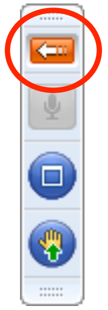
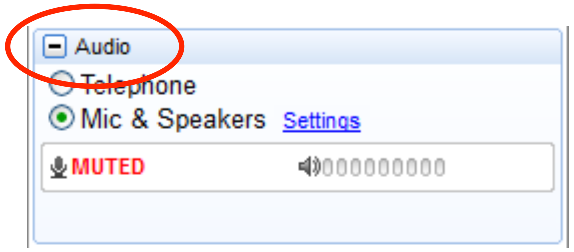
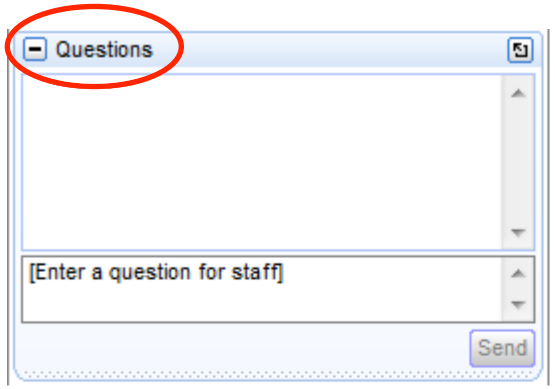
SparkR

The GoToWebinar Control Panel

1. The orange button
2. Audio Type
3. Close apps
4. Enlarge my screen
5. Headphones
6. Questions Pane







Today's Agenda

- Lecture
  - slides and/or video will be made available within one week
- Live Demonstration
- Q & A

"Intro to Spark" Training Course

To attend a hands-on Spark training course which runs every Saturday, please visit:

liondatasystems.com/courses

Thank You!

- This event has attracted nearly 900 registrants from various parts of the world.

- Thank you everyone for your support!

Today's Speaker

- Shivaram Venkataraman
- Co-Author of SparkR
- PhD Student @ UC Berkeley
- Former Google Engineer

Introduction to SparkR

Shivaram Venkataraman

amplab UC BERKELEY

Berkeley UNIVERSITY OF CALIFORNIA

Big Data & R

DataFrames
Visualization
Libraries

+

Data

Background

Engine for large-scale data processing

Fast, Easy to Use

Runs Everywhere - EC2, YARN, Mesos

SparkR

Interactive Shell

Batch Scripts

Outline

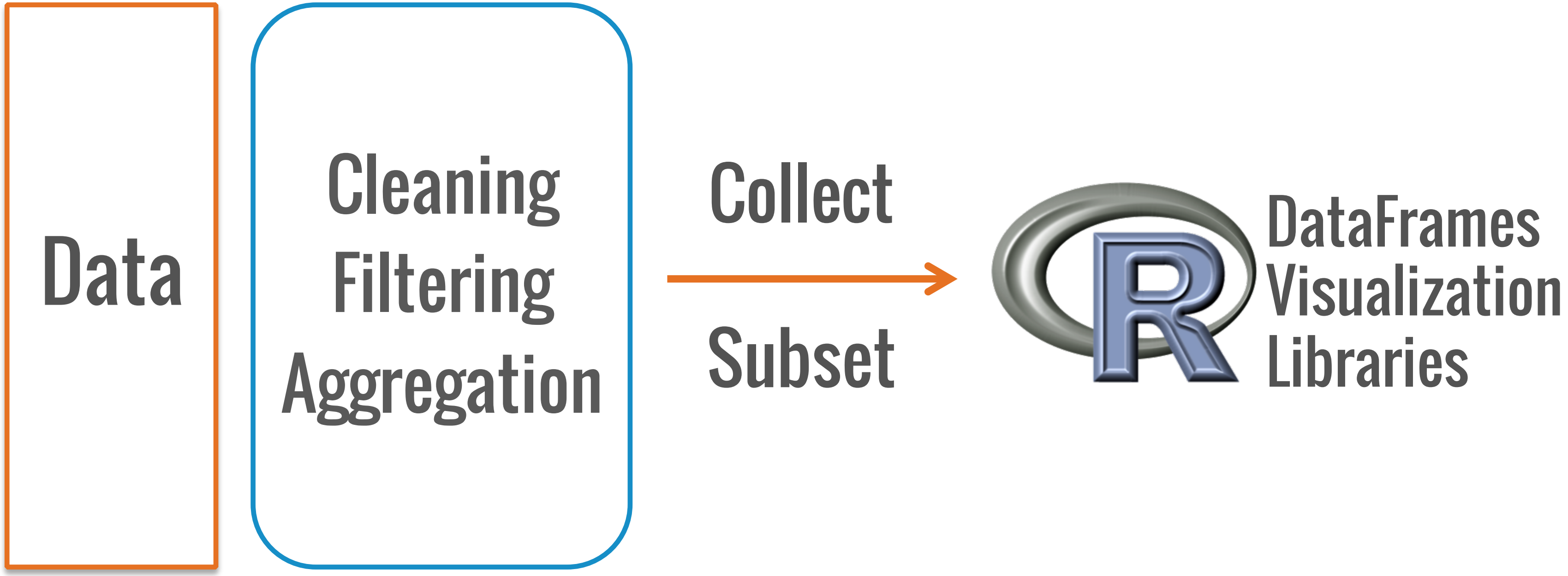
SparkR DataFrames

Architecture

Demo

SparkR Roadmap

Big Data Processing + R



SparkR DataFrames

High-level API for data manipulation

Read in CSV, JSON, JDBC etc.

dplyr-like syntax

Example

```
{"name":"Michael", "age":29}
{"name":"Andy", "age":30}
{"name":"Justin", "age":19}
{"name":"Bob", "age":22}
{"name":"Chris", "age":28}
{"name":"Garth", "age":36}
{"name":"Tasha", "age":24}
{"name":"Mac", "age":30}
{"name":"Neil", "age":32}
```

Example

```
people <- read.df(
    “hdfs://people.json”,
    “json”)

avgAge <- select(
    df,
    avg(df$age))

collect(avgAge)
```

Read input from HDFS

Collect to data.frame

DataFrame API

Filtering Data

- select, `$`, where, filter

Aggregating Data

- groupBy, summarize, arrange

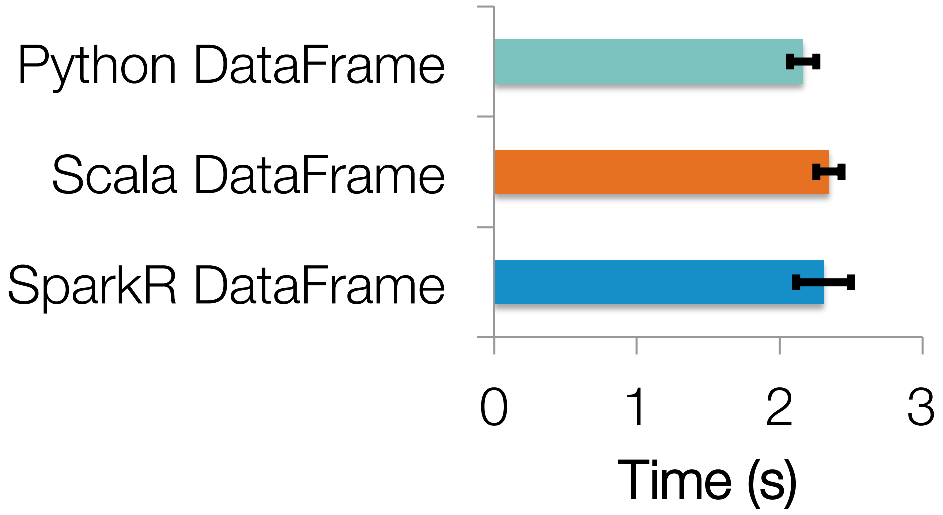
Input/Output

- read.df, write.df, sql

SparkR DataFrames

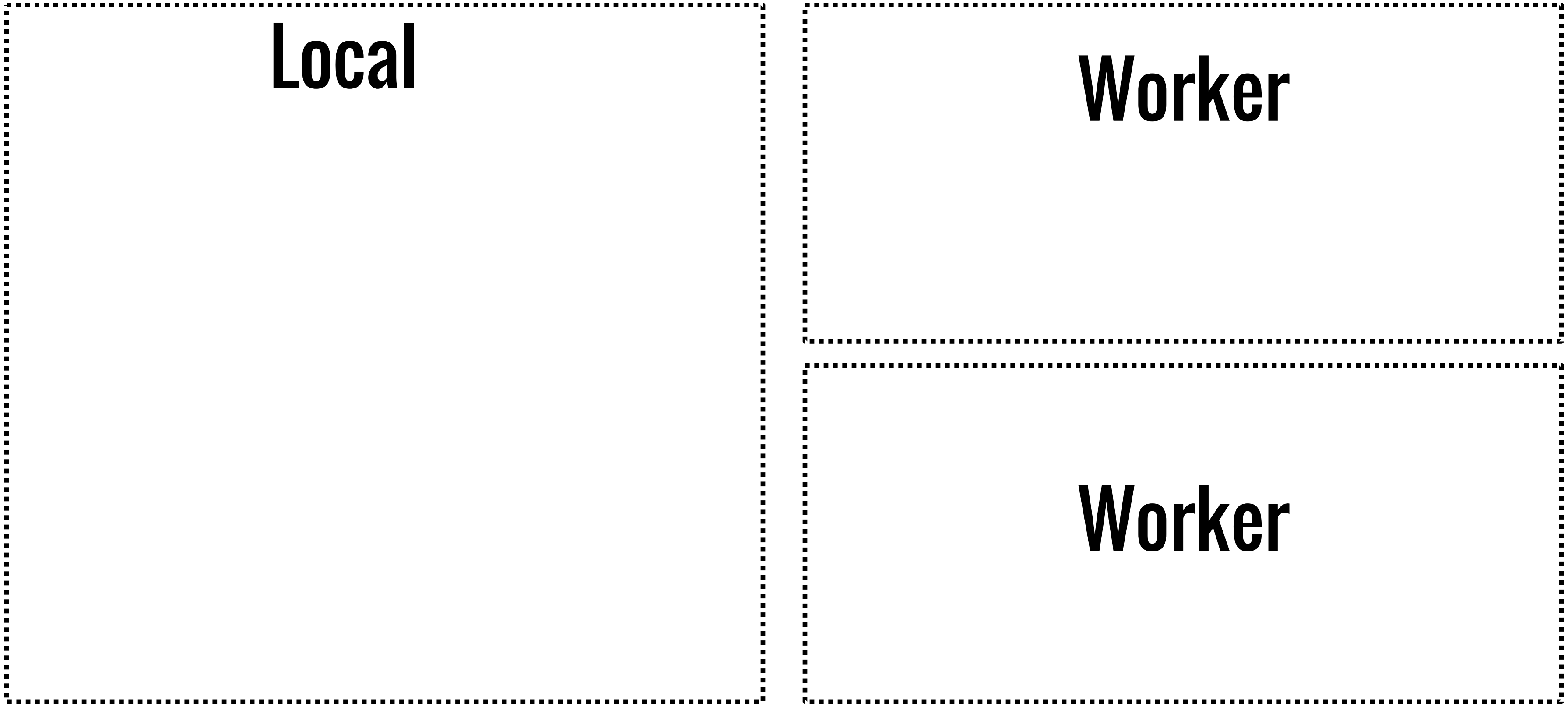
Query Planning

SQL Optimizations

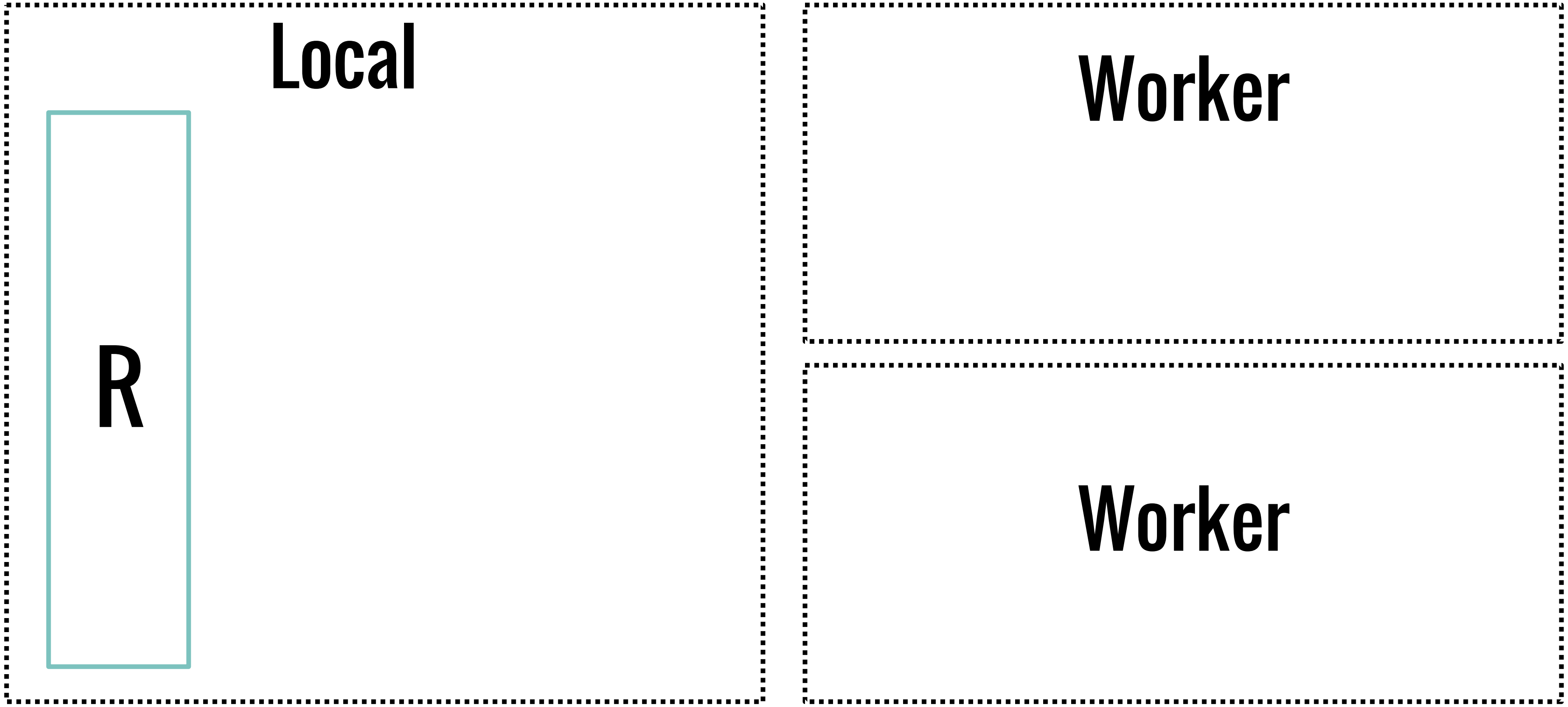


Architecture

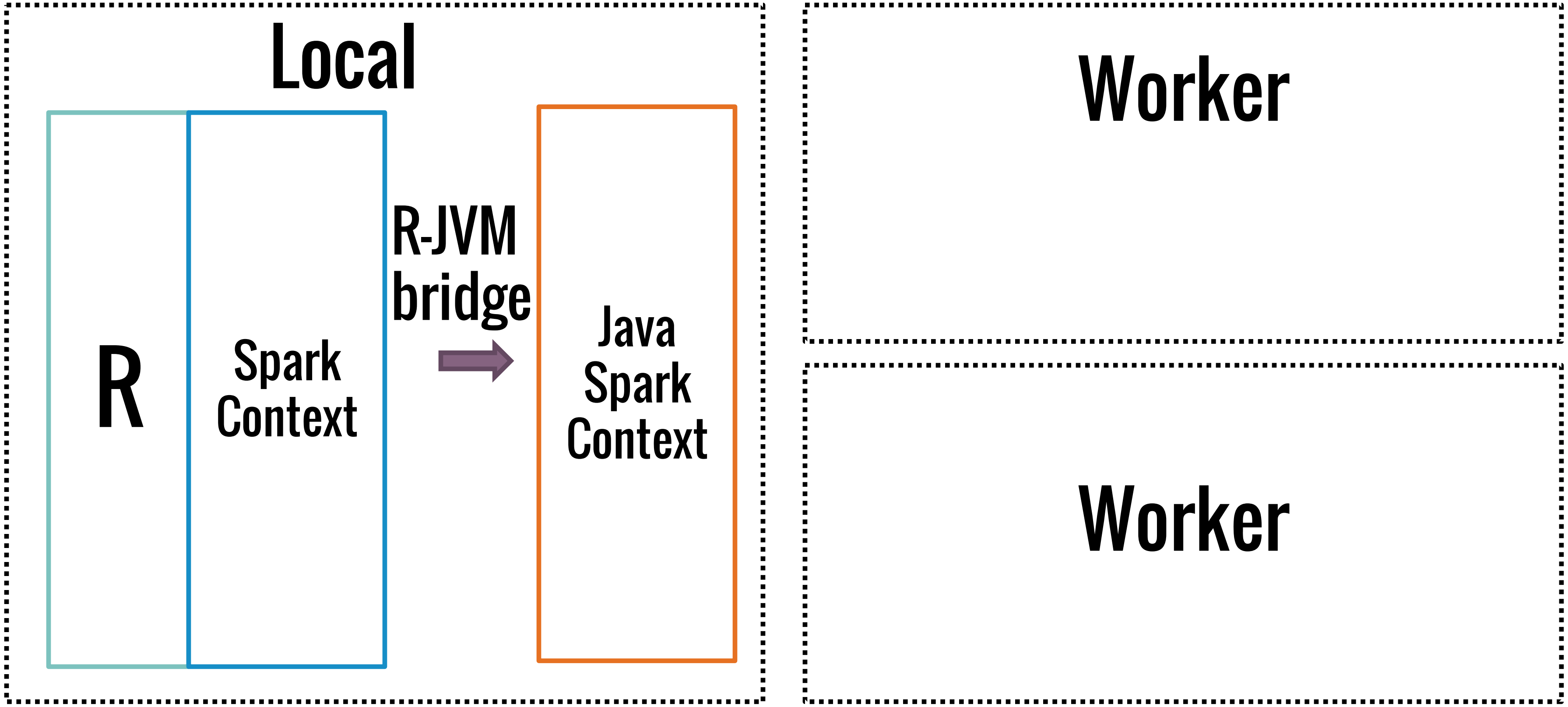
Architecture



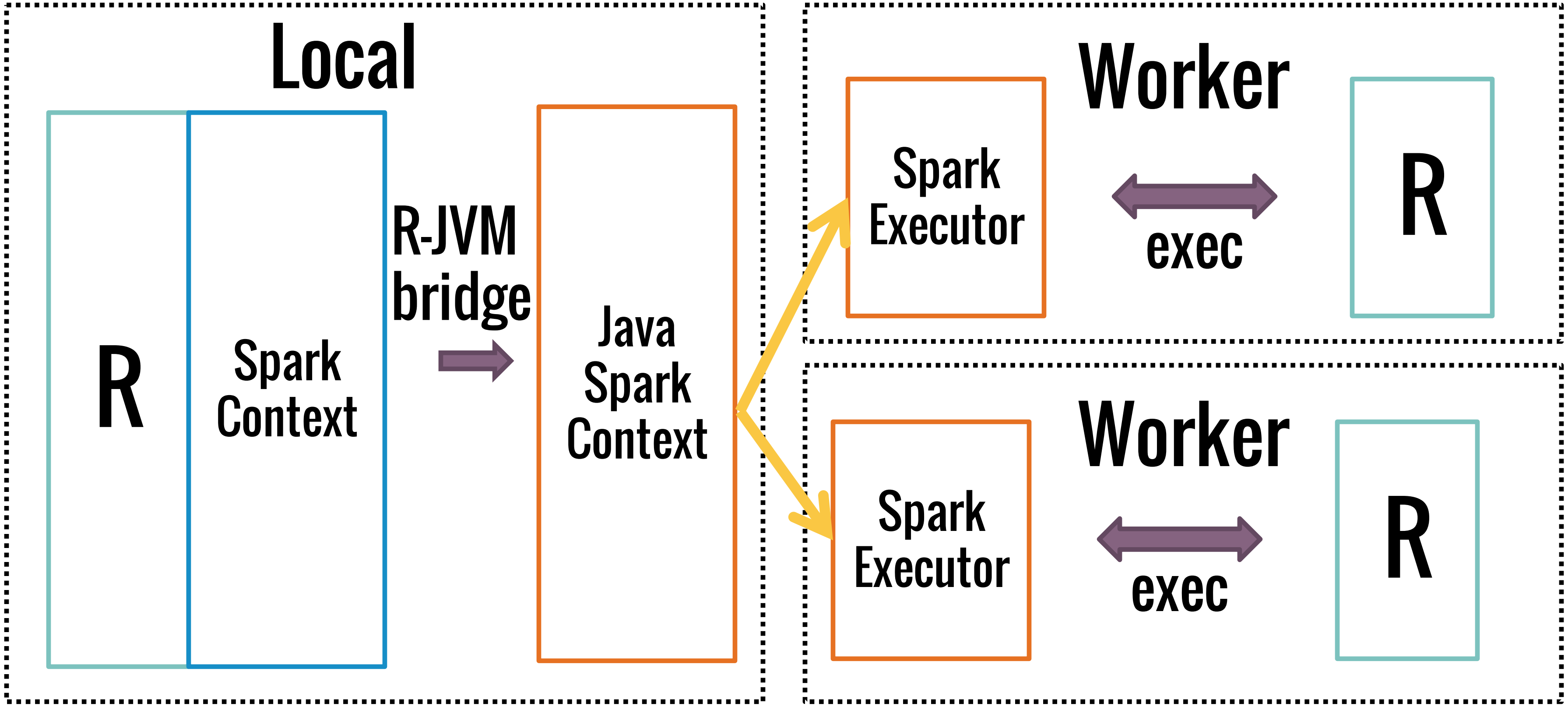
Architecture



Architecture



Architecture



Demo

Demo Overview

Launching SparkR

- On your laptop
- On EC2

SparkR DataFrames

Running SparkR Locally

Download from http://spark.apache.org/ (>1.4.0)

./bin/sparkR or RStudio

Useful for learning SparkR, demonstrations

SparkR on EC2

Launch cluster with Spark’s EC2 scripts

```
./spark-ec2 -s 2 -t r3.xlarge –i <pem> -k <key> sparkr
```

Follow r-bloggers.com/spark-1-4-for-rstudio/

Thanks Vincent Warmerdam !

SparkR Future

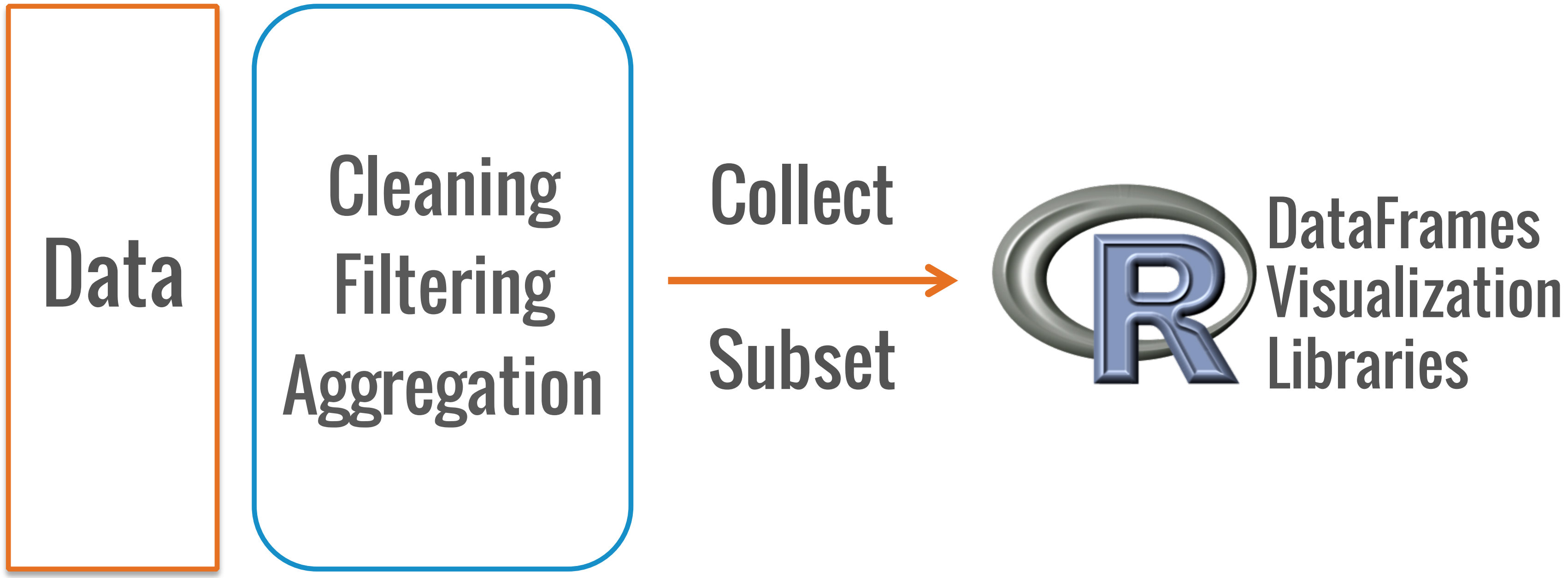
Big Data & R

Big Data
Small Learning

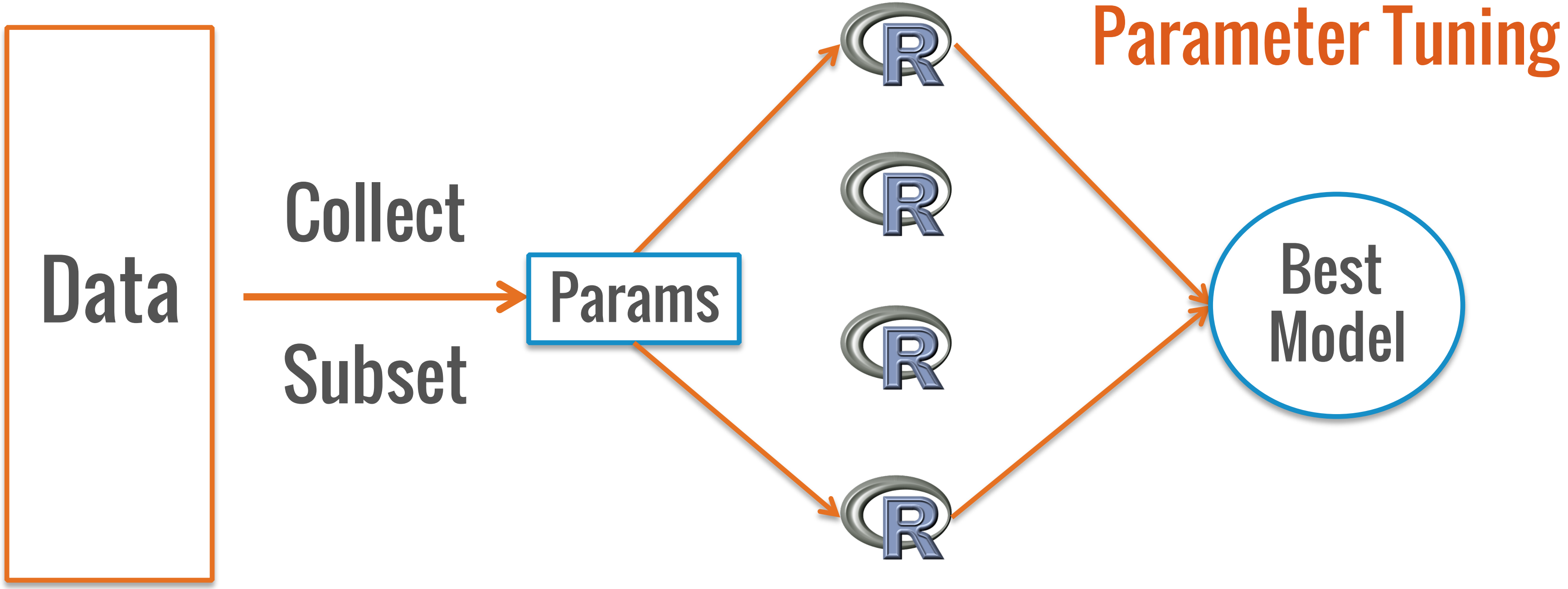
Partition
Aggregate

Large Scale
Machine Learning

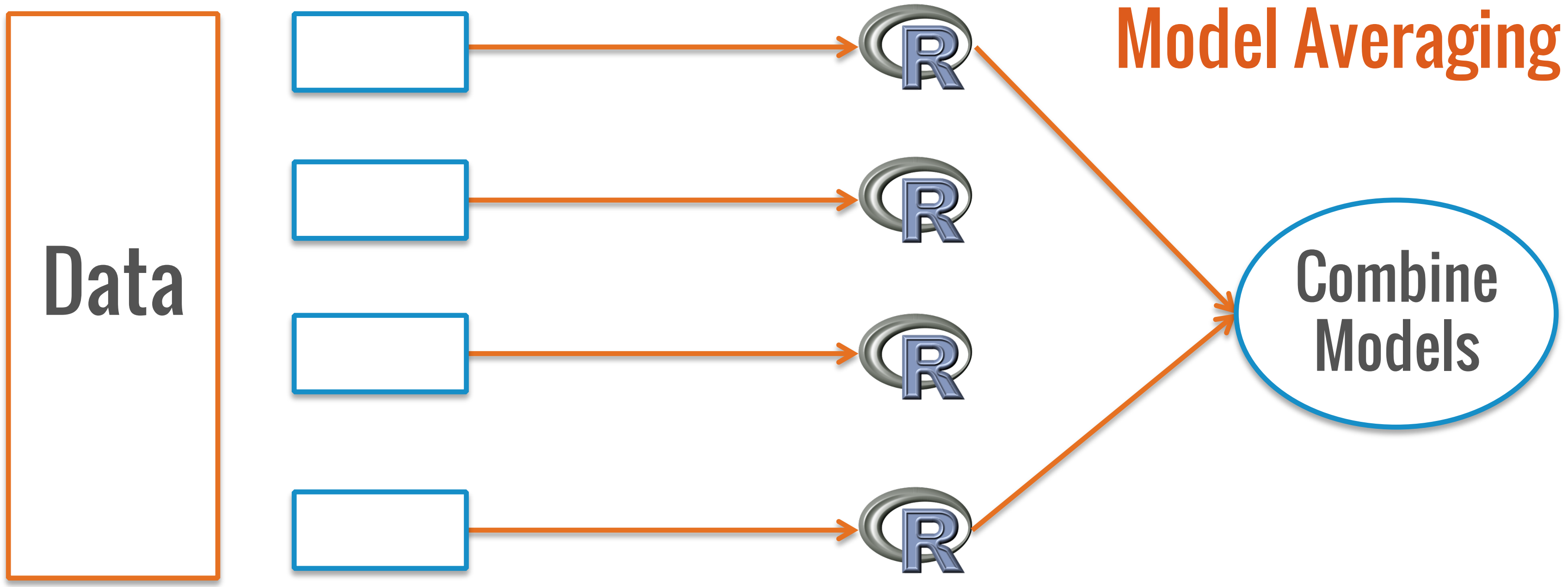
Big Data Processing + R



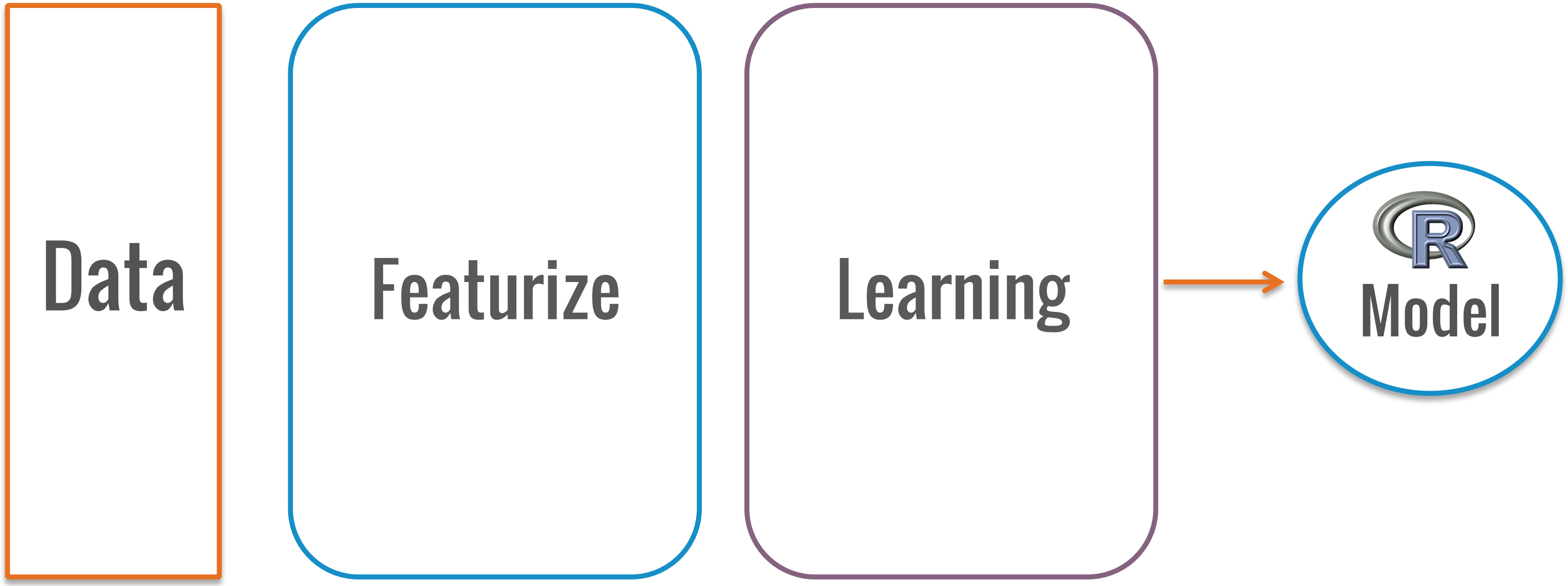
2(a). Partition Aggregate



2(b). Partition Aggregate



3. Large Scale Machine Learning



Big Data & R

Big Data
Small Learning

Partition
Aggregate

Large Scale
Machine Learning

SparkR:
Unified approach

Partition Aggregate

Upcoming feature:

- Simple, parallel API for SparkR
- Ex: Parameter tuning, Model Averaging

- Integrated with DataFrames
- Use existing R packages

Large Scale Machine Learning

Integration with MLLib

Support for GLM, KMeans etc.

```
model <- glm(
    a ~ b + c,
    data = df)
```

Large Scale Machine Learning

Key Features

- DataFrame inputs
- R-like formulas
- Model statistics

```
model <- glm(
    a ~ b + c,
    data = df)

summary(model)
```

Developer Community

>20 contributors including
AMPLab, Databricks, Alteryx, Intel

New contributions welcome !

SparkR

Big data processing from R

DataFrames in Spark 1.4

Future: Large Scale ML & more

```
# Local Demo

Sys.setenv(SPARK_HOME="/Users/shivaram/spark-1.4.1")
.libPaths(c(file.path(Sys.getenv("SPARK_HOME"), "R", "lib"), .libPaths()))
library(SparkR)
sc <- sparkR.init(master="local")
sqlContext <- sparkRSQL.init(sc)

df <- createDataFrame(sqlContext, faithful)
# Select one column
head(select(df, df$eruptions))

# Filter out rows
head(filter(df, df$waiting < 50))

# EC2 Demo

# If you are using Spark 1.4, then launch SparkR with the command
# ./bin/sparkR --packages com.databricks:spark-csv_2.10:1.0.3
# as the `sparkPackages=` flag was only added in Spark 1.4.1.

# # This will work in Spark 1.4.1.
sc <- sparkR.init(spark_link, sparkPackages = "com.databricks:spark-
csv_2.10:1.0.3")
sqlContext <- sparkRSQL.init(sc)

flights <- read.df(sqlContext, "s3n://sparkr-
data/nycflights13.csv","com.databricks.spark.csv", header="true")

# Print the first few rows
head(flights)

# Run a query to print the top 5 most frequent destinations from JFK
jfk_flights <- filter(flights, flights$origin == "JFK")

# Group the flights by destination and aggregate by the number of flights
dest_flights <- agg(group_by(jfk_flights, jfk_flights$dest), count = n(jfk_flights$dest))

# Now sort by the `count` column and print the first few rows
head(arrange(dest_flights, desc(dest_flights$count)))

##  dest count
##1  LAX 11262
##2  SFO  8204
##3  BOS  5898
```

```
# Running SQL Queries
registerTempTable(flights, "flightsTable")
delayDF <- sql(sqlContext, "SELECT dest, arr_delay FROM flightsTable")

# Creating new Columns, Deleting columns
flights$air_time_hr <- flights$air_time / 60
flights$air_time_hr <- NULL

# Combine the whole query into two lines using magrittr
library(magrittr)
dest_flights <- filter(flights, flights$origin == "JFK") %>%
  group_by(flights$dest) %>%
  summarize(count = n(flights$dest))

top_dests <- head(arrange(dest_flights, desc(dest_flights$count)))
barplot(top_dests$count, names.arg = top_dests$dest)
```