BigData TECHCON

San Francisco Bay Area • Oct. 27-29, 2014

"I want to die on Mars but not on impact"
– Elon Musk, interview with Chris Anderson

"There are no facts, only interpretations." - Friedrich Nietzsche

"The shrewd guess, the fertile hypothesis, the courageous leap to a tentative conclusion – these are the most valuable coin of the thinker at work" -- Jerome Seymour Bruner

The Hitchhiker's Guide to Machine Learning with Python & Apache Spark

@ksankar // doubleclix.wordpress.com

http://www.bigdatatechcon.com/classes.html#TheHitchhikersGuidetoMachineLearningwithPythonandApacheSparkPartI

October 29, 2014

Agenda

- Spark & Data Science DevOps
  - *Spark, Python & Machine Learning*
  - *Goals/non-goals*
  - *Intro to Spark*
    - *Stack, Mechanisms – RDD*
  - *Datasets : SOTU, Titanic, Frequent Flier*
  - *Statistical Toolbox*
    - *Summary, Correlations*
- "Mood Of the Union"
  - *State of the Union w/ Washington, Lincoln, FDR, JFK, Clinton, Bush & Obama*
  - *Map reduce, parse text*
- Clustering
  - *K-means for Gallactic Hoppers!*
- Break [3:15-3:45)
- Predicting Survivors with Classification
  - *Decision Trees*
  - *NaiveBayes (Titanic data set)*
- Linear Regression
- Recommendation Engine
  - *Collab Filtering w/movie lens*
- Discussions/Slack

Oct 29 2-3:15 (75min), 3:45-5:00 (75 min) = 150 min

[20] 2:00 – 2:20	[30] 2:20 – 2:50	
[25] 2:50 – 3:15	[30] 3:45 – 4:15	
[10] 4:15 – 4:25	[20] 4:25 – 4:45	[15] 4:45 – 5:00

Goals & non-goals

Goals

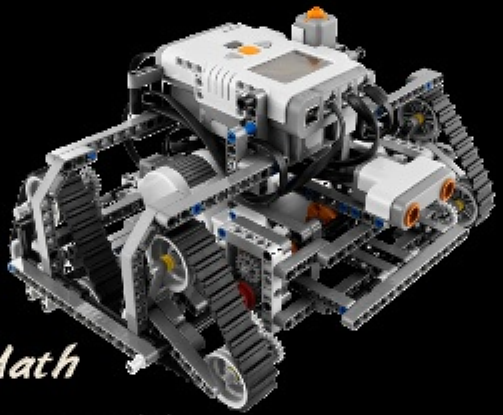
- Understand how to program Machine Learning with Spark & Python
- Focus on programming & ML application
- Give you a focused time to work thru examples
  - *Work with me. I will wait if you want to catch-up*
- Less theory, more usage - let us see if this works
- As straightforward as possible
  - *The programs can be optimized*

Non-goals

- Go deep into the algorithms
  - *We don' t have sufficient time. The topic can be easily a 5 day tutorial !*
- Dive into spark internals
  - *That is for another day*
- The underlying computation, communication, constraints & distribution is a fascinating subject
  - *Paco does a good job explaining them*
- A passive talk
  - *Nope. Interactive & hands-on*

About Me

- Chief Data Scientist at BlackArrow.tv
- Have been speaking at OSCON, PyCon, Pydata et al
- Reviewing Packt Book "Machine Learning with Spark"
- Picked up co-authorship Second Edition of "Fast Data Processing with Spark"
- Have done lots of things:
  - *Big Data (Retail, Bioinformatics, Financial, AdTech),*
  - *Written Books (Web 2.0, Wireless, Java,…)*
  - *Standards, some work in AI,*
  - *Guest Lecturer at Naval PG School,…*
  - *Planning MS-CFinance or Statistics or Computational Math*
- Volunteer as Robotics Judge at First Lego league World Competitions
- @ksankar, doubleclix.wordpress.com



2:00

Spark & Data Science DevOps

Close Encounters

- 1^{st}
  - *This Tutorial*
- 2^{nd}
  - *Do More Hands-on Walkthrough*
- 3^{nd}
  - *Listen To Lectures*
  - *More competitions …*

Spark Installation

- Install Spark 1.1.0 in local Machine
- https://spark.apache.org/downloads.html
  - *Pre-built For Hadoop 2.4 is fine*
- Download & uncompress
- Remember the path & use it wherever you see /usr/local/spark/
- I have downloaded in /usr/local & have a softlink spark to the latest version

Tutorial Materials

- Github : https://github.com/xsankar/cloaked-ironman
- Clone or download zip
- Open terminal
- cd ~/cloaked-ironman
- IPYTHON=1 IPYTHON_OPTS="notebook --pylab inline" /usr/local/spark/bin/pyspark
- Note :
- I have a soft link “spark” in my /usr/local that points to the spark version that I use. For example ln -s spark-1.1.0/ spark
- Click on ipython dashboard
- Just look thru the ipython notebooks

Data Science - Context

Data Management

Collect
- Volume
- Velocity
- Streaming Data

Store
- Metadata
- Monitor counters & Metrics
- Structured vs. Multi-structured

Transform
- Canonical form
- Data catalog
- Data Fabric across the organization
- Access to multiple sources of data
- Think Hybrid – Big Data Apps, Appliances & Infrastructure

Reason
- Flexible & Selectable
  - Data Subsets
  - Attribute sets

Model
- Refine model with
  - Extended Data subsets
  - Engineered Attribute sets
- Validation run across a larger data set

Deploy
- Scalable Model Deployment
- Big Data automation & purpose built appliances (soft/hard)
- Manage SLAs & response times

Data Science

- Bytes to Business a.k.a. Build the full stack
- Find Relevant Data For Business
- Connect the Dots

Visualize
- Performance
- Scalability
- Refresh Latency
- In-memory Analytics

Recommend

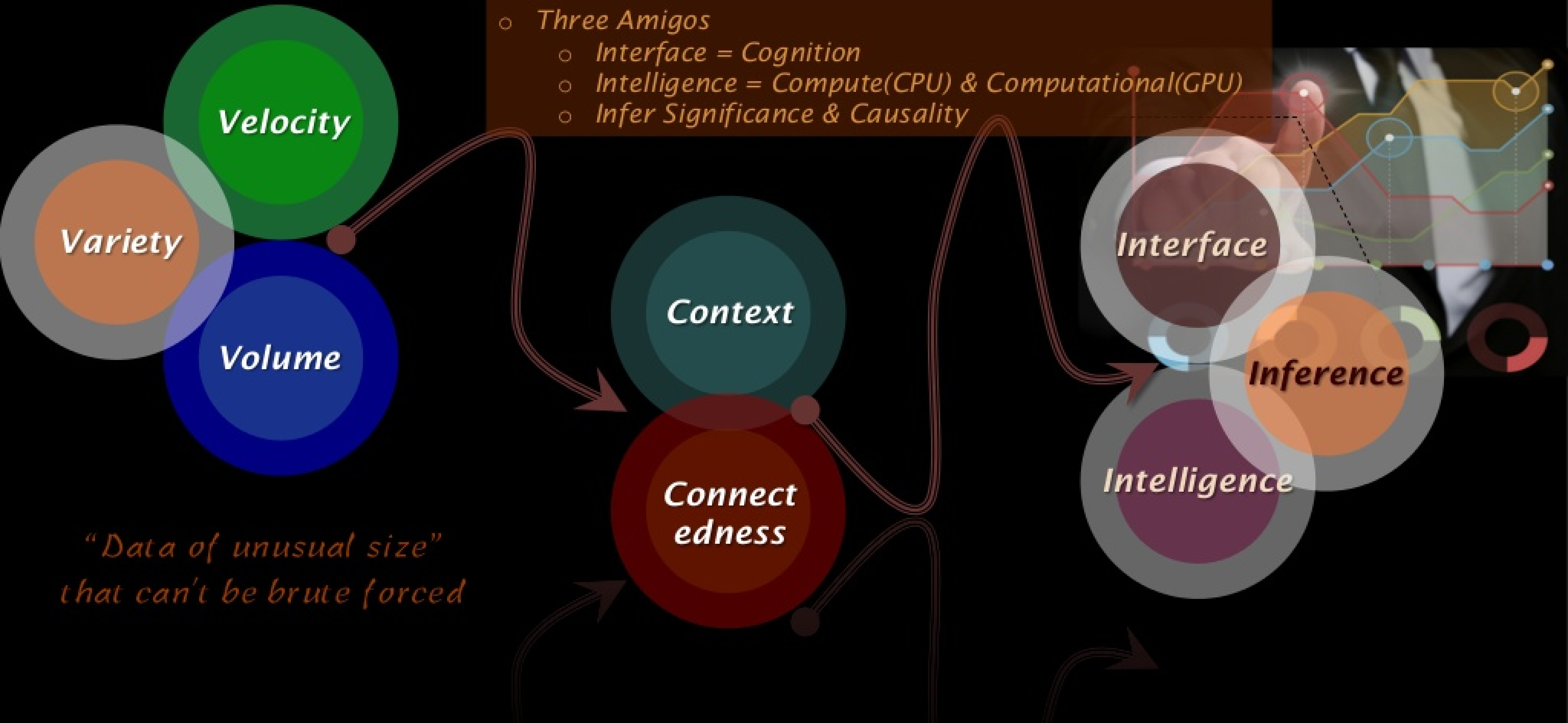
Predict
- Advanced Visualization
- Interactive Dashboards
- Map Overlay
- Infographics

Explore
- Dynamic Data Sets
- 2 way key-value tagging of datasets
- Extended attribute sets
- Advanced Analytics

Connect the Dots

Data Science - Context

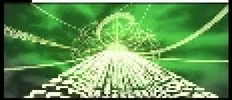
- *Three Amigos*
  - *Interface = Cognition*
  - *Intelligence = Compute(CPU) & Computational(GPU)*
  - *Infer Significance & Causality*



"Data of unusual size" that can't be brute forced

Day in the life of a (super) Model

Model Selection



Models

Algorithms

Intelligence

Reason & Learn

Dimensionality Reduction

Feature Selection

Model Assessment

Attributes

Parameters

Inference

Data (Scoring)

Data Representation

Visualize, Recommend, Explore

Interface

Data Science Maturity Model & Spark

	Isolated Analytics	*Integrated Analytics*	*Aggregated Analytics*	*Automated Analytics*
Data	Small Data	Larger Data set	Big Data	Big Data Factory Model
Context	Local	Domain	Cross-domain + External	Cross domain + External
Model, Reason & Deploy	• Single set of boxes, usually owned by the Model Builders • Departmental	• Deploy - Central Analytics Infrastructure • Models still owned & operated by Modelers • Partly Enterprise-wide	• Central Analytics Infrastructure • Model & Reason – by Model Builders • Deploy, Operate – by ops • Residuals and other metrics monitored by modelers • Enterprise-wide	• Distributed Analytics Infrastructure • AI Augmented models • Model & Reason – by Model Builders • Deploy, Operate – by ops • Data as a monetized service, extending to eco system partners
	• Reports	• Dashboards	• Dashboards + some APIs	• Dashboards + Well defined APIs + programming models
Type	• Descriptive & Reactive	• + Predictive	• + Adaptive	• Adaptive
Datasets	• All in the same box	• Fixed data sets, usually in temp data spaces	• Flexible Data & Attribute sets	• Dynamic datasets with well-defined refresh policies
Workload	• Skunk works	• Business relevant apps with approx SLAs	• High performance appliance clusters	• Appliances and clusters for multiple workloads including real time apps • Infrastructure for emerging technologies
Strategy	• Informal definitions	• Data definitions buried in the analytics models	• Some data definitions	• Data catalogue, metadata & Annotations • Big Data MDM Strategy

The Sense & Sensibility of a DataScientist DevOps

A Shift In Perspective

Analytics in the Lab

- Question-driven
- Interactive
- Ad-hoc, post-hoc
- Fixed data
- Focus on speed and flexibility
- Output is embedded into a report or in-database scoring engine

Analytics in the Factory

- Metric-driven
- Automated
- Systematic
- Fluid data
- Focus on transparency and reliability
- Output is a production system that makes customer-facing decisions

Factory = Operational

Factory = Operational

http://doubleclix.wordpress.com/2014/05/11/the-sense-sensibility-of-a-data-scientist-devops/

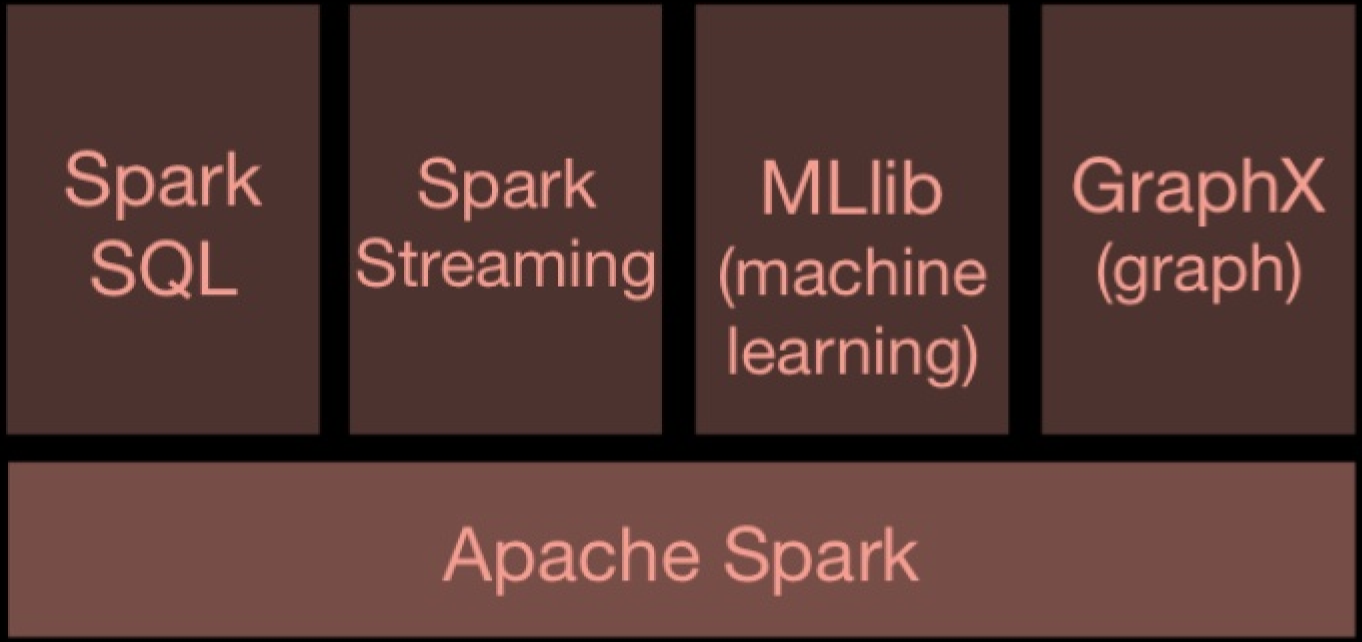
Lab = Investigative

Lab = Investigative

Spark

Investigative		Operational
Historical Subset Sample Workstation	**Data**	Production Data Large-Scale Shared Cluster
Ad Hoc Investigation Offline	**Context**	Continuous Operation Online
Accuracy	**Metrics**	Throughput, QPS
Many, Sophisticated	**Library**	Few, Simple
Scripting, High Level Ease of Development	**Language**	Systems Language Performance

Spark-The Stack



Spark Breaks Previous Large-Scale Sort Record

October 10, 2014 | by Reynold Xin

Tags: Spark

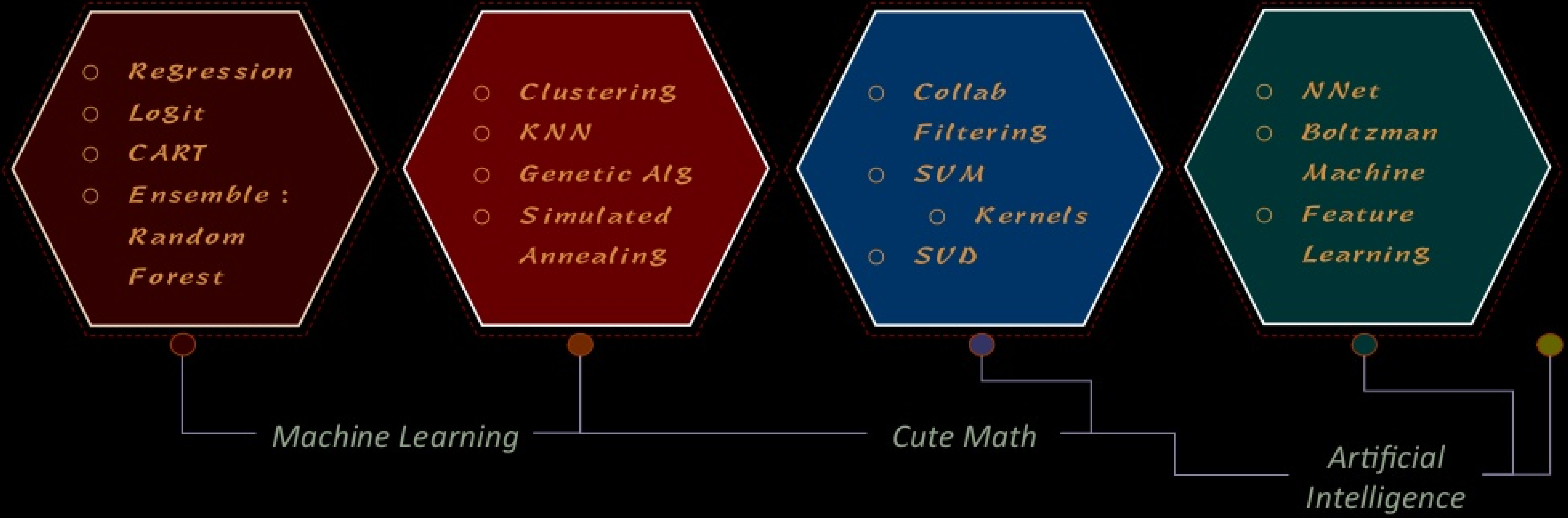
	Hadoop World Record	Spark 100 TB	Spark 1 PB
Data Size	102.5 TB	100 TB	1000 TB
Elapsed Time	72 mins	23 mins	234 mins
# Nodes	2100	206	190
# Cores	50400	6592	6080
# Reducers	10,000	29,000	250,000
Rate	1.42 TB/min	4.27 TB/min	4.27 TB/min
Rate/node	0.67 GB/min	20.7 GB/min	22.5 GB/min
Sort Benchmark Daytona Rules	Yes	Yes	No
Environment	dedicated data center	EC2 (i2.8xlarge)	EC2 (i2.8xlarge)

http://databricks.com/blog/2014/10/10/spark-breaks-previous-large-scale-sort-record.html

RDD – The workhorse of Spark

- Resilient Distributed Datasets
  - *Collection that can be operated in parallel*
- Transformations – create RDDs
  - *Map, Filter, …*
- Actions – Get values
  - *Collect, Take, …*
- We will apply these operations during this tutorial

Algorithm spectrum



ALL MLlib APIs are not available in Python (as of 1.1.0)

API	Spark 1.1.0		Spark 1.2.0
	Java/Scala	Python	
Basic Statistics	✔	✔	
Linear Models	✔	✔	
Decision Trees	✔	✔	
Random Forest	✖	✖	
Collab Filtering	✔	✔	
Clustering-KMeans	✔	✔	
Clustering-Hierarchical	✖	✖	
SVD	✔	✖	
PCA	✔	✖	
Standard Scaler, Normalizer	✔	✖	
Model Evaluation-PR/ROC			

Spark 1.2 MLlib JIRA
http://bit.ly/1ywotkm

Statistical Toolbox

- Sample data : Car mileage data

```
inp_file = sc.textFile("car-data/car-milage.csv")
car_rdd = inp_file.map(lambda line: line.split(','))
```

```
car_rdd.count()
```

```
33
```

```
car_rdd.first()
```

```
[u'mpg',
 u'displacement',
 u'hp',
 u'torque',
 u'CRatio',
 u'RARatio',
 u'CarbBarrells',
 u'NoOfSpeed',
 u'length',
 u'width',
 u'weight',
 u'automatic']
```

```
inp_file = sc.textFile("car-data/car-milage-no-hdr.csv")
car_rdd = inp_file.map(lambda line: array([float(x) for x in line.split(',')]))
car_rdd.first()
```

```
array([  1.89000000e+01,   3.50000000e+02,   1.65000000e+02,
         2.60000000e+02,   8.00000000e+00,   2.56000000e+00,
         4.00000000e+00,   3.00000000e+00,   2.00300000e+02,
         6.99000000e+01,   3.91000000e+03,   1.00000000e+00])
```

https://github.com/apache/spark/blob/master/examples/src/main/python/mllib/correlations.py

```
from pyspark.mllib.stat import Statistics
summary = Statistics.colStats(car_rdd)
```

```
print str(summary)
```

```
<pyspark.mllib.stat.MultivariateStatisticalSummary object at 0x1097a7310>
```

```
for x in summary.min():
    print "%4.4f " % x,
print
for x in summary.mean():
    print "%4.4f " % x,
print
for x in summary.max():
    print "%4.4f " % x,
print
```

```
11.2000  85.3000  70.0000  81.0000  8.0000  2.4500  1.0000  3.0000  155.7000  61.8000  1905.0000  0.0000
20.0383  286.0467  136.9667  217.9000  8.3133  3.0590  2.5667  3.3333  192.3400  71.4200  3625.8000  0.7333
36.5000  500.0000  223.0000  366.0000  9.0000  4.3000  4.0000  5.0000  231.0000  79.8000  5430.0000  1.0000
```

```
hp = car_rdd.map(lambda x: x[2])
weight = car_rdd.map(lambda x: x[10])
print '%2.3f' % Statistics.corr(hp, weight, method="pearson")
print '%2.3f' % Statistics.corr(hp, weight, method="spearman")
```

```
0.888
0.874
```

```
ra_ratio = car_rdd.map(lambda x: x[5])
width = car_rdd.map(lambda x: x[9])
print '%2.3f' % Statistics.corr(ra_ratio, width, method="pearson")
print '%2.3f' % Statistics.corr(ra_ratio, width, method="spearman")
```

```
-0.453
-0.244
```