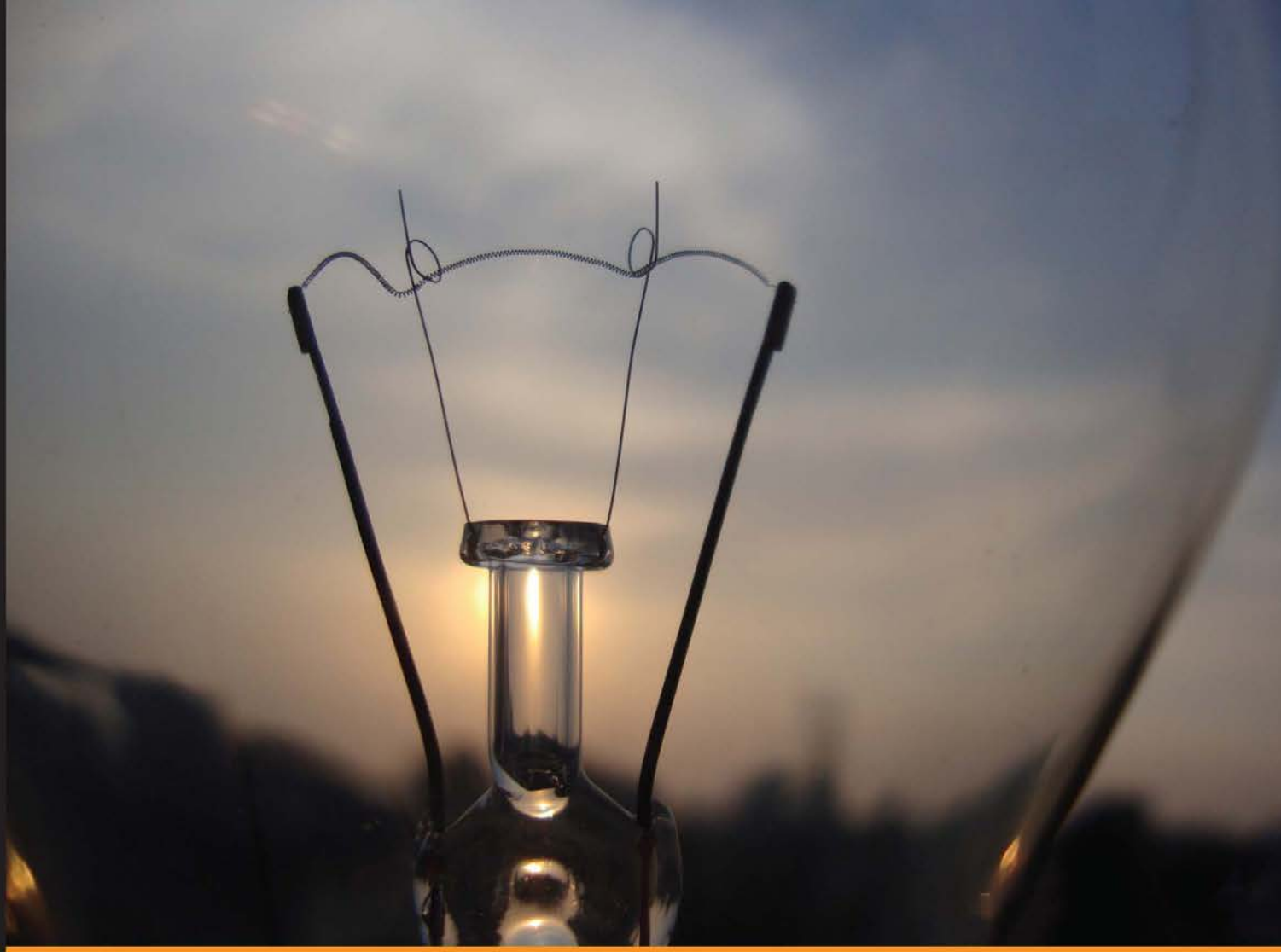

Community Experience Distilled

Machine Learning with Spark

Create scalable machine learning applications to power a modern data-driven business using Spark

Nick Pentreath

[PACKT] PUBLISHING open source* community experience distilled

Machine Learning with Spark

Create scalable machine learning applications to power a modern data-driven business using Spark

Nick Pentreath

BIRMINGHAM - MUMBAI

Machine Learning with Spark

First published: February 2015

Production reference: 1170215

Published by Packt Publishing Ltd.
Livery Place
35 Livery Street
Birmingham B3 2PB, UK.

ISBN 978-1-78328-851-9

www.packtpub.com

Cover image by Akshay Paunikar (akshaypaunikar4@gmail.com)

Credits

Author
Nick Pentreath

Reviewers
Andrea Mostosi
Hao Ren
Krishna Sankar

Commissioning Editor
Rebecca Youé

Acquisition Editor
Rebecca Youé

Content Development Editor
Susmita Sabat

Technical Editors
Vivek Arora
Pankaj Kadam

Copy Editor
Karuna Narayanan

Project Coordinator
Milton Dsouza

Proofreaders
Simran Bhogal
Maria Gould
Ameesha Green
Paul Hindle

Indexer
Priya Sane

Graphics
Sheetal Aute
Abhinash Sahu

Production Coordinator
Nitesh Thakur

Cover Work
Nitesh Thakur

About the Author

Nick Pentreath has a background in financial markets, machine learning, and software development. He has worked at Goldman Sachs Group, Inc.; as a research scientist at the online ad targeting start-up Cognitive Match Limited, London; and led the Data Science and Analytics team at Mxit, Africa's largest social network.

He is a cofounder of Graphflow, a big data and machine learning company focused on user-centric recommendations and customer intelligence. He is passionate about combining commercial focus with machine learning and cutting-edge technology to build intelligent systems that learn from data to add value to the bottom line.

Nick is a member of the Apache Spark Project Management Committee.

Acknowledgments

Writing this book has been quite a rollercoaster ride over the past year, with many ups and downs, late nights, and working weekends. It has also been extremely rewarding to combine my passion for machine learning with my love of the Apache Spark project, and I hope to bring some of this out in this book.

I would like to thank the Packt Publishing team for all their assistance throughout the writing and editing process: Rebecca, Susmita, Sudhir, Amey, Neil, Vivek, Pankaj, and everyone who worked on the book.

Thanks also go to Debora Donato at StumbleUpon for assistance with data- and legal-related queries.

Writing a book like this can be a somewhat lonely process, so it is incredibly helpful to get the feedback of reviewers to understand whether one is headed in the right direction (and what course adjustments need to be made). I'm deeply grateful to Andrea Mostosi, Hao Ren, and Krishna Sankar for taking the time to provide such detailed and critical feedback.

I could not have gotten through this project without the unwavering support of all my family and friends, especially my wonderful wife, Tammy, who will be glad to have me back in the evenings and on weekends once again. Thank you all!

Finally, thanks to all of you reading this; I hope you find it useful!

About the Reviewers

Andrea Mostosi is a technology enthusiast. An innovation lover since he was a child, he started a professional job in 2003 and worked on several projects, playing almost every role in the computer science environment. He is currently the CTO at The Fool, a company that tries to make sense of web and social data. During his free time, he likes traveling, running, cooking, biking, and coding.

I would like to thank my geek friends: Simone M, Daniele V, Luca T, Luigi P, Michele N, Luca O, Luca B, Diego C, and Fabio B. They are the smartest people I know, and comparing myself with them has always pushed me to be better.

Hao Ren is a software developer who is passionate about Scala, distributed systems, machine learning, and Apache Spark. He was an exchange student at EPFL when he learned about Scala in 2012. He is currently working in Paris as a backend and data engineer for ClaraVista—a company that focuses on high-performance marketing. His work responsibility is to build a Spark-based platform for purchase prediction and a new recommender system.

Besides programming, he enjoys running, swimming, and playing basketball and badminton. You can learn more at his blog `http://www.invkrh.me`.

Krishna Sankar is a chief data scientist at BlackArrow, where he is focusing on enhancing user experience via inference, intelligence, and interfaces. Earlier stints include working as a principal architect and data scientist at Tata America International Corporation, director of data science at a bioinformatics start-up company, and as a distinguished engineer at Cisco Systems, Inc. He has spoken at various conferences about data science (`http://goo.gl/9pyJMH`), machine learning (`http://goo.gl/sSem2Y`), and social media analysis (`http://goo.gl/D9YpVQ`). He has also been a guest lecturer at the Naval Postgraduate School. He has written a few books on Java, wireless LAN security, Web 2.0, and now on Spark. His other passion is LEGO robotics. Earlier in April, he was at the St. Louis FLL World Competition as a robots design judge.

www.PacktPub.com

Support files, eBooks, discount offers, and more

For support files and downloads related to your book, please visit `www.PacktPub.com`.

Did you know that Packt offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at `www.PacktPub.com` and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at `service@packtpub.com` for more details.

At `www.PacktPub.com`, you can also read a collection of free technical articles, sign up for a range of free newsletters and receive exclusive discounts and offers on Packt books and eBooks.

`https://www2.packtpub.com/books/subscription/packtlib`

Do you need instant solutions to your IT questions? PacktLib is Packt's online digital book library. Here, you can search, access, and read Packt's entire library of books.

Why subscribe?

- Fully searchable across every book published by Packt
- Copy and paste, print, and bookmark content
- On demand and accessible via a web browser

Free access for Packt account holders

If you have an account with Packt at `www.PacktPub.com`, you can use this to access PacktLib today and view 9 entirely free books. Simply use your login credentials for immediate access.

Table of Contents

Preface

In recent years, the volume of data being collected, stored, and analyzed has exploded, in particular in relation to the activity on the Web and mobile devices, as well as data from the physical world collected via sensor networks. While previously large-scale data storage, processing, analysis, and modeling was the domain of the largest institutions such as Google, Yahoo!, Facebook, and Twitter, increasingly, many organizations are being faced with the challenge of how to handle a massive amount of data.

When faced with this quantity of data and the common requirement to utilize it in real time, human-powered systems quickly become infeasible. This has led to a rise in the so-called big data and machine learning systems that learn from this data to make automated decisions.

In answer to the challenge of dealing with ever larger-scale data without any prohibitive cost, new open source technologies emerged at companies such as Google, Yahoo!, Amazon, and Facebook, which aimed at making it easier to handle massive data volumes by distributing data storage and computation across a cluster of computers.

The most widespread of these is Apache Hadoop, which made it significantly easier and cheaper to both store large amounts of data (via the Hadoop Distributed File System, or HDFS) and run computations on this data (via Hadoop MapReduce, a framework to perform computation tasks in parallel across many nodes in a computer cluster).

However, MapReduce has some important shortcomings, including high overheads to launch each job and reliance on storing intermediate data and results of the computation to disk, both of which make Hadoop relatively ill-suited for use cases of an iterative or low-latency nature. Apache Spark is a new framework for distributed computing that is designed from the ground up to be optimized for low-latency tasks and to store intermediate data and results in memory, thus addressing some of the major drawbacks of the Hadoop framework. Spark provides a clean, functional, and easy-to-understand API to write applications and is fully compatible with the Hadoop ecosystem.

Furthermore, Spark provides native APIs in Scala, Java, and Python. The Scala and Python APIs allow all the benefits of the Scala or Python language, respectively, to be used directly in Spark applications, including using the relevant interpreter for real-time, interactive exploration. Spark itself now provides a toolkit (called MLlib) of distributed machine learning and data mining models that is under heavy development and already contains high-quality, scalable, and efficient algorithms for many common machine learning tasks, some of which we will delve into in this book.

Applying machine learning techniques to massive datasets is challenging, primarily because most well-known machine learning algorithms are not designed for parallel architectures. In many cases, designing such algorithms is not an easy task. The nature of machine learning models is generally iterative, hence the strong appeal of Spark for this use case. While there are many competing frameworks for parallel computing, Spark is one of the few that combines speed, scalability, in-memory processing, and fault tolerance with ease of programming and a flexible, expressive, and powerful API design.

Throughout this book, we will focus on real-world applications of machine learning technology. While we may briefly delve into some theoretical aspects of machine learning algorithms, the book will generally take a practical, applied approach with a focus on using examples and code to illustrate how to effectively use the features of Spark and MLlib, as well as other well-known and freely available packages for machine learning and data analysis, to create a useful machine learning system.

What this book covers

Chapter 1, *Getting Up and Running with Spark*, shows how to install and set up a local development environment for the Spark framework as well as how to create a Spark cluster in the cloud using Amazon EC2. The Spark programming model and API will be introduced, and a simple Spark application will be created using each of Scala, Java, and Python.

Chapter 2, *Designing a Machine Learning System*, presents an example of a real-world use case for a machine learning system. We will design a high-level architecture for an intelligent system in Spark based on this illustrative use case.

Chapter 3, *Obtaining, Processing, and Preparing Data with Spark*, details how to go about obtaining data for use in a machine learning system, in particular from various freely and publicly available sources. We will learn how to process, clean, and transform the raw data into features that may be used in machine learning models, using available tools, libraries, and Spark's functionality.

Chapter 4, *Building a Recommendation Engine with Spark*, deals with creating a recommendation model based on the collaborative filtering approach. This model will be used to recommend items to a given user as well as create lists of items that are similar to a given item. Standard metrics to evaluate the performance of a recommendation model will be covered here.

Chapter 5, *Building a Classification Model with Spark*, details how to create a model for binary classification as well as how to utilize standard performance-evaluation metrics for classification tasks.

Chapter 6, *Building a Regression Model with Spark*, shows how to create a model for regression, extending the classification model created in *Chapter 5*, *Building a Classification Model with Spark*. Evaluation metrics for the performance of regression models will be detailed here.

Chapter 7, *Building a Clustering Model with Spark*, explores how to create a clustering model as well as how to use related evaluation methodologies. You will learn how to analyze and visualize the clusters generated.

Chapter 8, *Dimensionality Reduction with Spark*, takes us through methods to extract the underlying structure from and reduce the dimensionality of our data. You will learn some common dimensionality-reduction techniques and how to apply and analyze them, as well as how to use the resulting data representation as input to another machine learning model.

Chapter 9, *Advanced Text Processing with Spark*, introduces approaches to deal with large-scale text data, including techniques for feature extraction from text and dealing with the very high-dimensional features typical in text data.

Chapter 10, *Real-time Machine Learning with Spark Streaming*, provides an overview of Spark Streaming and how it fits in with the online and incremental learning approaches to apply machine learning on data streams.

What you need for this book

Throughout this book, we assume that you have some basic experience with programming in Scala, Java, or Python and have some basic knowledge of machine learning, statistics, and data analysis.

Who this book is for

This book is aimed at entry-level to intermediate data scientists, data analysts, software engineers, and practitioners involved in machine learning or data mining with an interest in large-scale machine learning approaches, but who are not necessarily familiar with Spark. You may have some experience of statistics or machine learning software (perhaps including MATLAB, scikit-learn, Mahout, R, Weka, and so on) or distributed systems (perhaps including some exposure to Hadoop).

Conventions

In this book, you will find a number of styles of text that distinguish between different kinds of information. Here are some examples of these styles, and an explanation of their meaning.

Code words in text, database table names, folder names, filenames, file extensions, pathnames, dummy URLs, user input, and Twitter handles are shown as follows: "Spark places user scripts to run Spark in the `bin` directory."

A block of code is set as follows:

```
val conf = new SparkConf()
.setAppName("Test Spark App")
.setMaster("local[4]")
val sc = new SparkContext(conf)
```

Any command-line input or output is written as follows:

```
>tar xfvz spark-1.2.0-bin-hadoop2.4.tgz
>cd spark-1.2.0-bin-hadoop2.4
```

New terms and **important words** are shown in bold. Words that you see on the screen, in menus or dialog boxes for example, appear in the text like this: "These can be obtained from the AWS homepage by clicking **Account** | **Security Credentials** | **Access Credentials**."

Warnings or important notes appear in a box like this.

Tips and tricks appear like this.

Reader feedback

Feedback from our readers is always welcome. Let us know what you think about this book—what you liked or may have disliked. Reader feedback is important for us to develop titles that you really get the most out of.

To send us general feedback, simply send an e-mail to `feedback@packtpub.com`, and mention the book title through the subject of your message.

If there is a topic that you have expertise in and you are interested in either writing or contributing to a book, see our author guide on `www.packtpub.com/authors`.

Customer support

Now that you are the proud owner of a Packt book, we have a number of things to help you to get the most from your purchase.

Downloading the example code

You can download the example code files for all Packt books you have purchased from your account at `http://www.packtpub.com`. If you purchased this book elsewhere, you can visit `http://www.packtpub.com/support` and register to have the files e-mailed directly to you.

Errata

Although we have taken every care to ensure the accuracy of our content, mistakes do happen. If you find a mistake in one of our books—maybe a mistake in the text or the code—we would be grateful if you would report this to us. By doing so, you can save other readers from frustration and help us improve subsequent versions of this book. If you find any errata, please report them by visiting `http://www.packtpub.com/support`, selecting your book, clicking on the **Errata Submission Form** link, and entering the details of your errata. Once your errata are verified, your submission will be accepted and the errata will be uploaded to our website or added to any list of existing errata under the Errata section of that title.

To view the previously submitted errata, go to `https://www.packtpub.com/books/content/support` and enter the name of the book in the search field. The required information will appear under the **Errata** section.

Piracy

Piracy of copyright material on the Internet is an ongoing problem across all media. At Packt, we take the protection of our copyright and licenses very seriously. If you come across any illegal copies of our works, in any form, on the Internet, please provide us with the location address or website name immediately so that we can pursue a remedy.

Please contact us at `copyright@packtpub.com` with a link to the suspected pirated material.

We appreciate your help in protecting our authors, and our ability to bring you valuable content.

Questions

You can contact us at `questions@packtpub.com` if you are having a problem with any aspect of the book, and we will do our best to address it.

1
Getting Up and Running with Spark

Apache Spark is a framework for distributed computing; this framework aims to make it simpler to write programs that run in parallel across many nodes in a cluster of computers. It tries to abstract the tasks of resource scheduling, job submission, execution, tracking, and communication between nodes, as well as the low-level operations that are inherent in parallel data processing. It also provides a higher level API to work with distributed data. In this way, it is similar to other distributed processing frameworks such as Apache Hadoop; however, the underlying architecture is somewhat different.

Spark began as a research project at the University of California, Berkeley. The university was focused on the use case of distributed machine learning algorithms. Hence, it is designed from the ground up for high performance in applications of an iterative nature, where the same data is accessed multiple times. This performance is achieved primarily through caching datasets in memory, combined with low latency and overhead to launch parallel computation tasks. Together with other features such as fault tolerance, flexible distributed-memory data structures, and a powerful functional API, Spark has proved to be broadly useful for a wide range of large-scale data processing tasks, over and above machine learning and iterative analytics.

For more background on Spark, including the research papers underlying Spark's development, see the project's history page at `http://spark.apache.org/community.html#history`.

Spark runs in four modes:

- The standalone local mode, where all Spark processes are run within the same **Java Virtual Machine** (**JVM**) process
- The standalone cluster mode, using Spark's own built-in job-scheduling framework
- Using Mesos, a popular open source cluster-computing framework
- Using YARN (commonly referred to as NextGen MapReduce), a Hadoop-related cluster-computing and resource-scheduling framework

In this chapter, we will:

- Download the Spark binaries and set up a development environment that runs in Spark's standalone local mode. This environment will be used throughout the rest of the book to run the example code.
- Explore Spark's programming model and API using Spark's interactive console.
- Write our first Spark program in Scala, Java, and Python.
- Set up a Spark cluster using Amazon's **Elastic Cloud Compute** (**EC2**) platform, which can be used for large-sized data and heavier computational requirements, rather than running in the local mode.

Spark can also be run on Amazon's Elastic MapReduce service using custom bootstrap action scripts, but this is beyond the scope of this book. The following article is a good reference guide: `http://aws.amazon.com/articles/Elastic-MapReduce/4926593393724923`.

At the time of writing this book, the article covers running Spark Version 1.1.0.

If you have previous experience in setting up Spark and are familiar with the basics of writing a Spark program, feel free to skip this chapter.

Installing and setting up Spark locally

Spark can be run using the built-in standalone cluster scheduler in the local mode. This means that all the Spark processes are run within the same JVM—effectively, a single, multithreaded instance of Spark. The local mode is very useful for prototyping, development, debugging, and testing. However, this mode can also be useful in real-world scenarios to perform parallel computation across multiple cores on a single computer.

As Spark's local mode is fully compatible with the cluster mode, programs written and tested locally can be run on a cluster with just a few additional steps.

The first step in setting up Spark locally is to download the latest version (at the time of writing this book, the version is 1.2.0). The download page of the Spark project website, found at `http://spark.apache.org/downloads.html`, contains links to download various versions as well as to obtain the latest source code via GitHub.

The Spark project documentation website at `http://spark.apache.org/docs/latest/` is a comprehensive resource to learn more about Spark. We highly recommend that you explore it!

Spark needs to be built against a specific version of Hadoop in order to access **Hadoop Distributed File System** (**HDFS**) as well as standard and custom Hadoop input sources. The download page provides prebuilt binary packages for Hadoop 1, CDH4 (Cloudera's Hadoop Distribution), MapR's Hadoop distribution, and Hadoop 2 (YARN). Unless you wish to build Spark against a specific Hadoop version, we recommend that you download the prebuilt Hadoop 2.4 package from an Apache mirror using this link: `http://www.apache.org/dyn/closer.cgi/spark/spark-1.2.0/spark-1.2.0-bin-hadoop2.4.tgz`.

Spark requires the Scala programming language (version 2.10.4 at the time of writing this book) in order to run. Fortunately, the prebuilt binary package comes with the Scala runtime packages included, so you don't need to install Scala separately in order to get started. However, you will need to have a **Java Runtime Environment** (**JRE**) or **Java Development Kit** (**JDK**) installed (see the software and hardware list in this book's code bundle for installation instructions).

Once you have downloaded the Spark binary package, unpack the contents of the package and change into the newly created directory by running the following commands:

```
>tar xfvz spark-1.2.0-bin-hadoop2.4.tgz
>cd spark-1.2.0-bin-hadoop2.4
```

Spark places user scripts to run Spark in the `bin` directory. You can test whether everything is working correctly by running one of the example programs included in Spark:

```
>./bin/run-example org.apache.spark.examples.SparkPi
```