(count)` > 10")\
  .count() # SQL => DF
```

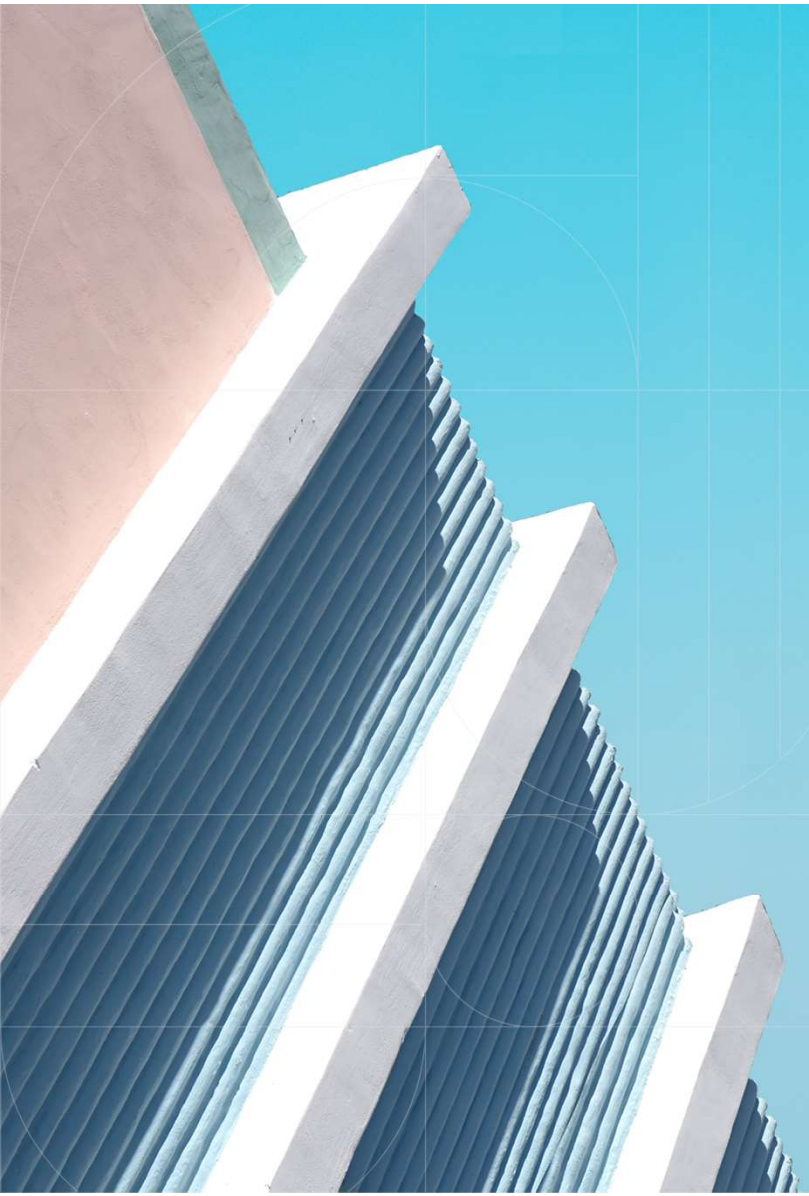
Magic SQL

```
SELECT * FROM person JOIN graduateProgram
ON person.graduate = graduateProgram.id
```

Demonstration

Python Code Walkthrough

Load to SparkSQL.ipynb



Developing ETL components with Python

ETL - Why not just use ADF ?

Cannot handle the data volume

Complicated transformations

Cost

Spark ETL

Incorporate into ADF

Create complex transformations

Decrease Azure Spend

Increase Speed of transformation

Extracting Data

Process streaming data as well as batch

Join data from disparate sources

Extract elements from objects

Transformation - Cleansing

Replace values

Deduplication

Derivation

Data Splitting

Validation

Create Standard reusable models

Transformation - Reformatting

Add keys

Change storage structures

Aggregation

Incorporate Machine Learning elements

Loading

Migrate data to multiple sources

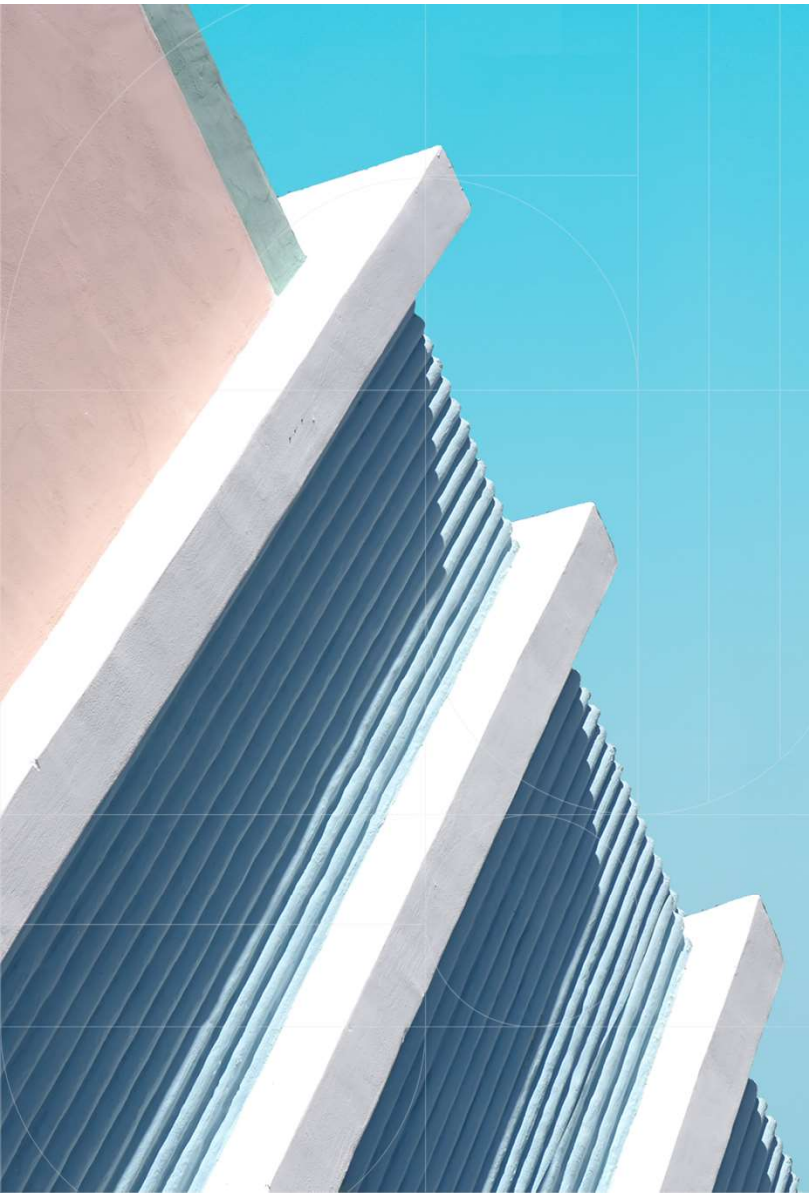
Write in multiple formats

Create Spark pipelines or call from other

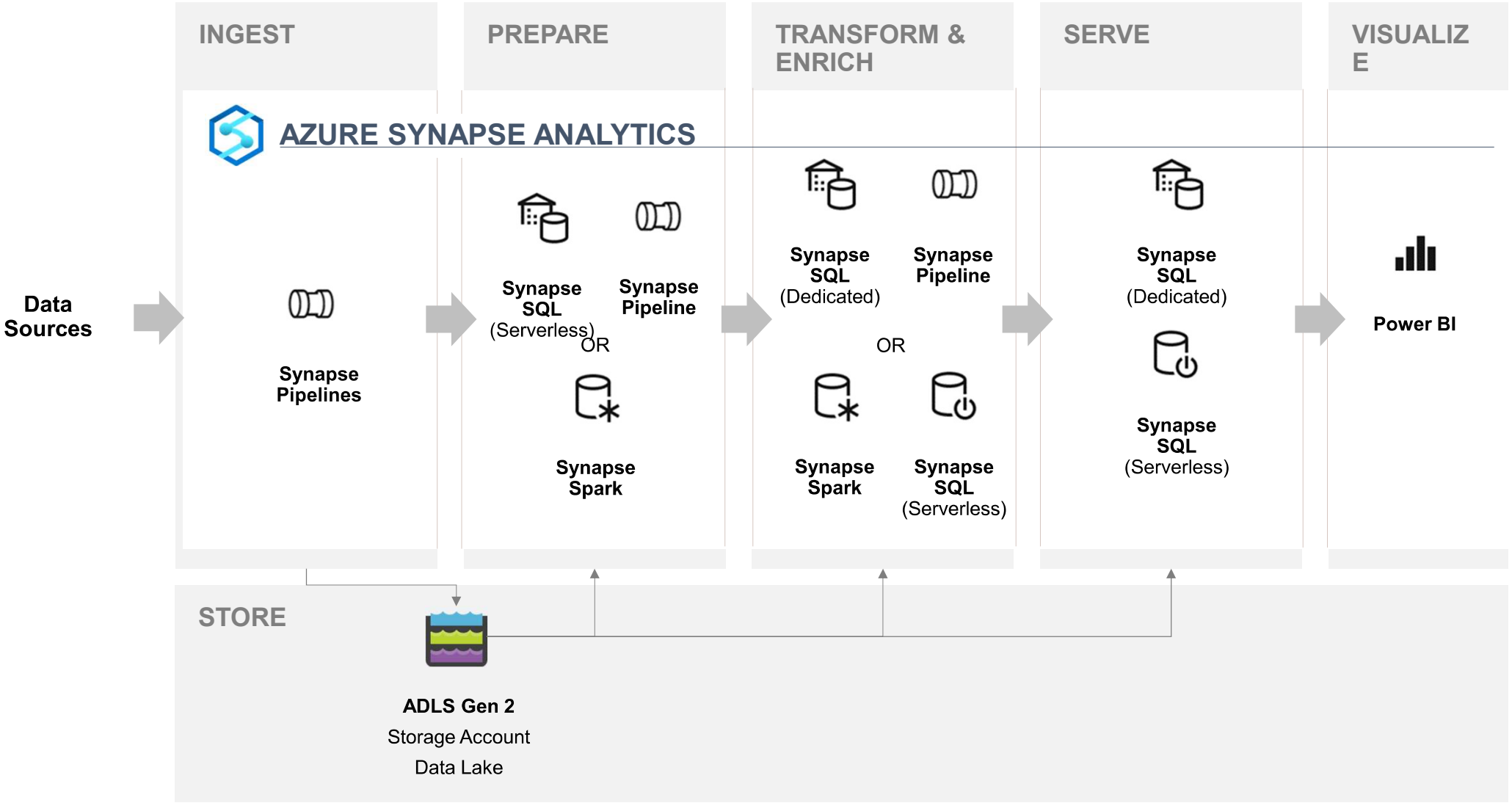
Demonstration

Python Code Walkthrough

ETL.ipynb



Organizing Data with Data Engineering


INGEST
PREPARE
TRANSFORM & ENRICH
SERVE
VISUALIZ E
AZURE SYNAPSE ANALYTICS
Data Sources
Synapse Pipelines
Synapse SQL (Serverless)
Synapse Pipeline
OR
Synapse Spark
Synapse SQL (Dedicated)
Synapse Pipeline
OR
Synapse Spark
Synapse SQL (Serverless)
Synapse SQL (Dedicated)
Synapse SQL (Serverless)
Power BI
STORE
ADLS Gen 2
Storage Account
Data Lake

What is a Data Lake

Cheap data storage location

Able to handle any kind of data

Integrates with many services and sources

Scalability at a low cost

Schema on Read

Beware the Data Swamp as data has no enforced organization

What is a Data Lakehouse?

Using a data lake as a data warehouse

Query flat file data as if it was in a warehouse

Decreased cost over Data Warehouse

Ideal for Power BI

Needs to be part of a well organized Data alkae

Data Lakehouse Considerations

Slower than a data warehouse

Needs to be well organized

Does the query speed matter?

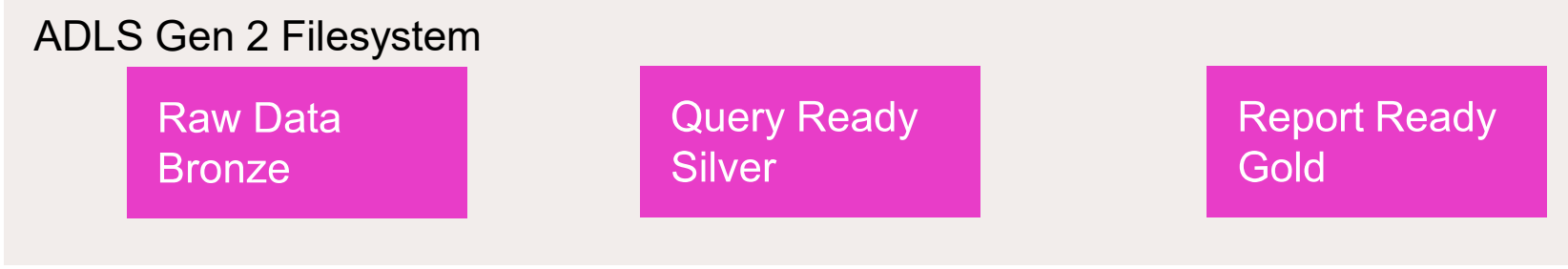
Azure Data Lake Gen 2

- Hierarchy
- Performance
- Delta Lake
- Flexible Pricing

Structuring ADLS Gen2

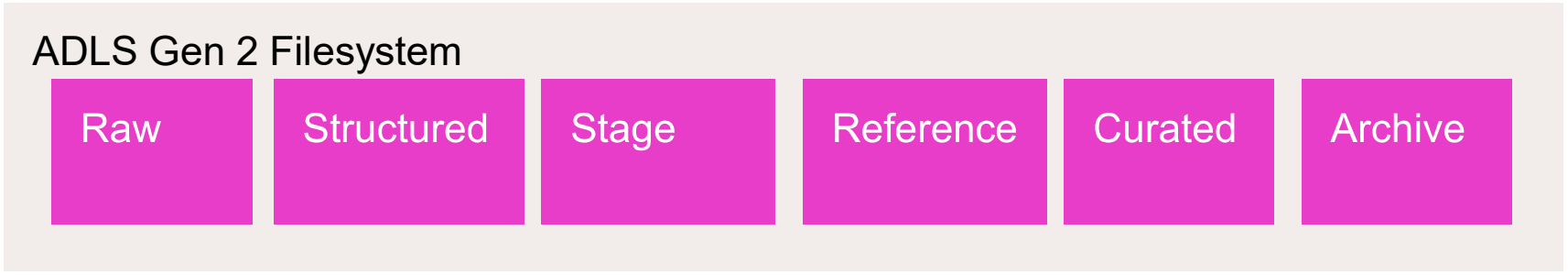
Separate storage accounts for each environment: dev, test, & production.

Use a common folder structure to organize data by degree of refinement.



Greater Flexiblity

Use a common folder structure to organize data by degree of refinement.



Optional Organizational Subareas

Subsystem

Data Sources

Data Feeds

Partitions

Used for data received incrementally Big Data

Users

Sand box for development

Azure Storage Explorer Many ways to explore

Application

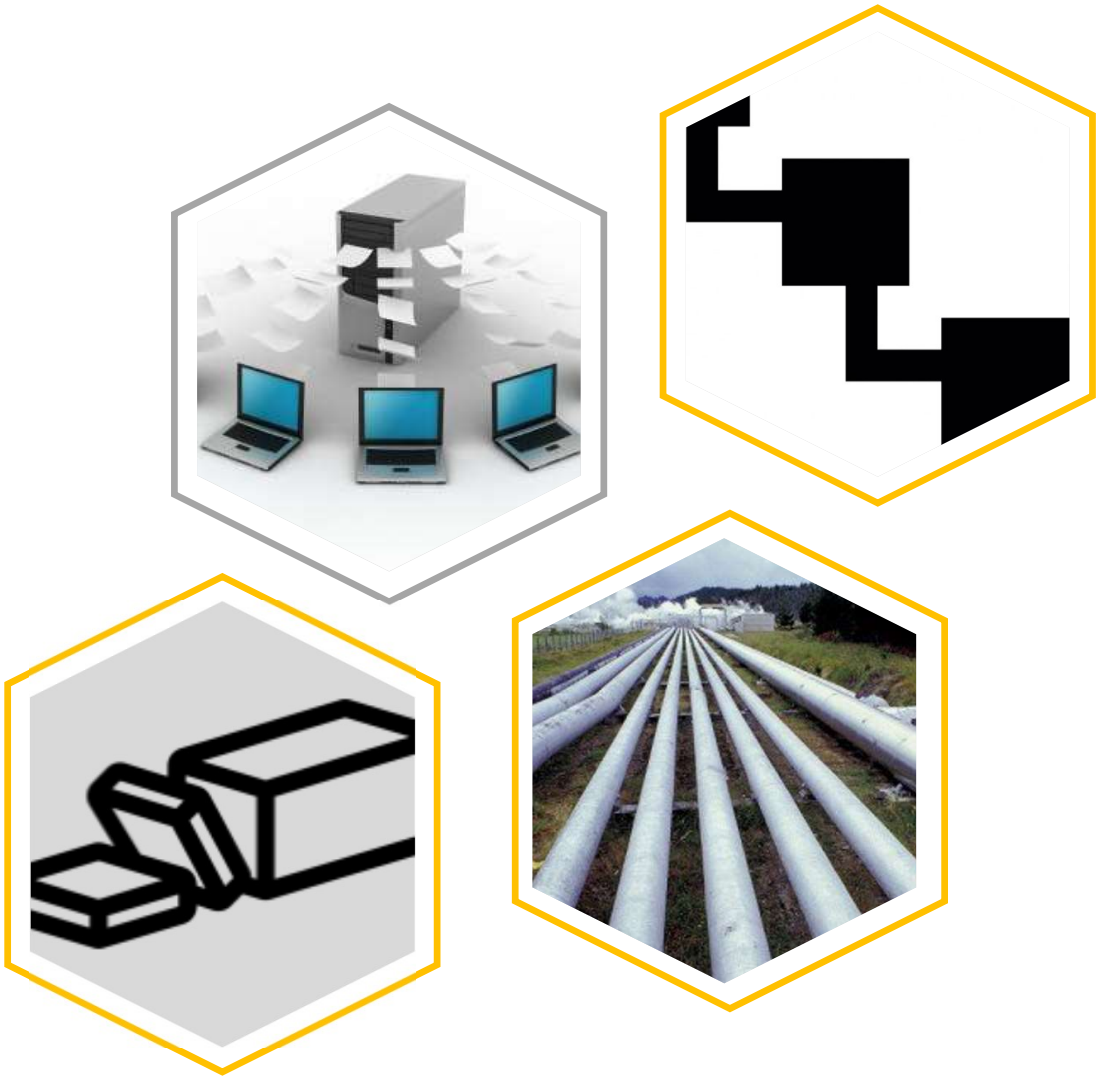
Within Azure

Limits on data transfer

Demonstration

Data Lake Review and Tools


Machine Learning with Spark



MLLib

Algorithms

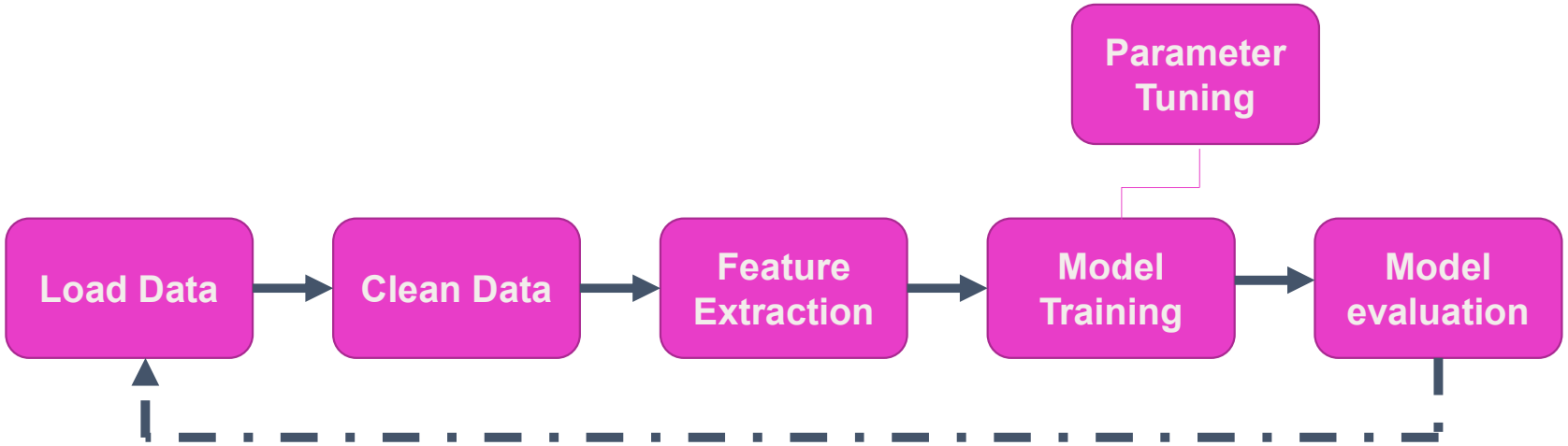
Featurization

Utilities

Pipelines

Machine Learning Pipeline

Standard Workflow

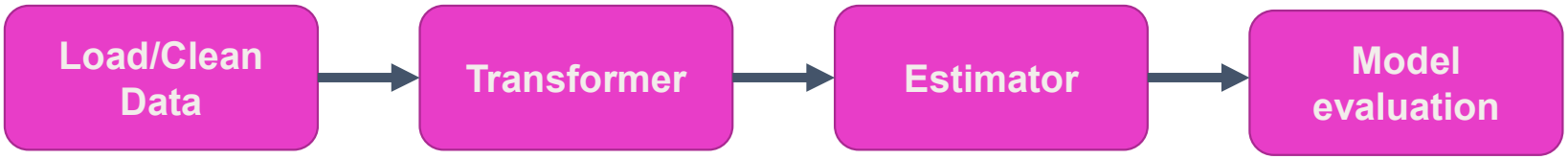


What is inside of Mllib?

- Algorithms
- Featurization
- Utilities
- Pipeline

ML Lib Pipeline

Spark



Sample MLlib Pipeline Python Code

```
from pyspark.ml import Pipeline
from pyspark.ml.feature import HashingTF, Tokenizer
from pyspark.ml.classification import LogisticRegression

# Configure an ML pipeline, which consists of three stages: tokenizer, hashingTF, and lr.
tokenizer = Tokenizer(inputCol="text", outputCol="words")
hashingTF = HashingTF(inputCol=tokenizer.getOutputCol(), outputCol="features")
lr = LogisticRegression(maxIter=10, regParam=0.001)
pipeline = Pipeline(stages=[tokenizer, hashingTF, lr])

# Fit the pipeline to training documents.
model = pipeline.fit(training)

# Make predictions
prediction = model.transform(test)
```

Pipeline Parameteter Tuning

Automate model tuning in the Pipeline

MLLib will select the best hyperparameters for the model

Use the individual algorithms in the Estimator or the whole Pipeline

Data is evaluated to determine the best result for you

Python Code for Pipeline Parameter tuning

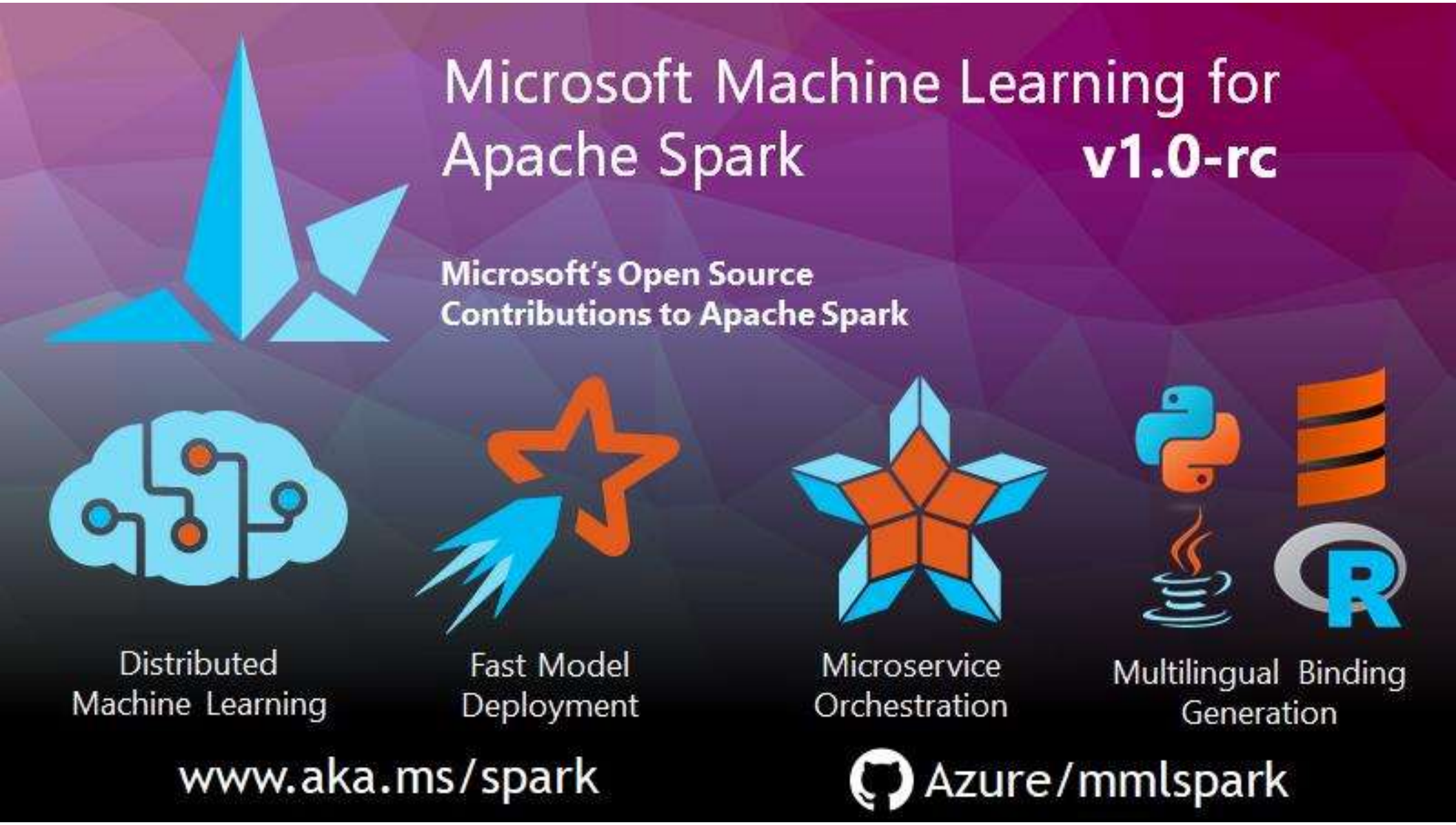
```
// Pipeline is treated as an Estimator, wrapping it in a CrossValidator instance.
// This will allow us to jointly choose parameters for all Pipeline stages.

val cv = new CrossValidator()
  .setEstimator(pipeline)
  .setEvaluator(new BinaryClassificationEvaluator)
  .setEstimatorParamMaps(paramGrid)
  .setNumFolds(2)  // Use 3+ in practice
  .setParallelism(2)  // Evaluate up to 2 parameter settings in parallel
```

Demonstration

ML Pipelines

ML Pipeline and Parameter Tuning.ipynb


Microsoft Machine Learning for Apache Spark
v1.0-rc
Microsoft's Open Source Contributions to Apache Spark
Distributed Machine Learning
Fast Model Deployment
Microservice Orchestration
Multilingual Binding Generation
www.aka.ms/spark
Azure/mmlspark

MMLSpark

Builds on top of MLLib in Spark

Includes Model Deployment with Web Services

Functionality to Improve Deep Learning

Structured Streaming

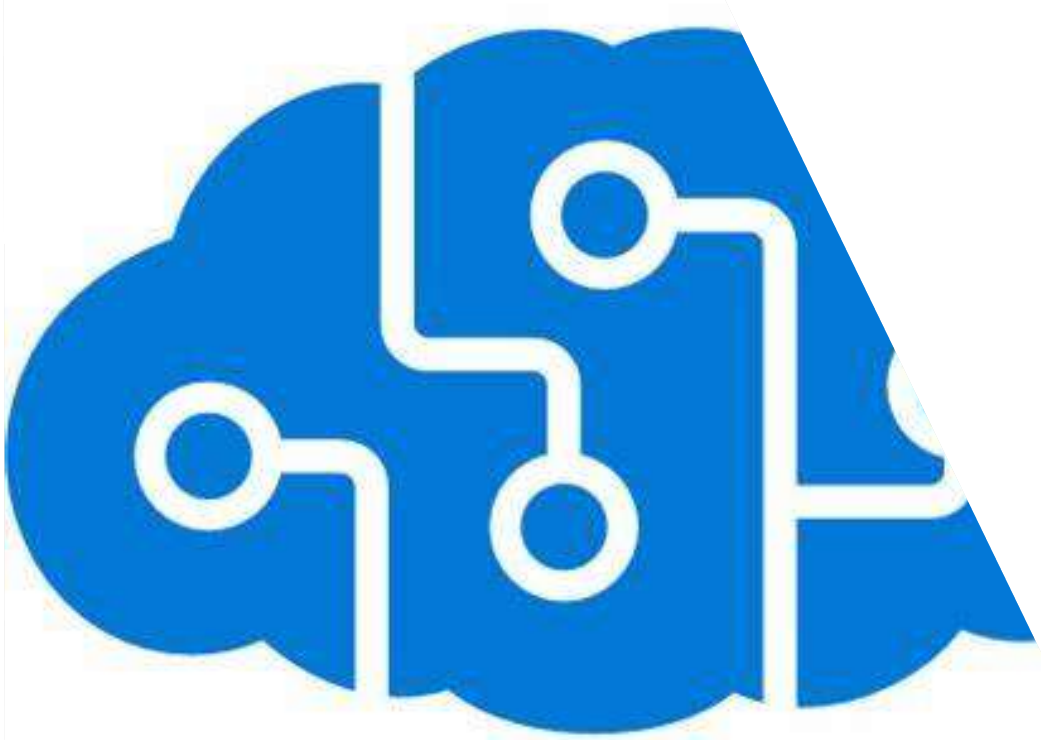
Easily Call Cognitive Services

Expanding ML

Writing in Spark

Cognitive Services

Incorporating Azure ML



Cognitive Services

Vision

Speech

Language

Decision

Search

Configuring Cognitive Services in Spark

- Cognitive Services Account
- Key Vault
- Secrets

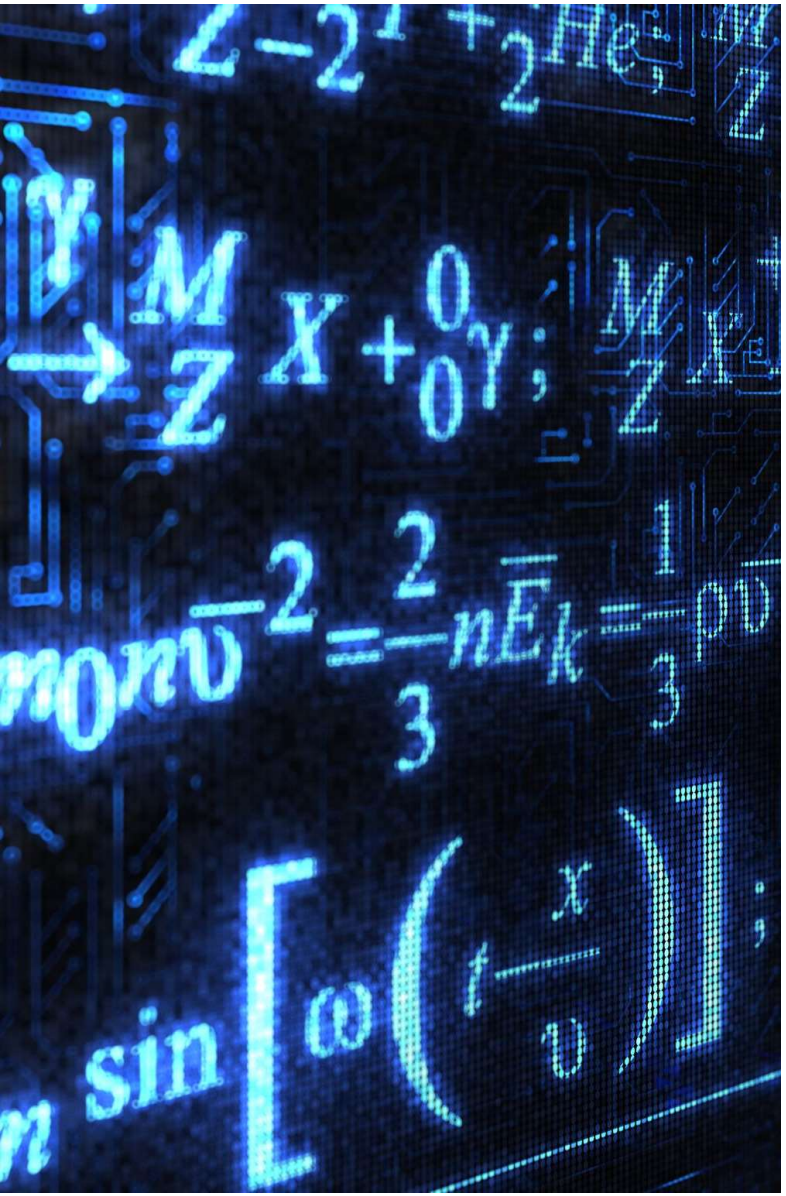
Demonstration

Spark and Cognitive Services

Cognitive Service TextSentiment in Spark.ipynb

Diving into Algorithms

- Classification
- Regresssion
- Anomaly Detection
- Time Series
- Clustering



Why does Algorithm Selection take so long?

Many different possible options

Many training runs need to be evaluated

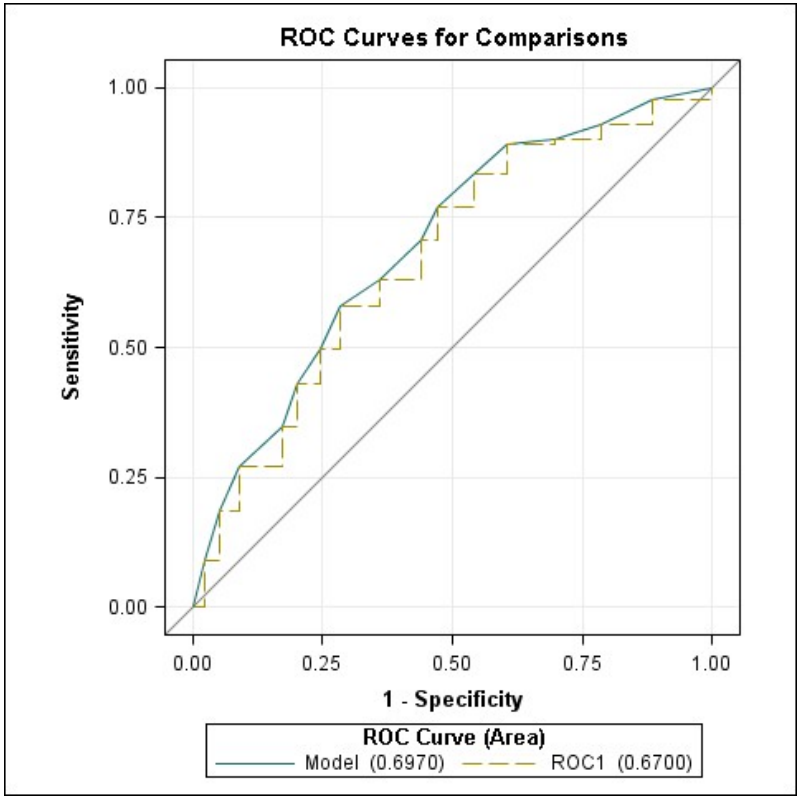
Need lots of data to prevent over fitting

Analyzing the Results

Graphical analysis

Numeric Analysis

Influencer Variables



Demonstration

Machine Learning with Python

ML Classification.ipynb

Time Sinks in Machine Learning

Cleaning your data

Determining the Best Algorithms

Hyperparameter Tuning

Automating Algorithm Selection

DataRobot

H20.ai

Azure AutoML

Azure ML

Wizards

Auto ML

ONNX Deployment

Azure Synapse using in with AzureML Libraries

- Incorporate Azure ML into Synapse
- Improve Algorithm Selection
- View results in Azure ML and Synapse

Demonstration

Incorporating Azure ML libraries

AutoML.ipynb

Summary

- Python 101
- Understanding Spark Basics
- Data Exploration and Analysis with Spark
- ETL with Python
- Engineering a well-organized Data lake
- Machine Learning with Spark ML Lib

Questions and Contact Info

Ginger Grant

ginger.grant@desertislesql.com

desertislesql.com

desertislesql