MARY ROSE LEGASPI DEVELOPMENT PROJECT

MR. MONIR (SUPERVISOR)

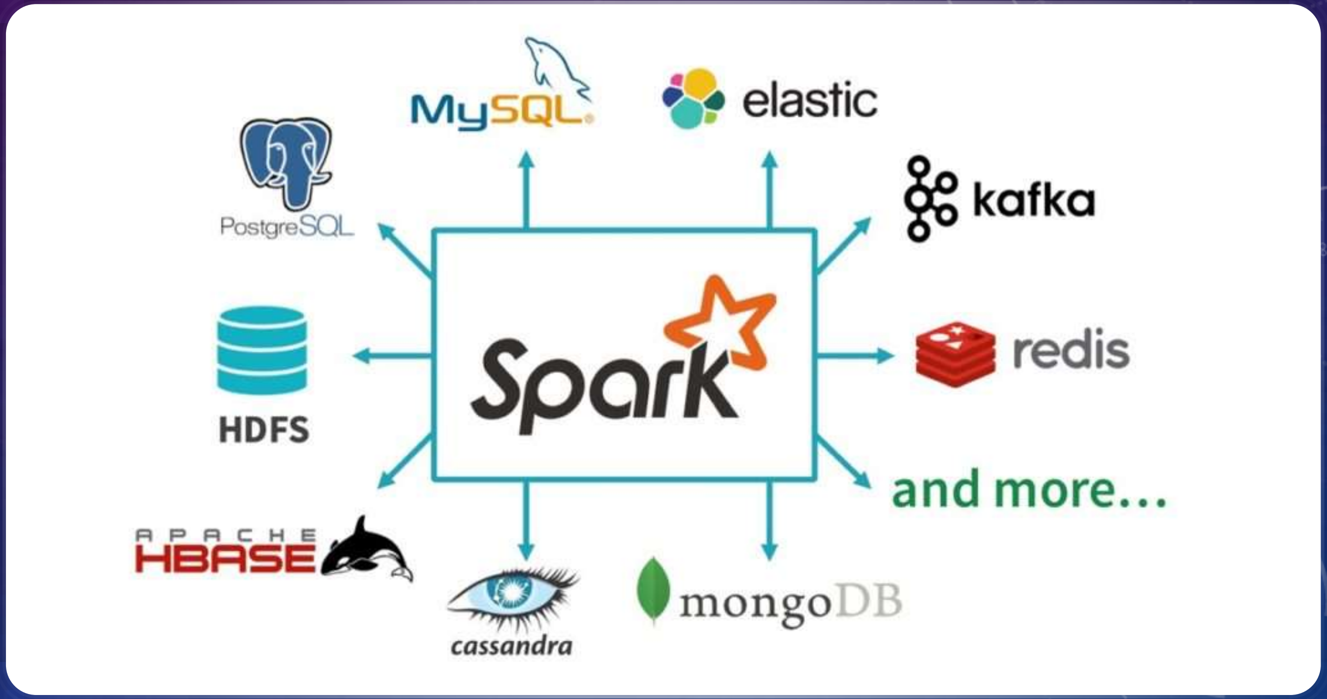
BUILDING SCALABLE SOLUTION FOR PREDICTING HEART DISEASE USING APACHE SPARK MLLIB IN STANDALONE CLUSTER MODE WITH MONGODB DATABASE

BACKGROUND AND CONTEXT

WHAT IS SPARK?

What is Apache Spark?

- Apache open sourced project originally developed at AMPLab (UC Berkeley)
- Unified general data processing engine that operates varied data workloads and platforms
- Built on top of Hadoop Map Reduce and it extends the MapReduce model to efficiently use more types of computations



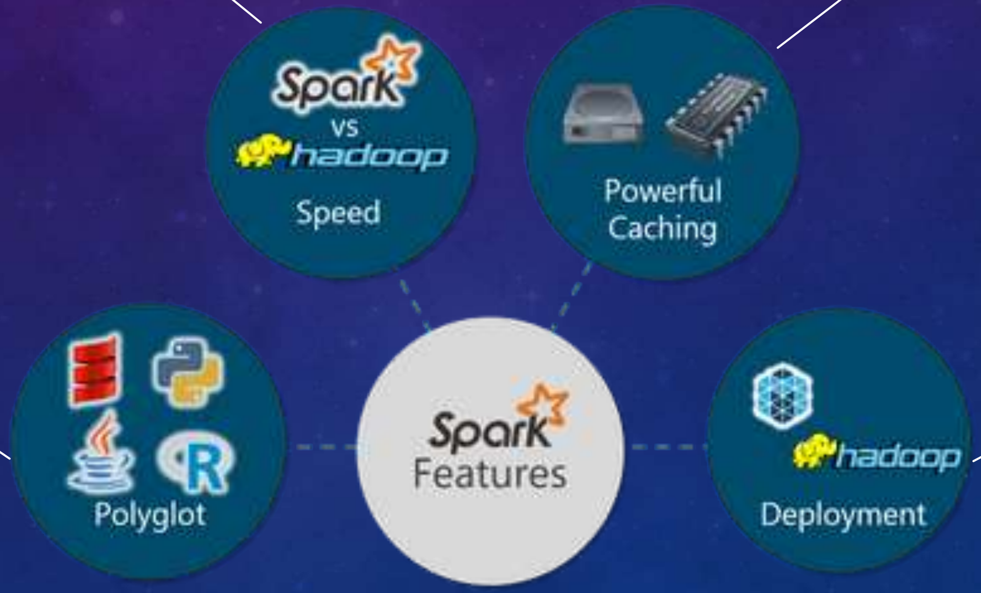
Spark features

100x faster than for large scale data processing

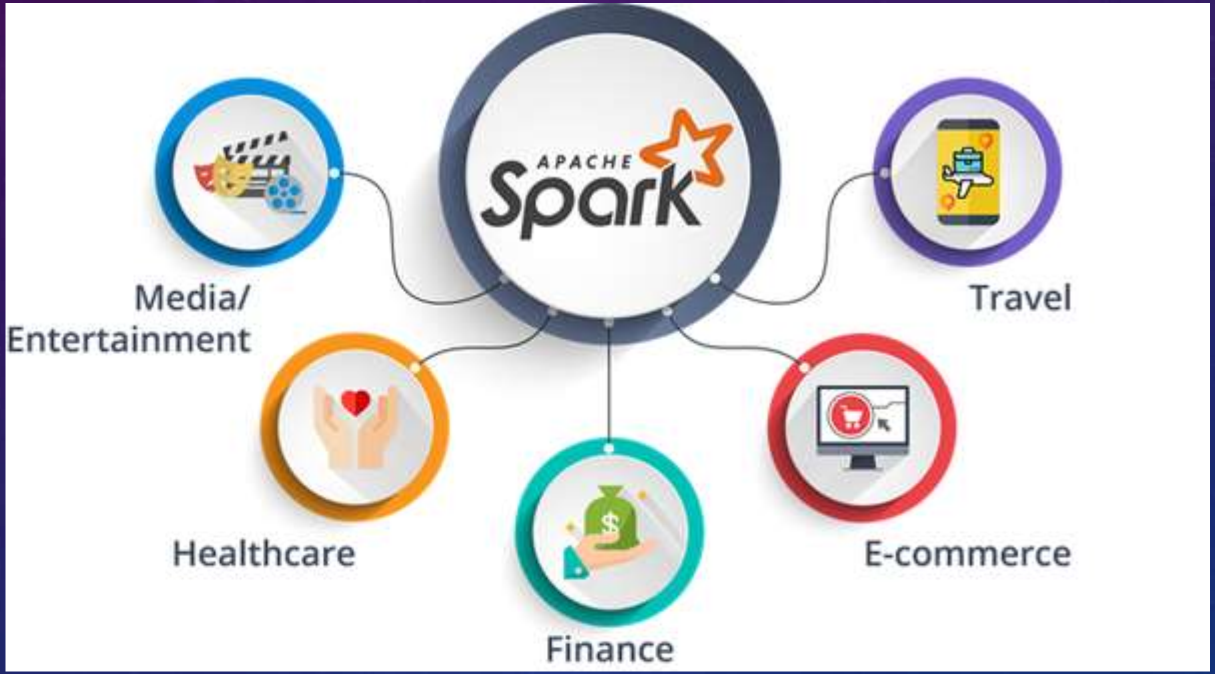
Simple programming layer provides powerful caching and disk persistence capabilities

Can be programmed in Scala, Java, Python, and R.

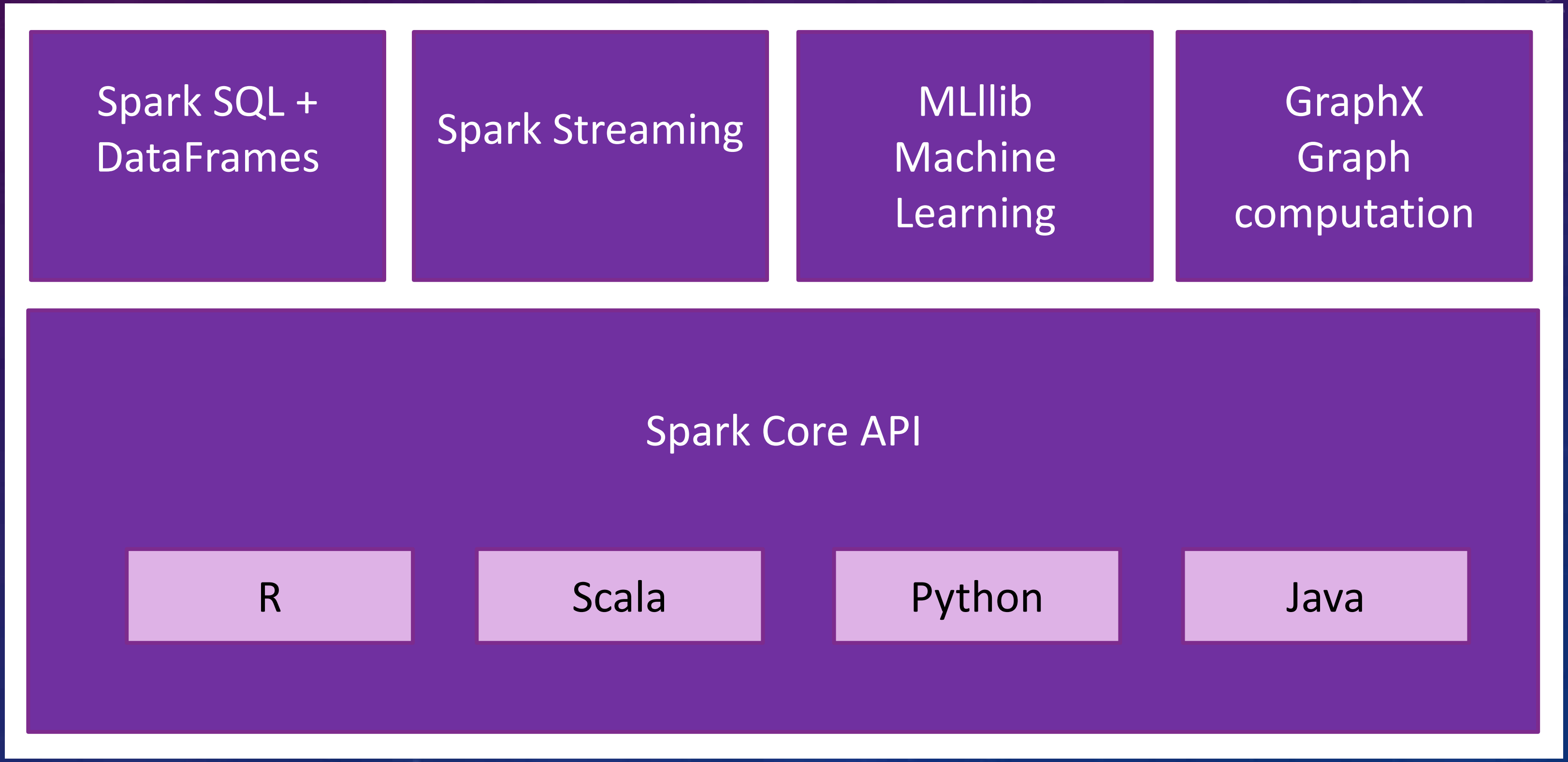
Can be deployed through Mesos, Yarn, EC2 or Sparks standalone cluster manager



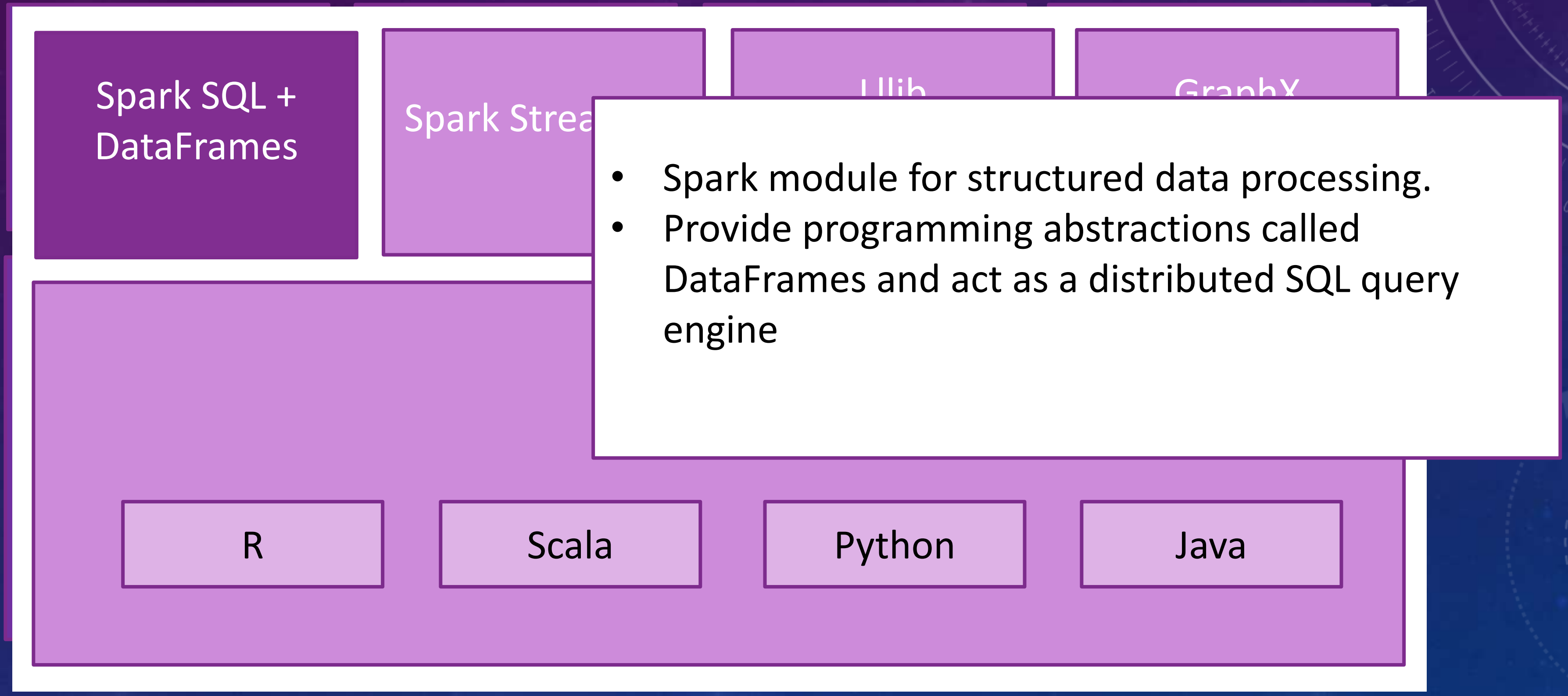
Apache Spark Domain scenarios



Spark Components

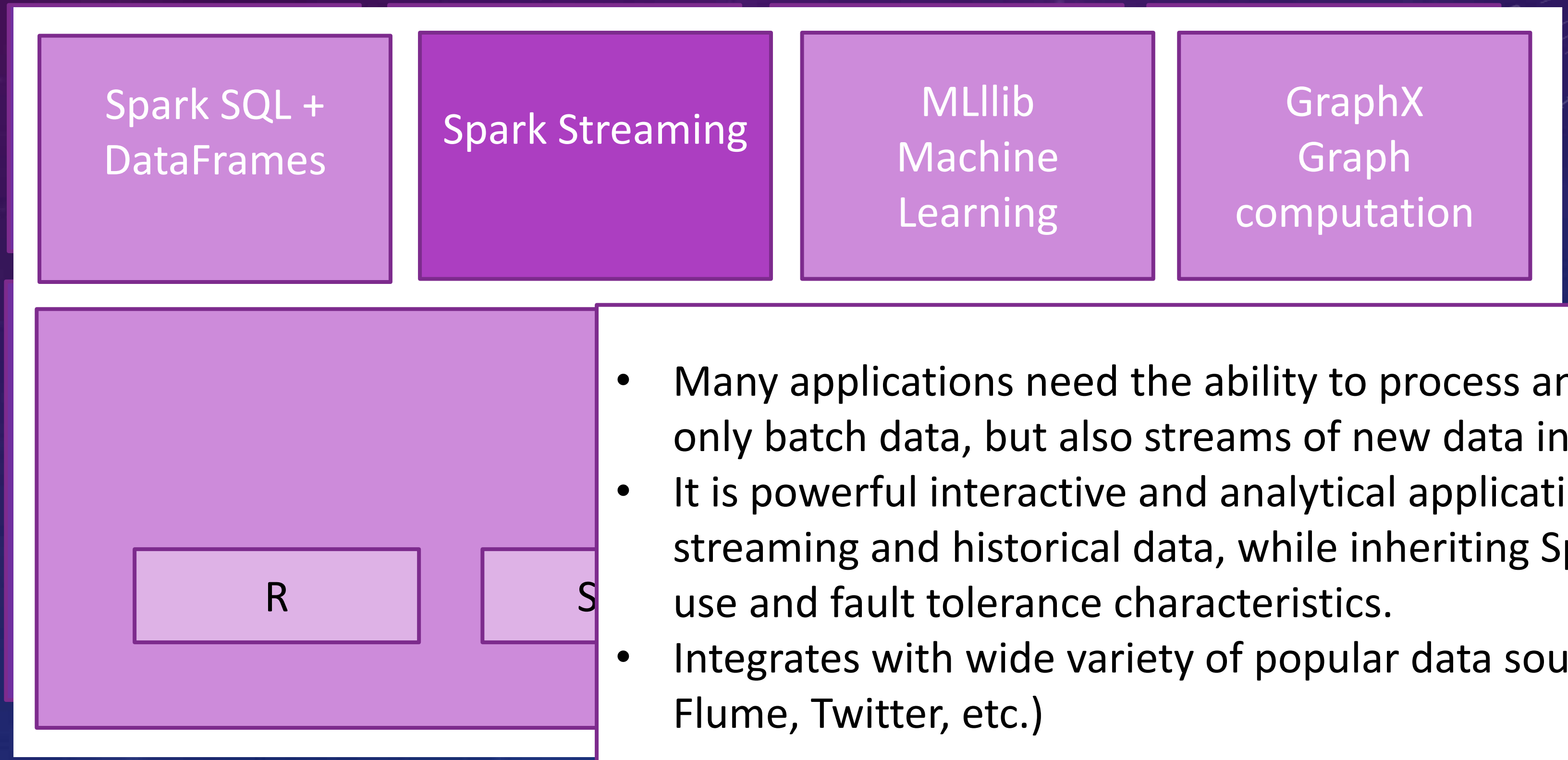


Spark Components



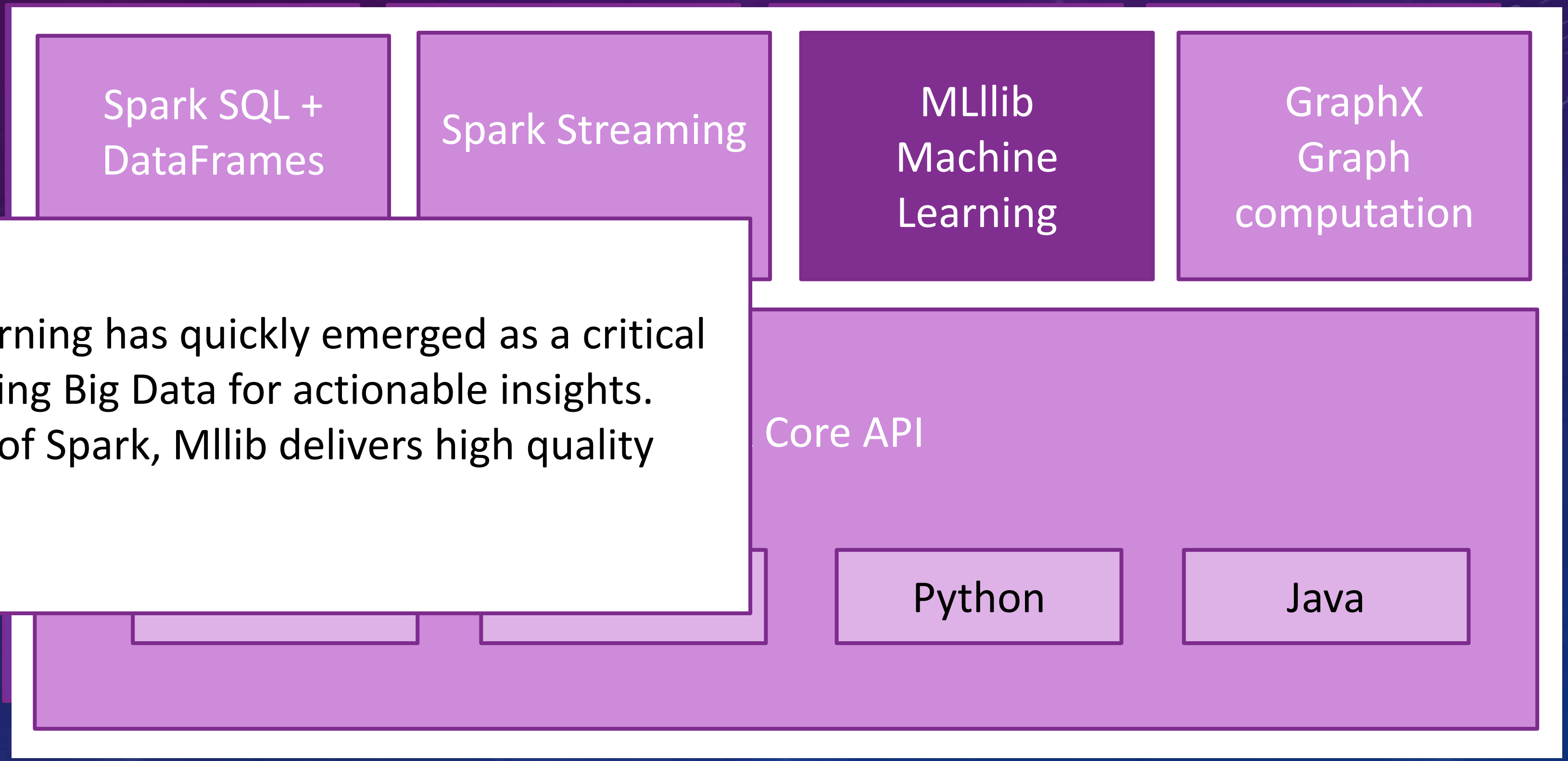
- Spark module for structured data processing.
- Provide programming abstractions called DataFrames and act as a distributed SQL query engine

Spark Components



- Many applications need the ability to process and analyse not only batch data, but also streams of new data in real-time
- It is powerful interactive and analytical applications across both streaming and historical data, while inheriting Spark’s ease of use and fault tolerance characteristics.
- Integrates with wide variety of popular data sources (HDFS, Flume, Twitter, etc.)

Spark Components



Spark SQL + DataFrames

Spark Streaming

MLllib Machine Learning

GraphX Graph computation

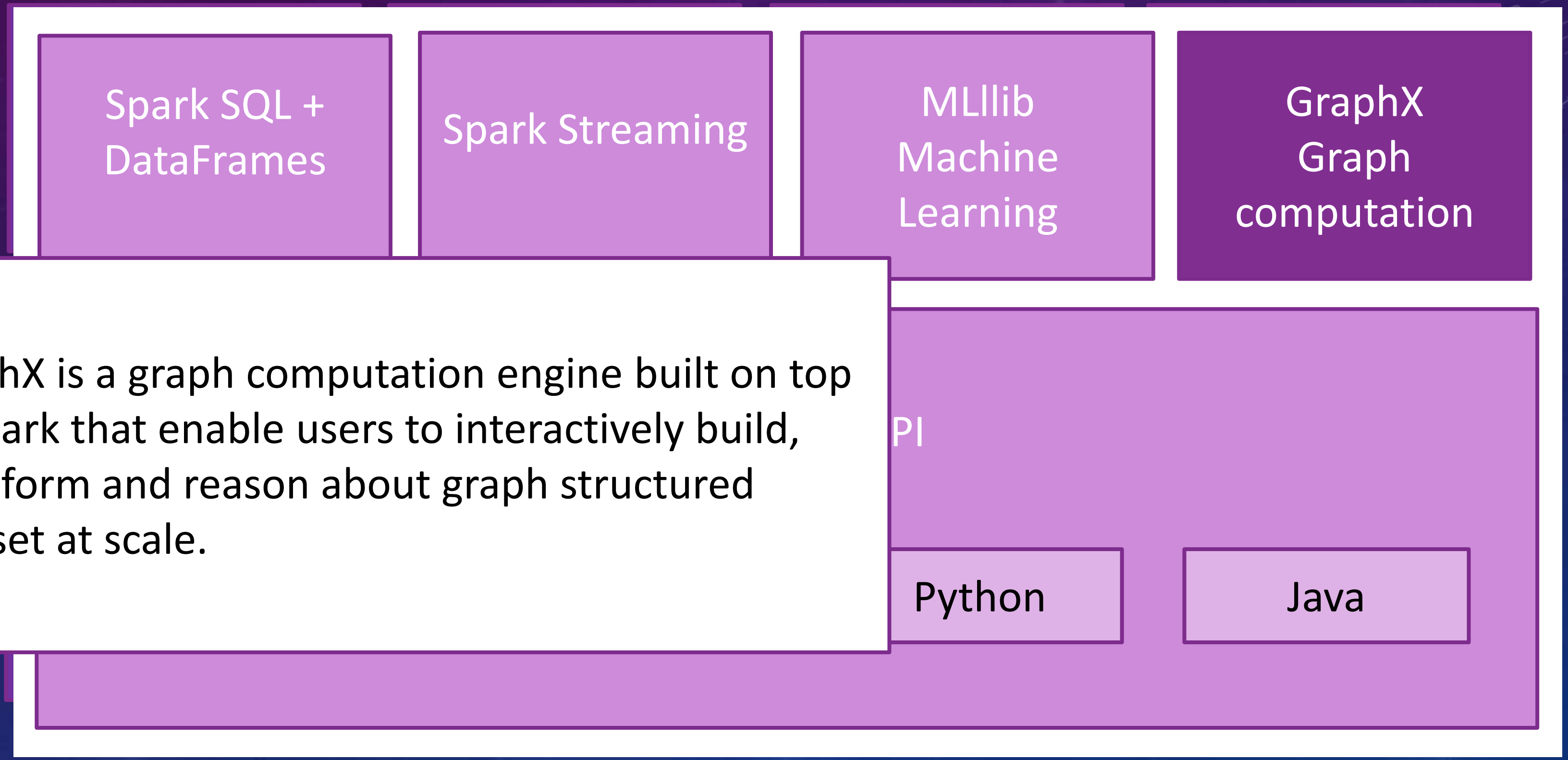
Core API

Python

Java

- Machine learning has quickly emerged as a critical piece in mining Big Data for actionable insights. Built on top of Spark, Mllib delivers high quality algorithms.

Spark Components

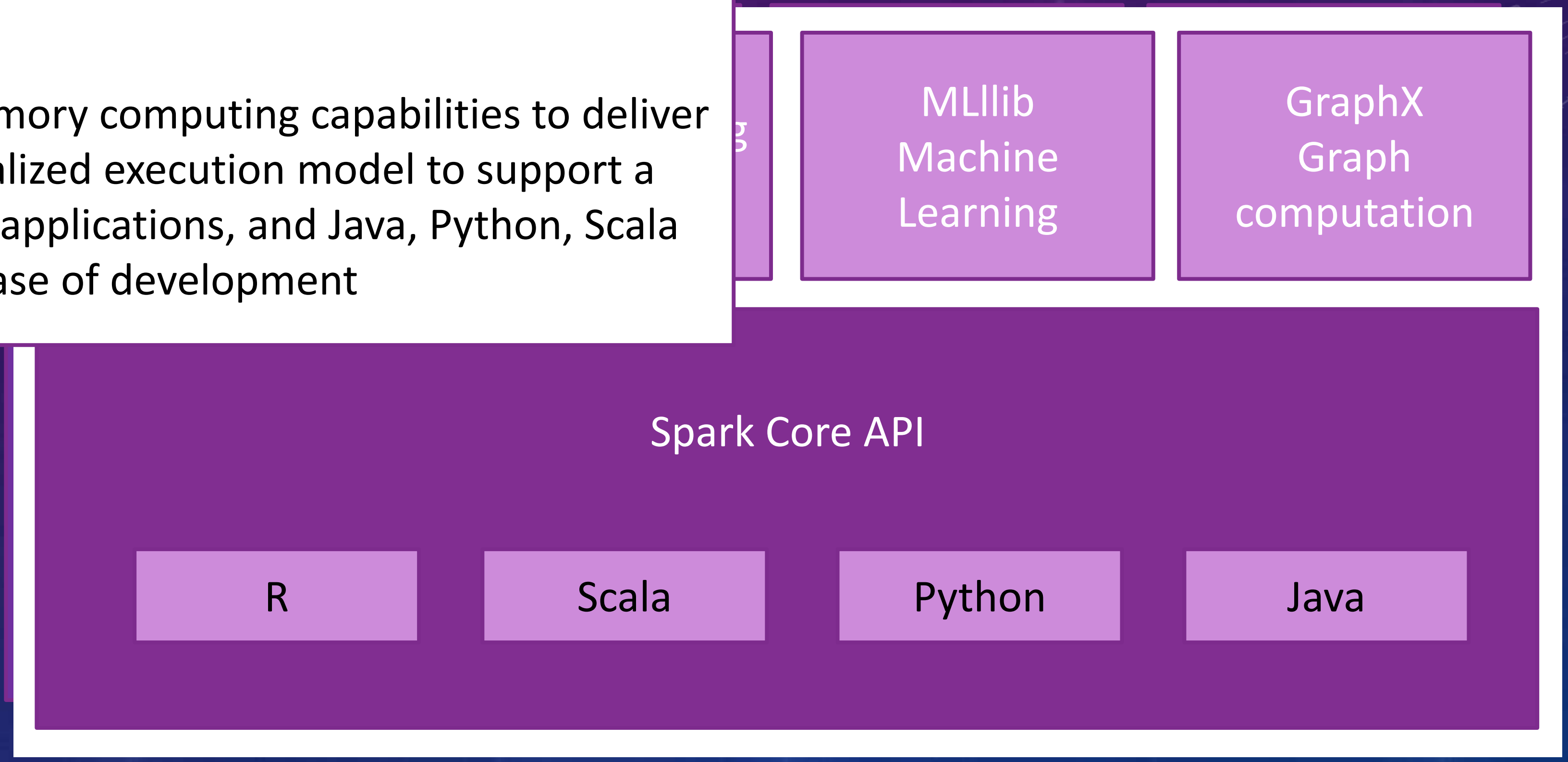


- GraphX is a graph computation engine built on top of Spark that enable users to interactively build, transform and reason about graph structured dataset at scale.

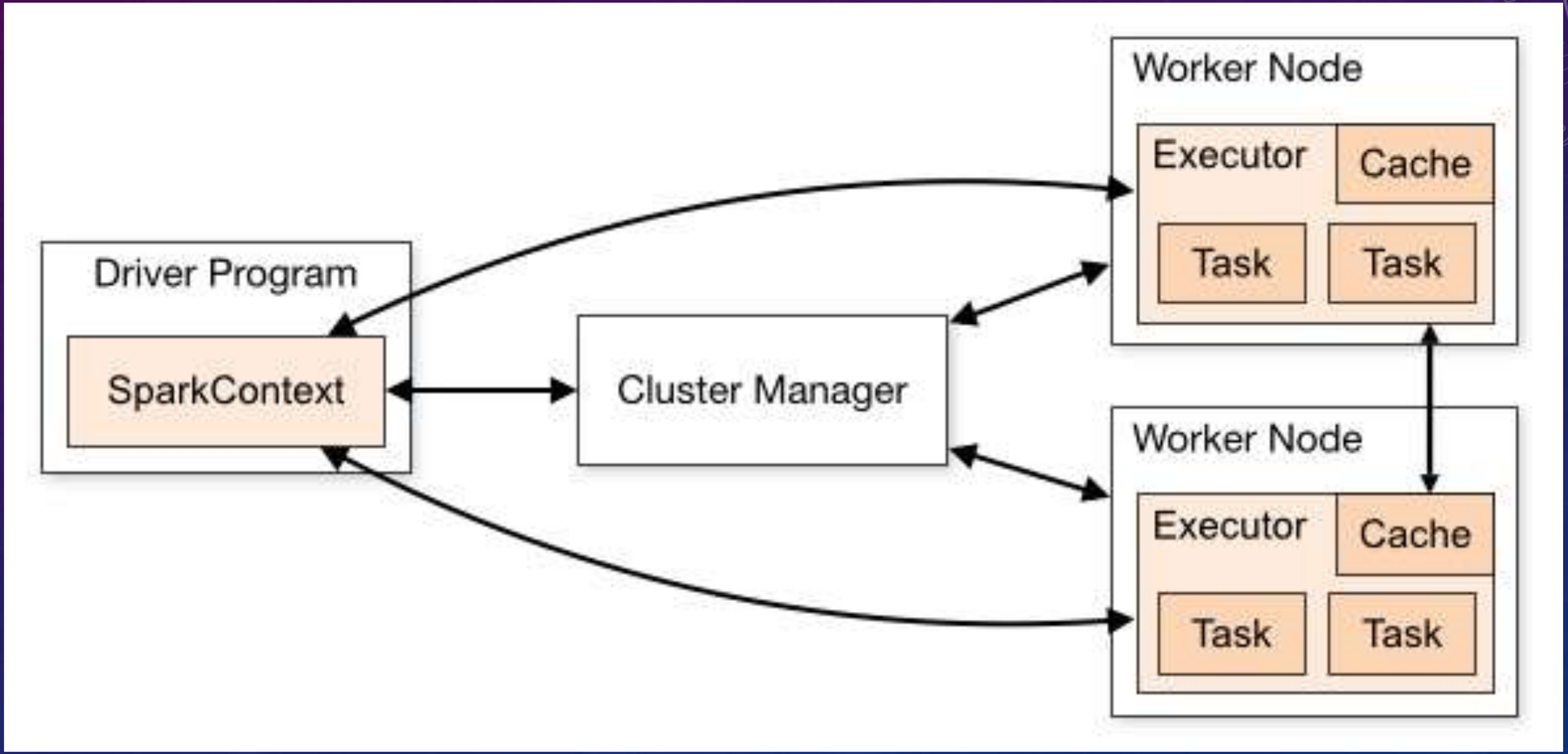
Spark Components

- Spark core is the underlying general execution engine for the Spark platform that all other functionality is built on top of.

- Provides in-memory computing capabilities to deliver speed, a generalized execution model to support a wide variety of applications, and Java, Python, Scala and R API for ease of development



Spark Architecture



CONTEXT

+

Context

- Big Data analytics
- Faster

- Health care data increasing rapidly
- Problems in health care
    - High cost
    - High waste
    - Low quality

Context

+

Health care data volume is increasing and is expected to skyrocket as new ways of collecting health data in various forms are continuously emerging (i.e. patient centered e-health record, and wearables)

- Health care data increasing rapidly
- Problems in health care
  - High cost
  - High waste
  - Low quality

According to AbuKhousa and Campbell, the healthcare system has a massive wealth of information but knowledge poor, " there is a lack of effectual analysis tools to discover knowledge contained in the databases of these systems "

Context

APACHE Spark

+

- Health care data increasing rapidly
- Problems in health care
  - High cost
  - High waste
  - Low quality

Overall spending ($170 billion spend on health in 2015-16)

Context

Poor quality of health (increasing mortality and morbidity rate on preventable diseases)

- Health care data increasing rapidly
- Problems in health care
  - High cost
  - High waste
  - Low quality

PURPOSE

- Understand big data processing in health care
- Learn factors that contribute to heart disease
- Build machine learning model to predict heart disease.
- Use the Spark framework to implement analysis.
- Find significant risk factors of coronary heart disease

DELIVERABLES

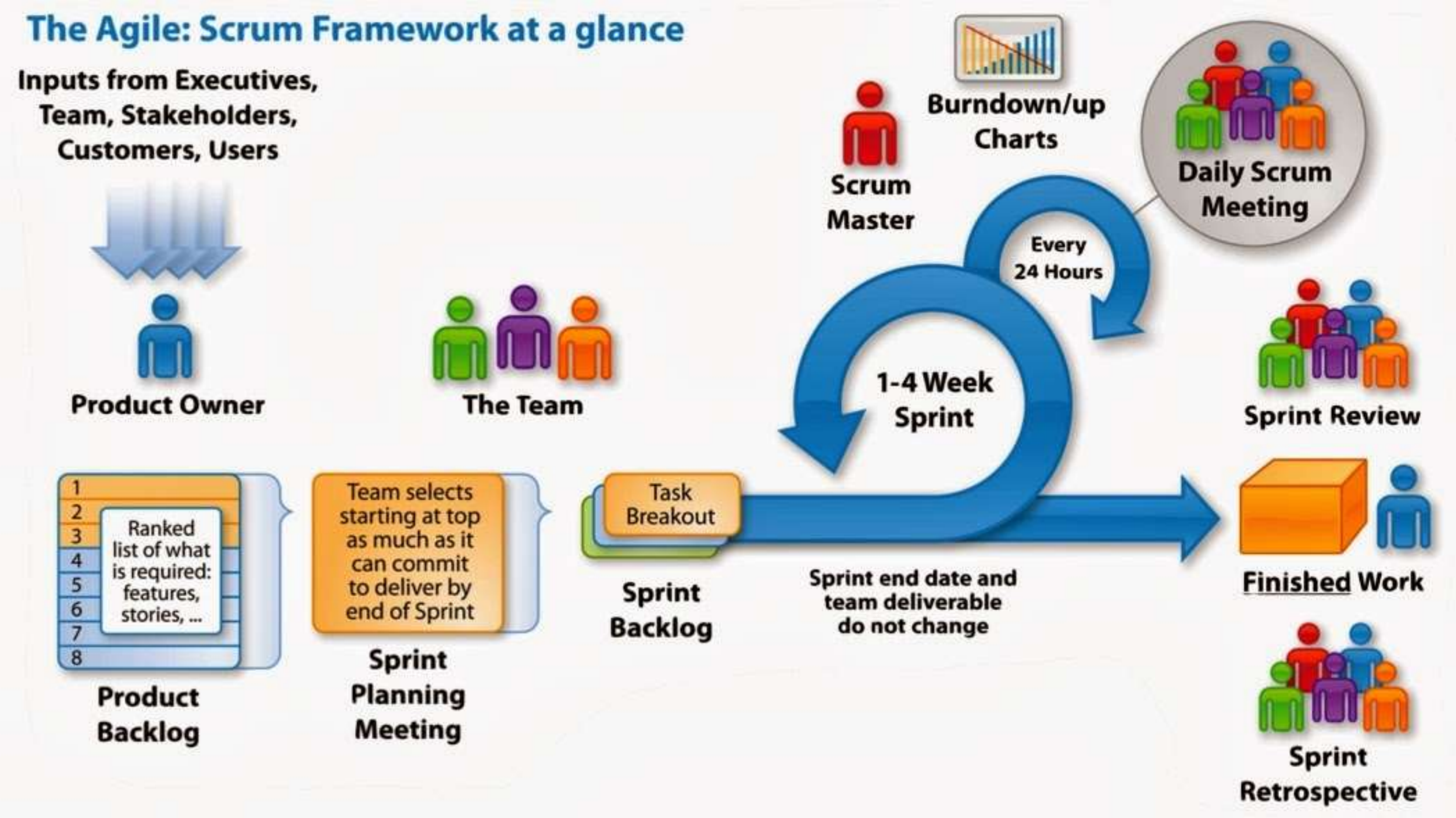
- Build a predictive analytics platform to predict heart disease using Spark's Machine Learning library module in Standalone cluster mode with Mongo DB as the database.

- Determine which features or feature subsets contribute to risk of heart disease.

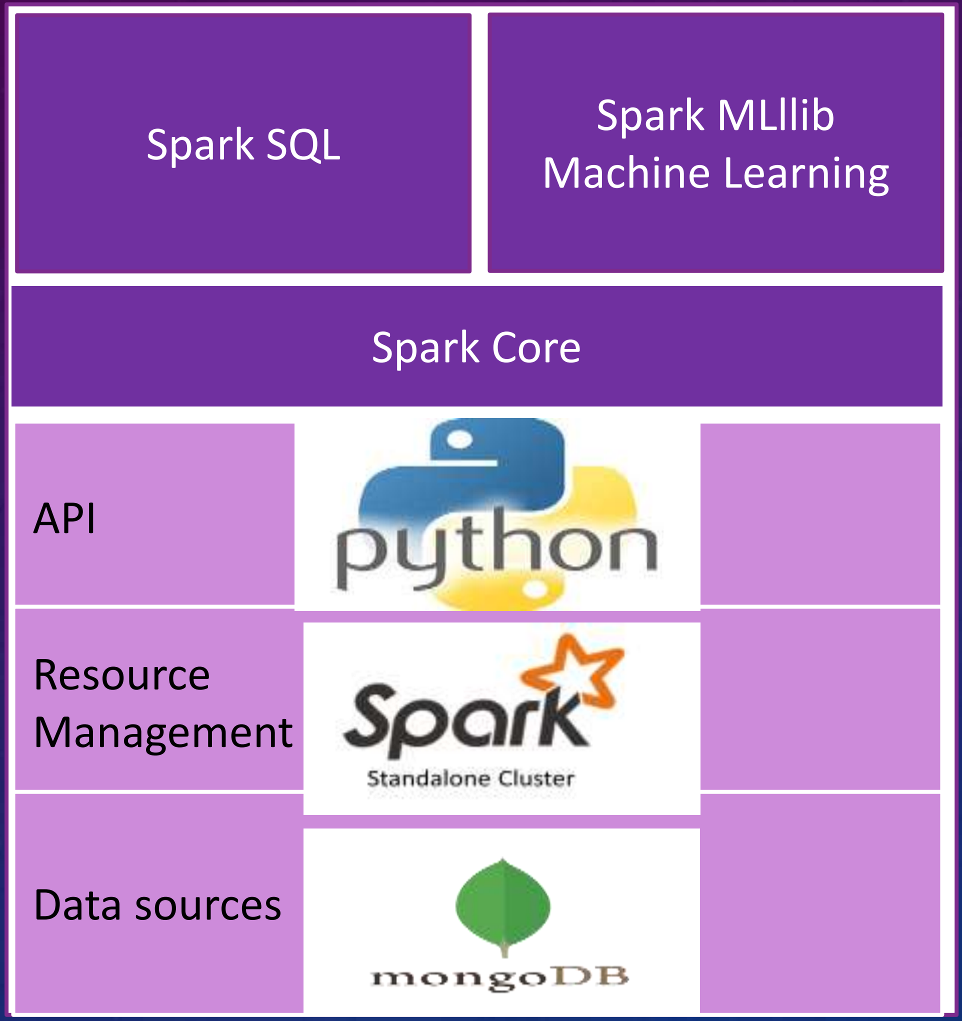
- Analyse result by comparing the ground truth to predicted outcome via graph or table.

OVERVIEW ON THE APPROACH USED

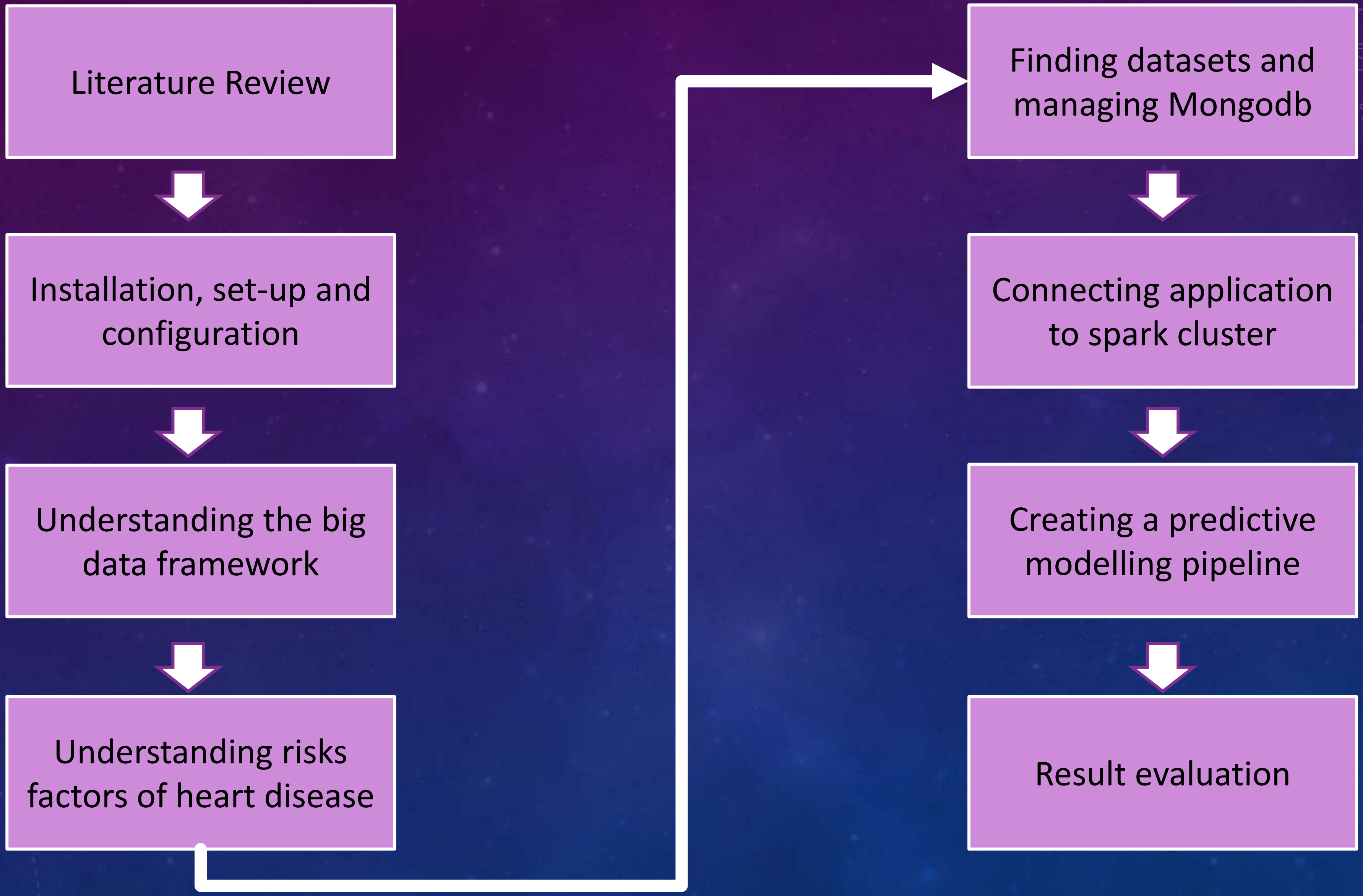
PROJECT MANAGEMENT APPROACH


The Agile: Scrum Framework at a glance
Inputs from Executives, Team, Stakeholders, Customers, Users
Product Owner
The Team
Scrum Master
Burndown/up Charts
Daily Scrum Meeting
Every 24 Hours
1-4 Week Sprint
Sprint Review
1
2
3
4
5
6
7
8
Ranked list of what is required: features, stories, ...
Product Backlog
Team selects starting at top as much as it can commit to deliver by end of Sprint
Sprint Planning Meeting
Task Breakout
Sprint Backlog
Sprint end date and team deliverable do not change
Finished Work
Sprint Retrospective

Project Architecture



Project Approach/Methodology



Literature Review

Installation, set-up and configuration

Understanding the big data framework

Understanding risks factors of heart disease

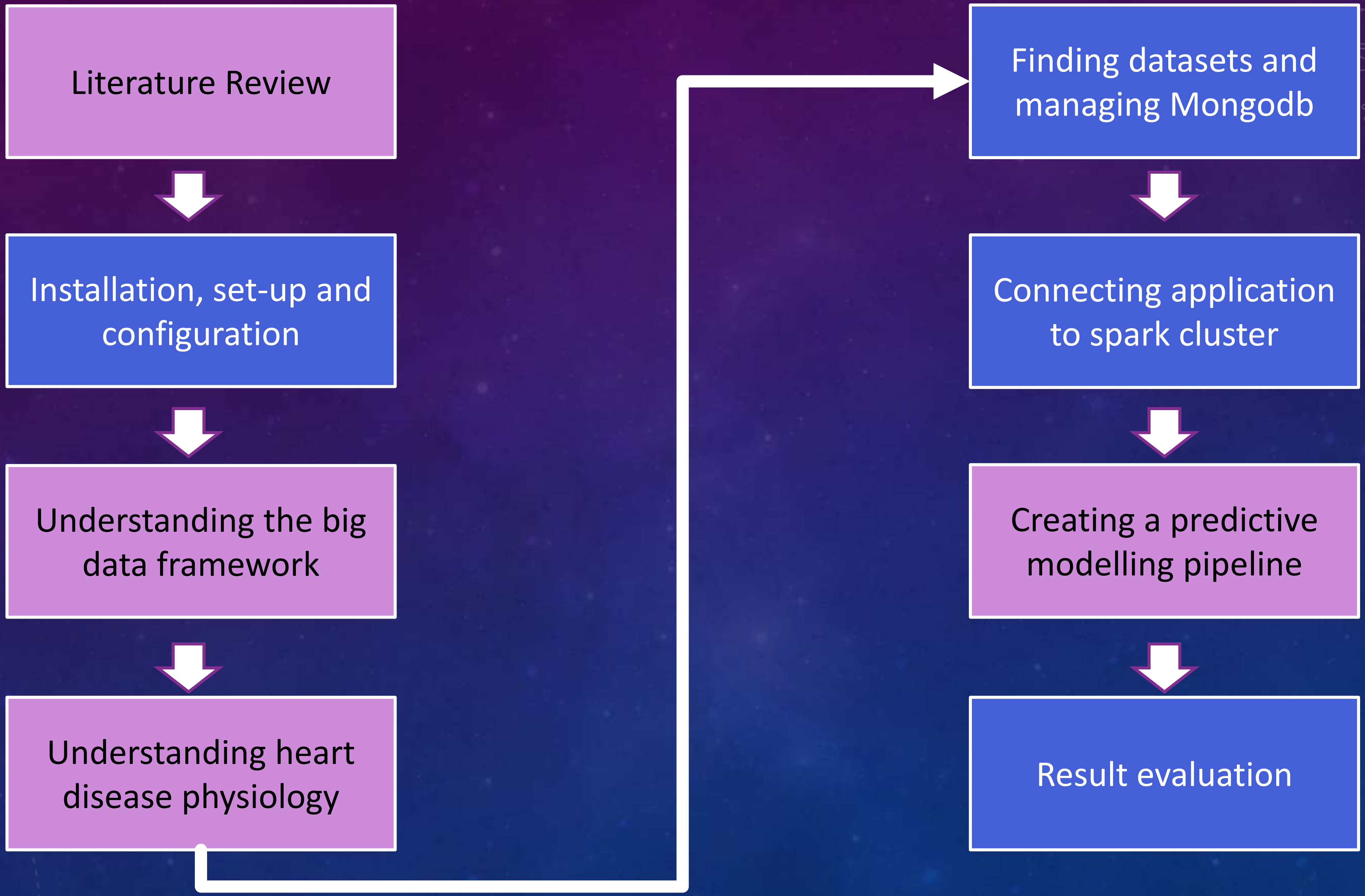
Finding datasets and managing Mongodb

Connecting application to spark cluster

Creating a predictive modelling pipeline

Result evaluation

Project Approach/Methodology



Literature Review

Installation, set-up and configuration

Understanding the big data framework

Understanding heart disease physiology

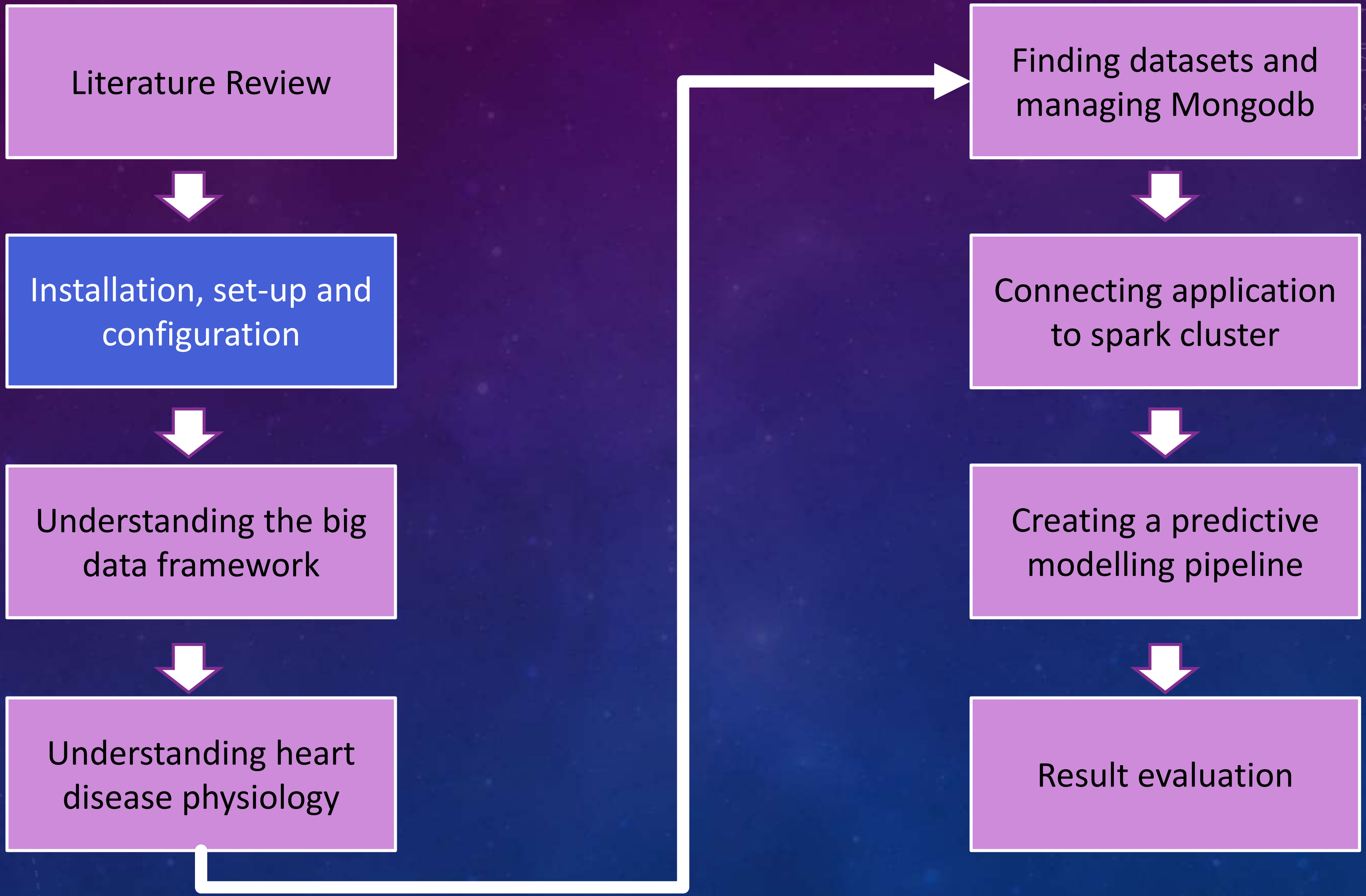
Finding datasets and managing Mongodb

Connecting application to spark cluster

Creating a predictive modelling pipeline

Result evaluation

Project Approach/Methodology



Literature Review

Installation, set-up and configuration

Understanding the big data framework

Understanding heart disease physiology

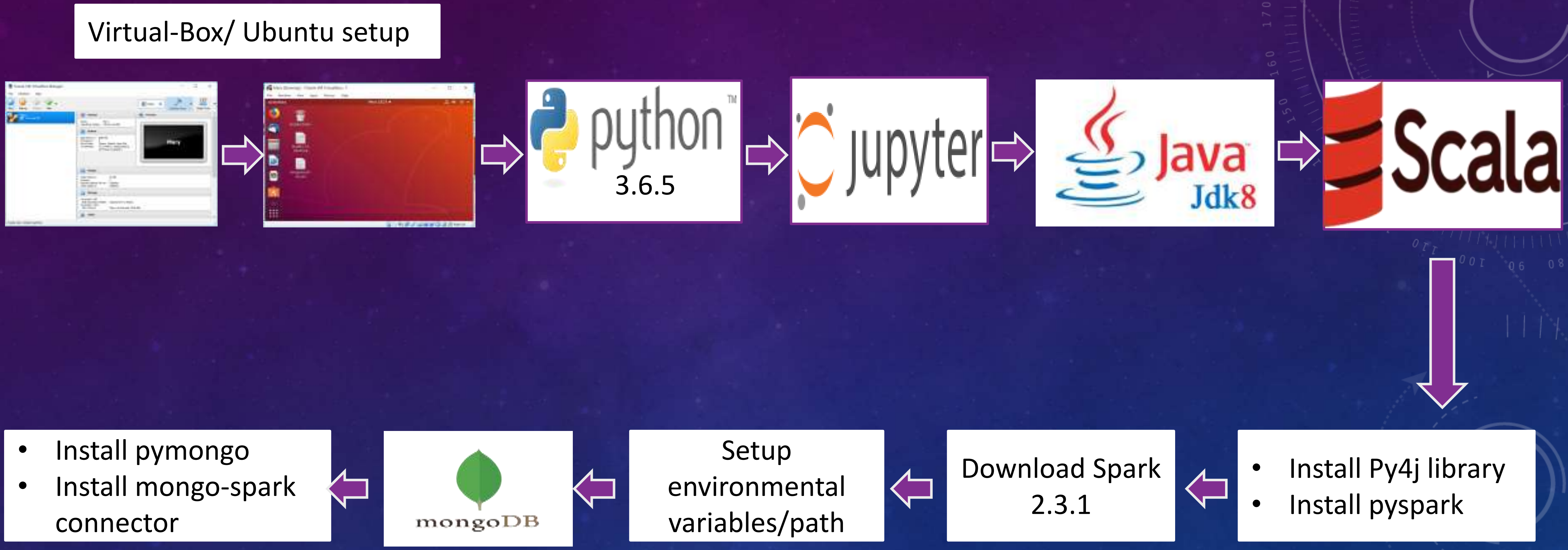
Finding datasets and managing Mongodb

Connecting application to spark cluster

Creating a predictive modelling pipeline

Result evaluation

Installation, set-up and configuration of machine and tools



Virtual-Box/ Ubuntu setup

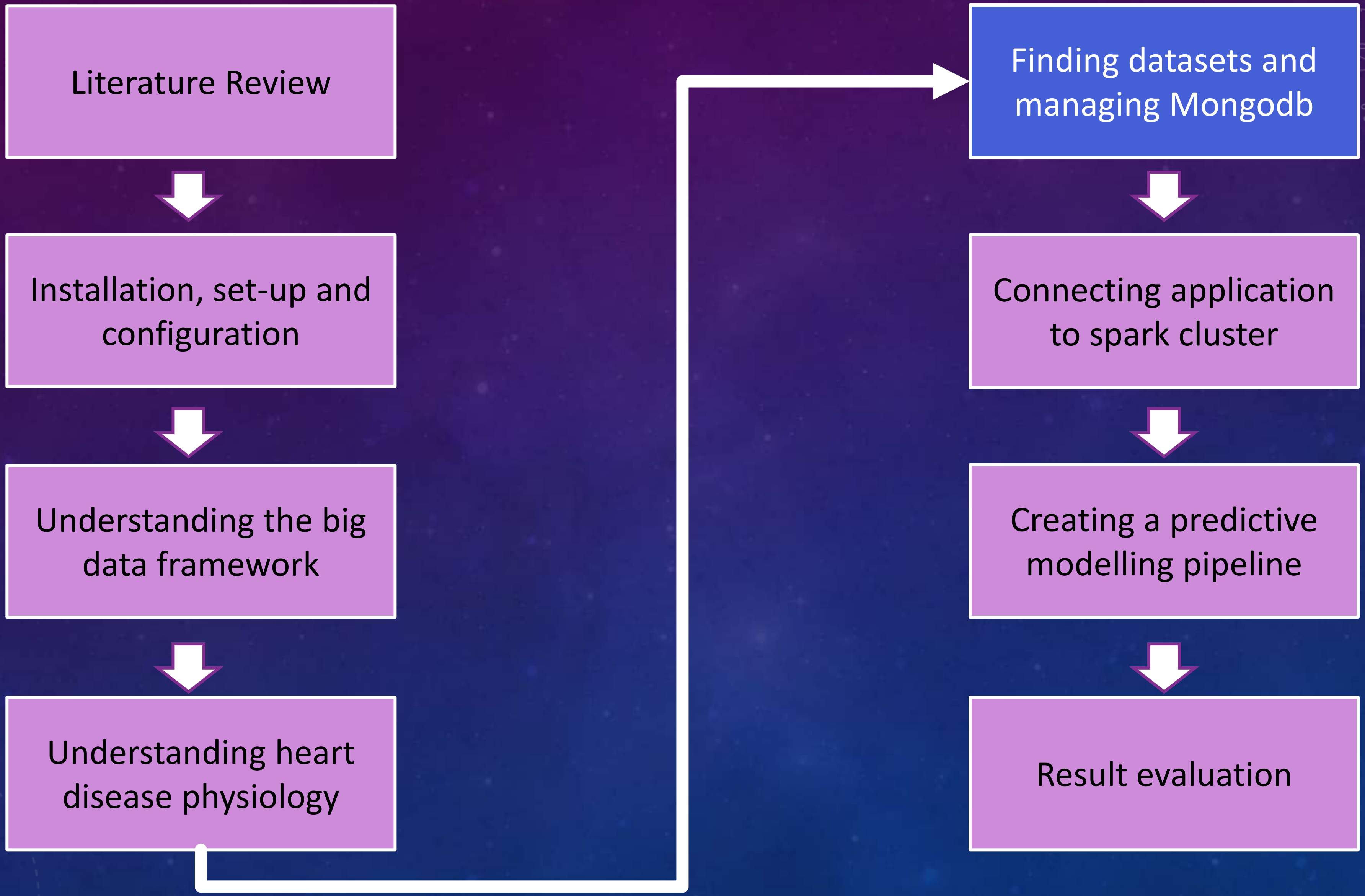
- Install Py4j library
- Install pyspark

Download Spark 2.3.1

Setup environmental variables/path

- Install pymongo
- Install mongo-spark connector

Project Approach/Methodology



Literature Review

Installation, set-up and configuration

Understanding the big data framework

Understanding heart disease physiology

Finding datasets and managing Mongodb

Connecting application to spark cluster

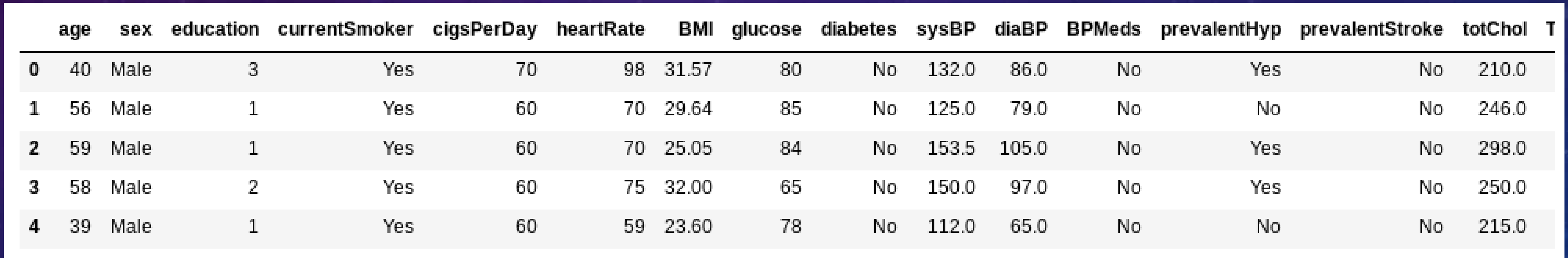
Creating a predictive modelling pipeline

Result evaluation

Project Approach/Methodology: Datasets and MongoDB

FRAMINGHAM HEART STUDY
Three Generations of Dedication
1948
1971
2002
A Project of the National Heart, Lung, and Blood Institute and Boston University

Framingham Heart Study datasets



	age	sex	education	currentSmoker	cigsPerDay	heartRate	BMI	glucose	diabetes	sysBP	diaBP	BPMeds	prevalentHyp	prevalentStroke	totChol	T
0	40	Male	3	Yes	70	98	31.57	80	No	132.0	86.0	No	Yes	No	210.0	
1	56	Male	1	Yes	60	70	29.64	85	No	125.0	79.0	No	No	No	246.0	
2	59	Male	1	Yes	60	70	25.05	84	No	153.5	105.0	No	Yes	No	298.0	
3	58	Male	2	Yes	60	75	32.00	65	No	150.0	97.0	No	Yes	No	250.0	
4	39	Male	1	Yes	60	59	23.60	78	No	112.0	65.0	No	No	No	215.0	

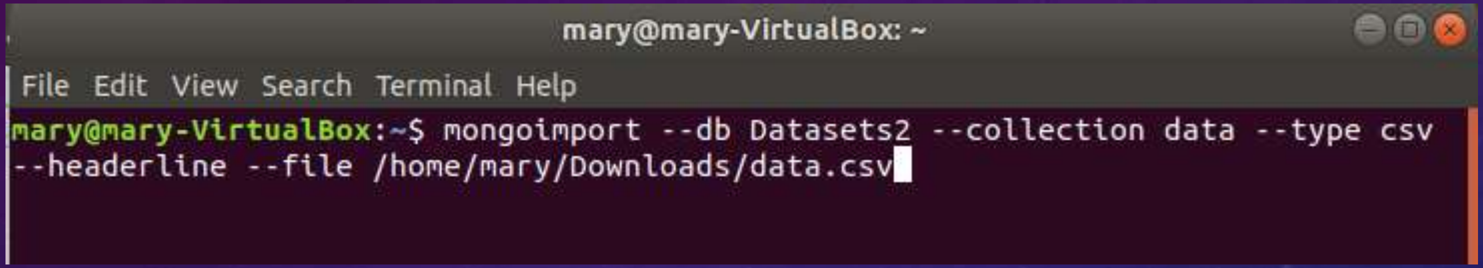
- csv format
- 4241 records

Project Approach/Methodology: Datasets and MongoDB

Framingham Heart Study datasets

Categorical variables	Numeric/continuous variables
Sex	age
Education	cigsPerDay
currentSmoker	heartrate
Diabetes	BMI
BPMeds	Glucose
prevalentHyp	sysBP
prevalentStroke	diaBP
label	totChol

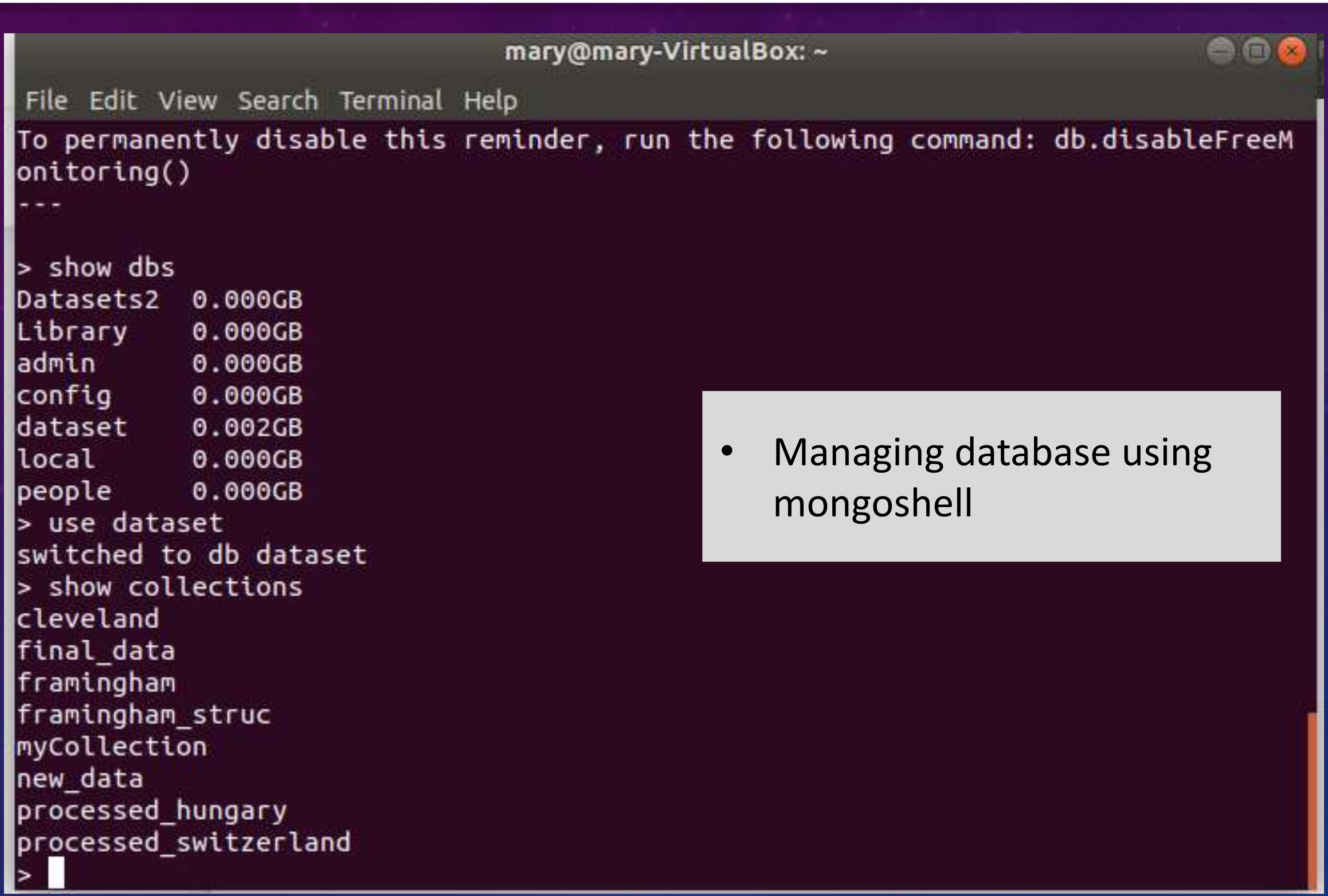
Project Approach/Methodology: Managing MongoDB



```
mary@mary-VirtualBox:~$ mongoimport --db Datasets2 --collection data --type csv
--headerline --file /home/mary/Downloads/data.csv
```

- Importing datasets

Project Approach/Methodology: Managing MongoDB

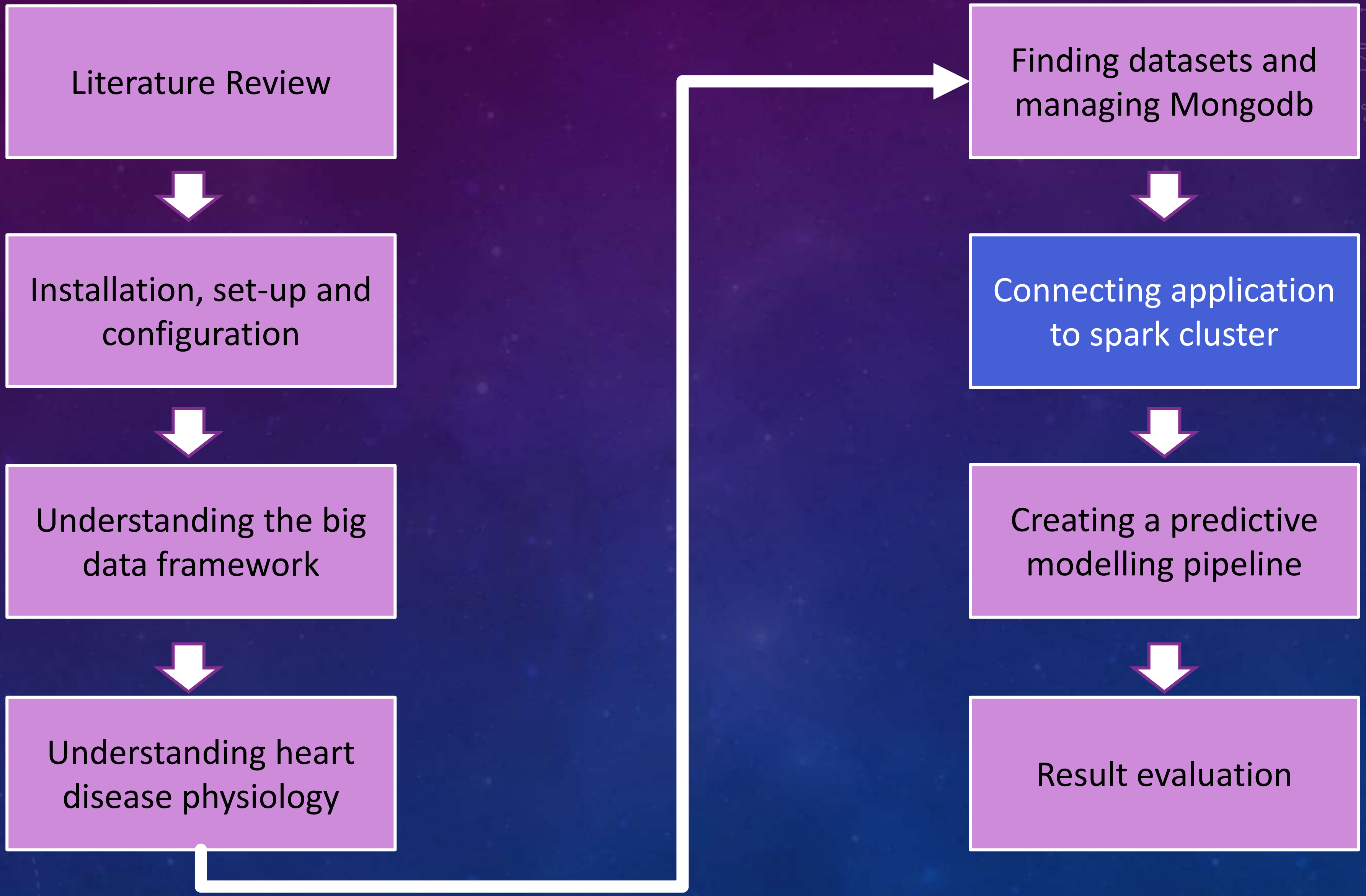


```
To permanently disable this reminder, run the following command: db.disableFreeM
onitoring()
---

> show dbs
Datasets2  0.000GB
Library    0.000GB
admin      0.000GB
config     0.000GB
dataset    0.002GB
local      0.000GB
people     0.000GB
> use dataset
switched to db dataset
> show collections
cleveland
final_data
framingham
framingham_struc
myCollection
new_data
processed_hungary
processed_switzerland
> 
```

- Managing database using mongoshell

Project Approach/Methodology



Literature Review

Installation, set-up and configuration

Understanding the big data framework

Understanding heart disease physiology

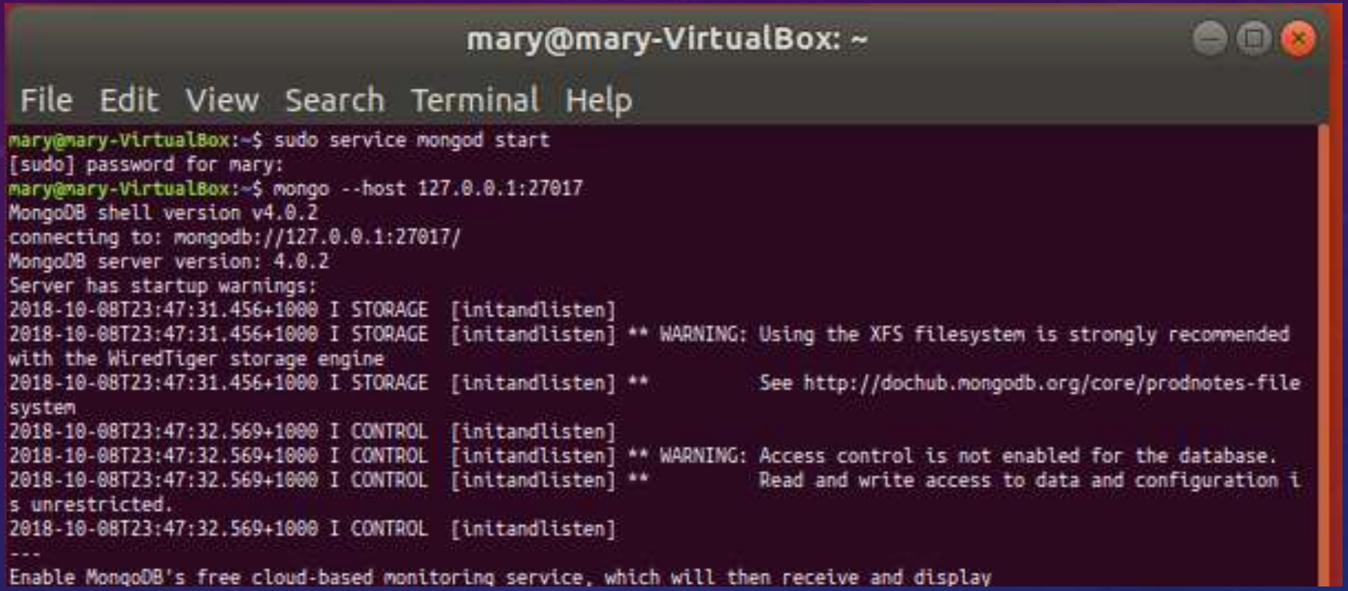
Finding datasets and managing Mongodb

Connecting application to spark cluster

Creating a predictive modelling pipeline

Result evaluation

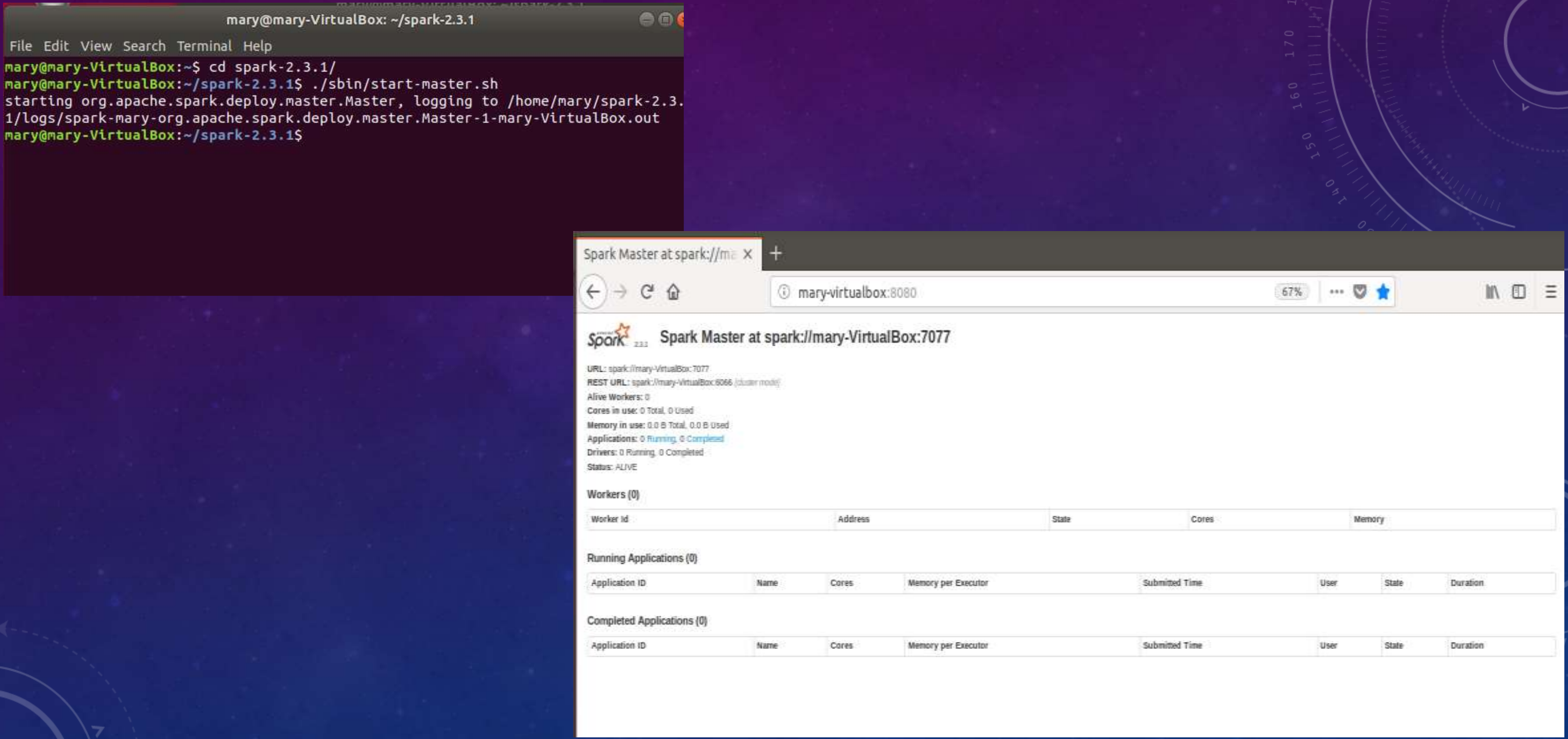
Deployment stage: Starting Mongodb service



```
mary@mary-VirtualBox:~$ sudo service mongod start
[sudo] password for mary:
mary@mary-VirtualBox:~$ mongo --host 127.0.0.1:27017
MongoDB shell version v4.0.2
connecting to: mongodb://127.0.0.1:27017/
MongoDB server version: 4.0.2
Server has startup warnings:
2018-10-08T23:47:31.456+1000 I STORAGE  [initandlisten]
2018-10-08T23:47:31.456+1000 I STORAGE  [initandlisten] ** WARNING: Using the XFS filesystem is strongly recommended
with the WiredTiger storage engine
2018-10-08T23:47:31.456+1000 I STORAGE  [initandlisten] **          See http://dochub.mongodb.org/core/prodnotes-file
system
2018-10-08T23:47:32.569+1000 I CONTROL  [initandlisten]
2018-10-08T23:47:32.569+1000 I CONTROL  [initandlisten] ** WARNING: Access control is not enabled for the database.
2018-10-08T23:47:32.569+1000 I CONTROL  [initandlisten] **          Read and write access to data and configuration i
s unrestricted.
2018-10-08T23:47:32.569+1000 I CONTROL  [initandlisten]
---
Enable MongoDB's free cloud-based monitoring service, which will then receive and display
```

Starting Mongodb service

Deployment stage : starting the spark master cluster



mary@mary-VirtualBox: ~/spark-2.3.1
File Edit View Search Terminal Help

```
mary@mary-VirtualBox:~$ cd spark-2.3.1/
mary@mary-VirtualBox:~/spark-2.3.1$ ./sbin/start-master.sh
starting org.apache.spark.deploy.master.Master, logging to /home/mary/spark-2.3.
1/logs/spark-mary-org.apache.spark.deploy.master.Master-1-mary-VirtualBox.out
mary@mary-VirtualBox:~/spark-2.3.1$
```

Spark Master at spark://ma
mary-virtualbox:8080
67%
Spark 2.3.1

Spark Master at spark://mary-VirtualBox:7077

URL: spark://mary-VirtualBox:7077
REST URL: spark://mary-VirtualBox:6066 (cluster mode)
Alive Workers: 0
Cores in use: 0 Total, 0 Used
Memory in use: 0.0 B Total, 0.0 B Used
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (0)

Worker Id	Address	State	Cores	Memory

Running Applications (0)

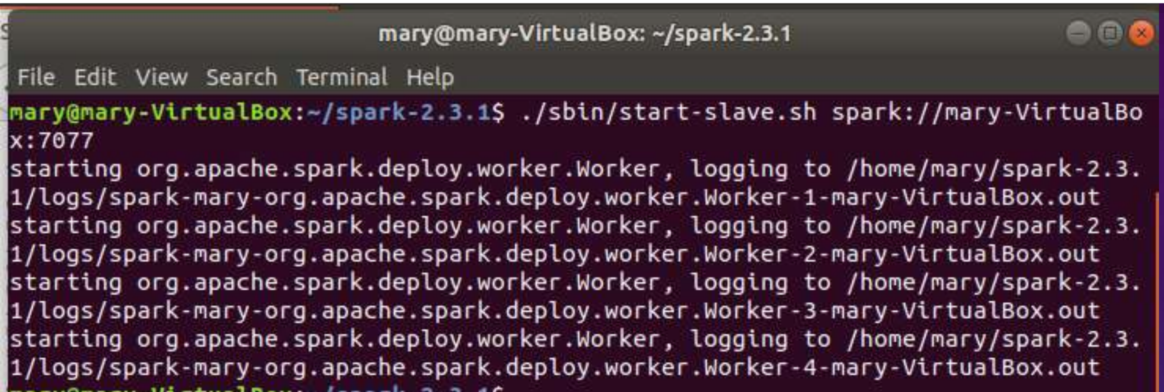
Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration

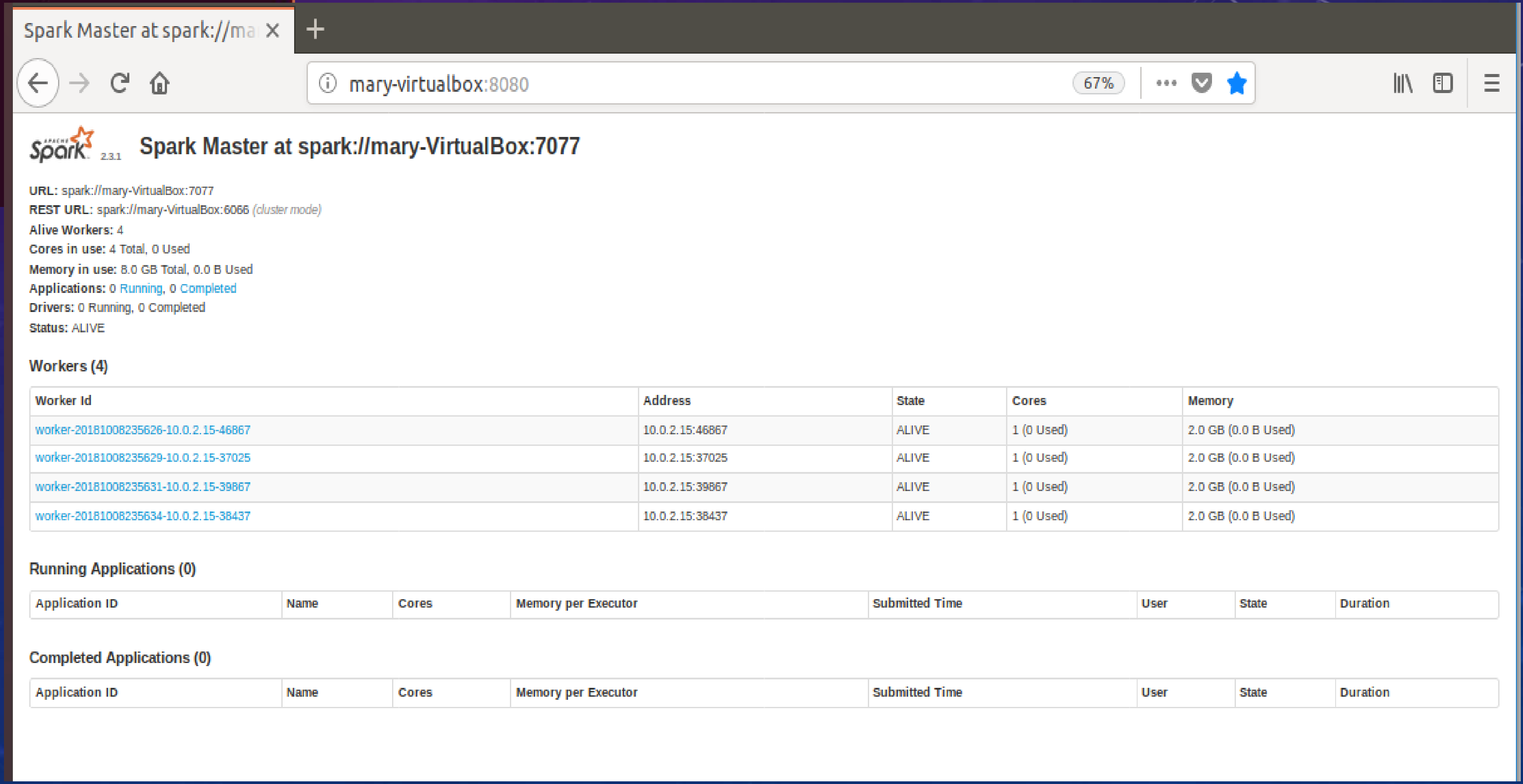
Deployment stage: starting 4 worker nodes

Starting the Spark cluster



mary@mary-VirtualBox: ~/spark-2.3.1
File Edit View Search Terminal Help

```
mary@mary-VirtualBox:~/spark-2.3.1$ ./sbin/start-slave.sh spark://mary-VirtualBo
x:7077
starting org.apache.spark.deploy.worker.Worker, logging to /home/mary/spark-2.3.
1/logs/spark-mary-org.apache.spark.deploy.worker.Worker-1-mary-VirtualBox.out
starting org.apache.spark.deploy.worker.Worker, logging to /home/mary/spark-2.3.
1/logs/spark-mary-org.apache.spark.deploy.worker.Worker-2-mary-VirtualBox.out
starting org.apache.spark.deploy.worker.Worker, logging to /home/mary/spark-2.3.
1/logs/spark-mary-org.apache.spark.deploy.worker.Worker-3-mary-VirtualBox.out
starting org.apache.spark.deploy.worker.Worker, logging to /home/mary/spark-2.3.
1/logs/spark-mary-org.apache.spark.deploy.worker.Worker-4-mary-VirtualBox.out
mary@mary-VirtualBox:~/spark-2.3.1$
mary@mary-VirtualBox:~/spark-2.3.1$
mary@mary-VirtualBox:~/spark-2.3.1$
mary@mary-VirtualBox:~/spark-2.3.1$
mary@mary-VirtualBox:~/spark-2.3.1$
mary@mary-VirtualBox:~/spark-2.3.1$
mary@mary-VirtualBox:~/spark-2.3.1$
mary@mary-VirtualBox:~/spark-2.3.1$
mary@mary-VirtualBox:~/spark-2.3.1$
```



Spark Master at spark://mary-VirtualBox:7077

URL: spark://mary-VirtualBox:7077
REST URL: spark://mary-VirtualBox:6066 *(cluster mode)*
Alive Workers: 4
Cores in use: 4 Total, 0 Used
Memory in use: 8.0 GB Total, 0.0 B Used
Applications: 0 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (4)

Worker Id	Address	State	Cores	Memory
worker-20181008235626-10.0.2.15-46867	10.0.2.15:46867	ALIVE	1 (0 Used)	2.0 GB (0.0 B Used)
worker-20181008235629-10.0.2.15-37025	10.0.2.15:37025	ALIVE	1 (0 Used)	2.0 GB (0.0 B Used)
worker-20181008235631-10.0.2.15-39867	10.0.2.15:39867	ALIVE	1 (0 Used)	2.0 GB (0.0 B Used)
worker-20181008235634-10.0.2.15-38437	10.0.2.15:38437	ALIVE	1 (0 Used)	2.0 GB (0.0 B Used)

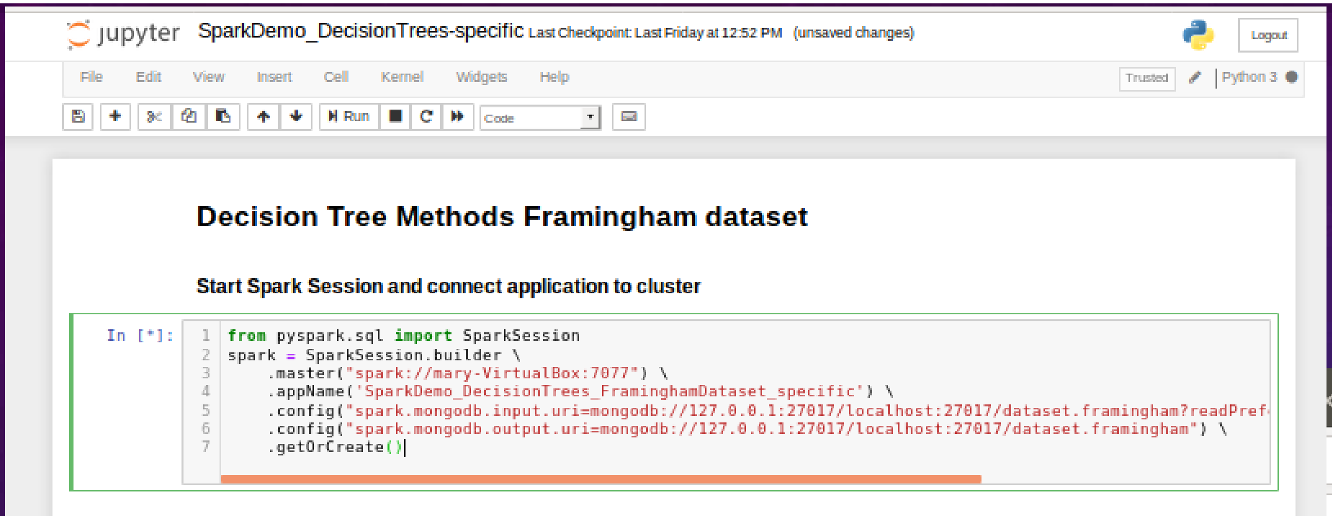
Running Applications (0)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration

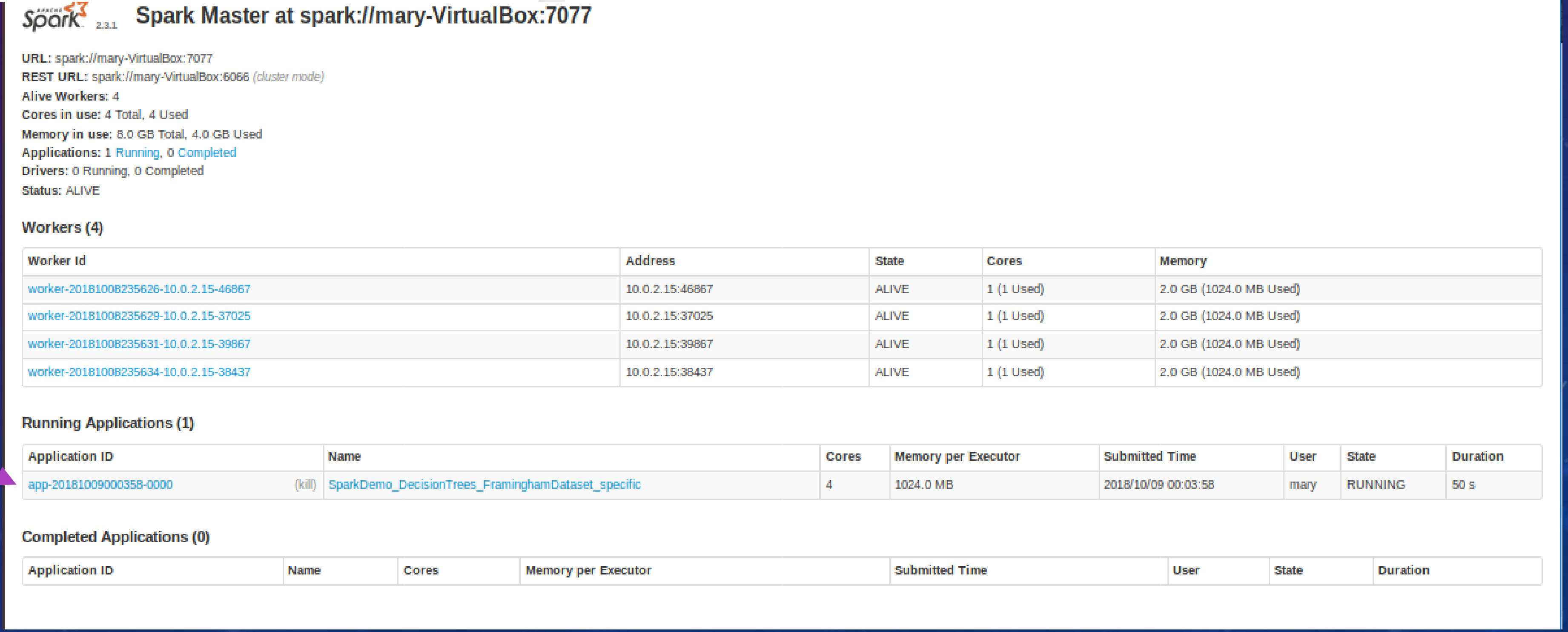
Deployment stage: starting 4 worker nodes


jupyter
SparkDemo_DecisionTrees-specific Last Checkpoint: Last Friday at 12:52 PM (unsaved changes)
Logout
File
Edit
View
Insert
Cell
Kernel
Widgets
Help
Trusted
Python 3
Run
Code

Decision Tree Methods Framingham dataset

Start Spark Session and connect application to cluster

```
In [*]:
from pyspark.sql import SparkSession
spark = SparkSession.builder \
    .master("spark://mary-VirtualBox:7077") \
    .appName('SparkDemo_DecisionTrees_FraminghamDataset_specific') \
    .config("spark.mongodb.input.uri=mongodb://127.0.0.1:27017/localhost:27017/dataset.framingham?readPref
    .config("spark.mongodb.output.uri=mongodb://127.0.0.1:27017/localhost:27017/dataset.framingham") \
    .getOrCreate()
```



Spark Master at spark://mary-VirtualBox:7077

URL: spark://mary-VirtualBox:7077
REST URL: spark://mary-VirtualBox:6066 *(cluster mode)*
Alive Workers: 4
Cores in use: 4 Total, 4 Used
Memory in use: 8.0 GB Total, 4.0 GB Used
Applications: 1 Running, 0 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

Workers (4)

Worker Id	Address	State	Cores	Memory
worker-20181008235626-10.0.2.15-46867	10.0.2.15:46867	ALIVE	1 (1 Used)	2.0 GB (1024.0 MB Used)
worker-20181008235629-10.0.2.15-37025	10.0.2.15:37025	ALIVE	1 (1 Used)	2.0 GB (1024.0 MB Used)
worker-20181008235631-10.0.2.15-39867	10.0.2.15:39867	ALIVE	1 (1 Used)	2.0 GB (1024.0 MB Used)
worker-20181008235634-10.0.2.15-38437	10.0.2.15:38437	ALIVE	1 (1 Used)	2.0 GB (1024.0 MB Used)

Running Applications (1)

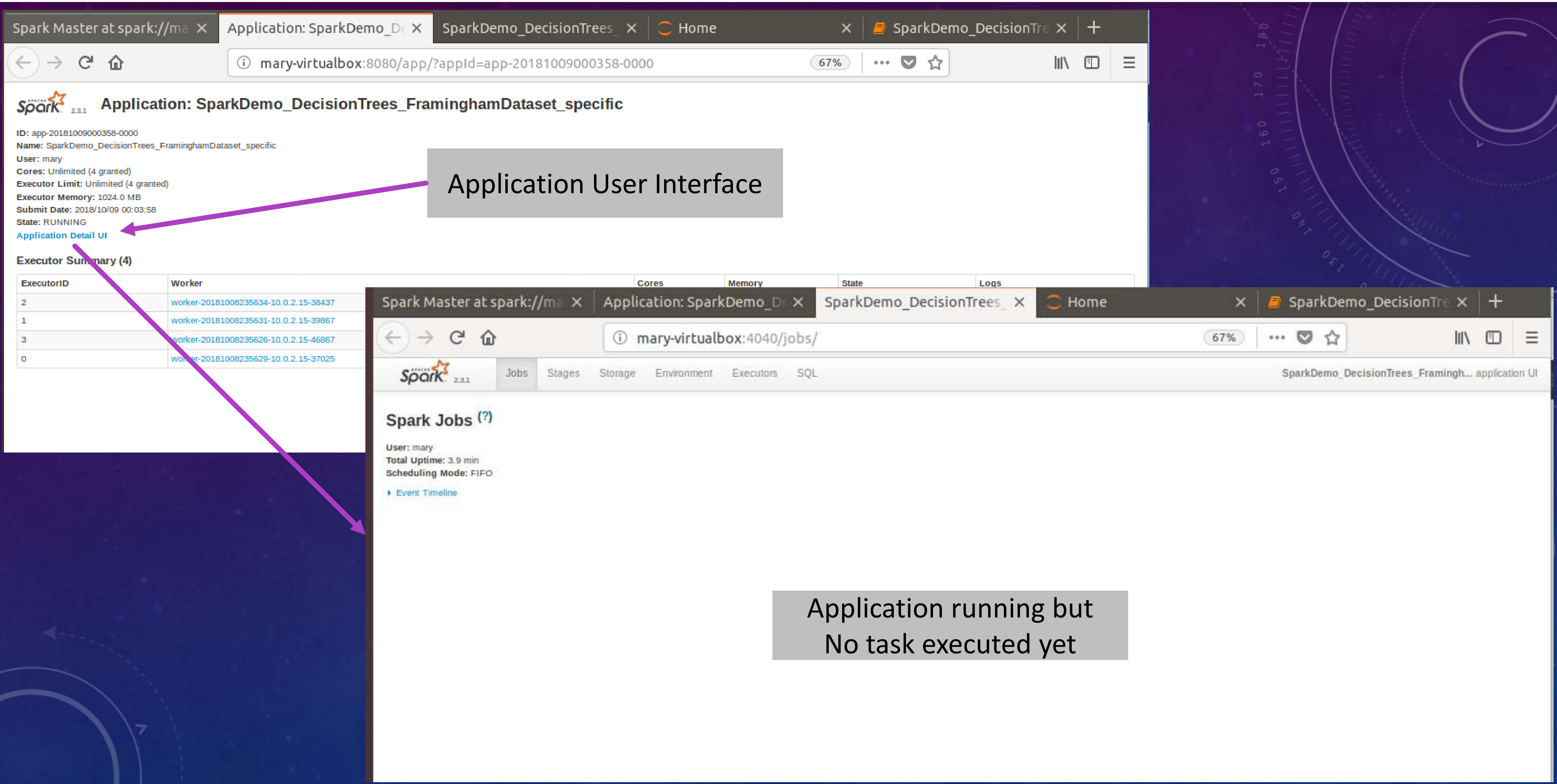
Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration
app-20181009000358-0000 (kill)	SparkDemo_DecisionTrees_FraminghamDataset_specific	4	1024.0 MB	2018/10/09 00:03:58	mary	RUNNING	50 s

Completed Applications (0)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration

Application running

Deployment stage:


Spark Master at spark://ma
Application: SparkDemo_De
SparkDemo_DecisionTrees_
Home
SparkDemo_DecisionTre
mary-virtualbox:8080/app/?appId=app-20181009000358-0000
67%
Spark 2.3.1

Application: SparkDemo_DecisionTrees_FraminghamDataset_specific

ID: app-20181009000358-0000
Name: SparkDemo_DecisionTrees_FraminghamDataset_specific
User: mary
Cores: Unlimited (4 granted)
Executor Limit: Unlimited (4 granted)
Executor Memory: 1024.0 MB
Submit Date: 2018/10/09 00:03:58
State: RUNNING
Application Detail UI

Application User Interface

Executor Summary (4)

ExecutorID	Worker	Cores	Memory	State	Logs
2	worker-20181008235634-10.0.2.15-38437				
1	worker-20181008235631-10.0.2.15-39867				
3	worker-20181008235626-10.0.2.15-46867				
0	worker-20181008235629-10.0.2.15-37025				

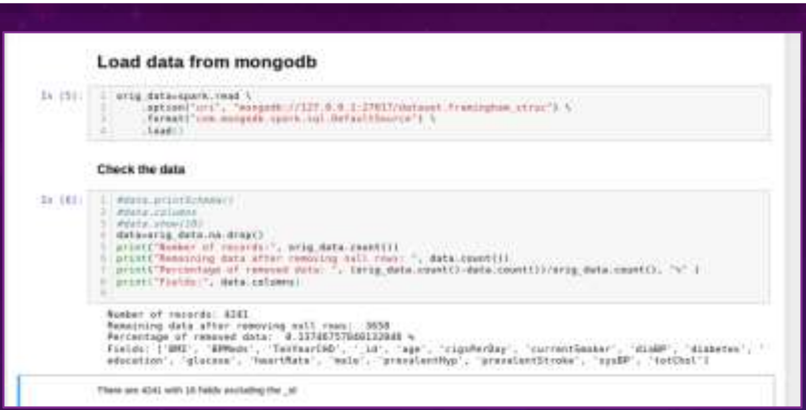
Spark Master at spark://ma
Application: SparkDemo_De
SparkDemo_DecisionTrees_
Home
SparkDemo_DecisionTre
mary-virtualbox:4040/jobs/
67%
Spark 2.3.1
Jobs
Stages
Storage
Environment
Executors
SQL
SparkDemo_DecisionTrees_Framingh... application UI

Spark Jobs (?)

User: mary
Total Uptime: 3.9 min
Scheduling Mode: FIFO
▸ Event Timeline

Application running but
No task executed yet

Deployment stage:



Load data from mongodb

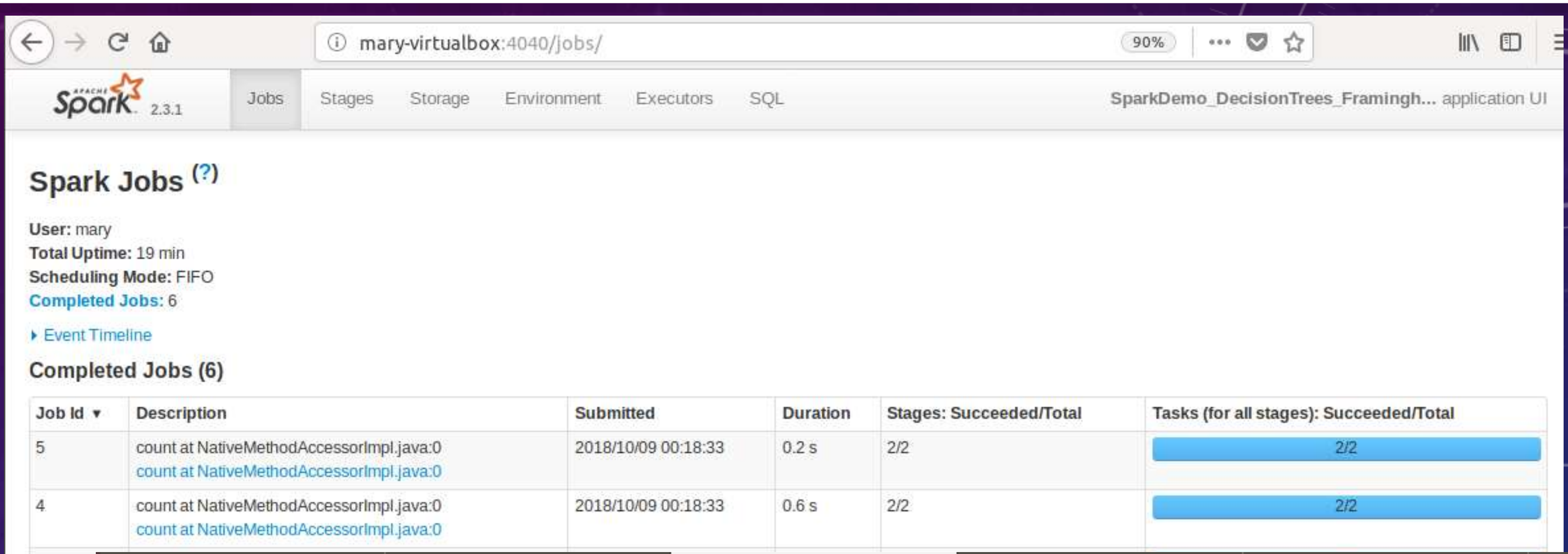
```
In [5]: orig_data=spark.read \
            .option("uri", "mongodb://127.0.0.1:27017/dataset.framingham_struc") \
            .format("com.mongodb.spark.sql.DefaultSource") \
            .load()
```

Check the data

```
In [6]: #data.printSchema()
        #data.columns
        #data.show(10)
        data=orig_data.na.drop()
        print("Number of records:", orig_data.count())
        print("Remaining data after removing null rows: ", data.count())
        print("Percentage of removed data: ", (orig_data.count()-data.count())/orig_data.count(), "%" )
        print("Fields:", data.columns)
```

```
Number of records: 4241
Remaining data after removing null rows:  3658
Percentage of removed data:  0.13746757860132048 %
Fields: ['BMI', 'BPMeds', 'TenYearCHD', '_id', 'age', 'cigsPerDay', 'currentSmoker', 'diaBP', 'diabetes', 'education', 'glucose', 'heartRate', 'male', 'prevalentHyp', 'prevalentStroke', 'sysBP', 'totChol']
```

There are 4241 with 16 fields excluding the _id



mary-virtualbox:4040/jobs/ 90%

SparkDemo_DecisionTrees_Framingh... application UI

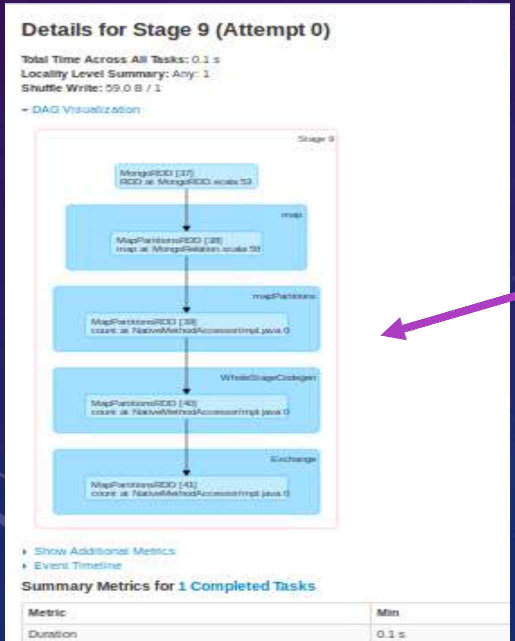
Spark Jobs (?)

User: mary
Total Uptime: 19 min
Scheduling Mode: FIFO
Completed Jobs: 6

▸ Event Timeline

Completed Jobs (6)

Job Id ▾	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
5	count at NativeMethodAccessorImpl.java:0 count at NativeMethodAccessorImpl.java:0	2018/10/09 00:18:33	0.2 s	2/2	2/2
4	count at NativeMethodAccessorImpl.java:0 count at NativeMethodAccessorImpl.java:0	2018/10/09 00:18:33	0.6 s	2/2	2/2
3					
2					



Details for Stage 9 (Attempt 0)

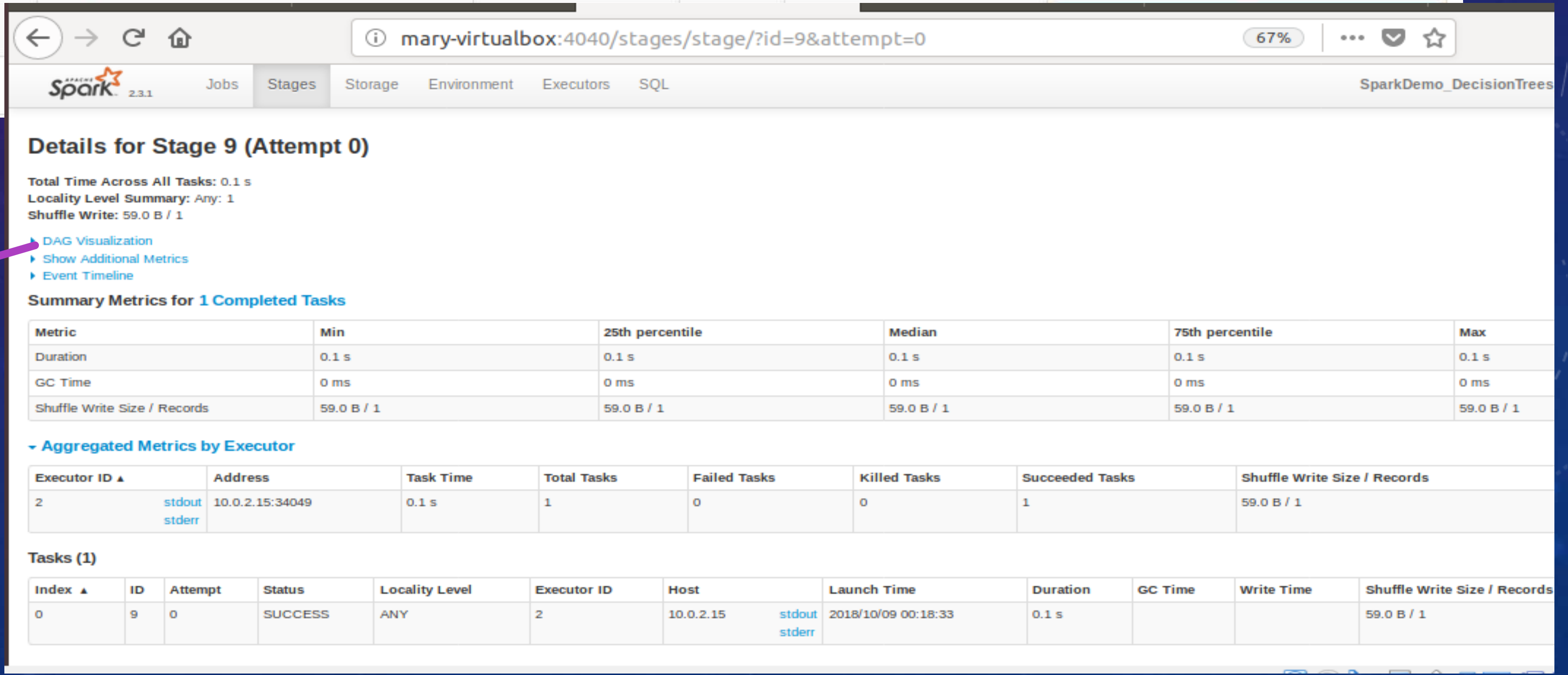
Total Time Across All Tasks: 0.1 s
Locality Level Summary: Any: 1
Shuffle Write: 59.0 B / 1

▾ DAG Visualization

▸ Show Additional Metrics
▸ Event Timeline

Summary Metrics for 1 Completed Tasks

Metric	Min
Duration	0.1 s



mary-virtualbox:4040/stages/stage/?id=9&attempt=0 67%

SparkDemo_DecisionTrees

Details for Stage 9 (Attempt 0)

Total Time Across All Tasks: 0.1 s
Locality Level Summary: Any: 1
Shuffle Write: 59.0 B / 1

▸ DAG Visualization
▸ Show Additional Metrics
▸ Event Timeline

Summary Metrics for 1 Completed Tasks

Metric	Min	25th percentile	Median	75th percentile	Max
Duration	0.1 s	0.1 s	0.1 s	0.1 s	0.1 s
GC Time	0 ms	0 ms	0 ms	0 ms	0 ms
Shuffle Write Size / Records	59.0 B / 1	59.0 B / 1	59.0 B / 1	59.0 B / 1	59.0 B / 1

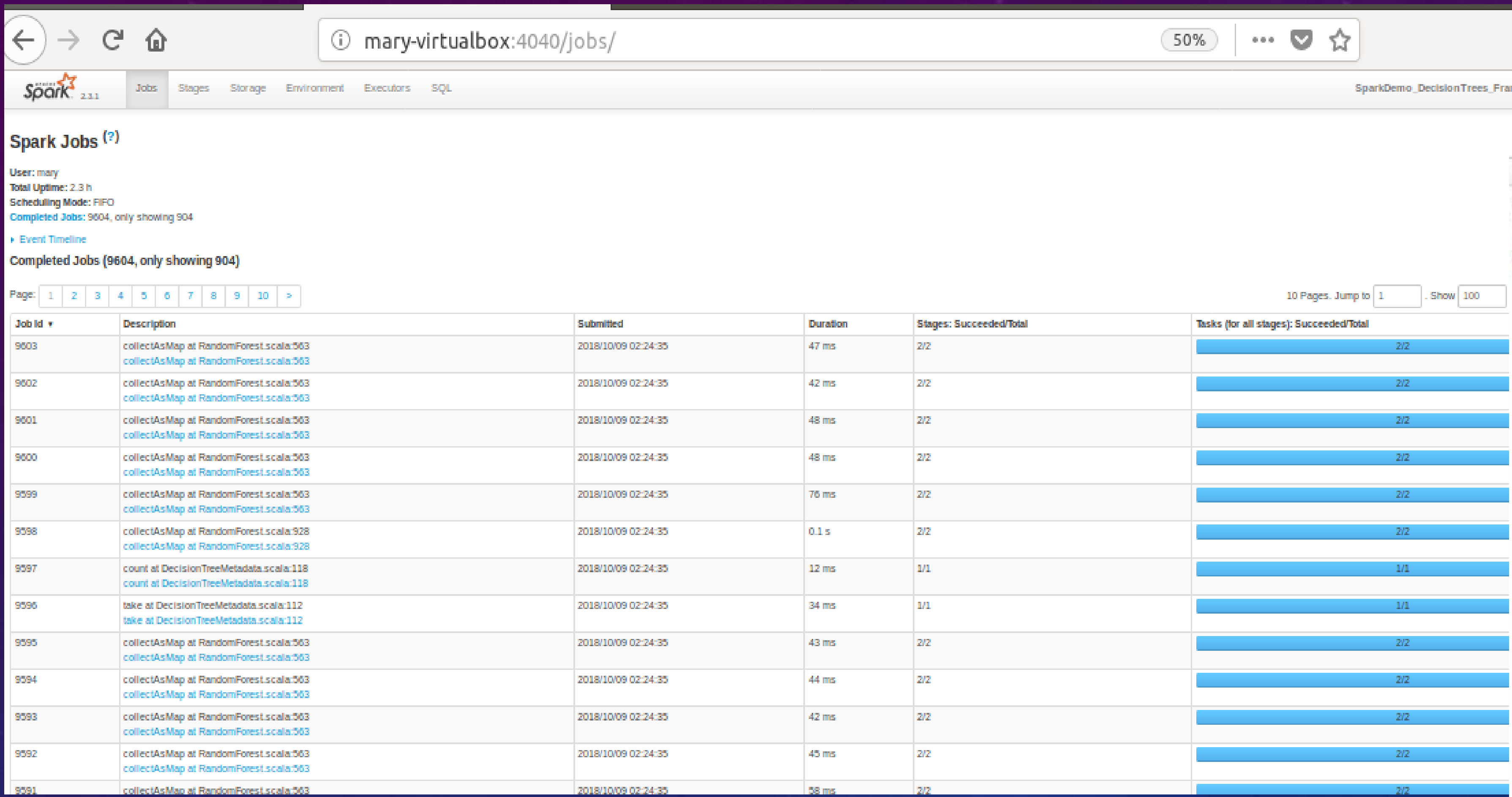
▾ Aggregated Metrics by Executor

Executor ID ▴	Address	Task Time	Total Tasks	Failed Tasks	Killed Tasks	Succeeded Tasks	Shuffle Write Size / Records
2 stdout stderr	10.0.2.15:34049	0.1 s	1	0	0	1	59.0 B / 1

Tasks (1)

Index ▴	ID	Attempt	Status	Locality Level	Executor ID	Host	Launch Time	Duration	GC Time	Write Time	Shuffle Write Size / Records
0	9	0	SUCCESS	ANY	2	10.0.2.15 stdout stderr	2018/10/09 00:18:33	0.1 s			59.0 B / 1

Deployment stage:



mary-virtualbox:4040/jobs/

50%

Spark 2.3.1 — Jobs | Stages | Storage | Environment | Executors | SQL

SparkDemo_DecisionTrees_Framingh... application UI

Spark Jobs (?)

User: mary
Total Uptime: 2.3 h
Scheduling Mode: FIFO
Completed Jobs: 9604, only showing 904

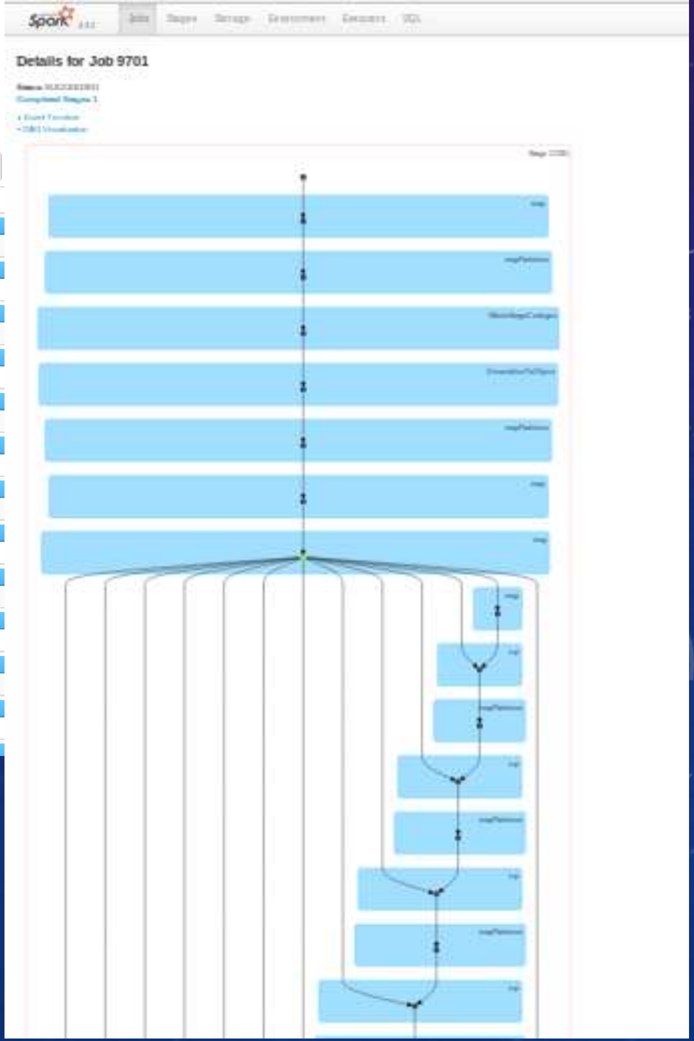
▸ Event Timeline

Completed Jobs (9604, only showing 904)

Page: 1 2 3 4 5 6 7 8 9 10 >

10 Pages. Jump to 1 . Show 100

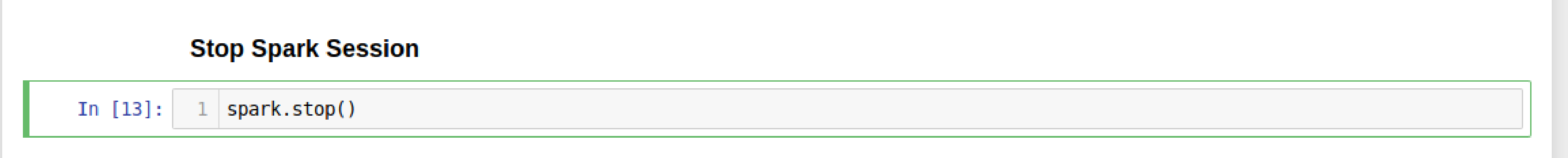
Job Id ▾	Description	Submitted	Duration	Stages: Succeeded/Total	Tasks (for all stages): Succeeded/Total
9603	collectAsMap at RandomForest.scala:563 collectAsMap at RandomForest.scala:563	2018/10/09 02:24:35	47 ms	2/2	2/2
9602	collectAsMap at RandomForest.scala:563 collectAsMap at RandomForest.scala:563	2018/10/09 02:24:35	42 ms	2/2	2/2
9601	collectAsMap at RandomForest.scala:563 collectAsMap at RandomForest.scala:563	2018/10/09 02:24:35	48 ms	2/2	2/2
9600	collectAsMap at RandomForest.scala:563 collectAsMap at RandomForest.scala:563	2018/10/09 02:24:35	48 ms	2/2	2/2
9599	collectAsMap at RandomForest.scala:563 collectAsMap at RandomForest.scala:563	2018/10/09 02:24:35	76 ms	2/2	2/2
9598	collectAsMap at RandomForest.scala:928 collectAsMap at RandomForest.scala:928	2018/10/09 02:24:35	0.1 s	2/2	2/2
9597	count at DecisionTreeMetadata.scala:118 count at DecisionTreeMetadata.scala:118	2018/10/09 02:24:35	12 ms	1/1	1/1
9596	take at DecisionTreeMetadata.scala:112 take at DecisionTreeMetadata.scala:112	2018/10/09 02:24:35	34 ms	1/1	1/1
9595	collectAsMap at RandomForest.scala:563 collectAsMap at RandomForest.scala:563	2018/10/09 02:24:35	43 ms	2/2	2/2
9594	collectAsMap at RandomForest.scala:563 collectAsMap at RandomForest.scala:563	2018/10/09 02:24:35	44 ms	2/2	2/2
9593	collectAsMap at RandomForest.scala:563 collectAsMap at RandomForest.scala:563	2018/10/09 02:24:35	42 ms	2/2	2/2
9592	collectAsMap at RandomForest.scala:563 collectAsMap at RandomForest.scala:563	2018/10/09 02:24:35	45 ms	2/2	2/2
9591	collectAsMap at RandomForest.scala:563	2018/10/09 02:24:35	58 ms	2/2	2/2



Spark — Jobs | Stages | Storage | Environment | Executors | SQL

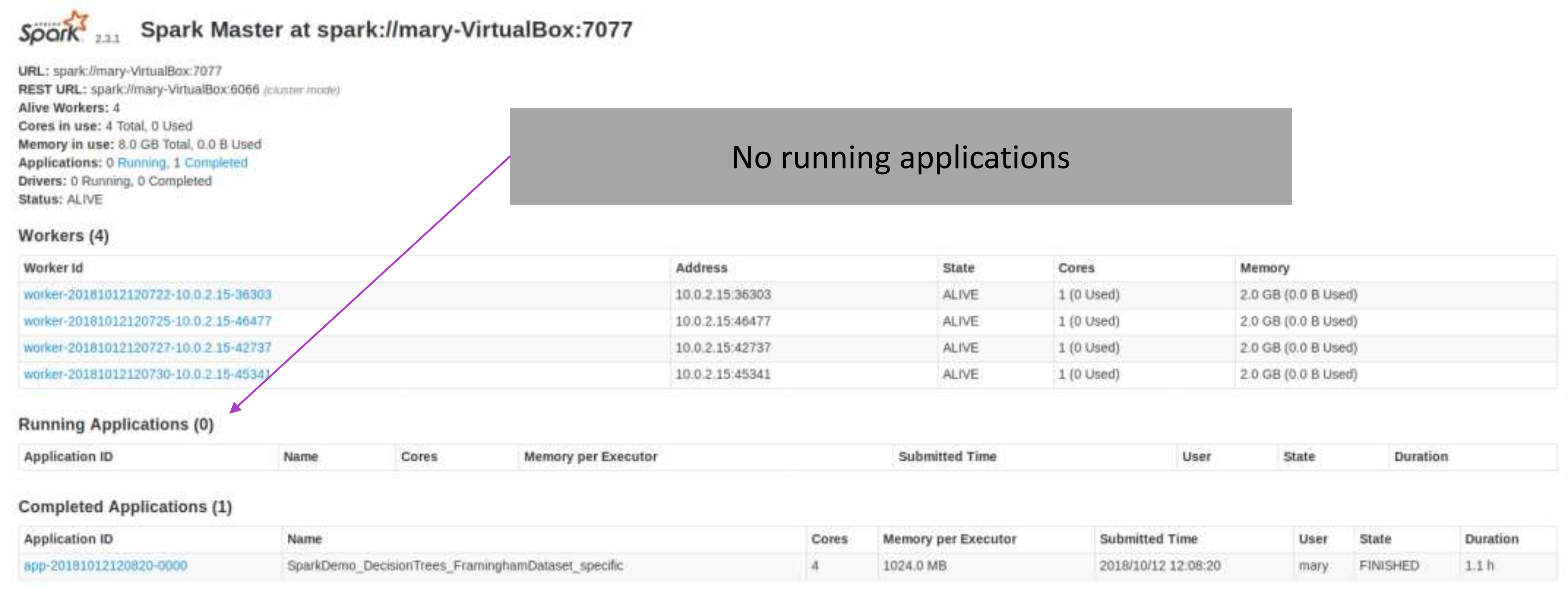
Details for Job 9701

Deployment stage: Stopping Spark Session



Stop Spark Session

```
In [13]: spark.stop()
```



Spark Master at spark://mary-VirtualBox:7077

URL: spark://mary-VirtualBox:7077
REST URL: spark://mary-VirtualBox:6066 *(cluster mode)*
Alive Workers: 4
Cores in use: 4 Total, 0 Used
Memory in use: 8.0 GB Total, 0.0 B Used
Applications: 0 Running, 1 Completed
Drivers: 0 Running, 0 Completed
Status: ALIVE

No running applications

Workers (4)

Worker Id	Address	State	Cores	Memory
worker-20181012120722-10.0.2.15-36303	10.0.2.15:36303	ALIVE	1 (0 Used)	2.0 GB (0.0 B Used)
worker-20181012120725-10.0.2.15-46477	10.0.2.15:46477	ALIVE	1 (0 Used)	2.0 GB (0.0 B Used)
worker-20181012120727-10.0.2.15-42737	10.0.2.15:42737	ALIVE	1 (0 Used)	2.0 GB (0.0 B Used)
worker-20181012120730-10.0.2.15-45341	10.0.2.15:45341	ALIVE	1 (0 Used)	2.0 GB (0.0 B Used)

Running Applications (0)

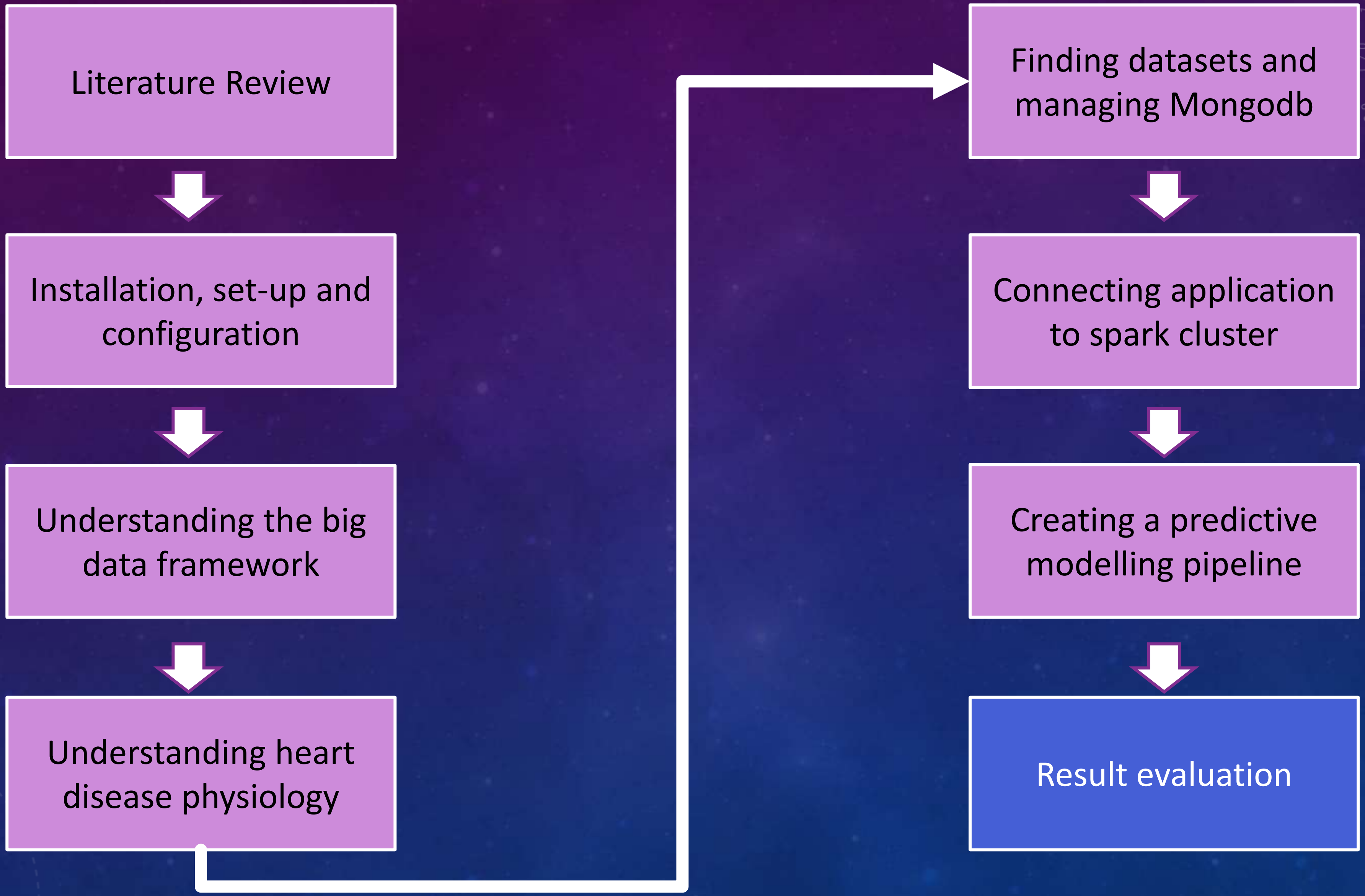
Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration

Completed Applications (1)

Application ID	Name	Cores	Memory per Executor	Submitted Time	User	State	Duration
app-20181012120820-0000	SparkDemo_DecisionTrees_FraminghamDataset_specific	4	1024.0 MB	2018/10/12 12:08:20	mary	FINISHED	1.1 h

FINDINGS/OUTCOMES

Project Approach/Methodology



Literature Review

Installation, set-up and configuration

Understanding the big data framework

Understanding heart disease physiology

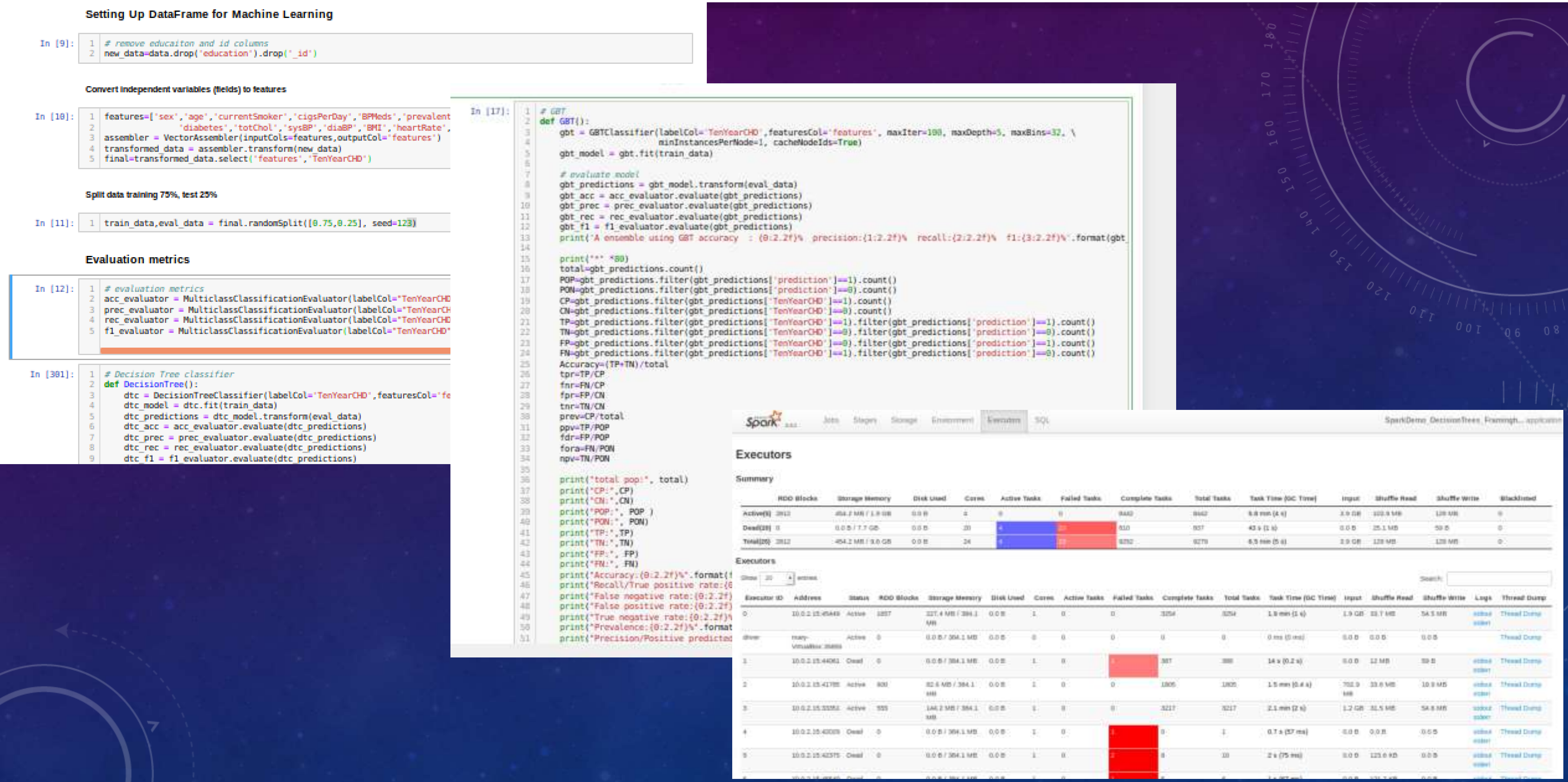
Finding datasets and managing Mongodb

Connecting application to spark cluster

Creating a predictive modelling pipeline

Result evaluation

Findings/outcome



Setting Up DataFrame for Machine Learning

```
In [9]:
# remove educaiton and id columns
new_data=data.drop('education').drop('_id')
```

Convert independent variables (fields) to features

```
In [10]:
features=['sex','age','currentSmoker','cigsPerDay','BPMeds','prevalent
          'diabetes','totChol','sysBP','diaBP','BMI','heartRate',
assembler = VectorAssembler(inputCols=features,outputCol='features')
transformed_data = assembler.transform(new_data)
final=transformed_data.select('features','TenYearCHD')
```

Split data training 75%, test 25%

```
In [11]:
train_data,eval_data = final.randomSplit([0.75,0.25], seed=123)
```

Evaluation metrics

```
In [12]:
# evaluation metrics
acc_evaluator = MulticlassClassificationEvaluator(labelCol="TenYearCHD
prec_evaluator = MulticlassClassificationEvaluator(labelCol="TenYearCH
rec_evaluator = MulticlassClassificationEvaluator(labelCol="TenYearCHD
f1_evaluator = MulticlassClassificationEvaluator(labelCol="TenYearCHD"
```

```
In [301]:
# Decision Tree classifier
def DecisionTree():
    dtc = DecisionTreeClassifier(labelCol='TenYearCHD',featuresCol='fe
    dtc_model = dtc.fit(train_data)
    dtc_predictions = dtc_model.transform(eval_data)
    dtc_acc = acc_evaluator.evaluate(dtc_predictions)
    dtc_prec = prec_evaluator.evaluate(dtc_predictions)
    dtc_rec = rec_evaluator.evaluate(dtc_predictions)
    dtc_f1 = f1_evaluator.evaluate(dtc_predictions)
```

```
In [17]:
# GBT
def GBT():
    gbt = GBTClassifier(labelCol='TenYearCHD',featuresCol='features', maxIter=100, maxDepth=5, maxBins=32, \
                        minInstancesPerNode=1, cacheNodeIds=True)
    gbt_model = gbt.fit(train_data)

    # evaluate model
    gbt_predictions = gbt_model.transform(eval_data)
    gbt_acc = acc_evaluator.evaluate(gbt_predictions)
    gbt_prec = prec_evaluator.evaluate(gbt_predictions)
    gbt_rec = rec_evaluator.evaluate(gbt_predictions)
    gbt_f1 = f1_evaluator.evaluate(gbt_predictions)
    print('A ensemble using GBT accuracy  : {0:2.2f}%  precision:{1:2.2f}%  recall:{2:2.2f}%  f1:{3:2.2f}%'.format(gbt_

    print("*" *80)
    total=gbt_predictions.count()
    POP=gbt_predictions.filter(gbt_predictions['prediction']==1).count()
    PON=gbt_predictions.filter(gbt_predictions['prediction']==0).count()
    CP=gbt_predictions.filter(gbt_predictions['TenYearCHD']==1).count()
    CN=gbt_predictions.filter(gbt_predictions['TenYearCHD']==0).count()
    TP=gbt_predictions.filter(gbt_predictions['TenYearCHD']==1).filter(gbt_predictions['prediction']==1).count()
    TN=gbt_predictions.filter(gbt_predictions['TenYearCHD']==0).filter(gbt_predictions['prediction']==0).count()
    FP=gbt_predictions.filter(gbt_predictions['TenYearCHD']==0).filter(gbt_predictions['prediction']==1).count()
    FN=gbt_predictions.filter(gbt_predictions['TenYearCHD']==1).filter(gbt_predictions['prediction']==0).count()
    Accuracy=(TP+TN)/total
    tpr=TP/CP
    fnr=FN/CP
    fpr=FP/CN
    tnr=TN/CN
    prev=CP/total
    ppv=TP/POP
    fdr=FP/POP
    fora=FN/PON
    npv=TN/PON

    print("total pop:", total)
    print("CP:",CP)
    print("CN:",CN)
    print("POP:", POP )
    print("PON:", PON)
    print("TP:",TP)
    print("TN:",TN)
    print("FP:", FP)
    print("FN:", FN)
    print("Accuracy:{0:2.2f}%".format(t
    print("Recall/True positive rate:{0
    print("False negative rate:{0:2.2f}
    print("False positive rate:{0:2.2f}
    print("True negative rate:{0:2.2f}%
    print("Prevalence:{0:2.2f}%".format
    print("Precision/Positive predicted
```

Executors

Summary

	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)	Input	Shuffle Read	Shuffle Write	Blacklisted
Active(5)	2812	454.2 MB / 1.9 GB	0.0 B	4	0	0	8442	8442	5.8 min (4 s)	2.9 GB	102.9 MB	128 MB	0
Dead(20)	0	0.0 B / 7.7 GB	0.0 B	20	4	23	810	837	43 s (1 s)	0.0 B	25.1 MB	59 B	0
Total(25)	2812	454.2 MB / 9.6 GB	0.0 B	24	4	23	9252	9279	6.5 min (5 s)	2.9 GB	128 MB	128 MB	0

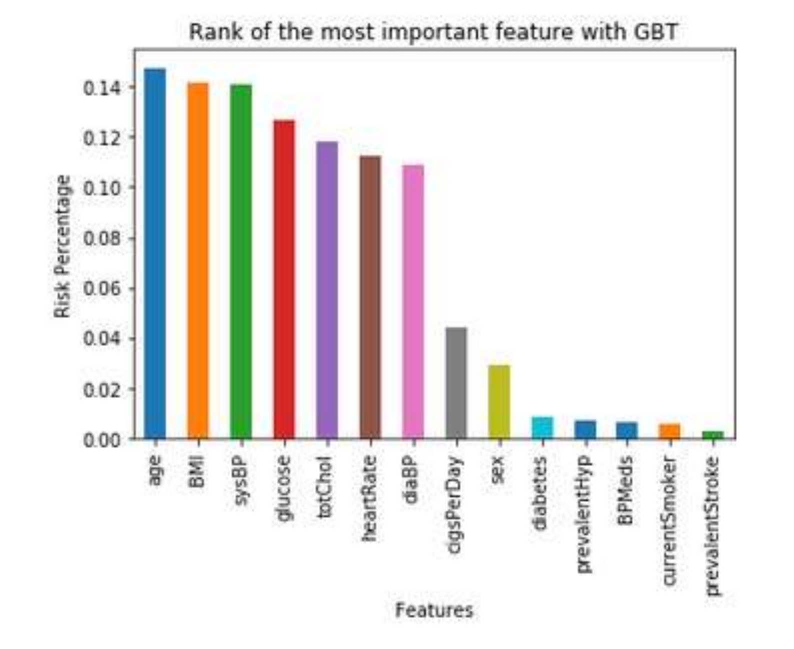
Executors

Show 20 entries

Search:

Executor ID	Address	Status	RDD Blocks	Storage Memory	Disk Used	Cores	Active Tasks	Failed Tasks	Complete Tasks	Total Tasks	Task Time (GC Time)	Input	Shuffle Read	Shuffle Write	Logs	Thread Dump
0	10.0.2.15:45449	Active	1857	327.4 MB / 384.1 MB	0.0 B	1	0	0	3254	3254	1.9 min (1 s)	1.9 GB	33.7 MB	54.5 MB	stdout stderr	Thread Dump
driver	mary-VirtualBox:35893	Active	0	0.0 B / 384.1 MB	0.0 B	0	0	0	0	0	0 ms (0 ms)	0.0 B	0.0 B	0.0 B		Thread Dump
1	10.0.2.15:44061	Dead	0	0.0 B / 384.1 MB	0.0 B	1	0	1	387	388	14 s (0.2 s)	0.0 B	12 MB	59 B	stdout stderr	Thread Dump
2	10.0.2.15:41785	Active	800	82.6 MB / 384.1 MB	0.0 B	1	0	0	1805	1805	1.5 min (0.4 s)	702.9 MB	33.6 MB	10.9 MB	stdout stderr	Thread Dump
3	10.0.2.15:33352	Active	555	144.2 MB / 384.1 MB	0.0 B	1	0	0	3217	3217	2.1 min (2 s)	1.2 GB	31.5 MB	54.8 MB	stdout stderr	Thread Dump
4	10.0.2.15:42009	Dead	0	0.0 B / 384.1 MB	0.0 B	1	0	1	0	1	0.7 s (57 ms)	0.0 B	0.0 B	0.0 B	stdout stderr	Thread Dump
5	10.0.2.15:42375	Dead	0	0.0 B / 384.1 MB	0.0 B	1	0	2	8	10	2 s (75 ms)	0.0 B	123.6 KB	0.0 B	stdout stderr	Thread Dump
6	10.0.2.15:46540	Dead	0	0.0 B / 384.1 MB	0.0 B	1	0	1	5	6	1 s (67 ms)	0.0 B	121.7 KB	0.0 B	stdout	Thread Dump

Findings/outcome



Accuracy	Precision	Recall	f1
85.08%	81.30%	85.08%	81.59%

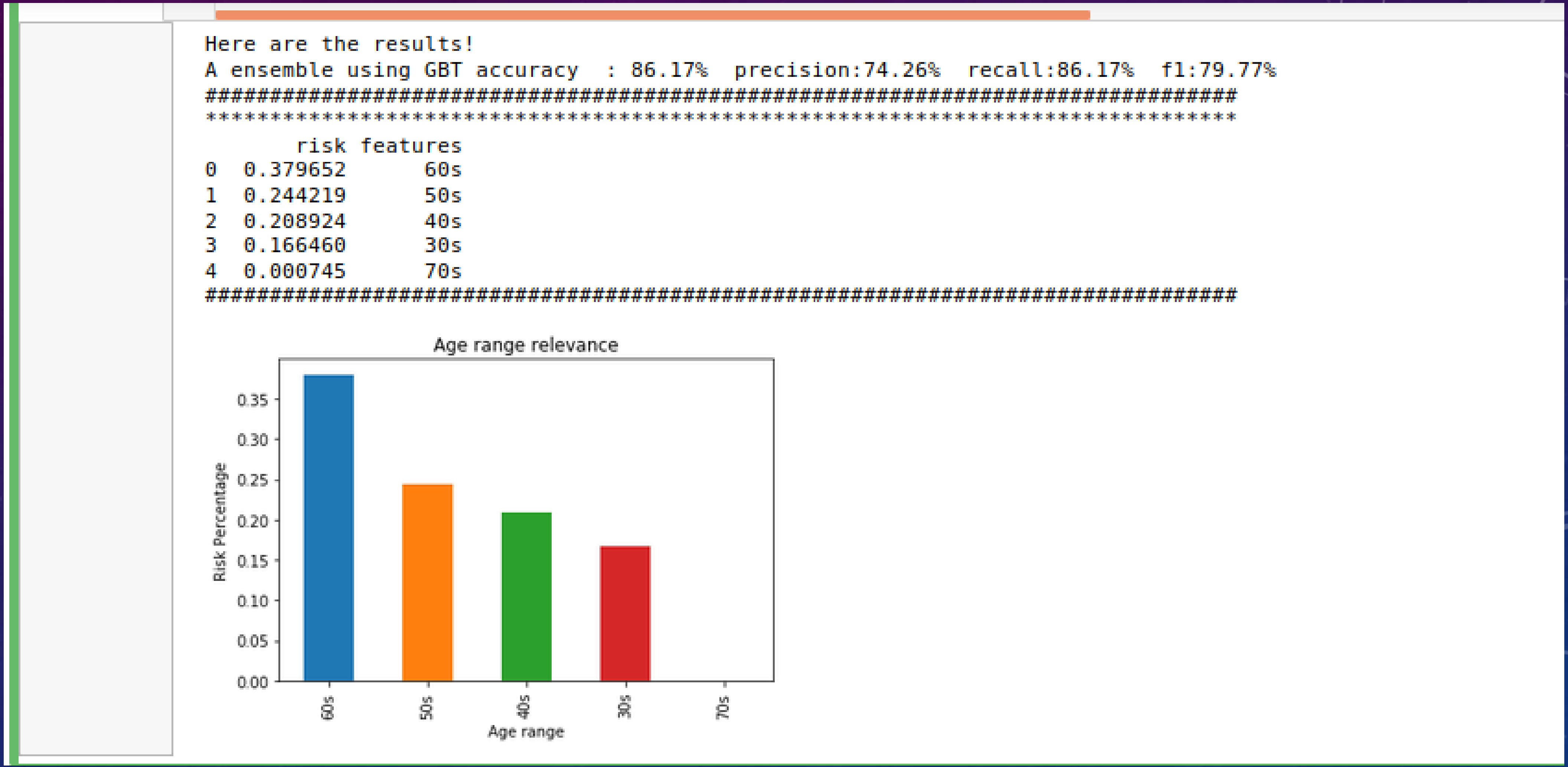
Age, body mass index, blood sugar level, systolic blood pressure and cholesterol are the most important determining factor to assess risk of having coronary heart disease

Findings/outcome

		Ground Truth			
	Total Population 905	Condition Positive 135	Condition Negative 770	Prevalence 14.92%	
Prediction	Predictive Outcome Positive 135	True Positive 114	False Positive 21	Positive Predictive Value 84.44%	False Discovery Rate 15.5%
	Predictive Outcome Negative 770	False Negative 21	True Negative 749	False Omission Rate 2.72%	Negative Predictive Value 97.27%
Accuracy 85.08%		True Positive Rate 85.08%	False Positive Rate 2.73%		
		False Negative Rate 15%	True Negative Rate 97.27%		

Digest features and determine which specific subsets or range is likely contributing to coronary heart disease.

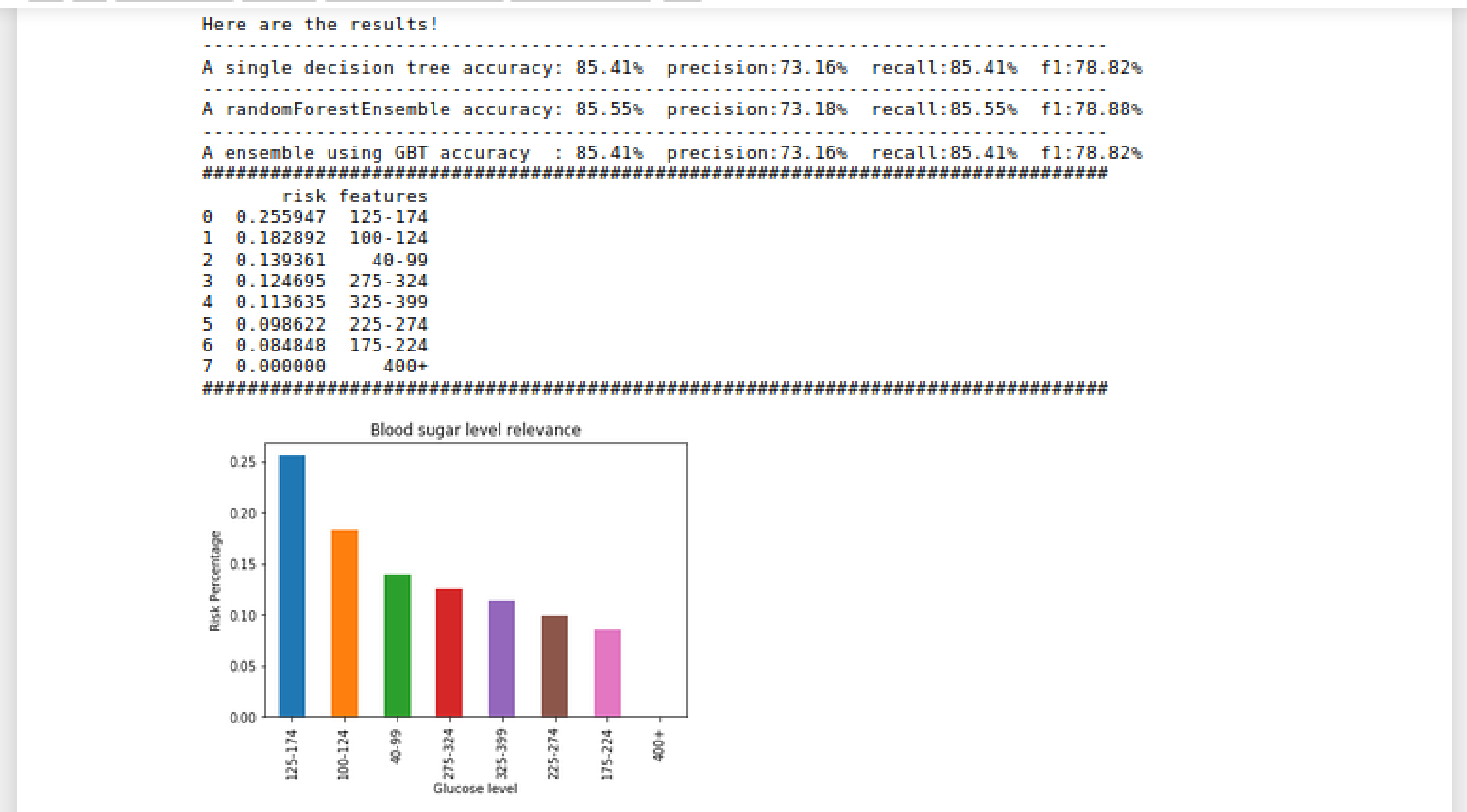
Findings/outcome: Age group



```
Here are the results!
A ensemble using GBT accuracy  : 86.17%  precision:74.26%  recall:86.17%  f1:79.77%
################################################################################
********************************************************************************
       risk features
0  0.379652      60s
1  0.244219      50s
2  0.208924      40s
3  0.166460      30s
4  0.000745      70s
################################################################################
```

Age range relevance

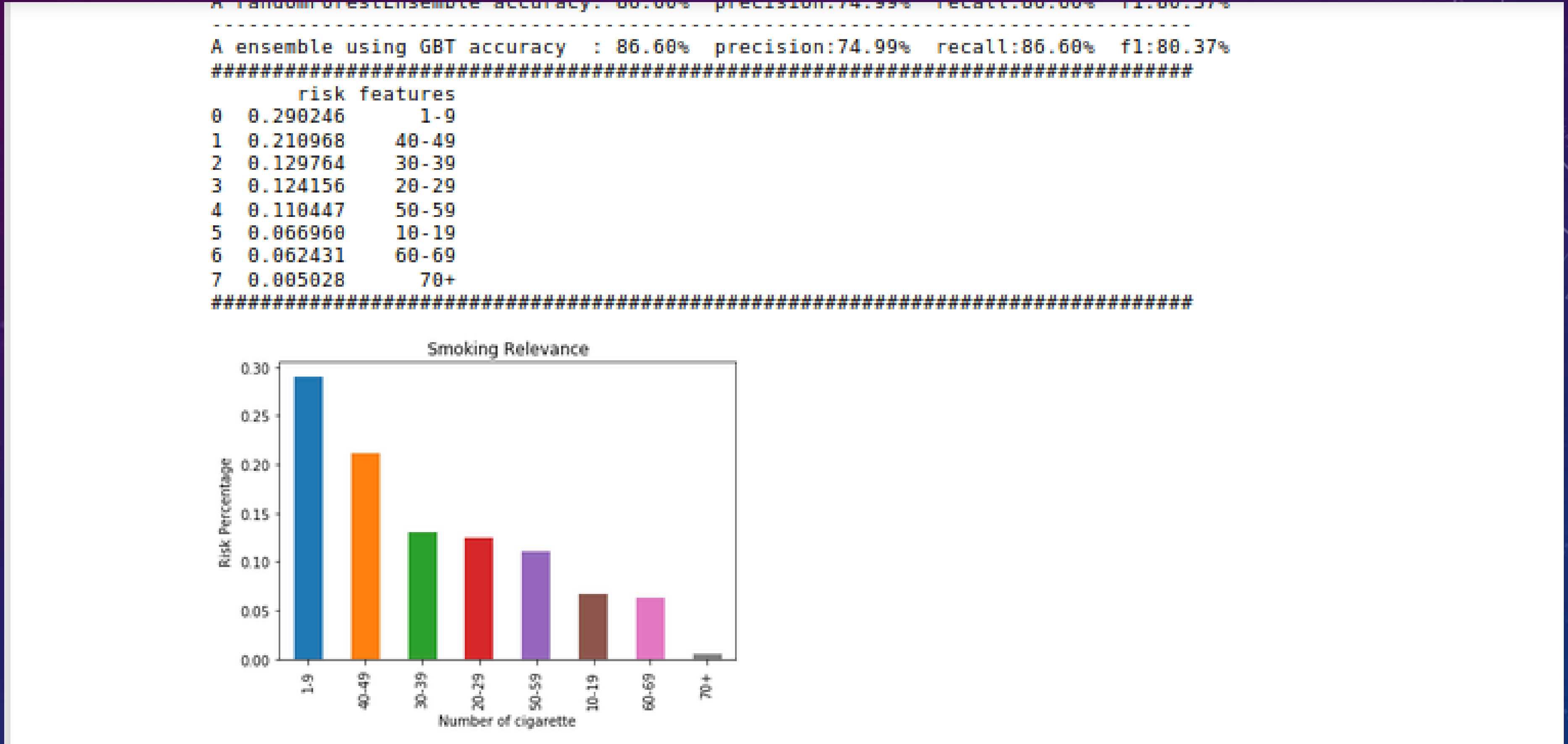
Findings/outcome: by Glucose level



```
Here are the results!
--------------------------------------------------------------------------------
A single decision tree accuracy: 85.41%  precision:73.16%  recall:85.41%  f1:78.82%
--------------------------------------------------------------------------------
A randomForestEnsemble accuracy: 85.55%  precision:73.18%  recall:85.55%  f1:78.88%
--------------------------------------------------------------------------------
A ensemble using GBT accuracy  : 85.41%  precision:73.16%  recall:85.41%  f1:78.82%
################################################################################
       risk features
0  0.255947  125-174
1  0.182892  100-124
2  0.139361    40-99
3  0.124695  275-324
4  0.113635  325-399
5  0.098622  225-274
6  0.084848  175-224
7  0.000000     400+
################################################################################
```

Blood sugar level relevance

Findings/outcome: by Glucose level



```
A randomForestEnsemble accuracy: 86.60%  precision:74.99%  recall:86.60%  f1:80.37%
--------------------------------------------------------------------------------
A ensemble using GBT accuracy  : 86.60%  precision:74.99%  recall:86.60%  f1:80.37%
################################################################################
       risk features
0  0.290246      1-9
1  0.210968    40-49
2  0.129764    30-39
3  0.124156    20-29
4  0.110447    50-59
5  0.066960    10-19
6  0.062431    60-69
7  0.005028      70+
################################################################################
```

IMPLICATIONS

Implications

- Health care authorities may use the findings to focus their health education not only to elderly but also the young individuals as the risk for heart disease increased significantly higher as they grow older.

- Regulatory commissions to study sugar content of consumable goods and reduce use of sugar.

- Promote healthy lifestyle by quitting cigarette smoking as oppose to cutting down.

- Although the dataset is not a big data, the framework could work using the big data. Thus, applicable for scalable projects in predicting heart disease.

THANK YOU FOR LISTENING