

# APPRENTISSAGE DISTRIBUE

- Débugger/Accélérer un programme distribué
- Comprendre la notion de Resilient Distributed Datasets
- Réaliser l'apprentissage distribué d'un modèle de machine learning
- Déployer et administrer une plateforme de calcul distribué dans le cloud
- Réaliser un calcul distribué avec Spark

# Sommaire

- 01 • Contexte, présentation du **projet**.
  - Le client.
  - Son besoin.
  - Notre mission.
- 02 • Présentation des choix **techniques réalisés**.
  - Technologies
  - Présentation technique de l'application
  - Déploiement de l'application sur AWS
  - Améliorations
- 03 • Présentation des **performances obtenues**.
  - Présentation des graphiques.
- 04 • **Conclusion**.

# 01

## PRESENTATION DU PROJET



## Le client

Notre client est la fondation "30 Millions d'Amis", qui lutte contre les abandons, l'expérimentation animale et les trafics d'animaux. Elle cherche à sensibiliser l'opinion et faire évoluer les lois et le statut de l'animal.

# Son besoin

Pour effectuer ses missions, et principalement l'aide aux refuges et la lutte contre le trafic d'animaux, la Fondation souhaite se doter d'un outil permettant de classifier des animaux (chiens et chats) selon leurs races. Cela permettra à des membres de la Fondation sur le terrain d'envoyer des photos dans l'application et de recevoir en retour la race de l'animal. Les data scientist et data architect de la Fondation nous ont fournis un dataset de 7390 images en provenance de 37 classes différentes.

# Notre mission

Nous avons été chargé de créer une application permettant:

- d'extraire des features d'images fournies
  - de réaliser l'apprentissage de modèle
  - de déployer l'application dans le cloud
  - de mesurer les performances
-



# 02

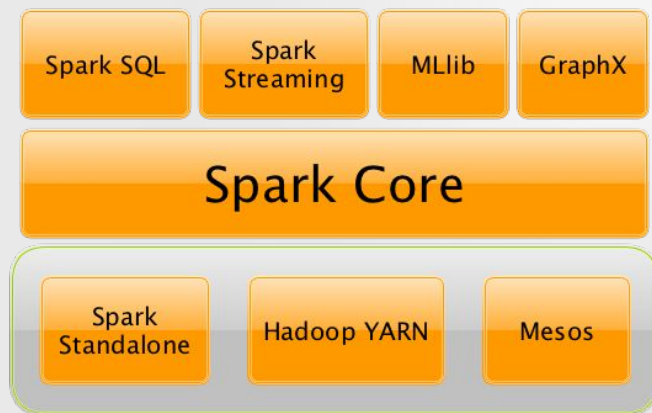
## CHOIX TECHNIQUES

---

# Technologies



- Apache Spark est un framework open source de calcul distribué avec (principalement) un moteur de traitement de données en mémoire pouvant traiter l'analyse, le datapumping, l'apprentissage automatique et le traitement de graphes sur de grands volumes de données stockées (traitement par lots) ou en mouvement. (traitement en continu) avec des API de haut niveau pour les langages de programmation: Scala, Python, Java, R et SQL.

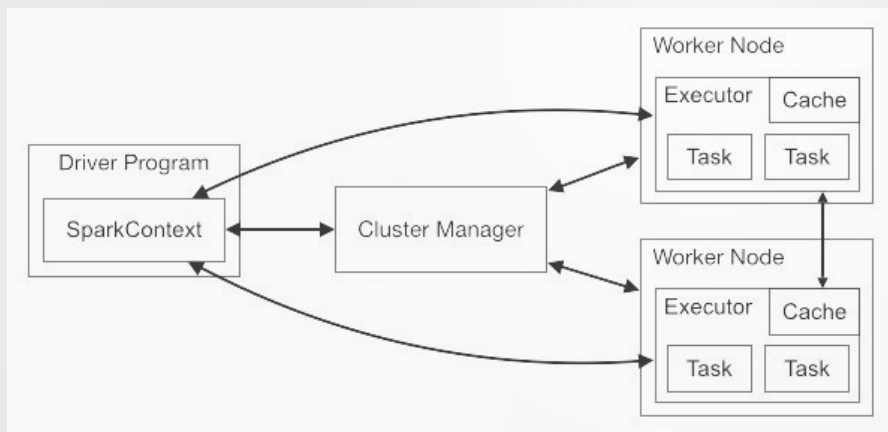




# Technologies



- Les applications Spark s'exécutent en tant qu'ensembles de processus indépendants sur un cluster, coordonnés par l'objet SparkContext dans le programme principal (une application est une instance de SparkContext). Pour s'exécuter sur un cluster, SparkContext peut se connecter à plusieurs types de gestionnaires de cluster (le gestionnaire de cluster autonome de Spark, Mesos ou YARN), qui allouent des ressources entre les applications. Une fois connecté, Spark acquiert les exécuteurs sur les nœuds du cluster, processus qui exécutent des calculs et stockent des données pour l'application. Ensuite, il envoie le code d'application aux exécuteurs. Enfin, SparkContext envoie des tâches aux exécuteurs.



# Technologies



## ■ **RDD** — Resilient Distributed Dataset (architecture et traitement):

Un RDD est une collection résiliente et distribuée d'enregistrements répartis sur une ou plusieurs partitions, c'est le concept central du framework Spark. un RDD représente de la donnée "distribuée", immuable, pouvant être traitée de façon parallèle.

Toute application Spark crée des RDD à partir de certaines entrées, exécute des transformations de ces RDD vers une autre forme, puis exécute des actions pour collecter ou stocker des données.

Il y a deux manières de créer des RDD: en **parallélisant** une collection existante dans le programme, ou en **référéncant un ensemble de données** dans un système de stockage externe, tel qu'un système de fichiers partagé, HDFS, HBase ou toute source de données offrant un format Hadoop compatible.

Les RDD possèdent deux types de méthodes :

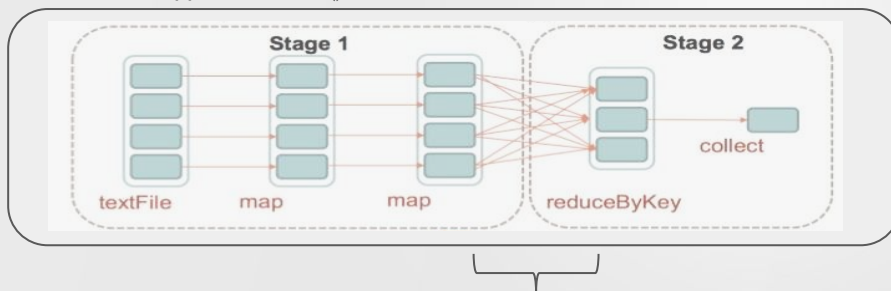
- > Les **transformations** évaluées de manière paresseuse ("lazy evaluation") qui donnent en sortie un autre RDD.
- > Les **actions** qui donnent en sortie... autre chose qu'un RDD.

# Technologies



- **RDD** – Resilient Distributed Dataset (architecture et traitement):
  - > chaque action sur un RDD crée un **job** spark
  - > chaque JOB comprend une succession d'étapes (**stages**) séparées par des **shuffles**
  - > chaque étape contient plusieurs tâches (**tasks**) (la plus petite unité de traitement de données)

JOB créé à l'appel de collect()



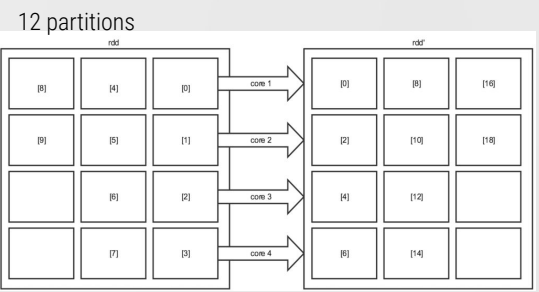
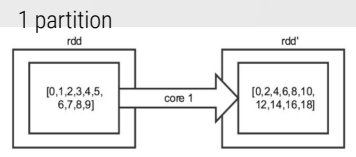
Les opérations pouvant provoquer un shuffle comprennent les opérations de répartition (repartition, coalesce), les opérations 'ByKey' -à l'exception du comptage- (groupByKey, reduceByKey), ainsi que les opérations de jointure (cogroup, join)

# Technologies

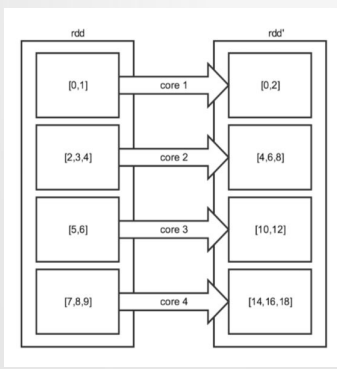


- **RDD** – Resilient Distributed Dataset (architecture et traitement):
  - > chaque tâche s'exécute sur une **partition** (une partition représente tout simplement une partie de la donnée d'un RDD)

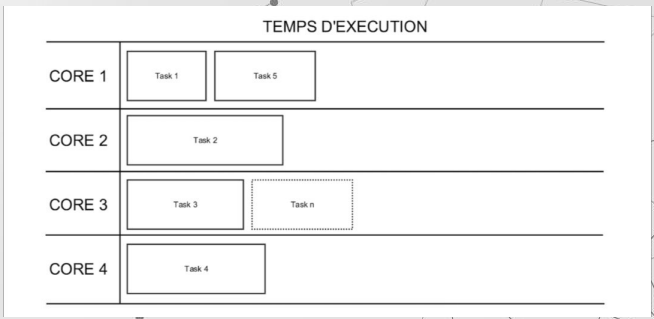
Exemple avec `sc.parallelize(0 to 9).map(_*2)` sur une machine avec 4 cœurs



4 partitions



Nota: 1 core n'exécute qu'une task a la fois

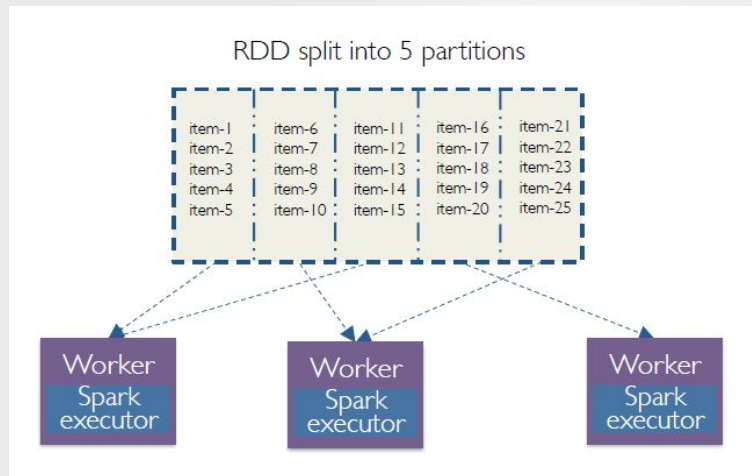


# Technologies



## ■ **RDD** – Resilient Distributed Dataset (architecture et traitement):

-> Les partitions sont réparties sur les **executors** (chacune des partitions sera à la charge d'un executor lors des traitements)



# Présentation technique de l'application

## ■ Extraction de features

L'extraction de features se fait avec le script fourni. Il utilise pour cela une application de la librairie Keras. Cette application est un modèle de deep learning pré-formé. Ce modèle peut être utilisé pour la prédiction et l'extraction de features.

La documentation de l'application utilisée:

<https://keras.io/applications/#extract-features-with-vgg16>



# Présentation technique de l'application

## ■ Application proposée

[Lien vers le notebook Jupyter \(html\) / représentation des données](#)

[Lien vers le script .py](#)

# Déploiement de l'application sur AWS

## ■ Spark en cluster

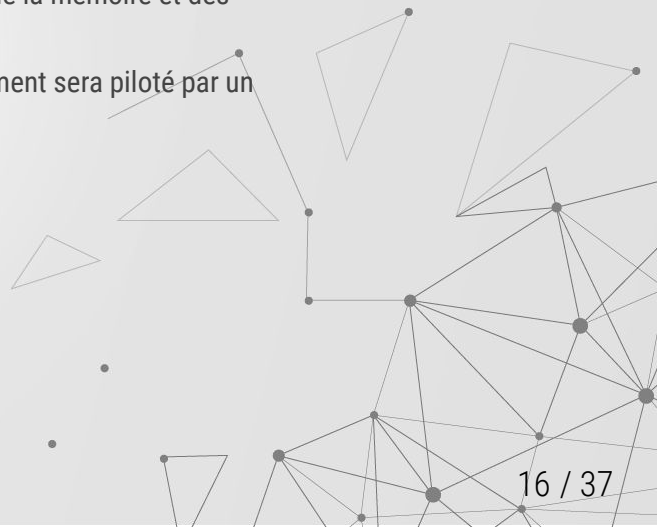
Un cluster Spark se compose d'un **master** et d'un ou plusieurs **workers**. Le cluster doit être démarré et rester actif pour pouvoir exécuter des **applications**.

Le master a pour seule responsabilité la gestion du cluster et il n'exécute donc pas de code MapReduce. Les workers, en revanche, sont les exécuteurs. Ce sont eux qui apportent des ressources au cluster, à savoir de la mémoire et des cœurs de traitement.

Pour exécuter un traitement sur un cluster Spark, il faut soumettre une **application** dont le traitement sera piloté par un **driver**. Deux modes d'exécution sont possibles :

mode client : le driver est créé sur la machine qui soumet l'application.

mode cluster : le driver est créé à l'intérieur du cluster.





# Déploiement de l'application sur AWS

## ■ EMR - S3

**S3** est une solution de stockage d'AWS à proximité des serveurs de calculs

**EMR** (Elastic Map Reduce) permet de coordonner des clusters de calculs **EC2** (Elastic Cloud Compute)

-> Les données ont été portées sur S3

-> Une connection en ssh permet le travail sur EMR en ligne de commande ainsi que l'accès aux outils WEB (Ganglia, Spark Web UI, Gestionnaire de ressources Hadoop...)

-> En ligne de commande, après quelques configurations, l'accès aux commandes s3 (aws s3) est possible (permet de copier par exemple le script main sur la machine)

-> utilisation du sdk d'AWS: boto3 qui fourni une API afin de manager les services aws (S3, EC2). Ici elle a été utilisée pour l'export des fichiers de logs et de résultats vers S3.

# Déploiement de l'application sur AWS

## ■ Lancement de l'application

Avec l'Aws CLI:

Mise a jour des paquetages -> `sudo yum update`

Utilisation de l'environnement python3 -> `sudo sed -i -e '$a\export  
PYSPARK_PYTHON=/usr/bin/python3' /etc/spark/conf/spark-env.sh`

Installation de boto3 -> `python3 -m pip install --user boto3`

Execution -> `spark-submit --conf spark.dynamicAllocation.enabled=false --num-executors 5  
script_fra-oc-p2.py s3://fra-oc-p2/features/ leonberger vs`

Le programme a mis **26 minutes pour du 1 vs 1** (leonberger vs (36 classes))

[Lien vers logs application](#)

[Lien vers tableur résultats](#)

# Déploiement de l'application sur AWS

## Executors

[Show Additional Metrics](#)

### 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) | 0          | 0.0 B / 11.5 GB | 0.0 B     | 8     | 6            | 0            | 352849         | 352855      | 1.2 h (1.8 min)     | 82.2 GB | 187.5 MB     | 187.5 MB      | 0           |
| Dead(0)   | 0          | 0.0 B / 0.0 B   | 0.0 B     | 0     | 0            | 0            | 0              | 0           | 0 ms (0 ms)         | 0.0 B   | 0.0 B        | 0.0 B         | 0           |
| Total(5)  | 0          | 0.0 B / 11.5 GB | 0.0 B     | 8     | 6            | 0            | 352849         | 352855      | 1.2 h (1.8 min)     | 82.2 GB | 187.5 MB     | 187.5 MB      | 0           |

### Executors

Show  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   |
|-------------|---|--------|------------|------------------|-----------|-------|--------------|--------------|----------------|-------------|---------------------|---------|--------------|---------------|--|
| driver      | ip-172-31-25-255.eu-west-1.compute.internal:43347 | Active | 0          | 0.0 B / 434.6 MB | 0.0 B     | 0     | 0            | 0            | 0              | 0           | 0 ms (0 ms)         | 0.0 B   | 0.0 B        | 0.0 B         |  |
| 1           | ip-172-31-27-116.eu-west-1.compute.internal:36651 | Active | 0          | 0.0 B / 2.8 GB   | 0.0 B     | 2     | 1            | 0            | 88245          | 88246       | 17 min (35 s)       | 15.6 GB | 43.6 MB      | 44.8 MB       | <a href="#">stdout</a><br><a href="#">stderr</a> |
| 2           | ip-172-31-17-188.eu-west-1.compute.internal:43375 | Active | 0          | 0.0 B / 2.8 GB   | 0.0 B     | 2     | 2            | 0            | 88293          | 88295       | 18 min (22 s)       | 34.2 GB | 50.4 MB      | 53.2 MB       | <a href="#">stdout</a><br><a href="#">stderr</a> |
| 3           | ip-172-31-17-116.eu-west-1.compute.internal:45033 | Active | 0          | 0.0 B / 2.8 GB   | 0.0 B     | 2     | 0            | 0            | 88078          | 88078       | 17 min (26 s)       | 16.2 GB | 44.2 MB      | 43.6 MB       | <a href="#">stdout</a><br><a href="#">stderr</a> |
| 4           | ip-172-31-17-188.eu-west-1.compute.internal:41065 | Active | 0          | 0.0 B / 2.8 GB   | 0.0 B     | 2     | 3            | 0            | 88233          | 88236       | 18 min (23 s)       | 16.3 GB | 49.3 MB      | 45.9 MB       | <a href="#">stdout</a><br><a href="#">stderr</a> |

Showing 1 to 5 of 5 entries

[Previous](#) [1](#) [Next](#)

# Déploiement de l'application sur AWS



## Nodes of the cluster

- Cluster
- About
- Nodes
- Node Labels
- Applications
- NEW
- NEW SAVING
- SUBMITTED
- ACCEPTED
- RUNNING
- FINISHED
- FAILED
- KILLED
- Scheduler
- Tools

**Cluster Metrics**

| Apps Submitted | Apps Pending | Apps Running | Apps Completed | Containers Running | Memory Used | Memory Total | Memory Reserved | VCores Used | VCores Total | VCores Reserved |
|----------------|--------------|--------------|----------------|--------------------|-------------|--------------|-----------------|-------------|--------------|-----------------|
| 17             | 0            | 0            | 17             | 0                  | 0 B         | 24 GB        | 0 B             | 0           | 8            | 0               |

**Cluster Nodes Metrics**

| Active Nodes | Decommissioning Nodes | Decommissioned Nodes | Lost Nodes | Unhealthy Nodes | Rebooted Nodes | Shutdown Nodes |
|--------------|-----------------------|----------------------|------------|-----------------|----------------|----------------|
| 2            | 0                     | 0                    | 0          | 0               | 0              | 0              |

**Scheduler Metrics**

| Scheduler Type     | Scheduling Resource Type | Minimum Allocation    | Maximum Allocation       | Maximum Cluster Application Priority |
|--------------------|--------------------------|-----------------------|--------------------------|--------------------------------------|
| Capacity Scheduler | [MEMORY]                 | <memory:32, vCores:1> | <memory:12288, vCores:4> | 0                                    |

Show 20 entries

| Node Labels | Rack          | Node State | Node Address                                     | Node HTTP Address                                | Last health-update             | Health-report | Containers | Mem Used | Mem Avail | VCores Used | VCores Avail | Version      |
|-------------|---------------|------------|--|--|--------------------------------|---------------|------------|----------|-----------|-------------|--------------|--------------|
| CORE        | /default-rack | RUNNING    | ip-172-31-17-188.eu-west-1.compute.internal:8041 | ip-172-31-17-188.eu-west-1.compute.internal:8042 | Fri Oct 04 13:09:40 +0000 2019 |               | 0          | 0 B      | 12 GB     | 0           | 4            | 2.8.5-amzn-4 |
| CORE        | /default-rack | RUNNING    | ip-172-31-27-116.eu-west-1.compute.internal:8041 | ip-172-31-27-116.eu-west-1.compute.internal:8042 | Fri Oct 04 13:09:50 +0000 2019 |               | 0          | 0 B      | 12 GB     | 0           | 4            | 2.8.5-amzn-4 |

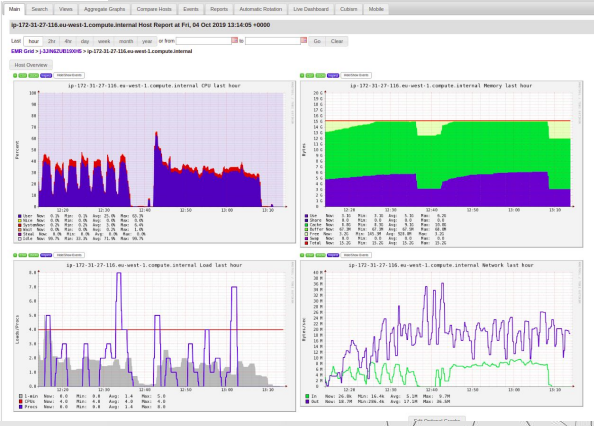
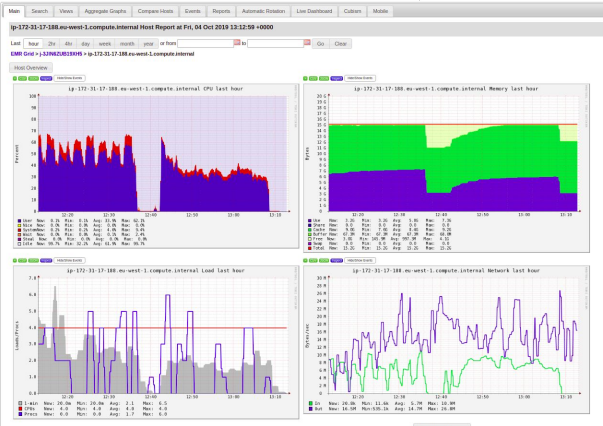
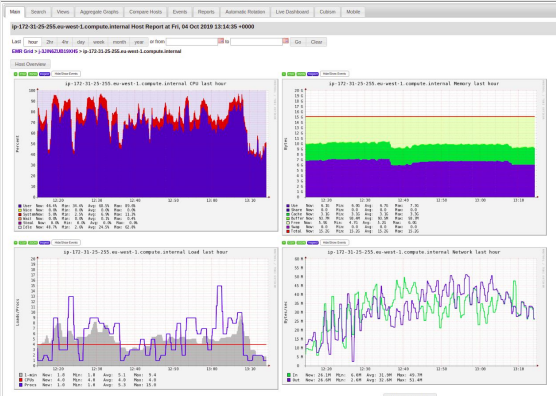
Showing 1 to 2 of 2 entries

# Déploiement de l'application sur AWS

MASTER

Worker1

Worker2

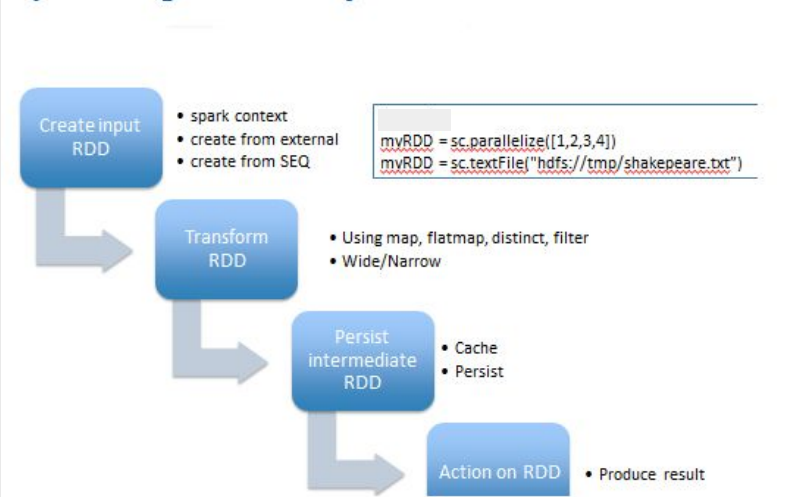


# Améliorations

## ■ Mise en cache (.persist())

La mise en cache de df\_data dans le script permet au RDD d'être stocké en mémoire lors de sa première action ce qui optimise notre script.

### Spark Program Flow by RDD



# Améliorations

## ■ Partitions / Allocation dynamique

Sans re-partitionnement, 1 seul executor est utilisé (cf schéma). Rajout dans le script: `rdd_raw_data = sc.wholeTextFiles(sys.argv[1:][0]+'*.json', minPartitions=24)`

De plus, en paramétrant l'allocation dynamique à "false", les executors ne sont pas 'killed' après 60 secondes de non utilisation. (INFO console= Removing executor 2 because it has been idle for 60 seconds)

Apache Spark 2.4.4 Jobs Stages Storage Environment Executors SQL projet2\_svm\_whith\_sgd application UI

### Executors

[Show Additional Metrics](#)

#### 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) | 0          | 0.0 B / 11.5 GB | 0.0 B     | 8     | 0            | 0            | 47089          | 47089       | 30 min (2.0 min)    | 156.7 GB | 58 KB        | 58 KB         | 0           |
| Dead(0)   | 0          | 0.0 B / 0.0 B   | 0.0 B     | 0     | 0            | 0            | 0              | 0           | 0 ms (0 ms)         | 0.0 B    | 0.0 B        | 0.0 B         | 0           |
| Total(5)  | 0          | 0.0 B / 11.5 GB | 0.0 B     | 8     | 0            | 0            | 47089          | 47089       | 30 min (2.0 min)    | 156.7 GB | 58 KB        | 58 KB         | 0           |

#### Executors

Show 20 entries


| 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   |
|-------------|---|--------|------------|------------------|-----------|-------|--------------|--------------|----------------|-------------|---------------------|----------|--------------|---------------|--|
| driver      | ip-172-31-25-255.eu-west-1.compute.internal:37501 | Active | 0          | 0.0 B / 434.6 MB | 0.0 B     | 0     | 0            | 0            | 0              | 0           | 0 ms (0 ms)         | 0.0 B    | 0.0 B        | 0.0 B         |  |
| 1           | ip-172-31-27-116.eu-west-1.compute.internal:37501 | Active | 0          | 0.0 B / 2.8 GB   | 0.0 B     | 2     | 0            | 0            | 154            | 154         | 7 s (0.3 s)         | 0.0 B    | 31.1 KB      | 0.0 B         | <a href="#">stdout</a><br><a href="#">stderr</a> |
| 2           | ip-172-31-17-188.eu-west-1.compute.internal:35543 | Active | 0          | 0.0 B / 2.8 GB   | 0.0 B     | 2     | 0            | 0            | 0              | 0           | 0 ms (0 ms)         | 0.0 B    | 0.0 B        | 0.0 B         | <a href="#">stdout</a><br><a href="#">stderr</a> |
| 3           | ip-172-31-27-116.eu-west-1.compute.internal:35235 | Active | 0          | 0.0 B / 2.8 GB   | 0.0 B     | 2     | 0            | 0            | 46935          | 46935       | 30 min (2.0 min)    | 156.7 GB | 26.9 KB      | 58 KB         | <a href="#">stdout</a><br><a href="#">stderr</a> |
| 4           | ip-172-31-17-188.eu-west-1.compute.internal:40325 | Active | 0          | 0.0 B / 2.8 GB   | 0.0 B     | 2     | 0            | 0            | 0              | 0           | 0 ms (0 ms)         | 0.0 B    | 0.0 B        | 0.0 B         | <a href="#">stdout</a><br><a href="#">stderr</a> |

Showing 1 to 5 of 5 entries [Previous](#) [Next](#)

# Améliorations

## ■ Executors (min)

Sans spécifier `--num-executors 5` : (durée 30 min soit +5min)



[Jobs](#)
[Stages](#)
[Storage](#)
[Environment](#)
[Executors](#)
[SQL](#)

projet2\_svm\_whith\_sgd application UI

### Executors

[Show Additional Metrics](#)

#### 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(3) | 0          | 0.0 B / 6 GB   | 0.0 B     | 4     | 0            | 0            | 619237         | 619237      | 1.4 h (1.4 min)     | 144.2 GB | 325.1 MB     | 325.1 MB      | 0           |
| Dead(0)   | 0          | 0.0 B / 0.0 B  | 0.0 B     | 0     | 0            | 0            | 0              | 0           | 0 ms (0 ms)         | 0.0 B    | 0.0 B        | 0.0 B         | 0           |
| Total(3)  | 0          | 0.0 B / 6 GB   | 0.0 B     | 4     | 0            | 0            | 619237         | 619237      | 1.4 h (1.4 min)     | 144.2 GB | 325.1 MB     | 325.1 MB      | 0           |

#### Executors

Show  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   |
|-------------|---|--------|------------|------------------|-----------|-------|--------------|--------------|----------------|-------------|---------------------|---------|--------------|---------------|--|
| driver      | ip-172-31-25-255.eu-west-1.compute.internal:33095 | Active | 0          | 0.0 B / 434.6 MB | 0.0 B     | 0     | 0            | 0            | 0              | 0           | 0 ms (0 ms)         | 0.0 B   | 0.0 B        | 0.0 B         |  |
| 1           | ip-172-31-17-188.eu-west-1.compute.internal:34307 | Active | 0          | 0.0 B / 2.8 GB   | 0.0 B     | 2     | 0            | 0            | 309336         | 309336      | 42 min (41 s)       | 89.5 GB | 164 MB       | 166.9 MB      | <a href="#">stdout</a><br><a href="#">stderr</a> |
| 2           | ip-172-31-27-116.eu-west-1.compute.internal:46323 | Active | 0          | 0.0 B / 2.8 GB   | 0.0 B     | 2     | 0            | 0            | 309901         | 309901      | 43 min (41 s)       | 54.7 GB | 161.1 MB     | 158.2 MB      | <a href="#">stdout</a><br><a href="#">stderr</a> |

Showing 1 to 3 of 3 entries
 

[Previous](#)
[1](#)
[Next](#)



# Améliorations

■ **Etat de l'application sans paramètre supplémentaire et avec le partitionnement par défaut:**

spark-submit script\_fra-oc-p2.py s3://fra-oc-p2/features/ leonberger vs (durée 21 min soit très proche de l'application optimisée finale). Mais les calculs sont mal ou peu distribués.

Executors

[Show Additional Metrics](#)

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(2) | 0          | 0.0 B / 3.2 GB  | 0.0 B     | 2     | 0            | 0            | 47088          | 47088       | 29 min (1.8 min)    | 156.7 GB | 58 KB        | 58 KB         | 0           |
| Dead(3)   | 0          | 0.0 B / 8.3 GB  | 0.0 B     | 6     | 0            | 0            | 1              | 1           | 4 s (0.3 s)         | 0.0 B    | 0.0 B        | 0.0 B         | 0           |
| Total(5)  | 0          | 0.0 B / 11.5 GB | 0.0 B     | 8     | 0            | 0            | 47089          | 47089       | 29 min (1.9 min)    | 156.7 GB | 58 KB        | 58 KB         | 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   |
|-------------|---|--------|------------|------------------|-----------|-------|--------------|--------------|----------------|-------------|---------------------|----------|--------------|---------------|--|
| driver      | ip-172-31-25-255.eu-west-1.compute.internal:43129 | Active | 0          | 0.0 B / 434.6 MB | 0.0 B     | 0     | 0            | 0            | 0              | 0           | 0 ms (0 ms)         | 0.0 B    | 0.0 B        | 0.0 B         |  |
| 1           | ip-172-31-17-188.eu-west-1.compute.internal:36079 | Dead   | 0          | 0.0 B / 2.8 GB   | 0.0 B     | 2     | 0            | 0            | 1              | 1           | 4 s (0.3 s)         | 0.0 B    | 0.0 B        | 0.0 B         | <a href="#">stdout</a><br><a href="#">stderr</a> |
| 2           | ip-172-31-27-116.eu-west-1.compute.internal:33671 | Dead   | 0          | 0.0 B / 2.8 GB   | 0.0 B     | 2     | 0            | 0            | 0              | 0           | 0 ms (0 ms)         | 0.0 B    | 0.0 B        | 0.0 B         | <a href="#">stdout</a><br><a href="#">stderr</a> |
| 3           | ip-172-31-17-188.eu-west-1.compute.internal:37461 | Active | 0          | 0.0 B / 2.8 GB   | 0.0 B     | 2     | 0            | 0            | 47088          | 47088       | 29 min (1.8 min)    | 156.7 GB | 58 KB        | 58 KB         | <a href="#">stdout</a><br><a href="#">stderr</a> |
| 4           | ip-172-31-27-116.eu-west-1.compute.internal:35635 | Dead   | 0          | 0.0 B / 2.8 GB   | 0.0 B     | 2     | 0            | 0            | 0              | 0           | 0 ms (0 ms)         | 0.0 B    | 0.0 B        | 0.0 B         | <a href="#">stdout</a><br><a href="#">stderr</a> |

Showing 1 to 5 of 5 entries

[Previous](#) [1](#) [Next](#)

# 03

## PERFORMANCES

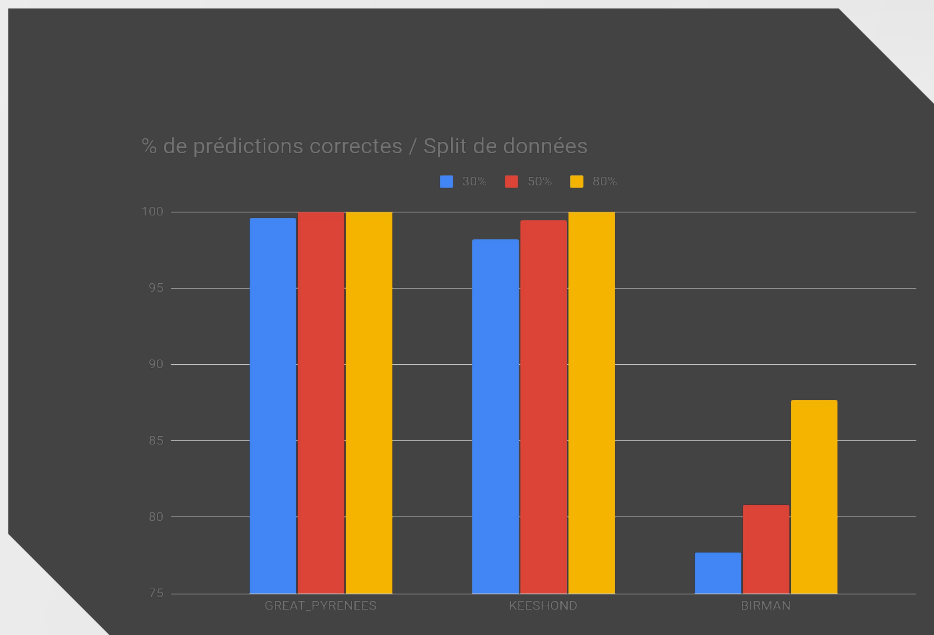
---

# Performances

## PERFORMANCES OBTENUES

| class 1 | class 2        | split | iteration | num_model | predict% | nb_train | nb_test | nb_correct |
|---------|----------------|-------|-----------|-----------|----------|----------|---------|------------|
| Ragdoll | great pyrenees | 0,3   | 100       | 1         | 99,64    | 122      | 278     | 277        |
| Ragdoll | great pyrenees | 0,5   | 100       | 2         | 100,00   | 212      | 188     | 188        |
| Ragdoll | great pyrenees | 0,8   | 100       | 3         | 100,00   | 327      | 73      | 73         |
| Ragdoll | keeshond       | 0,3   | 100       | 1         | 98,20    | 122      | 278     | 273        |
| Ragdoll | keeshond       | 0,5   | 100       | 2         | 99,47    | 212      | 188     | 187        |
| Ragdoll | keeshond       | 0,8   | 100       | 3         | 100,00   | 328      | 72      | 72         |
| Ragdoll | Birman         | 0,3   | 100       | 1         | 77,70    | 122      | 278     | 216        |
| Ragdoll | Birman         | 0,5   | 100       | 2         | 80,85    | 212      | 188     | 152        |
| Ragdoll | Birman         | 0,8   | 100       | 3         | 87,67    | 327      | 73      | 64         |
| Siamese | All            | 0,3   | 100       | 1         | 98,31    | 2244     | 5146    | 5059       |
| Siamese | All            | 0,5   | 100       | 2         | 98,42    | 3708     | 3682    | 3624       |
| Siamese | All            | 0,8   | 100       | 3         | 98,67    | 5884     | 1506    | 1486       |

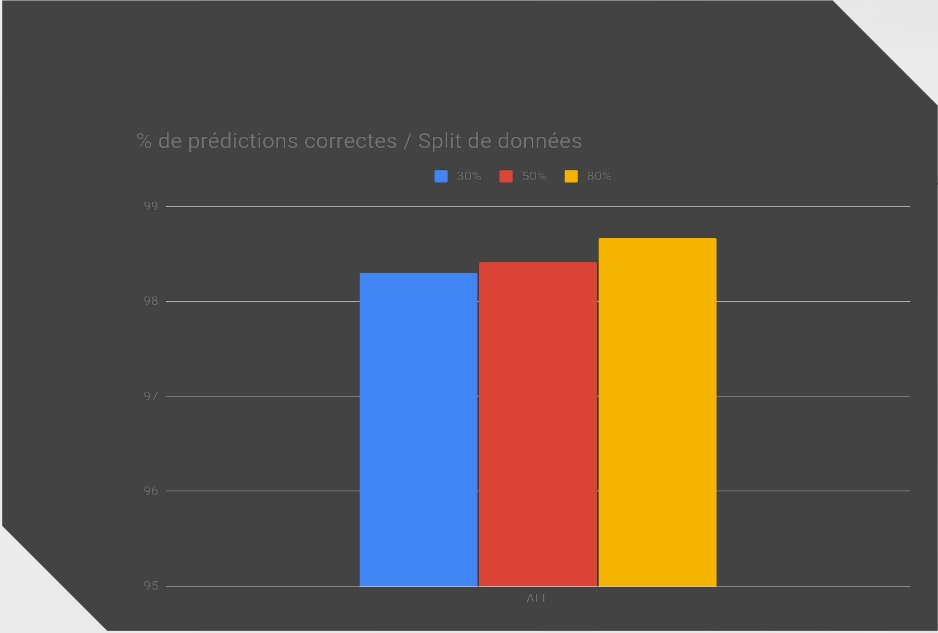
# Performances



## Ragdoll VS ...

On observe que le taux de prédictions correctes s'améliore lorsque les données d'entraînement augmentent.

# Performances



## Siamese VS All

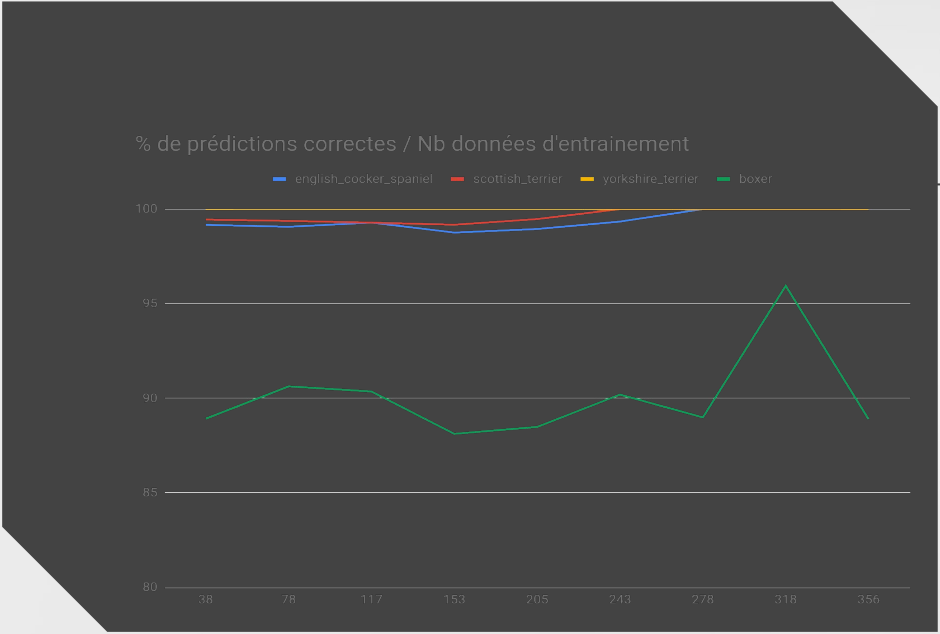
# Performances

## PERFORMANCES OBTENUES

Split 10/90 -> 90/10

| class 1          | class 2                | split | iteration | num_model | predict% | nb_train | nb_test | nb_correct |
|------------------|------------------------|-------|-----------|-----------|----------|----------|---------|------------|
| american_bulldog | english_cocker_spaniel | 0.1   | 100       | 1         | 99,17    | 39       | 361     | 358        |
| american_bulldog | english_cocker_spaniel | 0.2   | 100       | 2         | 99,06    | 81       | 319     | 316        |
| american_bulldog | english_cocker_spaniel | 0.3   | 100       | 3         | 99,29    | 120      | 280     | 278        |
| american_bulldog | english_cocker_spaniel | 0.4   | 100       | 4         | 98,76    | 158      | 242     | 239        |
| american_bulldog | english_cocker_spaniel | 0.5   | 100       | 5         | 98,94    | 211      | 189     | 187        |
| american_bulldog | english_cocker_spaniel | 0.6   | 100       | 6         | 99,34    | 249      | 151     | 150        |
| american_bulldog | english_cocker_spaniel | 0.7   | 100       | 7         | 100,00   | 286      | 114     | 114        |
| american_bulldog | english_cocker_spaniel | 0.8   | 100       | 8         | 100,00   | 327      | 73      | 73         |
| american_bulldog | english_cocker_spaniel | 0.9   | 100       | 9         | 100,00   | 367      | 33      | 33         |
| american_bulldog | scottish_terrier       | 0.1   | 100       | 1         | 99,44    | 39       | 360     | 358        |
| american_bulldog | scottish_terrier       | 0.2   | 100       | 2         | 99,37    | 81       | 318     | 316        |
| american_bulldog | scottish_terrier       | 0.3   | 100       | 3         | 99,28    | 121      | 278     | 276        |
| american_bulldog | scottish_terrier       | 0.4   | 100       | 4         | 99,17    | 157      | 242     | 240        |
| american_bulldog | scottish_terrier       | 0.5   | 100       | 5         | 99,47    | 209      | 190     | 189        |
| american_bulldog | scottish_terrier       | 0.6   | 100       | 6         | 100,00   | 247      | 152     | 152        |
| american_bulldog | scottish_terrier       | 0.7   | 100       | 7         | 100,00   | 282      | 117     | 117        |
| american_bulldog | scottish_terrier       | 0.8   | 100       | 8         | 100,00   | 325      | 74      | 74         |
| american_bulldog | scottish_terrier       | 0.9   | 100       | 9         | 100,00   | 363      | 36      | 36         |
| american_bulldog | yorkshire_terrier      | 0.1   | 100       | 1         | 100,00   | 39       | 361     | 361        |
| american_bulldog | yorkshire_terrier      | 0.2   | 100       | 2         | 100,00   | 80       | 320     | 320        |
| american_bulldog | yorkshire_terrier      | 0.3   | 100       | 3         | 100,00   | 120      | 280     | 280        |
| american_bulldog | yorkshire_terrier      | 0.4   | 100       | 4         | 100,00   | 157      | 243     | 243        |
| american_bulldog | yorkshire_terrier      | 0.5   | 100       | 5         | 100,00   | 210      | 190     | 190        |
| american_bulldog | yorkshire_terrier      | 0.6   | 100       | 6         | 100,00   | 248      | 152     | 152        |
| american_bulldog | yorkshire_terrier      | 0.7   | 100       | 7         | 100,00   | 282      | 118     | 118        |
| american_bulldog | yorkshire_terrier      | 0.8   | 100       | 8         | 100,00   | 326      | 74      | 74         |
| american_bulldog | yorkshire_terrier      | 0.9   | 100       | 9         | 100,00   | 364      | 36      | 36         |
| american_bulldog | boxer                  | 0.1   | 100       | 1         | 88,92    | 39       | 361     | 321        |
| american_bulldog | boxer                  | 0.2   | 100       | 2         | 90,63    | 80       | 320     | 290        |
| american_bulldog | boxer                  | 0.3   | 100       | 3         | 90,36    | 120      | 280     | 253        |
| american_bulldog | boxer                  | 0.4   | 100       | 4         | 88,11    | 156      | 244     | 215        |
| american_bulldog | boxer                  | 0.5   | 100       | 5         | 88,48    | 209      | 191     | 169        |
| american_bulldog | boxer                  | 0.6   | 100       | 6         | 90,20    | 247      | 153     | 138        |
| american_bulldog | boxer                  | 0.7   | 100       | 7         | 88,98    | 282      | 118     | 105        |
| american_bulldog | boxer                  | 0.8   | 100       | 8         | 95,95    | 326      | 74      | 71         |
| american_bulldog | boxer                  | 0.9   | 100       | 9         | 88,89    | 364      | 36      | 32         |

# Performances



American\_bulldog VS ...





# Performances

## Logs de l'application (extrait)

```

params OK
Get data - OK - (0.88 sec.)
Transform data - OK - (0.01 sec.)
Generate dataframe - OK - (6.35 sec.)
vs_list:
['english_cocker_spaniel', 'scottish_terrier', 'yorkshire_terrier', 'boxer']

*****
Preparing for "american_bulldog" vs "english_cocker_spaniel"...

  class1:"american_bulldog", nb rows=200
  class2:"english_cocker_spaniel", nb rows=200

... ready !

*** Split= 10.00 / 90.00 (39 rows in train dataset, 361 rows in test dataset)
... Evaluating model #1 (split:10.00 %, iteration:100)
--> correct prediction: 99.17 % (358 corrects / 361 test images)

*** Split= 20.00 / 80.00 (81 rows in train dataset, 319 rows in test dataset)
... Evaluating model #2 (split:20.00 %, iteration:100)
--> correct prediction: 99.06 % (316 corrects / 319 test images)

*** Split= 30.00 / 70.00 (120 rows in train dataset, 280 rows in test dataset)
... Evaluating model #3 (split:30.00 %, iteration:100)
--> correct prediction: 99.29 % (278 corrects / 280 test images)

*** Split= 40.00 / 60.00 (158 rows in train dataset, 242 rows in test dataset)
... Evaluating model #4 (split:40.00 %, iteration:100)
--> correct prediction: 98.76 % (239 corrects / 242 test images)

*** Split= 50.00 / 50.00 (211 rows in train dataset, 189 rows in test dataset)
... Evaluating model #5 (split:50.00 %, iteration:100)
--> correct prediction: 98.94 % (187 corrects / 189 test images)

*** Split= 60.00 / 40.00 (249 rows in train dataset, 151 rows in test dataset)
... Evaluating model #6 (split:60.00 %, iteration:100)
--> correct prediction: 99.34 % (150 corrects / 151 test images)

*** Split= 70.00 / 30.00 (286 rows in train dataset, 114 rows in test dataset)
... Evaluating model #7 (split:70.00 %, iteration:100)
--> correct prediction: 100.00 % (114 corrects / 114 test images)

*** Split= 80.00 / 20.00 (327 rows in train dataset, 73 rows in test dataset)
... Evaluating model #8 (split:80.00 %, iteration:100)
--> correct prediction: 100.00 % (73 corrects / 73 test images)

*** Split= 90.00 / 10.00 (367 rows in train dataset, 33 rows in test dataset)
... Evaluating model #9 (split:90.00 %, iteration:100)
--> correct prediction: 100.00 % (33 corrects / 33 test images)

```

```

class2:"boxer", nb rows=280

... ready !

*** Split= 10.00 / 90.00 (39 rows in train dataset, 361 rows in test dataset)
... Evaluating model #1 (split:10.00 %, iteration:100)
--> correct prediction: 88.92 % (321 corrects / 361 test images)

*** Split= 20.00 / 80.00 (80 rows in train dataset, 320 rows in test dataset)
... Evaluating model #2 (split:20.00 %, iteration:100)
--> correct prediction: 90.62 % (290 corrects / 320 test images)

*** Split= 30.00 / 70.00 (120 rows in train dataset, 280 rows in test dataset)
... Evaluating model #3 (split:30.00 %, iteration:100)
--> correct prediction: 90.36 % (253 corrects / 280 test images)

*** Split= 40.00 / 60.00 (156 rows in train dataset, 244 rows in test dataset)
... Evaluating model #4 (split:40.00 %, iteration:100)
--> correct prediction: 88.11 % (215 corrects / 244 test images)

*** Split= 50.00 / 50.00 (209 rows in train dataset, 191 rows in test dataset)
... Evaluating model #5 (split:50.00 %, iteration:100)
--> correct prediction: 88.48 % (169 corrects / 191 test images)

*** Split= 60.00 / 40.00 (247 rows in train dataset, 153 rows in test dataset)
... Evaluating model #6 (split:60.00 %, iteration:100)
--> correct prediction: 90.20 % (138 corrects / 153 test images)

*** Split= 70.00 / 30.00 (282 rows in train dataset, 118 rows in test dataset)
... Evaluating model #7 (split:70.00 %, iteration:100)
--> correct prediction: 88.98 % (105 corrects / 118 test images)

*** Split= 80.00 / 20.00 (326 rows in train dataset, 74 rows in test dataset)
... Evaluating model #8 (split:80.00 %, iteration:100)
--> correct prediction: 95.95 % (71 corrects / 74 test images)

*** Split= 90.00 / 10.00 (364 rows in train dataset, 36 rows in test dataset)
... Evaluating model #9 (split:90.00 %, iteration:100)
--> correct prediction: 88.89 % (32 corrects / 36 test images)

Best prediction: Model #8 whith 95.95 % of success !
Model params: split: 80.00%, iterations: 100
Save Model to: ./models/american_bulldog_VS_boxer/pythonSVMwithSGDModel ... Model saved

Image saved in : Accuracy_american_bulldog_VS_boxer.png
Classification "american_bulldog" vs "boxer" - END -
Took 71.45 sec.

***** END *****
Program took 427.91 sec. to perform.

... You can go to webUI ...

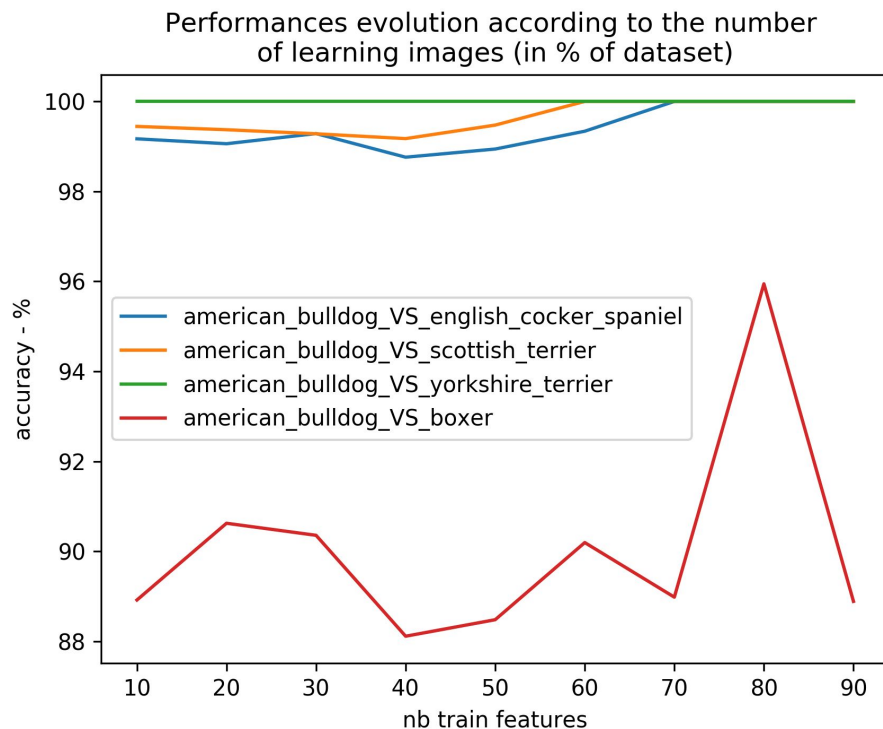
Press ctrl+c to exit

```



# Performances

## Graphique pyplot (matplotlib) généré



# 04

## CONCLUSION

---

# Un projet de machine learning

---

- Utilisation d'applications de librairies de ML
  - Réaliser, débbugger, accélérer un programme distribué avec Spark
  - Resilient Distributed Datasets
  - Apprentissage distribué d'un modèle de machine learning
  - Déployer et administrer une plateforme de calcul distribué dans le cloud
-

# Ressources

## Web

- <https://spark.apache.org/>
- <https://keras.io/models/about-keras-models/#about-keras-models>
- <http://imagenet.stanford.edu/synset?wnid=n02094433>
- <https://machinelearningmastery.com/use-pre-trained-vgg-model-classify-objects-photographs/>
- <https://spark.apache.org/docs/2.2.0/mllib-data-types.html>
- <https://meritis.fr/bigdata/introduction-partitioning-spark/>
- <https://blog.ippon.fr/2014/11/20/utiliser-apache-spark-en-cluster/amp/>
- <https://jaceklaskowski.gitbooks.io/mastering-apache-spark/spark-rdd-transformations.html>
- <https://blog.univalence.io/shuffle-dans-spark-reducebykey-vs-groupbykey/>
- <https://www.slideshare.net/LisaHua/spark-overview-37479609>
- <https://medium.com/@thejasbabu/spark-under-the-hood-partition-d386aaa26b7>
- <https://umbertogriffo.gitbooks.io/apache-spark-best-practices-and-tuning>

Et bien d'autres...



# MERCI

Avez-vous des **questions**?

f2buttet@gmail.com

06.84.19.58.69