Combining Machine Learning and Operations Research Methods to Advance the Project Management Practice

Nikos Kanakaris, Nikos Karacapilidis, Georgios Kournetas, and Alexis Lazanas(✉)

Industrial Management and Information Systems Lab, MEAD, University of Patras, 26504 Rio Patras, Greece
{nkanakaris, kournetag}@upnet.gr,
{karacap, alexlas}@upatras.gr

Abstract. Project Management is a complex practice that is associated with a series of challenges such as handling of conflicts and dependencies in resource allocation, fine tuning of projects to avoid fragmented planning, handling of potential opportunities or threats during the execution of a project, and alignment between projects and business objectives. Traditionally, methods and tools to address these issues are based on analytical approaches developed in the realm of the Operations Research discipline. Aiming to facilitate and augment the quality of the Project Management practice, this paper proposes a hybrid approach that builds on the synergy between contemporary Machine Learning and Operations Research techniques. Based on past data, Machine Learning techniques can predict undesired situations, provide timely warnings and recommend preventive actions regarding problematic resource loads or deviations from business priority lists. The applicability of our approach is demonstrated through two real examples elaborating two different datasets. In these examples, we comment on the proper orchestration of the associated Operations Research and Machine Learning algorithms, paying equal attention to both optimization and big data manipulation issues.

Keywords: Project Management · Machine Learning · Operations Research · Intelligent optimization

1 Introduction

Project Management (PM) is a complex practice that is highly fluid and hard to predict, thus imposing a series of challenges to organizations and experts [20]. Such challenges may concern alignment between projects and their business objectives, handling of conflicts and dependencies in resource allocation, fine tuning of multiple projects to avoid fragmented planning, as well as informed and diffused decision making to handle potential opportunities or threats during the execution of a project [39].

At the same time, PM is inherently collaborative and knowledge-intensive. Issues to be addressed are characterized by ever-increasing amounts of different types of data and knowledge, which may be obtained from various sources and vary in terms of

G. H. Parlier et al. (Eds.): ICORES 2019, CCIS 1162, pp. 135–155, 2020.
https://doi.org/10.1007/978-3-030-37584-3_7

subjectivity, ranging from individual opinions and estimations to broadly accepted practices and indisputable measurements and results [22]. Furthermore, their types can be of diverse level as far as human understanding and machine interpretation are concerned.

Up to now, the majority of methods and tools aiming to facilitate and augment the quality of PM are based on the application of advanced analytical approaches developed and elaborated in the realm of the Operations Research (OR) discipline. These approaches employ techniques such as mathematical optimization and statistical analysis to look for optimal or suboptimal solutions to diverse PM issues. In addition, the application of Artificial Intelligence (AI) techniques to automate project management has been first proposed more than 30 years ago. At that time, the proposed AI-leveraged project management systems used knowledge processing and procedural techniques to provide new kinds of decision support for project objective-setting and control [23].

Nowadays though, the adoption of AI in the data-intensive and cognitively-complex PM settings enables a series of advancements. AI - and in particular Machine Learning (ML) - techniques can aid project managers easily delegate thousands of tasks, while sustaining a holistic view of their resources and projects. This contributes to the achievement of the required accuracy and precision when dealing with bottlenecks or constraints that may obstruct business processes. At the same time, these techniques can aid managers and experts to interpret big volumes of data and gain valuable insights towards improving their overall PM practice. Based on past data, they can predict undesired situations, provide timely warnings and recommend preventive actions regarding problematic resource loads or deviations from business priority lists [20].

Admittedly, each of the abovementioned disciplines (OR and AI) has significantly contributed to the improvement of the PM practice, by addressing the associated issues from a different philosophy and research perspective. Moreover, both disciplines elaborate a mixture of problem modeling and problem solving methods. According to Radin [30], an OR analyst must trade off tractability (i.e. "the degree to which the model admits convenient analysis") and validity (i.e. "the degree to which inferences drawn from the model hold for real systems"). At a high level, the OR and ML analysts face the same validity and tractability dilemmas and it is not surprising that both can exploit the same optimization toolbox.

However, we argue that the joint consideration of these two disciplines has not been thoroughly explored yet, and has much potential to further augment PM-related business intelligence. Such an approach will concentrate on both planning and execution of individual projects, as well as on their association with past data and their impact on the wider business. Moreover, this approach can appropriately represent and process the associated data and knowledge, while at the same time remedy the underlying cognitive overload issues. Particular attention should be also given to the expression and maintenance of tacit knowledge (i.e. knowledge that employees do not know they possess or knowledge that they cannot express with the means provided), which predominantly exists and dynamically evolves in PM settings [20].

In line with the above remarks, this paper expands on [20] by attempting to shape a hybrid approach for better handling PM issues by meaningfully integrating tools

originally developed in the context of OR and AI. More specifically, in Example 1 (Sect. 4.1): (i) we describe in depth the Data attributes concerning Construction Projects as well as the involved constructors, (ii) we apply data pre-processing methods in order to retain data that contribute in gaining statically significant information, (iii) we perform the Kruskal-Wallis test to determine whether a statistically significant difference between constructors and projects' *Delay* attribute exists and (iv) we formulate the scoring function in order to provide more accurate results. Additionally, in Example 2 (Sect. 4.2): (i) we redefine the overall approach by using classification instead of clustering methods and we provide more in depth analysis of each step, (ii) we increase the success chances of the project by properly assigning the available developers to each project issue, (iii) we examine our approach using real data contrary to hypothetical data used in [20] and (iv) we evaluate our approach by utilizing the *Local Surrogate Models (LIME)* explanation method [15, 32] in order to get a solid understanding of the underlying mechanism of our trained model.

The remainder of the paper is organized as follows: Sect. 2 discusses background work considered in the context of our approach, which is analytically described in Sect. 3; the applicability of the proposed approach is demonstrated through two realistic examples in Sect. 4; the contribution of our approach is discussed in Sect. 5; finally, concluding remarks and future work directions are outlined in Sect. 6.

2 Background Issues

Numerous software solutions to project management exist in the market nowadays. The list of the most widely adopted ones includes Wrike (www.wrike.com), Asana (www.asana.com), Trello (www.trello.com), and Jira (www.atlassian.com/software/jira). These solutions offer a user-friendly environment that mainly enables issue tracking and supports various PM functions. In addition, by providing interactive graphics, issue boards and timelines, they simplify planning, collaboration, reporting and time management. It is broadly admitted that existing commercial PM solutions may increase an organization's productivity and prevent the teams from diverging from their actual goals. However, they unintentionally hide important PM-related information, due to the complex multidimensional data found in the hosted projects.

At the same time, by adopting an AI-perspective, a range of digital project management assistants has been already developed, including solutions such as Stratejos.ai (www.stratejos.ai), PMOtto.ai (www.pmotto.ai), and x.ai (www.x.ai). This category of solutions is based on seamless, easy-to-use interfaces that assist project managers in common tasks (e.g. a project's supervision). They rely on the expressiveness, immediacy, interactivity and descriptiveness that natural language provides to offer a 'zero-level' entrance environment. They are used to automate repetitive work such as creating project's tasks by analyzing textual conversations, to remind and organize important events such as meetings, to extract shallow insights (e.g. 'top contributors of the week'), and to answer simple queries (e.g. 'what is my team working on today?').

We argue that this second category of solutions offers narrow predictions and automations. In particular, their underlying reasoning mechanisms mainly build on rules to store and manipulate knowledge, and ignore contemporary AI technologies that

can uncover insights, perform more complex tasks, make explainable recommendations, and support informed decision making, sometimes in ways that outperforms what people are able to do today. Furthermore, each of these digital personal assistants is relevant to a specific project management need (e.g. reporting, scheduling meetings, organizing events); thus, they are unable to embrace a 'single-access-point' approach that mitigates the overall PM complexity.

From an OR perspective, a series of techniques and tools have been proposed and extensively used to solve various PM related issues. OR techniques provide solutions in problems such as prediction, resource allocation, forecasting, scheduling, task assignment, networking etc. These techniques are supported by very useful software libraries such as *pyschedule* (`github.com/timnon/pyschedule`), *PuLP* (`github.com/coin-or/pulp`), Google OR-tools (`developers.google.com/optimization`), JuMP.jl [12], Hungarian.jl (`github.com/Gnimuc/Hungarian.jl`) and CVXPY (www.cvxpy.org). These libraries support a variety of OR techniques including integer, linear, convex and dynamic programming. However, these techniques tend to add more complexity on the overall PM practice, mainly due to the complicated mathematical models needed to operate. Another drawback is that these techniques are unable to learn by the system's experience, which often results to the proposition of optimal or near-optimal solutions that are not realistically feasible.

With the advent of big data and cloud computing era, machine learning techniques gain ground in a variety of scientific and commercial sectors. These techniques (and corresponding algorithms) can categorize items, predict values, identify meaningful relationships, and detect data patterns or unexpected behavior (anomaly detection). ML approaches are usually grouped into four categories, namely supervised learning, semi-supervised learning, unsupervised learning and reinforcement learning [14].

Supervised learning refers to the process of learning aiming to predict values (e.g. house prices) or classify items into categories (e.g. categories of projects) by using labelled training data. Common algorithms and methods used in supervised learning include k-nearest neighbors, naive Bayes, decision trees, linear regression, and support vector machines. Semi-supervised learning combines both labeled and unlabeled input data for training, where in most cases there is a small amount of labeled data and a huge amount of unlabeled data available. Unsupervised learning analyzes unlabeled data to identify patterns or cluster similar items into groups using alternative distance metrics (e.g. Euclidean distance, Manhattan distance). Common algorithms used in unsupervised learning include k-means, DBSCAN, OPTICS, Apriori and hierarchical clustering [3, 40]. Finally, reinforcement learning approaches iteratively interact with their environment to identify specific actions that maximize a reward or minimize a risk. Common algorithms and methods used in this category include Q-learning, temporal difference, and deep adversarial networks. The above ML techniques and algorithms are fully supported today by various software libraries and environments, such as *scikit-learn* [28], *H2O.ai* [8], *Tensorflow* [1], *PyTorch* [27] and *WEKA* [17].

As a last note, it is worth mentioning that most AI-based approaches to PM build on artificial neural networks. Related works discuss how neural networks are capable to assist project managers in problems such as resource allocation, prediction, clustering, classification [7] and forecasting [44]. Neural network techniques have been also applied to predict construction cost and schedule success [43]. Other representative

works concern development of a neural network to estimate project performance [9], or to classify the level of a project's riskiness by exploiting the knowledge extracted from data concerning past successful and unsuccessful projects [10]. An interesting overview of the different types of neural network models applied in business can be found in [37].

3 The Proposed Approach

In ML, generalization is the most essential property used to validate a novel approach. For a practical ML problem, the analyst might pick one or more families of learning models and an appropriate training loss/regularization function, and then search for an appropriate model that performs well, according to some estimate of the generalization error based on the given training data [11]. This search typically involves some combination of data preprocessing, optimization and heuristics [34]. Every stage of the process can introduce errors that can degrade the quality of the resulting inductive functions. In the related literature, particular attention is paid to three sources of such errors. The first source of error is due to the fact that the underlying true function and error distribution are unknown, thus any choice of data representation, model family and loss functions may not be suitable for the problem and thus introduce inappropriate bias [35]. The second source of error stems from the fact that only a finite amount of (possibly noisy) data is available. Thus, even if we pick appropriate loss functions, models and out-of-sample estimates, the method may still yield inappropriate results [36]. The third source of error stems from the difficulty of the search problem that underlies the modeling problem under consideration. Reducing the problem to a convex optimization by appropriate choices of loss and constraints or relaxations can greatly help the search problem [33].

As far as ML algorithms are concerned, these can be distinguished in four categories concerning data classification, value prediction, structure discovery, and detection of anomalies or abnormal behavior. More specifically:

- Data classification aims to predict which category the input data belongs to. For example, in a software development project, a new task can be classified into distinct categories (e.g. story, bug, epic) based on its attributes using a decision tree classifier.
- Value prediction concerns regression algorithms to predict continuous numerical values. For example, in a common PM scenario, these algorithms can estimate the budget of a project by exploiting knowledge of similar, already accomplished projects using simple linear regression techniques, thus providing advice to the project manager during the planning phase on possible cost reduction decisions.
- Anomaly detection algorithms aim to identify unusual events or patterns that do not conform to usual or expected behavior. For example, in a certain maintenance setting, these algorithms can detect outages of some components before they occur and proactively act towards keeping the whole system functioning.
- Structure discovery aims to uncover data patterns, reveal hidden or not obvious relationships and divide data items into groups with similar traits (features). This is

achieved using widely-adopted ML techniques (e.g. k-means and Apriori algorithms). For example, in a construction PM problem, the Apriori algorithm can mine frequent itemsets concerning constructors and project durations to build useful association rules (e.g. constructor x is always late when delivering dam construction projects).

Considering the pros and cons of the techniques discussed in the previous section, in this paper we propose a hybrid approach to handle PM issues, which builds on a proper integration and orchestration of PM tools originally developed within the ML and OR disciplines. ML, which has become a buzzword nowadays [31], adopts a predictive analytics approach of the form 'if A happens, then B is likely to happen', which attempts to exploit available past data to create useful insights (i.e. make human-like decisions). On the other hand, OR adopts a prescriptive analytics approach to provide optimal solutions (courses of action) to problems of the form 'what does A need to be if we want B to happen' (i.e. make perfect decisions) [13].

We consider tools coming from the ML and OR fields as complementary, arguing that there is room for integration in a way that ML can create and refine A → B relationships that are often considered as optimal and remain unchanged upon the entry of new data in classical OR approaches. Despite the features that ML possesses in terms of data refinement and value prediction, it lacks algorithms aiming to provide optimal solutions, something that is inherent in OR techniques. Overall, our approach considers OR and ML as complementary to each other, and proposes an iterative interplay between them, where ML supplies OR algorithms with refined, accurate and up-to-date data (based on past records), while OR contributes to making optimal decisions with the continuously updated data input.

The proposed approach enables interpretation of big volumes of PM data to support preventive actions such as giving advice about resource assignments by identifying similar skills and expertise necessary to perform a task, make explainable recommendations about the capacity levels of certain resources based on historical performance data, and support informed decisions concerning a company's expansion to a new region or design of an efficient supply chain [20]. The proposed approach augments the overall PM decision-making process, by enabling the drawing of reliable conclusions about conditions and future events, while also identifying potential risks and opportunities.

4 Examples

As highlighted in the previous section, depending on the specific PM issue under consideration, our approach advocates a proper streamlining of ML and OR algorithms. In this section, we demonstrate its applicability through two realistic examples concerning resource assignment. Emphasis is given to the complementarity of ML and OR algorithms to advance the associated PM practice.

4.1 Example 1

Based on real data[1] concerning implementation of public construction projects in the Region of Attica, Greece, for the period 2003–2014, we consider the following problem [20]:

Let $P = \{P_1, P_2, ..., P_n\}$ be a set of future public construction projects. Each project (P_n) is described by a list of attributes, namely P_n= *[PID, {M_i}, category, est_cost, funding, duration]*, corresponding to a unique project identifier, the municipality to manage the project, the type of construction needed, the project's estimated cost, the source funding the project, and its estimated duration, respectively.

Similarly, let $C = \{C_1, C_2, ..., C_m\}$ be the set of registered construction companies, each of them being associated with the set of attributes *[CID, {$Location_i$}, {$Category_j$}, {$Cost\_Range_k$}, {$Duration\_Range_l$}, Cost_Overrun, Delay]*, corresponding to a unique constructor identifier, the municipality where the constructor is active, the type of projects the constructor deals with (e.g. flood control, health infrastructure), the projects' budget category the constructor is interested in (e.g. large scale (>1,5 M€), medium scale (0,5 M€–1,5 M€), the projects' duration range (e.g. short term (<6 months)), mid term (6–18 months)), the project's final budget overrun/underrun, and the delay (percentage) caused by the constructor, respectively. The abovementioned data attributes are summarized in Table 1.

Table 1. Data attributes for Projects (P_n) and Constructors (C_m).

Attribute	Description	Values	Attribute type
Location	The municipality to manage the project	*{M_i}*	Nominal
Category	The type of the project	*e.g. {Buildings, Roadworks, Athletics, ...}*	Nominal
Estimated_Cost	Initial budget available	*[3.000, 16.7 M] €*	Numerical
Cost_Range	The project's budget category	*{Small_Scale, Medium_Scale, Large_Scale}*	Ordinal
Cost_overrun	The project's final budget overrun/underrun	*[-76.18%, +38.55%]*	Numerical (%)
Funding	The source of funding	*{Region, EU, Third Party}*	Nominal
Duration	The estimated duration of the project	*[5, 2587] days*	Numerical
Duration_Range	The project's duration range	*{Short Term, Mid Term, Long Term}*	Ordinal
Delay	The delay in project's accomplishment	*[-402, 1485] days*	Numerical

[1] Data available in: https://drive.google.com/file/d/13oixL7QuKtE2NidtBJEEHaoFpvylRXlH/view?usp=sharing.

Data Pre-processing. In the dataset under consideration, there exist a number of project categories containing a limited number of occurrences with respect to the total projects accomplished, as shown in Table 2. To retain only data that contribute in gaining statically significant information, we remove all records with relative project categories frequency below 1%. Moreover, constructors with limited participation in projects' construction are also removed. As a result, the original dataset is reduced from 684 to 607 transaction records.

Table 2. Relative frequencies of project categories.

Project category	Relative frequency
Athletics	2,50%
Buildings	15,02%
Cultural	**0,15%**
Drains	**0,59%**
Education infrastructure	1,91%
Flood control	16,20%
Health infrastructures	2,95%
Nursery schools	**0,44%**
Other interventions	6,92%
Planning study	5,45%
Port works	1,62%
Roadworks	35,05%
Social infrastructures	1,47%
Urban reconstruction	8,84%
Water supply	**0,88%**
Total	100%

Next, to determine whether a statistically significant difference between $Constructors_m$ and projects' *Delay* attribute exists, we perform the Kruskal-Wallis test[2] using SPSS in the remaining records (see Table 3). The outcome of the Kruskal-Wallis test suggests that the null hypothesis should be retained, hence the *Delay* variable is excluded from our dataset.

Table 3. Kruskal-Wallis test for *Delay*.

Null hypothesis	Test	Sig.	Decision
The distribution of Delay % is the same across Constructors	Kruskal-Wallis test	0.289	**Retain the null hypothesis**

Additionally, we perform a Kruskal-Wallis test to determine whether a statistically significant difference between $Constructors_m$ and *Cost_Overrun* variable exists. As

[2] We selected Kruskal-Wallis test due to the fact that our data are not normally distributed.

shown in Table 4, the outcome of the test suggests that the null hypothesis should be rejected, hence the *Cost_Overrun* variable is included in our dataset.

Table 4. Kruskal-Wallis test for *Cost_Overrun*.

Null hypothesis	Test	Sig.	Decision
The distribution of Cost_Overrun % is the same across Constructors	Kruskal-Wallis test	0.000	**Reject the null hypothesis**

Application Scenario. Let a project management scenario where there are $n = 3$ projects of various categories and $m = 6$ available constructors. Obviously, each P_n requires a different expertise, while each C_m possesses a distinct number of skills extracted from past data. To determine the constructors that best fit to the projects' requirements, we need to populate a (P_n, C_m) score matrix (each entry taking values in the range [0, 1]). This is through the calculation of: (i) the Jaccard similarity index $J(P_n, C_m)$ [19], and (ii) an additional score value $Score_{C,M}$ for the attribute *CostOverrun* of each C_m (this attribute does not participate in the calculation of the Jaccard similarity index).

We define:

$$Score(C_m, P_n) = norm[avg(cost\_overrun(C_m, P_n))] \tag{1}$$

where: *norm[avg(cost_overrun(Cm, Pn))]* responds to the normalized average cost_overrun for C_m constructor in the specific P_n project category.

$$J(P_n, C_m) = |P_n \cap C_m| / |P_n \cap C_m| \tag{2}$$

where: $J(P_n, C_m)$ represents the Jaccard similarity index between C_m constructor and the corresponding P_n project category.

For the calculation of the constructor's final rating, we also normalized the output values of the Jaccard similarity index in the range [0, 1]. The final rating of the C_m constructor for the P_n project is calculated by the following formula:

$$Rating_{n,m} = [a * J(P_n, C_m) - b * Score(C_m, P_n)] \tag{3}$$

where: a and b are two coefficients (in the range [0..1]) aiming to promote the best combination of similarity and cost overrun. In the specific dataset, their values are 0.9 and 0.1, respectively.

Table 5. The (P_n, C_m) score matrix ($Rating_{n,m}$).

	Athletics	Roadworks	Buildings
C_2	**0.93**	0.00	0.00
C_3	0.00	0.00	0.15
C_9	0.00	0.00	**0.40**
C_4	0.00	0.00	0.19
C_{11}	0.00	**0.44**	0.03
C_{15}	0.00	0.24	0.06

By using formulas 1–3, we calculate the (P_n, C_m) score matrix (Table 5).

Aiming to minimize the total construction cost of these projects, the problem is considered as a typical linear assignment problem (LAP), which can be easily solved through tools available in widely used software packages such as Google OR-Tools (https://developers.google.com/optimization/assignment/simple_assignment). Using the linear assignment solver of the above software package, we get the outcome presented in Table 6.

Table 6. (Pn, C_m) assignment matrix.

Project	Athletics	Roadworks	Buildings
Constructor	C_2	C_{11}	C_9

Aiming to further improve the accuracy of our estimations, we next consider the exploitation of ML algorithms, which are capable to provide knowledge-based patterns of construction projects' data. More specifically, we propose the use of the Apriori Algorithm [3] in order to discover meaningful patterns (itemsets) relating P_n and C_m attributes.

We consider the transaction set $T = \{T_1, T_2, \ldots, T_{607}\}$ from a total of 607 transactions available in our dataset. The application of Apriori algorithm provides us with a "strong" supported i-itemset that has been generated for constructor C_m (see Table 7; it is noted that, due to space limitations, we present only the final step of the algorithm, omitting intermediate calculations of k-itemsets). As we notice, the Apriori algorithm confirms the proposed assignment presented in Table 6 by generating the corresponding L_4 itemsets (rules), which also contain the constructors C_2, C_{11}, C_9. Apart from the above outcomes, there is a strong indication suggested by the $L_2 \cup L_4$ that in case of *Long_Term* projects, constructor C_{15} might outperform constructor C_{11}. Comparatively, the OR and ML method produce the same results; however, the ML method does not necessitate the construction of a new variable (i.e. $Rating_{n,m}$). In addition, the ML method yields a more elaborated (multi-dimensional) picture of the reality without any further configuration. In the example shown above, this concerns indication of additional constructors that may be assigned to a specific project.

Table 7. Li itemsets for constructor C_m.

Constructor (C_m)	Large itemset	(L_i)
C_2	Athletics, Small_Scale, Short_Term, Region	(L_4)
C_{11}	Roadworks, Small_Scale, Short_Term, Region	(L_4)
C_9	Buildings, Small_Scale, Short_Term, Region	(L_4)
C_{15}	**Roadworks, Small_Scale, Mid_Term, Region**	**(L_4)**
C_{10}	Flood Control, Small_Scale, Short_Term	(L_3)
C_{15}	**Long_Term, Region**	**(L_2)**

Our approach is sketched in a pseudo-code form below:

Algorithm 1.1

```
1: for each (Pn) do
     {
2:     find similarity(Pm, Cm);
3:     calculate_Score(Pn,Cm) matrix;
     }
4: assign(Pn, Cm);
5: for each Cn do
     {
6:     Apply Apriori_Algorithm;
7:     Generate Li-itemsets;
     }
8: assign(Pn,Cm);
```

To summarize the basic concepts of the above example, we addressed a PM issue as a typical OR assignment problem (a group of constructing companies need to accomplish a set of construction projects) using a score matrix with estimations for each *(Pn, Cm)* element. The LAP solver algorithm provided a solution to the problem prescribing the optimal assignment matrix. Next, we exploited ML (the Apriori algorithm) to discover association rules between transactions' data to spot trends, relationships and structure similarity between data sets. In this way, we demonstrated that ML models and algorithms can be used to re-feed/alter initial OR solutions, integrating OR's prescriptive analytics with ML's predictive analytics orientation.

4.2 Example 2

In this example[3], we extract and analyze data[4] from the publicly accessible Jira instance of Apache Software Foundation (https://issues.apache.org/jira). This dataset concerns the development (i.e. issue tracking, bug fixing, implementation of new features, etc.) of the Apache Hadoop project [26] and contains information related to 1000 Jira issues. It has been retrieved using the open-source Python library 'jira' (https://github.com/pycontribs/jira), which relies on the official REST API of Jira. Each Jira issue in our dataset has, among others, the following important attributes (Table 8):

Table 8. Data attributes for Jira issues.

Attribute	Description	Values	Attribute type
Id	The identifier of the issue	*e.g. {1345}*	Integer
Key	The textual identifier of the issue	*e.g. {HADOOP-1345}*	String
Labels	The labels of the issue	*e.g. {KeyStore, security, tpm}*	String
Assignee	The assignee of the issue	*e.g. {john_doe, jane_doe}*	String
Status	The project's current status	*{Close, In Progress, Open, Patch Available, Reopened, Resolved}*	String
Components	The parental architectural components of the module that concerns the issue	*e.g. {Back-end, Front-end, Main-Framework}*	String
Description	The description of the issue	*Unstructured text*	String
Summary	The title of the issue	*Unstructured text*	String
Reporter	The reporter of the issue	*e.g. {john_doe, jane_doe}*	String
Resolution Date	The resolution date of the issue	*e.g.* {1560771216}	Timestamp
Created at	The date the issue has been created	*e.g. {1560771216}*	Timestamp

Data Cleansing. Similarly to the previous example, we retain only the Jira issues that: (i) have an assignee (in our dataset 677 out of 1000), and (ii) contribute in gaining statistically significant information (174 out of 677). As resulted, most of the remaining Jira issues have been assigned to only 4 developers (see Fig. 1 - for obvious reasons, the real names of the developers have been replaced by common first names).

[3] Code available at: https://github.com/nkanak/advance-project-management-practice.

[4] Data available at: https://github.com/nkanak/advance-project-management-practice/blob/master/data/hadoop_issues.json, retrieved at: 10 Dec 2018.

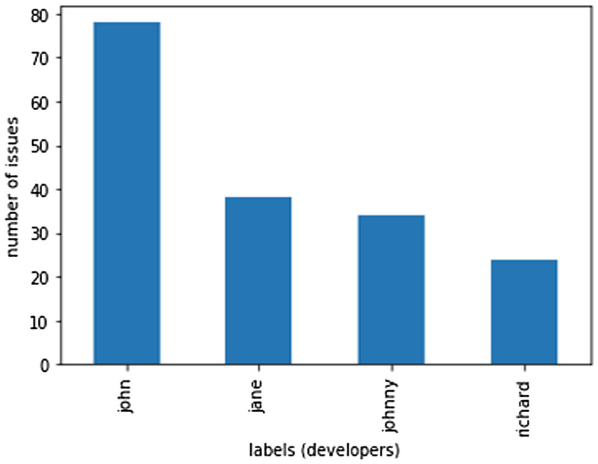

Fig. 1. Number of issues assigned per developer.

Feature Generation and Feature Extraction. To extract features needed in the next steps, a variety of Natural Language Processing (NLP) techniques, including feature generation and text normalization, is applied to the most important textual attributes of each issue, namely 'summary' and 'description'. This process can be accomplished through the following steps:

- Tokenization: the process of generating tokens from unstructured text;
- Lemmatization: the process of grouping together the different inflected forms of a word;
- Stemming: the process of reducing derived words to their root form;
- Feature/Tag generation: the process of transforming an unstructured text into a vector that contains word occurrences;
- Removal of stop words: the process of removing commonly used words, which usually add noise to the ML models;
- Removal of frequently occurred words: the process of removing insignificant words from a text, taking into consideration the document frequency value of each word; in our example, we remove the words with document frequency greater than 60%.

Application Scenario. We aim to increase the success chances of the project by properly assigning the available developers to Jira issues. We argue that a project has more chances to succeed if this assignment considers the skills of each developer and the nature of the job to be accomplished in each issue. For a given issue X and a developer Y, we define a relevance metric as the probability that the developer Y

possesses the skills required by issue X and, consequently, his/her capability of successfully completing the issue on time [16]. Our approach includes the following two steps:

- The estimation of how relevant each developer is to undertake (and successfully accomplish) each Jira issue;
- The assignment of developers to tasks, based on the linear assignment problem (LAP) algorithm [6], in a way that maximizes the total relevance metric.

As far as the calculation of relevance is concerned, we train a ML model (number of training records: 118, number of testing records: 56) that exploits the available textual information concerning each issue (i.e. 'summary' and 'description' attributes). Hence, we reduce the relevance calculation problem to the common and well-studied text classification problem. We adopt the multinomial Naive Bayes[5] classifier since it is suitable for classification with discrete features such as word counts [2]. The textual information of each issue is assigned to a class (i.e. the class with the highest probability) [25]. The set of classes of our model consists of the names of the available developers (i.e. 'assignee' attribute). Despite the fact that the multinomial Naive Bayes classifier assigns only one class (i.e. developer) to an issue, it also predicts the probability that the issue belongs to all other classes (i.e. how relevant is this issue to each available developer). It is clear that in our approach the relevance metric for each developer is equivalent to the probability calculated by our model.

For the needs of the example described in this section, we elaborate an application scenario using the textual information contained in the following four issues of our dataset:

Issue 1

Text: ''branch-2 site not building after ADL troubleshooting doc added Toc error on the ADL troubleshooting doc from HADOOP-15090{code}[ERROR] Failed to execute goal org.apache.maven.plugins:maven-site-plugin:3.5:site (default-cli) on project hadoop-azure-datalake: Error parsing 'hadoop-trunk/hadoop-tools/hadoop-azure-datalake/src/site/markdown/troubleshooting_adl.md': line [-1] Error parsing the model: Unable to execute macro in the document: toc -> [Help 1]{code}''
Actual Class: ''john''
Predicted Class: ''john''

[5] Python class used: `sklearn.naive_bayes.MultinomialNB`.

Issue 2

Text: ''S3 listing inconsistency can raise NPE in globber FileSystem Globber does a listStatus(path) and then, if only one element is returned, {{getFileStatus(path).isDirectory()}} to see if it is a dir. The way getFileStatus() is wrapped, IOEs are downgraded to null. On S3, if the path has had entries deleted, the listing may include files which are no longer there, so the getFileStatus(path),isDirectory triggers an NPE. While its wrong to glob against S3 when its being inconsistent, we should at least fail gracefully here.
Proposed
log all IOEs raised in Globber.getFileStatus @ debug
catch FNFEs and downgrade to warn
continue
The alternative would be fail fast on FNFE, but that's more traumatic''
Actual Class: ''john''
Predicted Class: ''john''

Issue 3

Text: ''ABFS: Code changes for bug fix and new tests
- add bug fixes.
- remove unnecessary dependencies.
- add new tests for code changes.''
Actual Class: ''johnny''
Predicted Class: ''johnny''

Issue 4

Text: ''Release Hadoop 2.7.7
Time to get a new Hadoop 2.7.x out the door.''
Actual Class: ''john''
Predicted Class: ''john''

Applying the LAP algorithm[6] to the elements of Table 9, the final step assigns developers to issues.

Table 9. Relevance of each developer per issue/task (%).

	T_0	T_1	T_2	T_3
John	96	95	8	53
Jane	0	1	0	43
Richard	4	0	0	3
Johnny	0	4	92	1

[6] https://developers.google.com/optimization/assignment/simple_assignment.

The outcome of the above process is shown in Table 10.

Table 10. The issue/task assignment matrix.

Issue	T_0	T_1	T_2	T_3
Developer	Richard	John	Johnny	Jane

Evaluation Measures. The above multinomial Naive Bayes classifier has a mean accuracy score[7] of 82.7%. In order to get a solid understanding of the underlying mechanism of our trained model, we explain the predictions of the model using the *Local Surrogate Models (LIME)* explanation method [15, 32]. In brief, this method tries to explain why single predictions of black-box ML classifiers were made by perturbing the dataset and building local interpretable models. In this case, the LIME text explainer[8] randomly removes words/features from the text of each issue and calculates the importance of a specific word to the decision made by the Naive Bayes classifier.

The provided explanations help us check the [21] of the trained ML model; they also confirm that the model selects the right label/class for the right reason (i.e. meaningful words/features). For instance, Fig. 2 explains why 'john' is (correctly and rationally) selected to work on a specific Jira issue with the following text:

Text: "`branch-2 site not building after ADL troubleshooting doc added. Toc error on the ADL troubleshooting doc from HADOOP-15090 {code} [ERROR] Failed to execute goal org.apache.maven.plugins:maven-site-plugin:3.5:site (default-cli) on project hadoop-azure-datalake: Error parsing 'hadoop-trunk/hadoop-tools/hadoop-azure-datalake/src/site/markdown/troubleshooting_adl.md': line [-1] Error parsing the model: Unable to execute macro in the document: toc -> [Help 1]{code}`".

As illustrated in Fig. 2, features such as '*hadoop'*, '*adl'*, '*doc'*, '*azure'* and '*troubleshoot'* play an important role in the decision made (choosing 'john') by the classifier. Additionally, features such as '*maven'*, '*tools'*, '*plugin'*, '*error'* and '*project'* increase the probability of 'richard' being the most suitable developer for the selected issue.

[7] Python method used: `sklearn.naive_bayes.MultinomailNB.score`.

[8] Python class used: `lime.lime_text.LimeTextExplainer`.

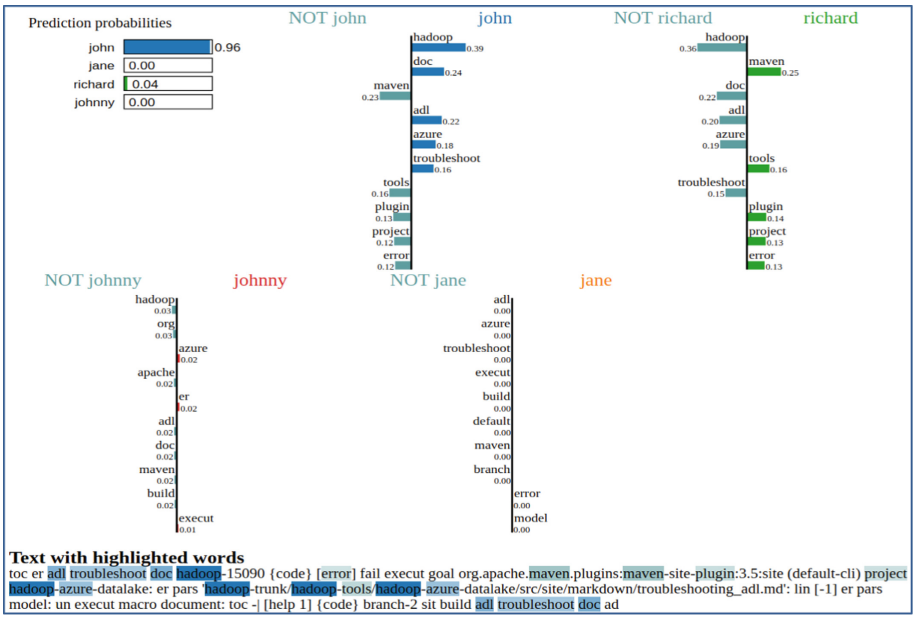

Fig. 2. Explaining the selection of developers.

Future Improvements. This example highlights the need for leveraging ML and OR techniques to augment the outcome of the classic linear assignment problem. However, during the development of our approach, various limitations and problems have been identified. First of all, the amounts of statistically significant data are limited. Secondly, the developed classifier, in some cases, predicts the correct developer for wrong reasons, as depicted in Fig. 3; features such as '*door*', '*new*', '*release*' and '*time*' affect the decision of our ML model. Finally, several important attributes remain unused; these attributes include the resolution time and the importance of an issue (e.g. '*blocker*', '*critical*', '*major*', '*minor*', '*trivial*').

To overcome the abovementioned shortcomings, we aim to enrich our approach; as far as the ML part of our approach is concerned, further removal of stop words, synonym identification [4], and part-of-speech tagging [24] is required. Also, as proposed in [38], the enrichment of the corpus using Wikipedia knowledge may improve the accuracy of the Naive Bayes classifier. Furthermore, contemporary knowledge representations such as knowledge graphs [5] and 'document to graph' [29, 41, 42] may mitigate problems that have arisen from the 'curse-of-dimensionality' phenomenon and improve the accuracy of the classifier. In the OR part of our approach, constraint satisfaction solvers can be integrated in order to enable the exploitation of usable features ranging from time and budget constraints to scheduling problems (e.g. top-ranking issues with the highest importance) [18].

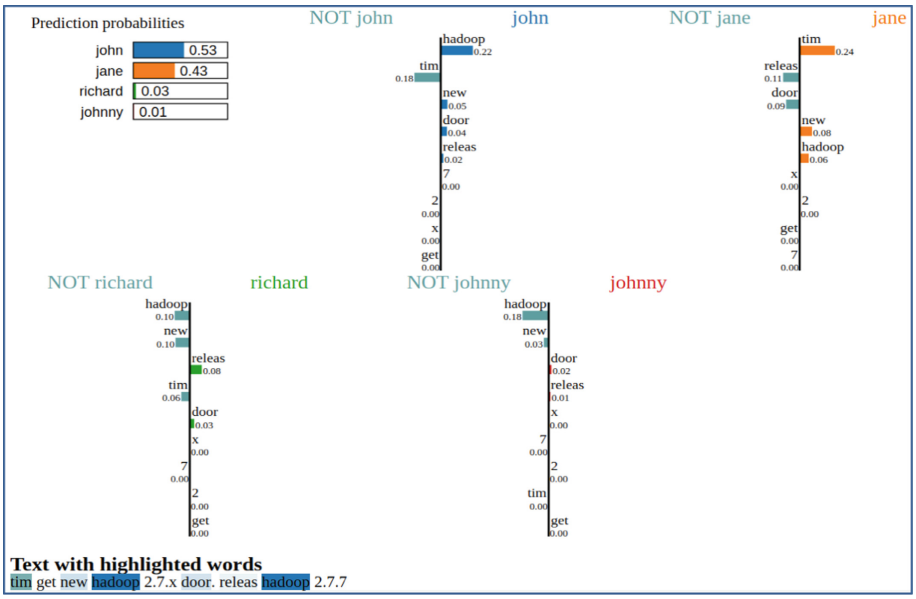

Fig. 3. Wrong classifier predictions.

5 Discussion

Key enablers that are driving the development of the proposed approach are the availability of huge computing power, the existence of big volumes of PM data and knowledge, as well as the accessibility of a range of well-tried and powerful OR and ML software libraries. Undoubtfully, there is more computing power available today than ever before, something that contributes significantly in making OR and ML algorithms extremely powerful, in ways that were not possible even a few years ago. In fact, this computing power enables us today to process massive amounts of PM data and extract valuable knowledge needed to make our models more intelligent. At the same time, as discussed in Sect. 2, software needed to process the diversity of PM data is open and freely available; it is also noted here that PM-related AI algorithms become available and get commoditized via dedicated APIs (Application Programming Interfaces) and cloud platforms.

Despite the above advancements, much work must still be carried out on the proper manipulation of PM data and knowledge, as far as its labeling, interrelation, modeling and assessment are concerned; this has mainly to be done by humans. Especially in the context of project management, one should always take into account that valuable data and knowledge emerge continuously during an organization's lifecycle, and concern both the organization *per se* (e.g. a project's duration, overall budget, KPIs etc.) and its employees (e.g. one's competences and performance, knowledge shared during a decision-making process etc.).

Building on a meaningful and flexible integration of OR and ML techniques and associated tools, our approach enables organizations to reap the benefits of the AI revolution. It allows for new working practices that may convert information overload and cognitive complexity to a benefit of knowledge discovery. This is achieved through properly structured data that can be used as the basis for more informed decisions. Simply put, our approach improves the quality of PM practice, while enabling users to be more productive and focus on creative activities. However, diverse problems and limitations still exist; these concern the value and veracity of existing data, as well as the availability of the massive amounts of data required to drive contemporary AI approaches.

6 Conclusions

This paper presents a hybrid approach aiming to assist the overall PM practice. We demonstrated that in ML and OR exist a variety of techniques enhancing a vast number of issues and tasks such as: resource assignment problems, task(s) duration estimation and task accomplishment prediction. Optimization and big data manipulation both a key issue in ML and OR correspondingly, are handled with equal attention in order to reach the desired outcome in PM tasks. It is worth to notice that our work further extends [20] by embedding explainability features in the recommendations provided in the presented examples.

From a broader point of view, we also argue that there is a major lack of scientific research that relates to AI solutions being developed for specific functions such as project management [30]. Thus, a key issue for future research is to study AI-based solutions for specific functions purely from a scientific perspective. On the other hand, there are no scientific theories that help one understand how a technology that has not been fully understood or developed yet can affect functions that traditionally rely on cognitive input or human interactions [36]. Therefore, another research direction should contribute to the development of theoretical models in order to help understand the impact of technologies on such functions.

References

1. Abadi, M., et al.: Tensorflow: a system for large-scale machine learning. In: Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation, Savannah, pp. 265–283 (2016)
2. Aggarwal, C.C., Zhai, C.: Mining Text Data. Springer, Heidelberg (2012). https://doi.org/10.1007/978-1-4614-3223-4
3. Agrawal, R., Srikant, R.: Fast algorithms for mining association rules. In: Proceedings of 20th International Conference on Very Large Data Bases (VLDB 1215), San Francisco, pp. 487–499 (1994)
4. Alkhraisat, H.: Issue tracking system based on ontology and semantic similarity computation. Int. J. Adv. Comput. Sci. Appl. **7**(11), 248–251 (2016)
5. Betts, C., Power, J., Ammar, W.: GrapAL: Querying Semantic Scholar's Literature Graph. arXiv preprint arXiv:1902.05170 (2019, to appear)

6. Burkard, R.E., Dell'Amico, M., Martello, S.: Assignment Problems, Philadelphia (2009)
7. Burke, L.I., Ignizio, J.P.: Neural networks and operations research: an overview. Comput. Oper. Res. **19**(3), 179–189 (1992)
8. Candel, A., Parmar, V., LeDell, E., Arora, A.: Deep Learning with H2O, 6th edn. H2O.ai Inc. http://h2o.ai/resources/. Accessed 19 Oct 2018
9. Cheung, S.O., Wong, P.S.P., Fung, A.S., Coffey, W.: Predicting project performance through neural networks. Int. J. Project Manag. **24**(3), 207–215 (2006)
10. Costantino, F., Gravio, G.D., Nonino, F.: Project selection in project portfolio management: an artificial neural network model based on critical success factors. Int. J. Proj. Manag. **33** (8), 1744–1754 (2015)
11. Dolan, E., More, J.: Benchmarking optimization software with performance profiles. Math. Program. **91**(2), 201–213 (2002)
12. Dunning, I., Huchette, J., Lubin, M.: Jump: a modeling language for mathematical optimization. SIAM Rev. **59**(2), 295–320 (2017)
13. Evans, J.R., Lindner, C.H.: Business analytics: the next frontier for decision sciences. Decis. Line **43**(2), 4–6 (2012)
14. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. The MIT Press, Cambridge (2016)
15. Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Giannotti, F., Pedreschi, D.: A survey of methods for explaining black box models. ACM Comput. Surv. (CSUR) **51**(5), 93 (2018)
16. Hill, J., Thomas, L.C., Allen, D.B.: Experts' estimates of task durations in software development projects. Int. J. Proj. Manag. **18**(1), 13–21 (2000)
17. Holmes, G., Donkin, A., Witten, I.H.: Weka: a machine learning workbench. In: Proceedings of the 1994 Second Australian and New Zealand Conference, Adelaide, pp. 357–361 (1994)
18. Hooker, J.N., Van Hoeve, W.J.: Constraint programming and operations research. Constraints **23**(2), 172–195 (2018)
19. Jaccard, P.: Étude comparative de la distribuition florale dans une portion des Alpes et des Jura. Bull. Soc. Vandoise Sci. Nat. **37**, 547–579 (1901)
20. Kanakaris, N., Karacapilidis, N., Lazanas, A.: On the advancement of project management through a flexible integration of machine learning and operations research tools. In: 8th International Conference on Operations Research and Enterprise Systems (ICORES), Prague, pp. 362–369 (2019)
21. Karacapilidis, N., Malefaki, S., Charissiadis, A.: A novel framework for augmenting the quality of explanations in recommender systems. Intell. Decis. Technol. J. **11**(2), 187–197 (2017)
22. Karacapilidis, N.: Mastering Data-Intensive Collaboration and Decision Making: Cutting-Edge Research and Practical Applications in the Dicode Project. Studies in Big Data Series, vol. 5. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-02612-1
23. Levitt, R.E., Kunz, J.C.: Using artificial intelligence techniques to support project management. Artif. Intell. Eng. Des. Anal. Manuf. **1**(1), 3–24 (1987)
24. Maurya, A., Telang, R.: Bayesian multi-view models for member-job matching and personalized skill recommendations. In: 2017 IEEE International Conference on Big Data, pp. 1193–1202. IEEE (2017)
25. Mooney, R.J., Roy, L.: Content-based book recommending using learning for text categorization. In: Proceedings of the fifth ACM conference on Digital libraries, pp. 195–204. ACM (2000)
26. O'Malley, O.: Terabyte sort on apache hadoop. Yahoo, pp. 1–3 (2008). http://sortbenchmark.org/Yahoo-Hadoop.pdf

27. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lerer, A.: Automatic differentiation in pytorch. In: 31st Conference on Neural Information Processing Systems, Long Beach, pp. 1–4 (2017)
28. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**(1), 2825–2830 (2011)
29. Pittaras, N., Giannakopoulos, G., Tsekouras, L., Varlamis, I.: Document clustering as a record linkage problem. In: Proceedings of the ACM Symposium on Document Engineering, p. 39. ACM (2018)
30. Radin, R.L.: Optimization in Operations Research. Prentice-Hall, New Jersey (1998)
31. Raschka, S.: Python Machine Learning. Packt Publishing Ltd., Birmingham (2015)
32. Ribeiro, M.T., Singh, S., Guestrin, C.: Why should I trust you? Explaining the predictions of any classifier. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 1135–1144. ACM (2016)
33. Rifkin, R., Klautau, A.: In defense of one-vs-all classification. J. Mach. Learn. Res. **5**(1), 101–141 (2004)
34. Rousu, J., Saunders, C., Szedmak, S., Shawe-Taylor, J.: Kernel-based learning of hierarchical multilabel classification models. J. Mach. Learn. Res. **7**(1), 1601–1626 (2006)
35. Rummelhart, D., Hinton, G., Williams, R.: Learning Internal Representations by Error Propagation. Parallel Distributed Processing. MIT Press, Cambridge (1986)
36. Shivaswamy, P.K., Bhattacharyya, C., Smola, A.J.: Second order cone programming approaches for handling missing and uncertain data. J. Mach. Learn. Res. **7**(1), 1283–1314 (2006)
37. Smith, K.A., Gupta, J.N.: Neural networks in business: techniques and applications for the operations researcher. Comput. Oper. Res. **27**(11), 1023–1044 (2000)
38. Spanakis, G., Siolas, G., Stafylopatis, A.: Exploiting Wikipedia knowledge for conceptual hierarchical clustering of documents. Comput. J. **55**(3), 299–312 (2012)
39. Svejvig, P., Andersen, P.: Rethinking project management: a structured literature review with a critical look at the brave new world. Int. J. Proj. Manag. **33**(2), 278–290 (2015)
40. Trupti, M.K., Prashant, R.M.: Review on determining number of cluster in K-means clustering. Int. J. Adv. Res. Comput. Sci. Manag. Stud. **1**(6), 90–95 (2013)
41. Tsekouras, L., Varlamis, I., Giannakopoulos, G.: Graph-based text similarity measure that employs named entity information. In: RANLP, pp. 765–771 (2017)
42. Vazirgiannis, M., Malliaros, F.D., Nikolentzos, G.: GraphRep: boosting text mining, NLP and information retrieval with graphs. In: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, pp. 2295–2296. ACM (2018)
43. Wang, Y.R., Yu, C.Y., Chan, H.H.: Predicting construction cost and schedule success using artificial neural networks ensemble and support vector machines classification models. Int. J. Proj. Manag. **30**(4), 470–478 (2012)
44. Zhang, G., Patuwo, B.E., Hu, M.Y.: Forecasting with artificial neural networks: the state of the art. Int. J. Forecast. **14**(1), 35–62 (1998)