Big Data to Enable Global Disruption of the Grapevine-powered Industries

D4.3 - Models and Tools for Predictive Analytics over Extremely Large Datasets

DELIVERABLE NUMBER	D4.3
DELIVERABLE TITLE	Models and Tools for Predictive Analytics over Extremely Large Datasets
RESPONSIBLE AUTHOR	Franco Maria Nardini (CNR)

Co-funded by the Horizon 2020
Framework Programme of the European Union

GRANT AGREEMENT N.	780751
PROJECT ACRONYM	BigDataGrapes
PROJECT FULL NAME	Big Data to Enable Global Disruption of the Grapevine-powered industries
STARTING DATE (DUR.)	01/01/2018 (36 months)
ENDING DATE	31/12/2020
PROJECT WEBSITE	http://www.bigdatagrapes.eu/
COORDINATOR	Panagiotis Zervas
ADDRESS	110 Pentelis Str., Marousi, GR15126, Greece
REPLY TO	pzervas@agroknow.com
PHONE	+30 210 6897 905
EU PROJECT OFFICER	Ms. Annamária Nagy
WORKPACKAGE N. | TITLE	WP4 | Analytics and Processing Layer
WORKPACKAGE LEADER	CNR
DELIVERABLE N. | TITLE	D4.3 | Models and Tools for Predictive Analytics over Extremely Large Datasets
RESPONSIBLE AUTHOR	Franco Maria Nardini (CNR)
REPLY TO	francomaria.nardini@isti.cnr.it
DOCUMENT URL	http://www.bigdatagrapes.eu/
DATE OF DELIVERY (CONTRACTUAL)	30 September 2018 (M9), 31 March 2019 (M15, Updated version)
DATE OF DELIVERY (SUBMITTED)	28 September 2018 (M9), 15 April 2019 (M16, Updated version)
VERSION | STATUS	2.0 | Final
NATURE	DEM (Demonstrator)
DISSEMINATION LEVEL	PU (Public)
AUTHORS (PARTNER)	Franco Maria Nardini (CNR), Vinicius Monteiro de Lira (CNR), Nicola Tonellotto (CNR), Raffaele Perego (CNR), Matteo Catena (CNR), Cristina Muntean (CNR), Ida Mele (CNR), Salvatore Trani (CNR)
CONTRIBUTORS	Panagiotis Zervas (Agroknow), Milena Yankova (ONTOTEXT), Stefan Scherer (Geocledian)
REVIEWER	Simone Parisi (ABACO), Nyi-Nyi Htun (KULeuven)

VERSION	MODIFICATION(S)	DATE	AUTHOR(S)
0.1	Initial contributions	04/09/2018	Vinicius Monteiro de Lira (CNR), Franco Maria Nardini (CNR), Nicola Tonellotto (CNR)
0.2	Added contributions	07/09/2018	Vinicius Monteiro de Lira (CNR), Franco Maria Nardini (CNR)
0.3	Input from partners integrated, document almost finalized	14/09/2018	Franco Maria Nardini (CNR), Raffaele Perego (CNR), Cristina Muntean (CNR), Ida Mele (CNR), Salvatore Trani (CNR)
0.4	Internal Review	16/09/2018	Simone Parisi (ABACO)
1.0	Partners review, final comments and edits	28/09/2018	Franco Maria Nardini (CNR), Raffaele Perego (CNR), Cristina Muntean (CNR), Ida Mele (CNR), Salvatore Trani (CNR)
1.1	Added contributions for updated version	15/04/2019	Vinicius Monteiro de Lira (CNR), Franco Maria Nardini (CNR), Raffaele Perego (CNR), Cristina Muntean (CNR), Nicola Tonellotto (CNR), Salvatore Trani (CNR)
1.5	Internal Review	12/04/2019	Nyi-Nyi Htun (KULeuven)
1.8	Partners review, final comments and edits	15/04/2019	Vinicius Monteiro de Lira (CNR), Franco Maria Nardini (CNR), Raffaele Perego (CNR), Cristina Muntean (CNR), Nicola Tonellotto (CNR), Salvatore Trani (CNR)
2.0	Updated version	16/04/2019	Vinicius Monteiro de Lira (CNR), Franco Maria Nardini (CNR), Raffaele Perego (CNR), Cristina Muntean (CNR), Nicola Tonellotto (CNR), Salvatore Trani (CNR)

PARTICIPANTS		CONTACT
Agroknow IKE (Agroknow, Greece)	Agroknow	Panagiotis Zervas Email: pzervas@agroknow.com
Ontotext AD (ONTOTEXT, Bulgaria)		Todor Primov Email: todor.primov@ontotext.com
Consiglio Nazionale Delle Ricerche (CNR, Italy)		Raffaele Perego Email: raffaele.perego@isti.cnr.it
Katholieke Universiteit Leuven (KULeuven, Belgium)	KATHOLIEKE UNIVERSITEIT LEUVEN	Katrien Verbert Email: katrien.verbert@cs.kuleuven.be
Geocledian GmbH (GEOCLEDIAN Germany)	geo|cledian	Stefan Scherer Email: stefan.scherer@geocledian.comm
Institut National de la Recherché Agronomique (INRA, France)	INRA SCIENCE & IMPACT	Pascal Neveu Email: pascal.neveu@inra.fr
Agricultural University of Athens (AUA, Greece)		Katerina Biniari Email: kbiniari@aua.gr
Abaco SpA (ABACO, Italy)		Simone Parisi Email: s.parisi@abacogroup.eu
Symbeeosis (Symbeeosis, Greece)		Eleni Foufa Email: foufa-e@symbeeosis.com

ACRONYMS LIST

BDG	BigDataGrapes
HDFS	Hadoop Distributed File System
RDD	Resilient Distributed Dataset
TF-IDF	Term Frequency – Inverse Document Frequency
GBT	Gradient Boosting Trees
DT	Decision Trees
RMSE	Root Mean Squared Error
GGMS	Gaussian Mixture Modelling

EXECUTIVE SUMMARY

This accompanying document for deliverable D4.3 (Models and Tools for Predictive Analytics over Extremely Large Datasets) describes the first version of the mechanisms and tools supporting efficient and effective predictive data analytics over the BigDataGrapes (BDG) platform in the context of grapevine-related assets.

The BDG software stack employs efficient and fault-tolerant tools for distributed processing, aimed at providing scalability and reliability for the target applications. On top of this stack, the BDG platform enables distributed predictive big data analytics by effectively exploiting scalable Machine Learning algorithms using the computational resources of the underlying infrastructure efficiently. The software components enabling BDG predictive data analytics have been designed and deployed using Docker containers[1]. They thus include everything needed to run the supported predictive data analytics tools on any system that can run a Docker engine.

The document first introduces the main technologies currently used in the first version of the BDG component for performing efficient and scalable analytics over extremely large dataset. The docker component provided in this deliverable relies on the BDG software stack discussed in Deliverable 2.3: “BigDataGrapes Software Stack Design” and exploits the distributed execution environment provided by the Persistence and Processing Layers of the BDG architecture contributed in Deliverable 4.1: “Methods and Tools for Scalable Distributed Processing”.

The document details the steps to be followed to download and deploy the first version of the BDG platform and provides the reader with practical examples of usage of its scalable predictive analytics component. Specifically, we provide four demonstrators released as Jupyter Notebooks[2] implementing four different machine learning tasks by exploiting the BDG infrastructure. The first one shows how to train two kinds of regressors, i.e., linear and random forest regressors, to fit synthetically generated data. We present these results by adding a visualization of the result to allow the reader to understand the limitations of each specific solution. The second demonstrator employs a well-known dataset, i.e., the KDD CUP 1999 dataset[3], to train a binary logistic regression classifier. We show how to train and evaluate the performance of the classifier by means of a standard metric, i.e., Accuracy. This second demonstrator also shows how the distributed file system can be exploited to directly feed the machine learning platform with data. The third demonstrator extends the second one by showing how to train a multi-label classifier on a Red Wine Quality Dataset[4], a public dataset employed on Kaggle for a machine learning competition. We show how to learn a multi-label logistic regression classifier and how to evaluate its performance in terms of Accuracy. The fourth demonstrator is much more complex and structured and has been added to this document as an update done at M15. The demonstrator focuses on the application of machine learning methods on wine data collected from online social networks of wine passionate users. The dataset analyzed contains: 489,417 wine reviews by 195,678 users, written in 86 languages, related to 51,579 different wines, from 58 wine countries and 2272 wine regions. The predictive analysis conducted on this dataset allows us to show the potential of the machine learning layer of the BDG infrastructure providing efficient and effective methods for assessing the potential market penetration of a given wine in a new country. We estimate this penetration capability by learning a model from user-generated contents about wines in a target country.

[1] https://www.docker.com/resources/what-container

[2] https://jupyter.org/

[3] http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html

[4] https://www.kaggle.com/uciml/red-wine-quality-cortez-et-al-2009

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 INTRODUCTION

The BigDataGrapes platform aims at providing Predictive Data Analytics tools that go beyond the state-of-the-art for what regards their application to large and complex grapevine-related data assets. Having this ambitious goal in mind, even the first version of the predictive analytics component described in this document has been designed by relying on the core technologies and frameworks for efficient processing of large datasets, e.g., Apache Spark, employed on the lower levels of the BDG platform. Machine learning largely benefits from the distributed execution paradigm that serves as the basis for addressing efficiently the analytics and scalability challenges of grapevines-powered industries.

The document first introduces the main technologies currently used in the first version of the *BDG* component for performing efficient and scalable analytics over extremely large dataset. The dockerized component provided in this deliverable relies on the BDG software stack discussed in Deliverable 2.3 "BigDataGrapes Software Stack Design" and exploits the distributed execution environment provided by the Persistence and Processing Layers of the BDG architecture contributed in Deliverable 4.1 "Methods and Tools for Scalable Distributed Processing".

The BDG platform allows the user to learn and apply predictive data analytics over extremely large dataset. We show these functionalities by discussing four demonstrators implemented as Jupyter Notebooks. The first three demonstrators deal with different kinds of machine learning tasks: regression, binary classification and multi-label classification. We present the three applications on three different datasets. The first one is synthetically generated while the other two are the KDD CUP 1999 dataset and the Red Wine Quality, both public. We first present how to train several kinds of models addressing the three tasks, i.e., *linear regressors*, *random forest regressors*, *logistic regression classifiers* by interacting with the BDG distributed infrastructure. We then present how to assess the performance of a learned model in terms of a well-known quality metric, i.e., Accuracy. Moreover, we present some basic visualization of the data to provide the user with a useful visual feedback.

The fourth demonstrator is much more complex and structured. It has been added to this document as an update done at M15. The demonstrator focuses on the application of machine learning methods on wine data collected from online social networks of wine passionate users. The dataset analysed contains: 489,417 wine reviews by 195,678 users, written in 86 languages, related to 306,856 different wines, from 57 wine countries and 2,120 wine regions. The predictive analysis conducted on this dataset allows us to show the potential of the machine learning layer of the BDG infrastructure providing efficient and effective methods for assessing the potential market penetration of a given wine on a new country. We estimate this penetration capability by learning a model from user-generated content about wines in the target country.

The rest of this document is organized as follows: Section 2 highlights the main technologies used in the BDG software stack to support and implement scalable predictive analytics. Section 3 describes the design, implementation and application of the BDG predictive analytics component. It also details how to set up and run it on the top of the current version of the BDG platform. Moreover, it discusses the four demonstrators that are provided in the deliverable as running examples showing the functionalities of the BDG platform. Finally, Section 4 concludes the document.

2 DISTRIBUTED MACHINE LEARNING OVER BIG DATA

In this Section we introduce the main components allowing BigDataGrapes to perform predictive analytics over extremely large dataset. To provide the reader with a self-contained view of the predictive data analytics functionality supported by the first version of the BDG platform, we first introduce Apache Spark, a scalable big data processing framework that is the de-facto technological state-of-the-art for parallel and distributed batch computations, and Apache HDFS, a distributed file system enabling persistence of information. We finally introduce MLLib, the machine learning library working on Apache Spark that allows distributed and parallel learning from big data of effective models for regression and classification tasks.

2.1 APACHE SPARK

Apache Spark (https://spark.apache.org/) is a well-known and open-source framework for big data processing. It has been designed at the University of California, Berkeley's AMPLab in 2009. Later, the Spark project moved to the Apache Software Foundation, which has maintained it since 2010. As previous big data processing frameworks, Spark provides an interface for programming clusters with implicit data parallelism and fault tolerance. The novel key feature that motivates its success and popularity is the introduction of the "Resiliend Distributed Dataset" (RDD), a data structure implementing a read-only multiset of data items distributed over a cluster of machines and guaranteeing fault-tolerance. The introduction of the RDD was needed in response to the limitations of the MapReduce cluster computing paradigm, which forces a particular linear dataflow structure on distributed programs. With MapReduce, programs simply read input data from disk, map a function across the data, reduce the results of the map, and store reduction results on the file system. The functions of the RDD in Spark allows distributed programs to interact with a (deliberately) restricted form of distributed shared memory. This novel approach allows Spark to ease the implementation of both iterative algorithms, that visit the data multiple times in a loop, and interactive/exploratory data analysis, i.e., the repeated database-style querying of data. As a consequence, the latency of this kind of applications is extremely reduced, i.e., by several orders of magnitude compared to a MapReduce implementation (as was common in Apache Hadoop stacks).

2.2 APACHE HDFS

The Apache Hadoop software library (http://hadoop.apache.org/) is a framework supporting the distributed processing of large datasets across clusters of computers using simple programming models. It is designed to scale out from single servers to thousands of machines, each offering local computation and storage. Among the Apache Hadoop's core components, there is the Hadoop File System (HDFS). HDFS or Hadoop File System is the file system in which the data is stored. The stored data may be structured, semi-structured or unstructured thus containing for example either the tables of a relational database, or a set of json or log files. HDFS is designed for massive scalability, it stores unlimited amounts of data in a single platform. As the data grow, it is possible to simply add more servers to scale linearly. Furthermore, a key feature of HDFS is reliability. It automatically performs multiple copies of the data stored, letting it always available for access and protection from data loss. Built-in fault tolerance means servers can fail but a system will remain available for all workloads.

2.3 MLLIB

MLlib (https://spark.apache.org/mllib/) is the machine learning library for Apache Spark. It is a fully-fledged tool containing many algorithms and utilities for several tasks ranging from classification (logistic regression, naive Bayes, etc.) to regression (generalized linear regression, survival regression, isolation regression, etc.) and gradient-boosted algorithms based on trees, e.g., random forests, multiple additive regression trees. It also provides distributed methods for clustering (K-means, Gaussian mixtures (GMMs), etc.) and for mining frequent item sets, association rules, and sequential patterns. MLLib also provides utilities for data pre-processing exploiting the underlying distributed Spark environment. Pre-processing of data includes feature transformations (standardization, normalization, hashing) and easy statistics on data (summaries, hypothesis testing, etc.). All this processing blocks can be also combined together toward the definition of a "Pipeline" of machine learning techniques that enable model evaluation and efficient hyper-parameter tuning. The persistence of the results is also implemented by the definition of methods for saving and loading models and pipelines to/from the file system.

2.4 H20 SPARKING WATER

Sparkling Water (https://www.h2o.ai/products/h2o-sparkling-water/) allows users to combine the fast, scalable machine learning algorithms of H2O with the capabilities of Apache Spark. Sparking Water stems from H2O, an in-memory platform for machine learning, the potential of performing effective and efficient machine learning on big data. Sparkling Water and Apache Spark allows for a seamless experience for users who want to interact with distributed databases/ filesystems, build a model and make predictions, and then use the results again in Apache Spark. It is used for exploring and analysing datasets held in cloud computing systems and in the Apache HDFS as well as in the conventional operating-systems Linux, macOS, and Microsoft Windows. The H2O software is written in Java, Python, and R. Its graphical-user interface is compatible with four browsers: Chrome, Safari, Firefox, and Internet Explorer.

3 BIGDATAGRAPES PREDICTIVE DATA ANALYTICS SETUP

The BDG platform relies on MLLib, the Apache HDFS file system and Apache Spark to allow the BDG users to develop methods for predictive data analytics, i.e., classification, regression, and clustering algorithms, and to interact and use them on the BDG distributed infrastructure. In the following sections we detail how to setup and install the BDG Predictive Data Analytics component and how to run the toy demonstrators provided in the first version of the deliverable. Demonstrators are implemented in Python as Jupyter Notebooks (http://jupyter.org/).

3.1 REQUIREMENTS

The Predictive Big Data Analytics functionality of the BDG platform is available after setting up the BDG architecture as detailed in D4.1. Interested users should refer to D4.1 and follow all the steps reported in Section 3 to setup a working BDG architecture.

3.2 DOCKER COMPONENTS SCHEMA

Figure 1 shows the components used by the Predictive Data Analytics functionality of BDG and their implementation as Docker components. In the Figure we made transparent all the components of the BDG architecture not explicitly needed by this functionality. As detailed above, the Predictive Data Analytics functionality relies on MLLib, Spark and HDFS to perform machine learning tasks over extremely large datasets. We also report the addresses and ports used by these containers. Particularly, the Spark component is composed of one master node/container and many workers nodes/containers linked to the master. Similarly, the HDFS component has one *namenode* container and many *datanodes* containers.

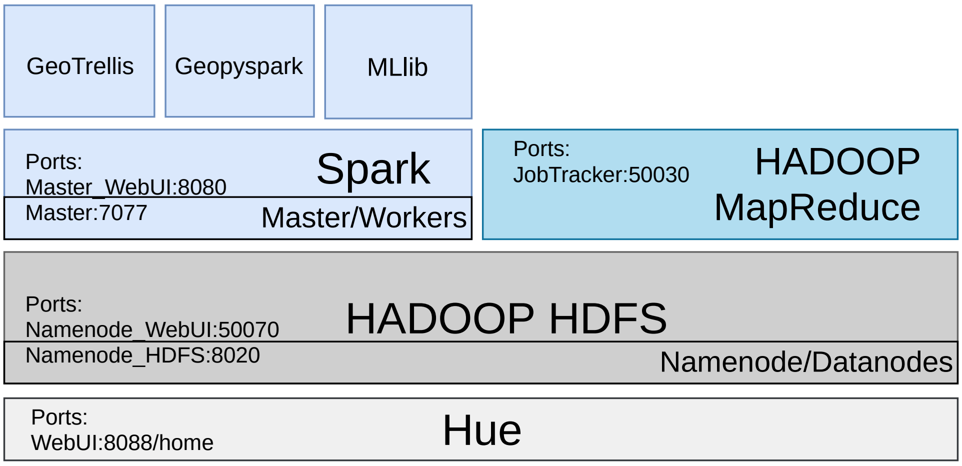


Figure 1: BigDataGrapes Components used by the Predictive Data Analytics functionality and their mapping on Docker Containers

3.3 HOW TO GET THE CODE

To get the code demonstrating the Predictive Data Analytics functionality of BDG, clone the following GitHub repository

```
$ git clone https://github.com/BigDataGrapes-EU/deliverable-D4.3.git
```

The repository contains three Jupyter Notebooks and a Bash shell script that should be used to initialize the Big Data Grapes Platform. The script should be executed after cloning the repository by running the Bash command below:

```
$ ./run-components.sh
```

The Bash script above downloads the Docker images and builds the environment according to the predefined configuration settings. The Bash script also starts the Docker containers of the BigDataGrapes software stack components.

Finally, to execute the demonstrators, run the following Bash command:

```
$ ./run-jupyter_notebooks.sh
```

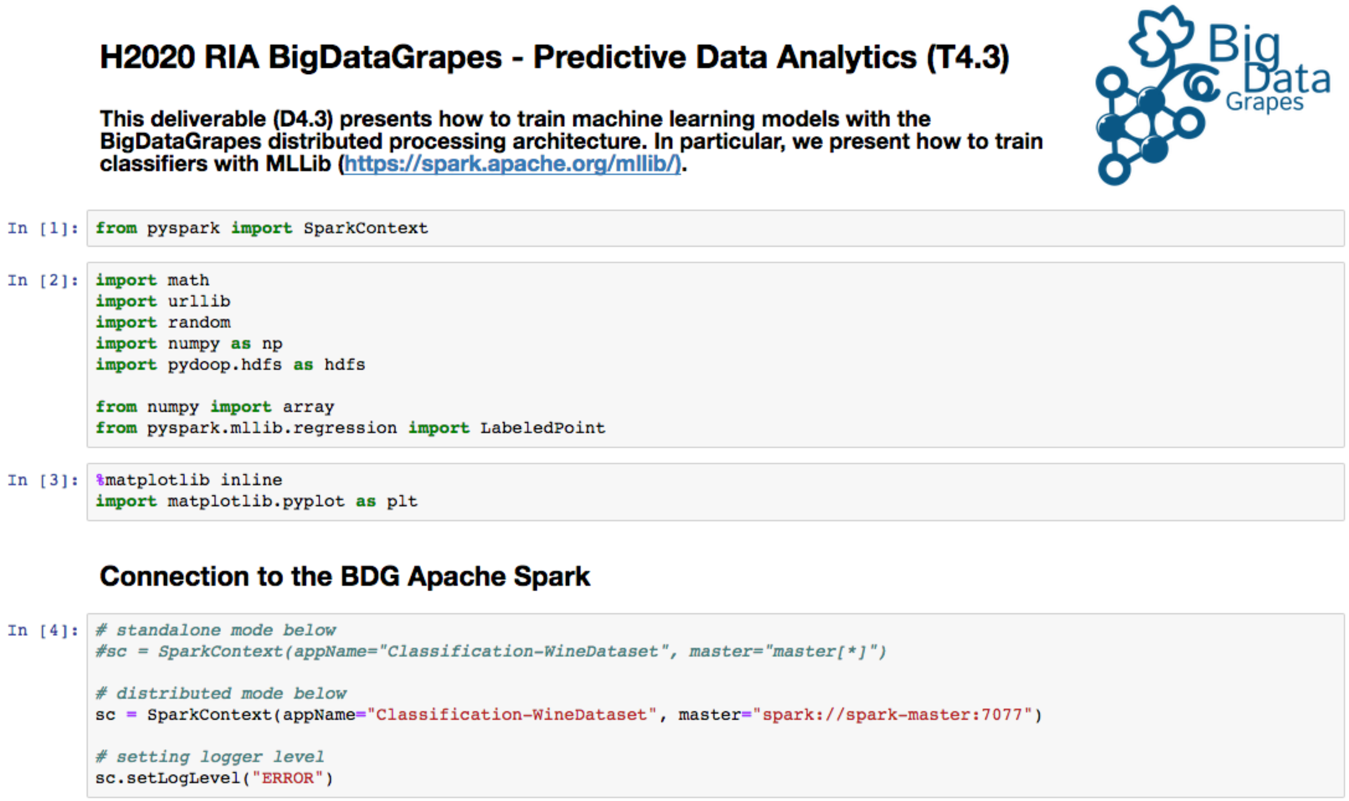
The Bash script above starts the Jupyter notebooks for predictive data analytics using the BigDataGrapes platform.

3.4 DESCRIPTION OF THE DEMONSTRATORS

The Predictive Data Analytics demonstrator consists of Python code made available in three Jupyter Notebook files. The first one shows how to train linear and random forest regressors with MLLib on synthetic data. The second one details how to perform a classification task with MLLib on a real-world public dataset, the KDD Cup 1999 challenge. The third one shows how to train multi-label classifiers with MLLib on the Red Wine Quality Dataset.

Jupyter Notebook (http://jupyter.org/) is an open-source Web application that allows the user to create and share documents that contain live code, equations, visualizations and narrative text. They are extremely useful for data cleaning and transformation, numerical simulation, statistical modeling, data visualization, machine learning, and much more. The Notebook supports over 40 programming languages. Code inside a notebook can produce rich, interactive output. More importantly, Jupyter Notebook can exploit big data tools, such as Apache Spark, from Python, R and Scala.

Figure 2 shows a snapshot of one of the Jupyter Notebooks of the demo. A Jupyter Notebook is a sequence of cells containing code, text, images, etc. The execution of each cell allows to run the code inside it. The output of each cell is then shown in the notebook so to allow easy interaction with the user.



H2020 RIA BigDataGrapes - Predictive Data Analytics (T4.3)

This deliverable (D4.3) presents how to train machine learning models with the BigDataGrapes distributed processing architecture. In particular, we present how to train classifiers with MLLib (https://spark.apache.org/mllib/).

```
In [1]: from pyspark import SparkContext
```

```
In [2]: import math
        import urllib
        import random
        import numpy as np
        import pydoop.hdfs as hdfs

        from numpy import array
        from pyspark.mllib.regression import LabeledPoint
```

```
In [3]: %matplotlib inline
        import matplotlib.pyplot as plt
```

Connection to the BDG Apache Spark

```
In [4]: # standalone mode below
        #sc = SparkContext(appName="Classification-WineDataset", master="master[*]")

        # distributed mode below
        sc = SparkContext(appName="Classification-WineDataset", master="spark://spark-master:7077")

        # setting logger level
        sc.setLogLevel("ERROR")
```

Figure 2: Snapshot of a Jupyter Notebook taken from the Predictive Data Analytics demonstrator.

The three demonstrators of the Predictive Data Analytics functionality of BDG are both structured around four different steps that are detailed in the following.

3.4.1 Connection to the BDG Apache Spark enabling distributed computation

This is a preliminary operations allowing Jupyter Notebook to connect to the Spark context used for the computation, i.e., the master node that manage the distributed computation on the cluster. Data generation/download for the regression/classification demonstrators works as follows:

- In the regression demonstrator we generate synthetic data and we preliminary plot them to allow the user to understand their shape and complexity.
- The binary classification demonstrator exploits real data from a well-known data challenge: KDD Cup 1999. The dataset is used for the Third International Knowledge Discovery and Data Mining Tools Competition, which was held in conjunction with KDD-99 The Fifth International Conference on Knowledge Discovery and Data Mining. The competition task was to build a network intrusion detector, a predictive model capable of distinguishing between "bad" connections, called intrusions or attacks, and "good" normal connections. This database contains a standard set of data to be audited, which includes a wide variety of intrusions simulated in a military network environment. The demonstrator shows how to build a Logistic Regression Classifier with MLLib. We also show how to interact with the BDG HDFS distributed file system by downloading and storing the dataset on HDFS.
- The multi-label classification demonstrator is built around the Red Wine Quality Dataset made available on Kaggle (https://www.kaggle.com/piyushgoyal443/red-wine-dataset/discussion). The dataset is related to red variants of the Portuguese "Vinho Verde" wine. Due to privacy and logistic issues, only physicochemical (inputs) and sensory (the output) variables are available (e.g. there is no data about grape types, wine brand, wine selling price, etc.). The dataset is made up of 1599 instances. Each instance has 11 attributes and one label, i.e., the quality of the wine produced expressed on a ten-grade scale, i.e., {1, 2, … , 10}. The classes are ordered and not balanced.

3.4.2 Reading of the data with Apache Spark (to create a Resilient Distributed Dataset (RDD)

The three demonstrators read the datasets used in two different ways. While the regression demonstrator shows how to build Spark RDDs from NumPy arrays the second and the third one, i.e., the ones dealing with classification read data from HDFS showing an example of interaction with the BDG distributed file system.

3.4.3 Training three different kinds of machine learning applications with MLLib

- Regression (Linear and Random Forest Regressors): We intentionally generate data by perturbating results from linear, square, and sine functions with gaussian noise. The visualization of the results of the regression clearly show that the three lines fitted follow the shape of the data used to fit them. Indeed, this kind of regressors are very simple yet not totally able to deal with the complexity of square and sine data (Figure 3).

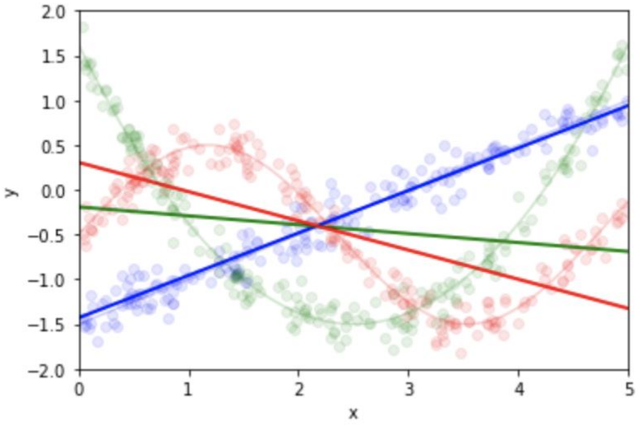


Figure 3: Application of three linear regression models (red, green and blue solid lines) trained on synthetic data generated by perturbating linear, square and sine functions with gaussian noise.

We then add a more complex (non-linear) regressor, i.e., random forest, to deal with the complexity of the data. The prediction produced is now more accurate as Figure 4 shows.

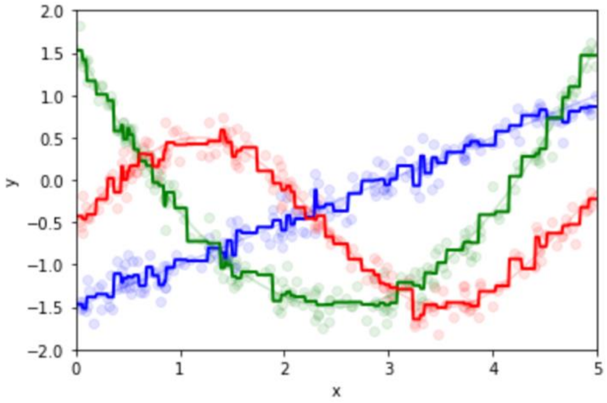


Figure 4: Application of three random forest regressors (red, green and blue solid lines) trained on synthetic data generated by perturbating linear, square and sine functions with gaussian noise.

- Binary Classification (Logistic Regression Classifier): In the binary classification demonstrator we train a binary classifier by employing logistic regression. The demonstrator is built around the KDD 1999 dataset that comes split in train/test. Train and test data are read from HDFS to allow Apache Spark to build two RDDs that are then used to build the classifier. We first build the classifier on training data.

We then perform an evaluation step by computing the accuracy metric, i.e., the proportion of true results (both true positives and true negatives) among the total number of cases examined, on the test set of the dataset. This information is then returned to the used to allow an easy comparison of the performance with other classifiers.

- Multi-label Classification (Logistic Regression Classifier): In the multi-label classification demonstrator we train a multi label classifier by employing logistic regression. The demonstrator is built around the Red Wine Quality Dataset that we split in train/test. The dataset is made up of examples labeled with a "quality" label expressed on a ten-grade scale. We thus learn a classifier to guess the correct class, i.e., quality of each wine. We learn the classifier after loading data from HDFS. We then perform an evaluation step by computing the accuracy metric, i.e., the proportion of true results (both true positives and true negatives) among the total number of cases examined, on the test set of the dataset. This information is then returned to the used to allow an easy comparison of the performance with other classifiers.

3.4.4 Disconnection from the BDG Apache Spark environment

This is the last, yet very important, phase needed to disconnect the Jupyter Notebook from the BDG big data platform. The operation allows for freeing the resources allocated and used by the computation. Each step above is documented inside the Jupyter Notebooks. We clearly identify each part by adding a textual cell describing when a specific step starts. We also add several cells allowing for visualizing the dataset and the intermediate results.

3.5 ASSESSMENT ON ONLINE WINE DATA

This section discusses the updates of this deliverable added at M15. The aim of this update is to assess on a large scale the machine learning methods for predictive data analytics provided on top of the BDG infrastructure. Specifically, we present the results of an investigation focusing on the application of machine learning methods on wine data collected from online social networks of wine passionate users. The data available consist of a general description of the wine (wine name, producer, year of production, wine grape varieties, etc.), user-generated content associated with it (user ratings and reviews), and user information (unique Id, nationality, etc.). The dataset used contains: 489,417 wine reviews by 195,678 users, written in 86 languages, related to 51,579 different wines, from 58 wine countries and 2272 wine regions.

The predictive analysis conducted on this dataset allows us to show the potential of the machine learning layer of the BDG infrastructure exploited in this case for descriptive analytics and for deploying efficient and effective methods for assessing the potential market penetration of a given wine in a new country. We estimate this penetration capability by learning a model from user-generated contents for estimating wine ratings in a target country from wine characteristics. In details, the investigation conducted aims at answering the four main research questions:

1. Is it possible to derive "aroma fingerprints" of a wine from user-generated content?
2. Is it possible to derive "taste fingerprints" of a wine from user-generated content?
3. Is it possible to exploit the fingerprints above to assess how wines are perceived in a specific country? Is there any similarity between countries in the way they perceive a specific wine?
4. Is it possible to build machine learning models that predict the user rating of a wine given its aromas and taste fingerprint?

3.5.1 Preliminary investigation of the data

We first perform a preliminary analysis of the data available. To this end, the following Table shows the number of reviews in the dataset after data cleaning grouped by language.

Table 1: Distribution of wine reviews by language.

Language	# Reviews
EN	357,347
FR	17,166
PT	13,007
IT	10,949
DE	8,388

We note that English is the most popular language. We thus focus our analysis on the reviews written in English. We now detail the reviews above by user country and plot the distribution obtained in the Figure below.

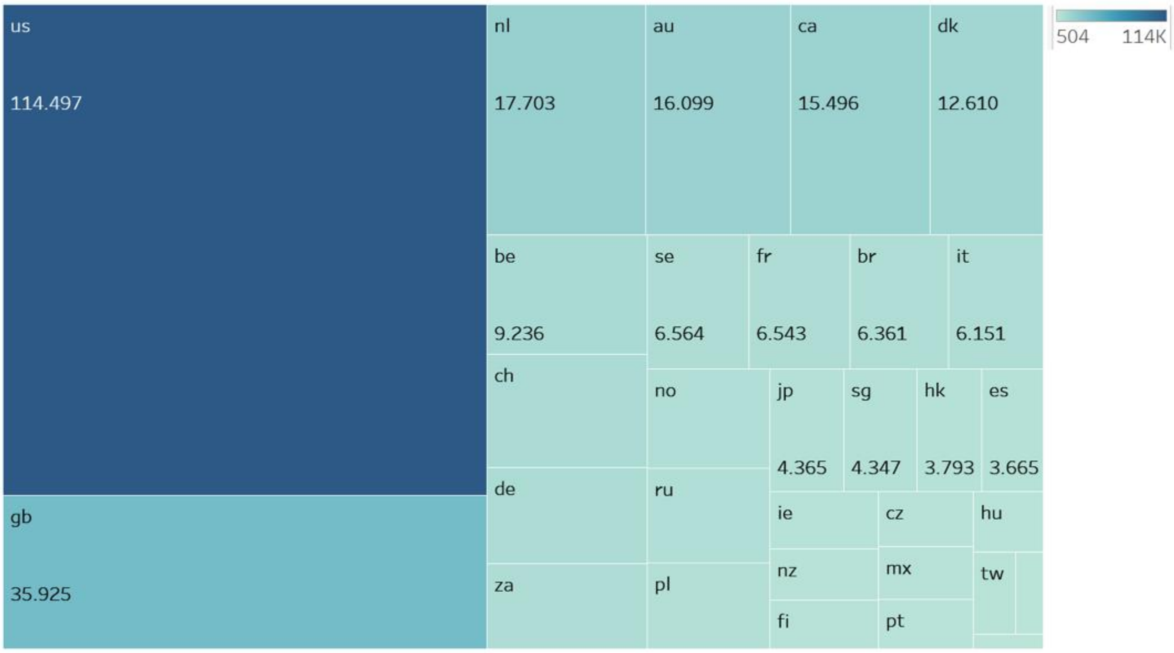


Figure 5: Distribution of wine reviews by country.

We observe that the countries with the highest number of reviews are US and GB. Netherland, Canada and Australia also contribute significantly. The distribution of reviews per country is highly skewed. We observe that non-English-speaking countries contribute with a significant number of reviews to the total even if the number of English reviews is larger than the sum of all the remaining non-English reviews. Figure 5 plots the distribution per user country of the reviews written in English.

Another interesting analysis regards the distribution of the wine rating that is expressed on a nine-graded scale from one to five, where one means “very bad” while five means “excellent”. We plot the distribution of the user ratings to understand what are the most popular classes. The figure below shows the number of reviews (Y axis) per rating value (X axis).

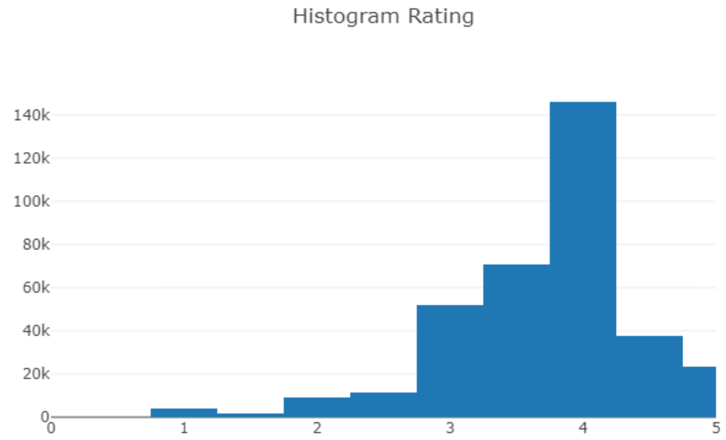


Figure 6: Distribution of wine ratings by score value.

We observe that the most popular classes are 3, 3.5, 4. These three classes cover together up to 270K reviews consisting in the majority of the reviews present in the dataset. We also observe that wines in the dataset are provided with an aggregation by wine grapes, i.e., wines are grouped together using the grape that characterizes them the most. We call this aggregation "wine type". We observe that the first four wine types are: cabernet-sauvignon, chardonnay, merlot, pinot-noir, sauvignon blanc. This aggregation of wines by wine grape allows us to perform the analysis we present in the next section.

3.5.2 Building aromas fingerprints of a wine from user-generated content

We now show how to exploit user reviews to build an aroma profile of a wine. To this end we select from our dataset the user reviews written in English. The generation of the aroma profile is done by identifying in the text of the reviews the occurrences of terms belonging to a taxonomy of wine aromas provided by one of the most popular social networks of wine lovers.

By using the user reviews, we build a multi-dimensional fingerprint of aromas perceived as characterizing a specific wine group on the basis of the user country. Each dimension of the fingerprint measures the presence of a specific aroma in the reviews of each group of wines. It is thus a term frequency vector where each element of the vector is the frequency of a specific aroma in the user reviews. After a normalization process, the per-country multi-dimensional vectors enable interesting similarity operations thus measuring the perception of the same wine type in different countries.

In the following figures, i.e., Figure 7 and 8, we show as examples the fingerprints of the most popular wine group, i.e., cabernet-sauvignon. The top half of Figures 7 and 8 present a heat-map of the vectors built for the top five most-popular countries (US, GB, NL, AU and CA). On the X-axis we report the different aromas taken from the taxonomy described before. The heat-map plots the frequency of each aroma in the user reviews of each specific country. The US is the country with the highest number of reviews showing also the highest variance in terms of aromas. On the contrary, GB, i.e., the second most popular country shows a narrow set of well-perceived aromas with a long tail of low-frequency aromas. The same consideration holds also for Canada and Australia, while the Netherlands and Denmark show a more selective fingerprint with no tail of low-frequency aromas.

The bottom half of Figures 7 and 8 present a heat map showing the similarity between countries computed starting from the vectors above for cabernet-sauvignon. The heat map visually shows how similarly cabernet sauvignon is perceived in different countries. The similarity is computed using the cosine distance. Interestingly, USA, Australia and Canada perceive cabernet sauvignon in a very similar way (similarity > 0.6). Indeed, the similarity between Canada and USA is even stronger (similarity > 0.8). On the contrary, the Netherlands has a different perception of this kind of wine as its similarity with all other English-speaking countries (USA, Canada, Australia, Great Britain) is always lower than 0.2. In terms of specific aromas, users from Canada and USA identify the presence of cherry in the cabernet sauvignon-based wines while users from Australia and Great Britain identify the prevalence of oak. Instead, users from the Netherlands report the presence of fruit/red fruits aromas. The software shared in the GitHub repository of the project (details in Section 3.3) proposes the same analysis for the five biggest wine groups, i.e., cabernet-sauvignon, chardonnay, merlot, pinot-noir, sauvignon-blanc. For each one we provide the two heat maps, the first one visualizing the aromas fingerprints per country while the second one visualizing the similarity between countries.

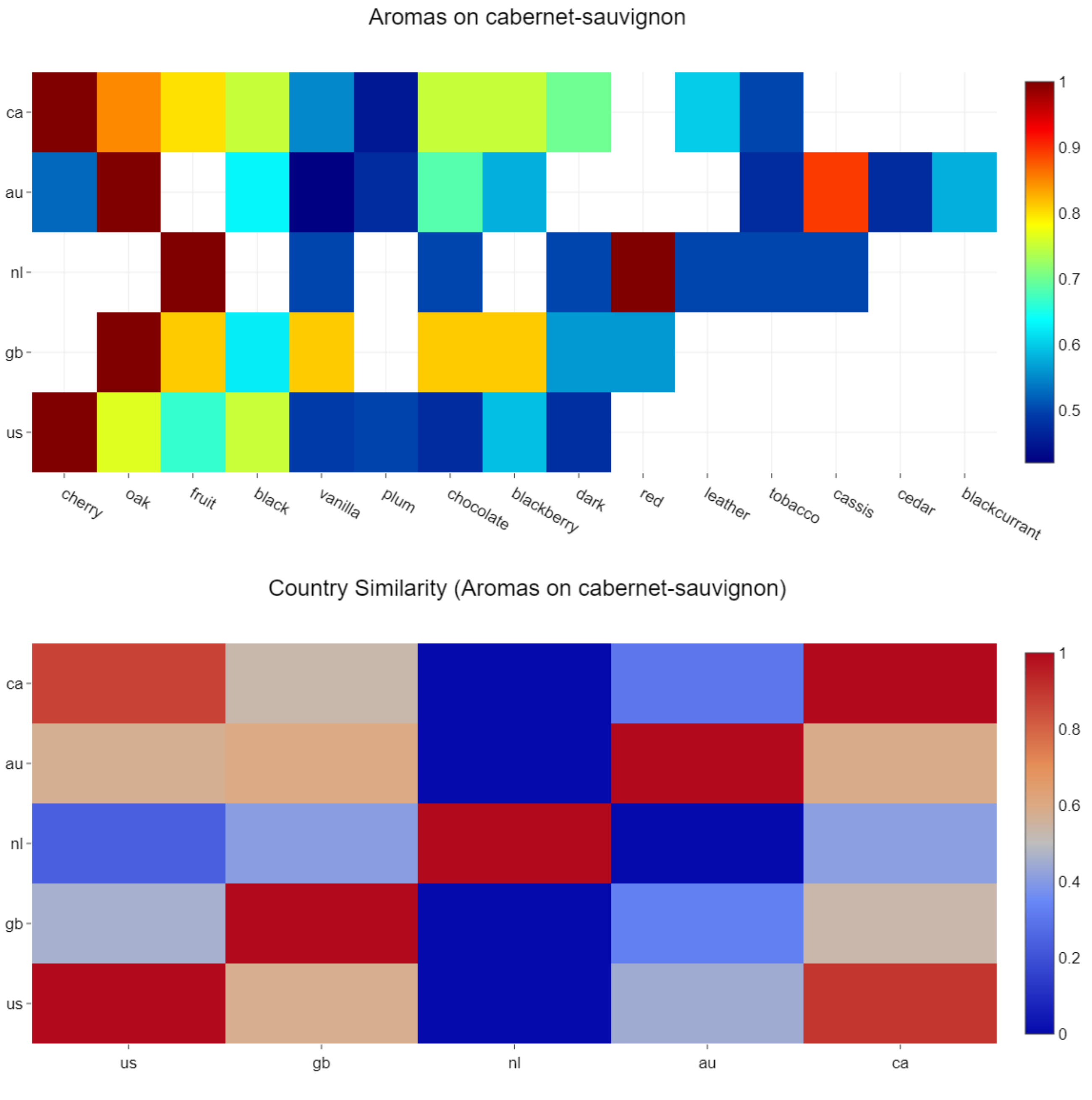


Figure 7: Aroma fingerprints perceived for cabernet-sauvignon by country and their similarities.

3.5.3 Building tastes fingerprints of a wine from user-generated content

We apply the same methodology described in Section 3.5.2 to derive a "tastes fingerprint" of a wine from user-generated content. In this case we used the taxonomy of wine tastes available here: https://winefolly.com/tutorial/wine-aroma-wheel-100-flavors/. As before, we report the results for the largest wine group, i.e., cabernet sauvignon in Figure 8. The analysis confirms the results presented before for aromas. USA shows a very high similarity with Canada (similarity > 0.8) while it is also similar with Australia (similarity > 0.6). The Netherland has its own way of perceiving cabernet sauvignon while Great Britain shows slight similarities with Canada and the Netherland (similarity > 0.5). In terms of tastes, English-speaking countries (USA, GB, Australia and Canada) identify smoothness in cabernet sauvignon while GB and the Netherlands frequently report its fruity characteristics.

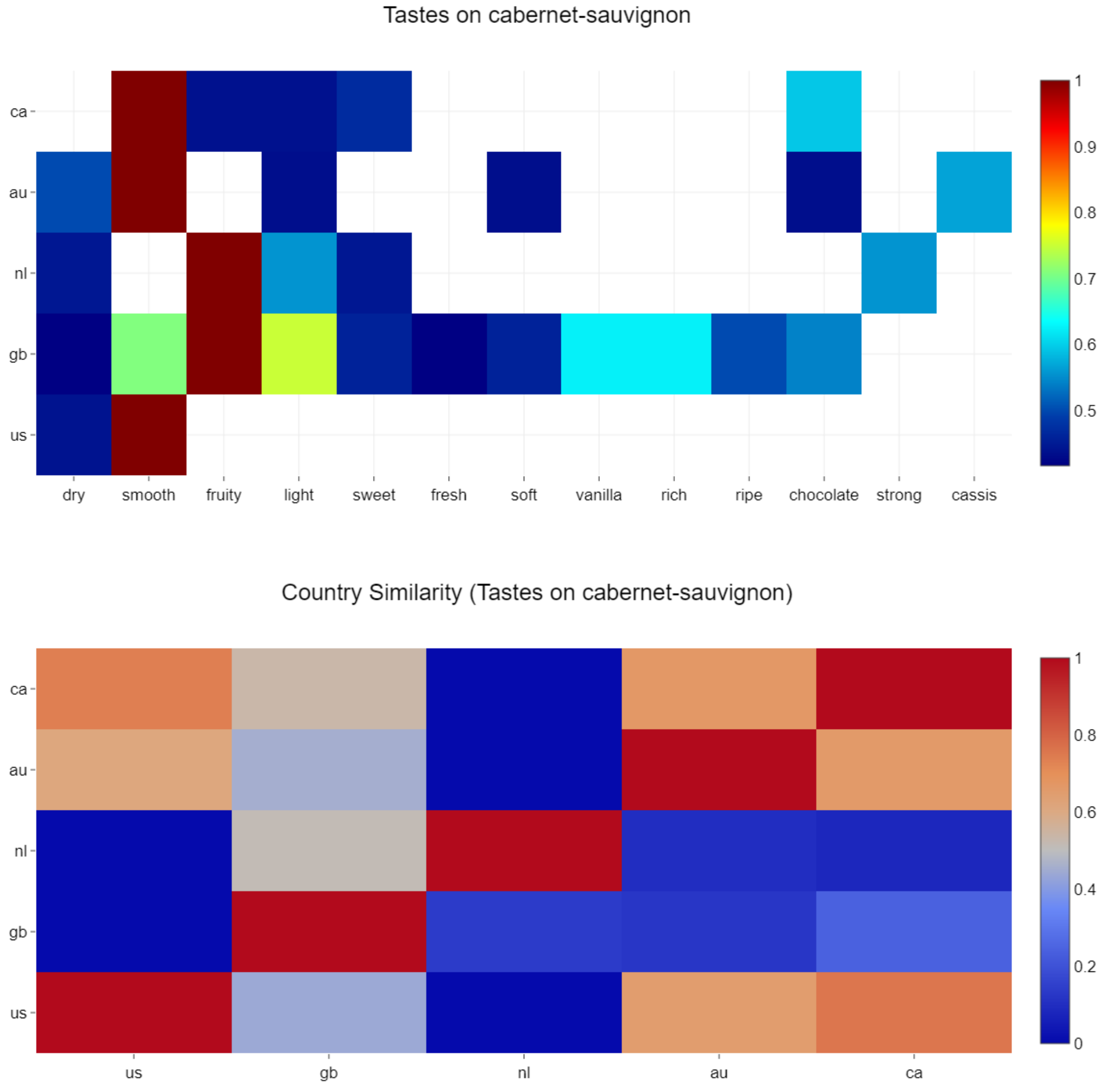


Figure 8: Taste fingerprints perceived for cabernet-sauvignon by country and their similarities.

The analysis presented in Sections 3.5.2 and 3.5.3 answer the first three research question: we derive a way to synthetize aromas and tastes fingerprints from wine user reviews. The fingerprints are based on a public taxonomy of aromas from domain experts. We also show how to understand the per-country perception of a wine by exploiting these fingerprints. The fingerprints allow to identify the most important aromas that users from a given country identify in a wine. We believe this is an interesting tool for wine makers that allow them to better target their products for specific markets. In addition, the proposed fingerprints built from user-generated content can also be exploited to compute the similarity of perception of users of different countries thus allowing to better understand the market.

3.5.4 Machine Learning for User Rating Prediction

We now present the application of supervised machine learning techniques to the data. We apply machine learning to build a user rating predictor, i.e., predict the rating of a wine in a given country. We focus this analysis on user reviews from the USA. We exploit the BDG platform to learn three different kind of predictors, i.e., Decision Trees, Random Forest and Gradient Boosted Trees regressors. We also employ two different feature representation of the wine:

1. We first employ the metadata of each wine collected from online social networks, i.e., user country, wine country, name of the wine region, year of the wine, wine alcohol, wine acidity, wine body, wine grapes.
2. We include in the above representation the TF-IDF representation of user reviews, i.e., we add the user feedback expressed in terms of reviews.

The three models are optimized by using grid search with a standard three-fold validation methodology. The software provided on GitHub details how to learn the three regressors by using the BDG platform. Figure 9 shows the performance in terms of RMSE of the three models each one trained with the two different feature sets. The best prediction performance in predicting the user rating is achieved by the Gradient Boosted Trees model. The exploitation of the user reviews in the feature set (GBT-TFIDF) allows this predictor to achieve an RMSE of 0.553. On the contrary, the Decision Tree predictor is the worst performing predictor with a performance that is up to 6% worse than the one provided by GBT-TFIDF. Indeed, the decision tree regressor is not able to exploit the user reviews as the performance achieved by DT-TFIDF (0.588) is almost the same the one obtained by the plain DT (0.59). Random Forest shows a similar behavior even if the performance of the two predictors is better than the one obtained by Decision Tree regressors.

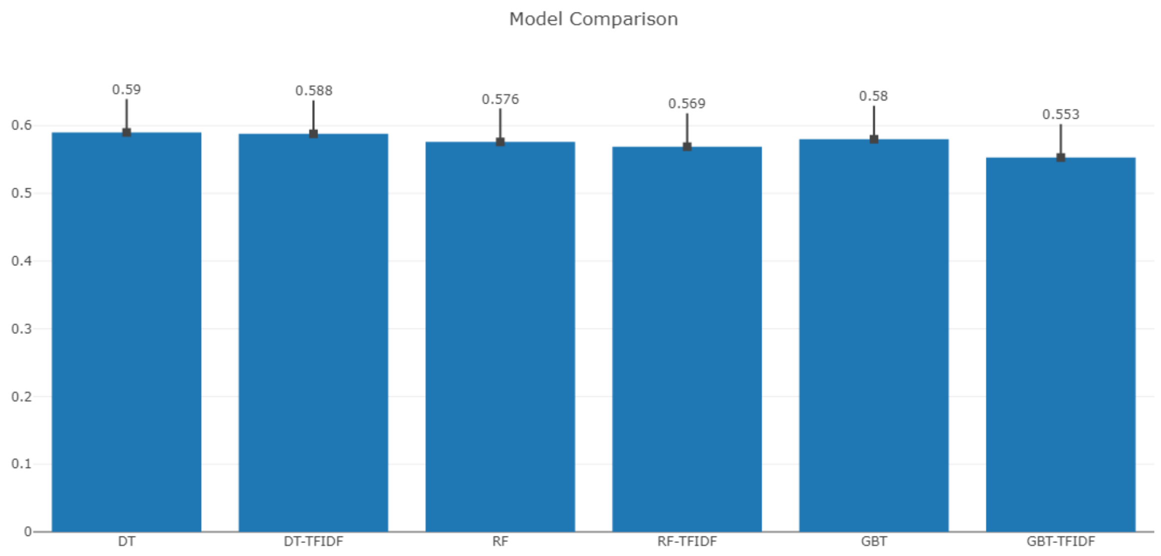


Figure 9: RMSE of the various predictors on wine rating estimation.

This section shows that the BDG infrastructure allows to run state-of-the-art machine learning methods to build user rating predictors. We present two different feature sets derived from our data and show that the exploitation of the user reviews as features for wines allows to improve the predictive power of the models. Specifically, one of them, Gradient Boosted Trees predictor, shows a good improvement (up to 4.9%) when exploiting textual information from reviews. We conclude this section by presenting an analysis of the feature importance, i.e., what predictive power are the features providing for the model. As before, this analysis is done on the two feature sets by training a random forest with a maximum depth equal to 15 and number of trees equal to 15. The analysis performed on the first feature set, i.e., the one not exploiting the review text, (Figure 10) identifies that two out of the three most important features for predicting the rating of a wine for the US users are: the name of the region and the country where the wine is produced. This means that the provenience of the wine is an important signal to consider when predicting the rating of a wine. The second and, from the fifth feature on, all features belong to characteristics of the wine itself, i.e., acidity, alcoholic degree, body, composition in terms of grapes, and year of production. This is an expected result as these are all important dimensions to consider when reviewing a wine. Another important feature highlighted by this analysis regards the list of foods to pair with the wine (the fourth most important feature) meaning that users evaluate also this aspect when expressing their rating. From Figure 9, we learn that user reviews are important to better focus the user rating prediction. We then evaluate the importance of the features when employing the second feature set, i.e., the one combining user reviews with the metadata of each wine.

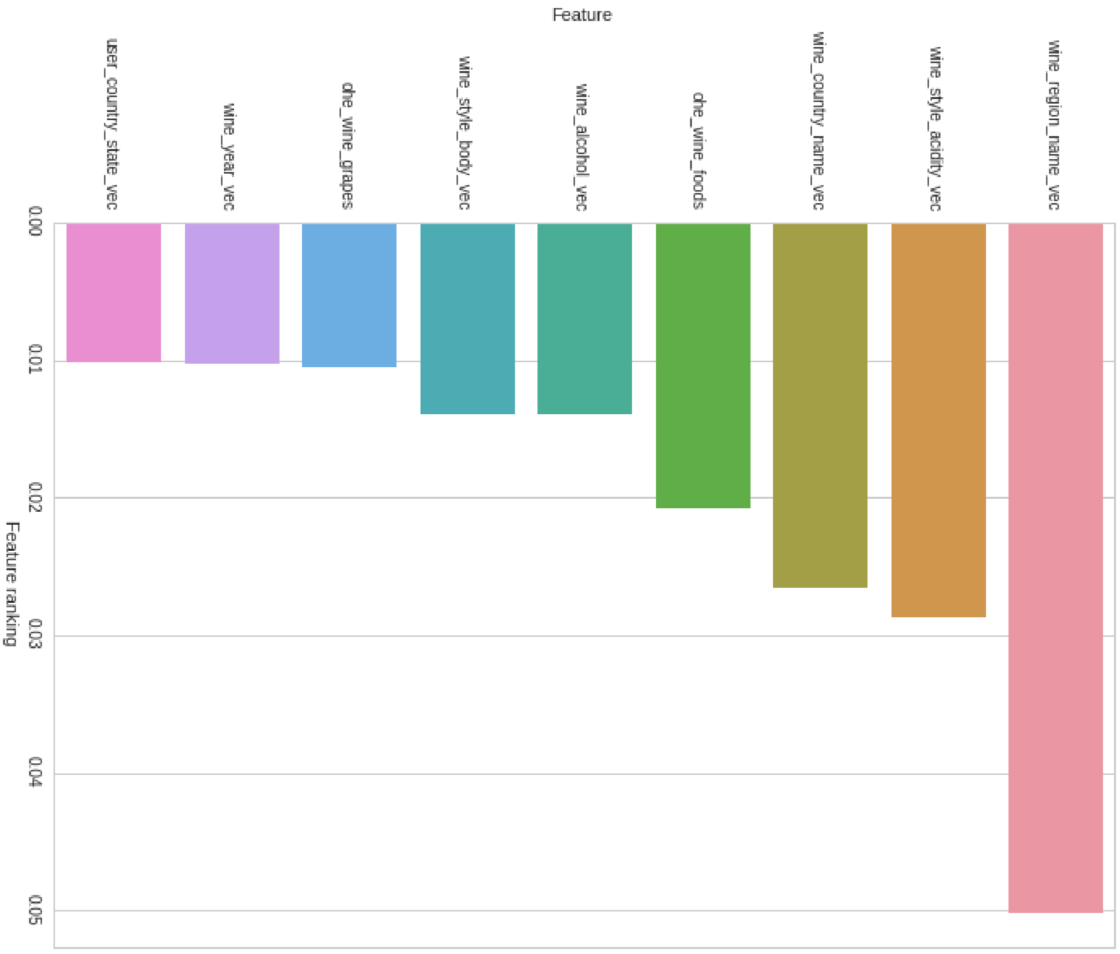


Figure 10: Feature importance for the first feature set.

The feature importance of the second feature set used is shown in Figure 11. Here we observe a swap in the feature importance. The first most important feature is, in fact, the country where the wine is produced followed by the region of the wine. This confirms that geographical information of the wine are important signals in determining the prediction of the user rating. Differently from before, the fourth most important feature used by the model here is the TF-IDF weight of the terms composing the review. This behavior confirms what has been observed before (Figure 9), i.e., user reviews are a significant source of information for predicting the user rating of a wine. Some features characterizing the nature of the specific wine, i.e., acidity, style, and the foods to pair with are also present here confirming their importance in determining the prediction.

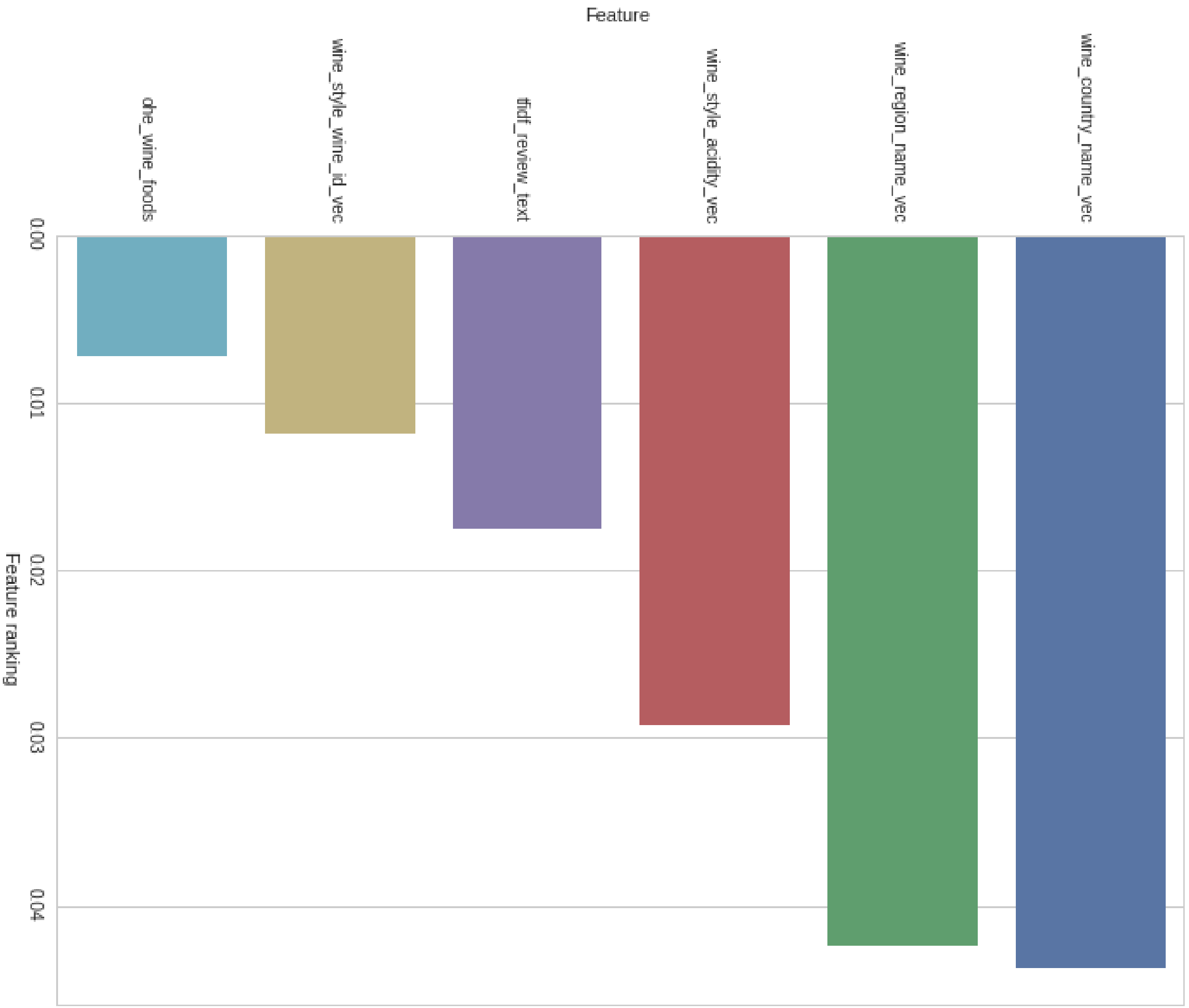


Figure 11: feature importance for the second feature set.

This investigation is an answer to the fourth research question: it is possible to exploit online wine data (both metadata and user reviews) to build effective predictors of the user rating. Experiments show that the exploitation of user reviews allows for a better prediction model. Indeed, this investigation confirms the validity of the BDG platform to learn predictive analytics effectively over large and heterogeneous datasets.

3.6 HOW TO RUN THE CODE

To run the demonstrators, after following the instructions detailed in Section 3.3, the user should point her browser to the following Jupyter Notebook URL: http://<server_address>:9999

The password used to protect the Jupyter Notebook instance is “bigdatagrapes”.

After a successful login the user can open the files below:

- D4.3-PredictiveDataAnalyticsWithMLLib-Regression-Py2.ipynb
- D4.3-PredictiveDataAnalyticsWithMLLib-Classification-Py2-KDDCUP1999.ipynb
- D4.3-PredictiveDataAnalyticsWithMLLib-MultiLabelClassification-Py2-WineDataset.ipynb
- D4.3-WineDataAnalysis-FlavorsTastes-PredictionUserRating.ipynb

The execution of the code can be done by running each cell from the beginning to the end of each notebook and wait for the result.

4 SUMMARY

This accompanying document for deliverable D4.3 (Models and Tools for Predictive Analytics over Extremely Large Datasets) describes the first version of the mechanisms and tools supporting efficient and effective predictive data analytics over the BigDataGrapes platform in the context of grapevine-related assets.

The BDG software stack employs efficient and fault-tolerant tools for distributed processing, aimed at providing scalability and reliability for the target applications. On top of this stack, the BDG platform enables distributed predictive big data analytics by effectively exploiting scalable Machine Learning algorithms using the computational resources of the underlying infrastructure efficiently. The software components enabling BDG predictive data analytics have been designed and deployed using Docker containers. They thus include everything needed to run the supported predictive data analytics tools on any system that can run a Docker engine.

In this first version of D4.3 we assessed the effectiveness of different predictive data analytics tools on very simple datasets in terms of standard and popular metrics such as Accuracy. In the updates delivered in this document, we added a rich and complex analysis of a large dataset of wine reviews highlighting the potentiality of the BDG tools used to assess the potential market penetration of a given wine in a new country. We estimate this penetration capability by learning a model from a very large dataset of user-generated wine content containing 489,417 wine reviews by 195,678 users, written in 86 languages, related to 51,579 different wines, from 58 wine countries and 2272 wine regions.