An effective approach for software project effort and duration estimation with machine learning algorithms

Working Paper · January 2017

DOI: 10.13140/RG.2.2.15248.30724

CITATIONS
35

READS
4,065

3 authors:

Przemek Pospieszny
SGH Warsaw School of Economics
10 PUBLICATIONS **330** CITATIONS
SEE PROFILE

Beata Czarnacka-Chrobot
SGH Warsaw School of Economics
26 PUBLICATIONS **369** CITATIONS
SEE PROFILE

Andrzej Kobyliński
SGH Warsaw School of Economics
13 PUBLICATIONS **296** CITATIONS
SEE PROFILE

An effective approach for software project effort and duration estimation with machine learning algorithms

Przemysław Pospieszny , Beata Czarnacka-Chrobot , Andrzej Kobyliński

PII: S0164-1212(17)30294-7
DOI: 10.1016/j.jss.2017.11.066
Reference: JSS 10084

To appear in: *The Journal of Systems & Software*

Received date: 24 January 2017
Revised date: 26 November 2017
Accepted date: 27 November 2017

Please cite this article as: Przemysław Pospieszny , Beata Czarnacka-Chrobot , Andrzej Kobyliński , An effective approach for software project effort and duration estimation with machine learning algorithms, *The Journal of Systems & Software* (2017), doi: 10.1016/j.jss.2017.11.066

Highlights

- Effective approach for building and deploying effort and duration estimation.
- Three ensemble machine learning algorithms (SVM, MLP, GLM) and cross validation.
- Software projects sourced from ISBSG dataset and processed using smart data prep.
- The results indicate very good prediction accuracy and suitability for deployment.
- Provided a practical guidance for models' implementation and maintenance within organisations.

An effective approach for software project effort and duration estimation with machine learning algorithms

Przemysław Pospieszny*, Beata Czarnacka-Chrobot, and Andrzej Kobyliński

Institute of Information Systems and Digital Economy, Warsaw School of Economics, Warsaw, Poland

Abstract

During the last two decades, there has been substantial research performed in the field of software estimation using machine learning algorithms that aimed to tackle deficiencies of traditional and parametric estimation techniques, increase project success rates and align with modern development and project management approaches. Nevertheless, mostly due to inconclusive results and vague model building approaches, there are few or none deployments in practice.

The purpose of this article is to narrow the gap between up-to-date research results and implementations within organisations by proposing effective and practical machine learning deployment and maintenance approaches by utilization of research findings and industry best practices. This was achieved by applying ISBSG dataset, smart data preparation, an ensemble averaging of three machine learning algorithms (Support Vector Machines, Neural Networks and Generalized Linear Models) and cross validation. The obtained models for effort and duration estimation are intended to provide a decision support tool for organisations that develop or implement software systems.

Keywords: Software project estimation, machine learning, effort and duration estimation, ensemble models, ISBSG

1. Introduction

Software estimation is one of the most challenging areas of project management. For decades, project professionals have struggled with correct estimation of effort, cost and duration of initiatives that is required for development of schedules and budgets. The difficulty lies in forecasting those parameters at the initial stages of the project lifecycle, when boundaries of every initiative need to be established and when uncertainty regarding functionalities of the final product is substantial (Boehm, 1981). Oftentimes, limited knowledge about influencing factors and risks which may occur, pressure from client or management, and legacy software estimation techniques based on expert judgment may lead to imprecise and usually overoptimistic estimates. As a result, they may severely impact delivering project outcomes within a defined timeframe, budget and of acceptable quality

(Alami, 2016; Mieritz, 2012; Spalek, 2005; Tan, 2011). Despite continuous improvements in project and software development methodologies, recent studies of the Standish Group (2015) indicate that still only one third of initiatives are successful. For the remainder, either cost or time overrun occurs, or the product does not meet the customer's satisfaction, which can ultimately result in project termination. Wide propagation of an agile approach to software development slightly decreased the previously mentioned trend but the project failure rate is still substantial. Although the Standish Group is often criticised by researchers (Eveleens and Verhoef, 2009; Glass, 2006), mostly due to their restricting access to the survey data and limiting their research to US based companies, the problem with achieving project success definitely exists and is observed on a daily basis by project professionals (Carvalho et al., 2015; Lehtinen et al., 2014; Savolainen et al., 2012). The reason is mostly associated with a lack of soft and hard skills of project teams (Richman, 2011), deficiency of communication within stakeholders (Schwalbe, 2014), and foremost: poor project estimation and planning (Trendowicz et al., 2011). Precise forecasting of effort and duration required to implement software solution at the initial stage significantly increases the probability of successful project completion, as once budget and schedule are defined it may not be feasible to alter them due to costs exceeding benefits or it not being a competitive time to market. Hence this may lead to reduction in features or even project failure.

Regardless of whether the waterfall or agile approach is chosen to perform a project each initiative at the beginning of its lifecycle requires definition of cost and time frames to determine a business case and receive approval from a sponsor. For that purpose, software estimation techniques based on expert knowledge or by analogy to a similar initiative, like PERT, decomposition, wideband delphi and planning poker, are most widely used due to their simplicity and almost effortless application (Wysocki, 2014). Nevertheless, these methods are usually error prone, therefore for decades alternative approaches based on lines of code (SLOC) and later on function points (FP) have been developed. Both SLOC and FP models are constantly being updated to adapt to new trends in programming, software architecture and software development methodologies. However, in the fast-paced world of software development those techniques are struggling to keep up to date (Galorath and Evans, 2006), especially with progressing code reuse and customized or configurable deployments of software, which come almost out of the box. Moreover, they tend to be subjective, particularly those based on function points (Kemerer, 1993), and require substantial effort for their utilization and maintenance.

Therefore, in order to tackle the mentioned deficiencies, for the last two decades extensive research has been conducted for software estimation data mining techniques (Sehra et al., 2017), in particular: 'state of art' predictive machine learning (ML) algorithms (Wen et al., 2012). They are considered highly effective for tackling uncertainty and the obtained results present their powerful prediction capabilities for effort and duration estimation at the initial stages of the project lifecycle (Berlin et al., 2009; de Barcelos Tronto et al., 2008; Lopez-Martin et al., 2012). Additionally, through their automated prediction process based on historic information, they tend to reduce human biases and psychological or political influences. Nevertheless, few if any up to date implementations can be found in practice. The reason behind this may lie in narrowed research that focused on finding the most accurate data mining algorithm and tailoring it for best performance. This was frequently performed on small legacy and outdated data sets of completed projects that may lead to overfitting, often with inclusion of complex ensemble approaches (Kocaguneli et al., 2012; Pai et al., 2013). Additionally, for data preparation, which plays a substantial role in developing effective models, various, often contradictory methods were applied (García et al., 2016; J. Huang et al., 2015). Because of the limitations outlined above and additional ones, indicated further in this article in the literature review section, there are inconclusive results in relation to the

accuracy of individual algorithms, even if they were applied to the same dataset. This could be an outcome of different approaches utilized by researchers for data preparation and building data mining models for effort and duration estimation of software projects. Moreover, deployment aspects are often omitted, such as implementation and maintenance methodology and integration with existing processes and project management software to provide an effective estimation support tool.

The aim of this article is to tackle these limitations and narrow the gap between up to date research findings and potential deployment of robust machine learning algorithms in practice for effort and duration estimation at the initial project lifecycle of software initiatives. Therefore, a comprehensive approach is presented, beginning from data preparation to the models' implementation and maintenance, that ensures their usability as well as outstanding estimation accuracy and robustness for noise within data. For that purpose, a practical and effective approach for preparing data and building models is applied and presented based on the ISBSG dataset, which provides the most reliable source of a large volume of recent software projects from multiple industries (International Software Benchmarking Standards Group, 2013), and ensemble three machine learning predictive algorithms. Additionally, this research paper intends to provide guidance for practitioners in terms of models' deployment, integration and maintenance.

2. Theoretical background

2.1 Limitations of software estimation techniques

Regardless of the project and software methodology applied, every initiative requires definition of budget and a specific timeframe necessary to deliver a final outcome. These are obtained during the early stages of the project lifecycle through the process of estimation, which aims to provide an approximation of the amount of resources required to complete project activities and produce a product or service in accordance to specified functional and non-functional characteristics (Project Management Institute, 2013). According to Bohem's cone of uncertainty (Boehm, 1981), this is still valid. Decades of advances in project software environment (McConnell, 2009) show the highest risk involved in estimation and probability of error occurrence is at the initial project stages and decrease with its progression when product functionalities and influencing factors are more perceptible. Therefore, for project practitioners, initial estimation is the most challenging, especially considering that during that time they are obliged to define a budget and timeframe of initiatives which are perceived as constraints by project sponsors. Any further changes to them may imbalance the iron triangle (triple constraints), impact functionalities of a product, disrupt a business case and ultimately affect successful completion of a project.

In principle, software estimation relies on forecasting effort, from which cost and duration is derived. As a supporting measure, the size of a system to be delivered is often used foremost when applying estimation techniques based on the source line of code (SLOC) or function points, and according to various research (Hill, 2010; Wilkie et al., 2011) is has the most significant impact on accurate estimation of effort and duration. Other factors affecting estimation include soft and hard skills of the project team, applied methodology to deliver a product and technical software features such as the architecture or programming language (Galorath and Evans, 2006; Laird and Brennan, 2006). The obtained forecast of effort and duration required to perform a project is considered acceptable when it reflects the reality of a project and ensures successful completion by development and deployment of a final outcome – product or service (McConnell, 2009). More empirical approaches indicate that a precise

estimate should be within 10% of the threshold of real values (Laird and Brennan, 2006) or within 25% in 75% of cases (Conte et al., 1986).

According to Flyvbjerg (2006), there are three fundamental reasons for potential estimates' biases, which cause them to be overoptimistic and impact successful completion of a project: technical, psychological and political. The first refers to limitation of estimation methods and founding estimates on imperfect information. The other two focus on the human factor, in particular the inability to plan properly and factors influencing expert judgment such as pressure from management, sponsor or client. Both traditional and parametric software estimation techniques are constantly developed and tuned to address and limit the aforementioned biases. Nevertheless, with increasing complexity of software systems, demand for rapid time-to-market, distributed project environment and modern programming language (Trendowicz, 2014), the existing methods often do not conform to modern estimation requirements. They also depend excessively on expert knowledge, which with imperfect information and vast uncertainty leads to inaccurate estimates. Those based on size measured with SLOC or function points often require trained personnel, substantial estimation effort and time, and it is difficult to utilize them at an early project stage due to limited knowledge regarding product or service requirements (Galorath and Evans, 2006; Lavazza, 2015).

The table below provides an overview of main advantages and limitations of three widely applied traditional estimation methods and those based on source line of code and function points.

Table 1.
Examples of software estimation techniques and their limitations.

Estimation method	Limitations
Expert estimation (PERT, Delphi, Planning Poker)	• Relies on expert knowledge, experience and perception which may be biased (often lead to overoptimistic estimates) • Unable to quantify and justify all factors applied by experts in the process of estimation
Estimation by analogy	• Subjective choice of comparison criterion and process of difference identification (level of confidence) • Requires analogues project for comparison which is rarely achievable in software development
Decomposition and bottom-up (WBS-based)	• For medium and large projects, it may be time-consuming • High risk of overlooking systems related tasks such as integration, configuration and testing • Due to lack of information at early project stage may lead to underestimation
Parametric models (COCOMO II, SLIM, SEER-SEM)	• Does not take into consideration skill set of project team, specific to organization software and project management culture • May not be feasible for modern approaches in software development – code reuse, codeless programming and agile development • Dependency on programming language
Size-based estimation models (FPA, Use Case, Story Points)	• Requires trained personnel • For large projects their application requires great effort and cost • If applied at early project stage, due to limited information, may lead to inaccurate estimates

2.2 Project knowledge discovery

At every stage throughout the project lifecycle, various information is generated and stored within project repositories. Project Management Institute (2013)groups project knowledge into 10 areas, including cost, time, scope, quality and resource management, which represent sets of processes, techniques and activities required and which create or collect knowledge needed to perform a project. Additionally, two dimensions of knowledge associated with initiatives can be distinguished – micro and macro (Gasik, 2011). The first is related to knowledge required to perform an individual task or solve a given problem. Through the process of its acquisition, creation, application, transfer and sharing, it is transformed into macro-knowledge, which is the total knowledge required to conduct the project. Moreover, it represents an organisational ability to implement new systems and services, or enhance existing ones.

During the process of project knowledge management, the generated information is usually stored within project databases, maintained by the Project Management Office (PMO)with support of project management software. The information reflects not only data related to attributes of an initiative captured during its lifecycle, such as effort and time required to perform individual activities, the methodology applied and resources allocated, but also product characteristics such as features, architecture, size, programming language or defects identified during testing (Kobyliński and Pospieszny, 2015). These databases are used mainly by organisations for monitoring and reporting purposes. Additionally, the historic information related to project and product is applied for estimation purposes, especially for estimation by analogy and for adjusting parametric methods based on SLOC and FP.

Apart from these mentioned applications, project databases constitute a perfect candidate as an input for knowledge discovery process in databases (Fayyad et al., 1996) and can be mined for various project management and software development purposes. In general, data mining (DM) aims to extract meaningful patterns and rules from databases (Larose, 2007) by applying interdisciplinary techniques derived from statistics, machine learning and pattern recognition. Due to its efficiency for dealing with uncertainty, data mining is widely applied for disciplines with high probability and impact occurrence, such as credit scoring or clinical trials. In terms of software project management, in practice it is rarely implemented. Nevertheless, research done in the last two decades indicate data mining's effectiveness in the areas of effort and duration initial estimation (Balsera et al., 2012; de Barcelos Tronto et al., 2008; López-Martín, 2015), during project monitoring as an alternative approach for earned value analysis (Azzeh et al., 2010; Iranmanesh and Mokhtari, 2008), for quality and risk management (Moeyersoms et al., 2015; Nagwani and Bhansali, 2010) and software maintenance cost (Shukla et al., 2012).

2.3 Machine learning algorithms

As a broad discipline, data mining utilizes state of art machine learning algorithms for extracting useful knowledge from databases through the process of automated learning based on input data (Ben-David and Shalev-Shwartz, 2014). The main advantage over statistical and mathematical DM algorithms is additivity of ML algorithms to a changing environment. This is an especially important factor for software estimation, where there are frequent technology advances, new tools and programing languages available, and where methodologies improvements and changing skillsets of project teams may bias existing efforts and duration of prediction approaches. This can be achieved by their ability to learn and improve based on generated predictions through determining estimation that closest matches the observed data and any prior knowledge held by the learner (Mitchell, 1997). Hence ML algorithms are

suitable for modelling complex problems which can hardly be programmed and up to a certain point mimic the human learning process.

Mainly, the ML learning process can take from an unsupervised or supervised dependable the chosen descriptive or predictive task. The first relies on fining patterns that are not tied to any variable in the training dataset (Linoff and Berry, 2011) for such purposes as clustering, association rules or anomaly detection. Supervised learning must have explicitly defined inputs corresponding to known outputs by a determined target variable (Cios et al., 2007) which is used for classification or prediction. For both there are numerous algorithms available that can be applied depending on the desired outcome. This article is focused on predicting the continuous effort and duration target variable, and for that purpose three effective ML predictive algorithms were used: Support Vector Machines (SVM), Multi-Layer Perceptron Artificial Neural Network (MLP-ANN) and Generalized Linear Models (GLM).

SVM has an ability to model complex linear and nonlinear problems and generate highly accurate results, even on noisy data, due to use of kernels and the absence of local minima (Han et al., 2006). Nevertheless, their training process may be time-consuming on a large volume of data which in terms of project databases, usually contain up to thousands of historic initiatives which are insignificant. In contrast MLP is a parametric algorithm and converge on identifying local minima rather than a global one. Because it consists of hidden layers and bias parameters the network is very robust in relation to noisy data (Larose and Larose, 2015). Although it is prone to overfitting in case of an extended training process, the last effective algorithm chosen for this article was GLM. It is a generalization of ordinary linear regression, that depending on the response variable type, distribution and variance and appropriate link function, is used to represent the relationship between attributes (Clarke et al., 2009). Due to its mentioned flexibility, it is efficient in handling non-linear variables and provides a wide array of inference tools.

3. Related work

For the last two decades, extensive research has been performed in terms of applying data mining statistical and machine learning algorithms for software estimation. Emphasis was put on initial effort and duration estimation since accurate forecast of these during the early project stage is the most challenging due to uncertainty and limited knowledge. Any significant deviation of those constraints during the project lifecycle may severely impact features of a product or service, its quality and ultimately the project's successful completion.

Wen et al. (2012) performed the most comprehensive review of machine learning algorithms used for effort estimation. For that purpose, 84 studies were investigated from the last two decades. According to the results obtained, the researchers focused mainly on tailoring individual algorithms for best performance, particularly; artificial neural networks (ANN), case-based reasoning (CBR) models and decision trees. The accuracy of machine learning models were of an acceptable level and better than statistical ones, with mean magnitude relative error MMRE ranging 35-55%, percentage relative error deviation PRED(25) 45-75% and median magnitude relative error MdMRE 30-55%. The researchers also indicated that depending on the dataset applied for building models and the data preprocessing approach taken, machine learning algorithms may generate disparate results, due to outliers, missing values and possibility of overfitting.

The discrepancy in utilizing different approaches for building ML is even more perceptible when analysing individual publications. For instance, de Barcelos Tronto et al. (2008) compared accuracy of artificial neural networks with multiple regression models for effort estimation using COCOMO dataset (Reifer et al., 1998), demonstrating the superiority

of the first one. For evaluation purposes, MMRE and PRED was used. López-Martín and Abran (2015) also focused his research only on effort estimation, investigating prediction precision of different neural network types. Nevertheless, for that purpose an ISBSG (International Software Benchmarking Standards Group, 2013) dataset was applied, normalisation of dependent variable, cross-validation approaches and MAR with MdAR were used as an accuracy criterion. Berlin et al. (2009) took a more comprehensive approach in terms of scope and examined accuracy of ANN and linear regression (LR) not only for effort but also for duration prediction. The models were based on two datasets: ISBSG and Israeli companies' dataset. Based on results it can be noted that ANN slightly outperformed LR and log transformation of output variables improved the accuracy. Additionally, it can be concluded from this research that duration estimation tends to be less precise than effort due to a stronger correlation between size and effort.

In spite of different approaches taken for building ML, meaningful recommendations that support their implementation in practice for effort and duration estimation at early project stages can be extracted. Due to sensitivity of ML for noises within the data sets, models should not rely on individual algorithms but used in conjunction, which furthermore boosts prediction accuracy (Minku and Yao, 2013). Researchers proposed various ensemble methods, such as boosting, bagging and complex random sampling techniques (Kocaguneli et al., 2012), mostly for the same type of ML algorithms. Nevertheless, ensemble methods may introduce substantial performance overhead (Azhar et al., 2013) if applied too excessively. Therefore, for building ML effort and duration models a set of different but limited number of algorithms and simple ensemble approach should be used such as averaging of obtained estimates (Ho, 1998; Xu et al., 1992).

Throughout the processes of training ML models an emphasis should be put on data preprocessing, especially in dealing with outliers, the missing values that have a large impact on the accuracy of ML algorithms. According to Huang et al. (2015), apart from different techniques available, such as deletion (listwise, pairwise), imputation (mean, regression), their use depends on the dataset. However, where possible, missing values should be discarded in order to remove biases that may alter ML prediction accuracy, as inputting them may reduce variability of data (Strike et al., 2001). In relation to outliers the common rule of three standard deviations from a mean is used for their removal (Ruan et al., 2005). Additionally, although ML algorithms do not require normalisation of dependent variables, results of studies (Berlin et al., 2009; Keung et al., 2013; López-Martín and Abran, 2015) indicate that log transformation of effort and duration tend to generate more accurate estimates.

For evaluation of effort and duration ML prediction models, the majority of researchers utilize MMRE and PRED (Wen et al., 2012). The measures are subject to various critical studies, especially MMRE, that is considered to be an asymmetric measure (Foss et al., 2003; Myrtveit and Stensrud, 2012) and sensitive to outliers (Port and Korte, 2008). Nevertheless, MMRE and PRED constitute an evaluation standard, enable comparison of results and often with support of mean magnitude relative error to estimate (MMER), mean of balanced relative error (MBRE) and mean absolute residual error (MAR) are still widely used by researchers. Moreover, for validation purposes k-fold cross validation procedure is performed in order to avoid overfitting of ML models (Idri et al., 2016).

To conclude, it needs to be mentioned that apart from a large amount of research performed in the last two decades, of which only a fraction was presented above, few implementations of ML predictive algorithms for effort and duration estimation within organisations can be found. The research emphasis was put on tailoring and comparing individual algorithms' performance, with attention to deployment practices and framework which were unsatisfactory. Moreover, the inconclusive results, in spite of the mentioned reasons, were largely caused by applying small, outdated datasets (COCOMO, NASA) for

ML algorithms, which often led to overfitting. Additionally, estimation models should conform to current software methodologies and modern development approaches, as using legacy projects that aim to develop software using inefficient methods, and basic programming languages may be considered as unsuitable.

Taking the above into consideration, this paper aims to address the mentioned limitations, recommendations and best practices proposed by researchers through outlining a practical and effective approach for building and deploying ML models for effort and duration estimation to enable their implementation in practice.

4. Proposed approach

In this section, an effective approach based on best practices and up to date practical research findings for building effort and duration models by use of machine learning algorithms is presented. For this purpose, an ISBSG dataset is utilized that is processed through smart data preparation. Furthermore, the obtained input dataset is used for modelling with application of three ML algorithms (SVM, MLP, GLM), ensemble averaging and cross validation. The building process follows cross industry standard process for data mining (CRISP-DM) methodology.

4.1 Data preparation

Noisy and unreliable data may severely influence the predictive accuracy of machine learning models. Poor quality of data, especially the significant occurrence of missing values and outliners may lead to inconsistent and unreliable results. Therefore, data preparation is a critical task in the process of building ML models, in which data is preprocessed through selection, cleaning, reduction, transformation and feature selection (García et al., 2016; J. . Huang et al., 2015). Although there are numerous approaches and techniques for cleaning data and preparing the dataset, for this article, based on preliminary modelling and literature review, only those were applied that were efficient, that do not bias the estimates, and contribute for building accurate ML models for effort and duration prediction.

The most comprehensive source of a historic software project incorporates the ISBSG dataset (International Software Benchmarking Standards Group, 2013), which was applied for the purpose of this article. Version R12 which was used for this research consists of 6006 initiatives conducted during the last two decades by numerous companies from 25 countries. The database is used mostly by organisations and researchers for building and tailoring function points' estimation models, hence project size is calculated by ranges of that measure. Other available datasets of software projects, such as PROMISE Software Engineering Repository, COCOMO and SourceForge are either outdated by a limited number of initiatives or unreliable. Nevertheless, the drawback of the ISBSG dataset is its heterogeneity due to sourcing from a variety of organisations and projects. Another aspect is the large volume of missing values that in conjunction with heterogeneity may provide a challenge for data preparation and building ML models (Fernández-Diego and González-Ladrón-De-Guevara, 2014). However, from an application point of view it may reflect the real case scenario. It needs to be emphasized that in the last decade ISBSG dataset is widely used by researchers for software estimation with data mining algorithms (González-Ladrón-de-Guevara et al., 2016). Despite that, due to utilization of different versions of ISBSG dataset and various objectives there is no standard approach for its processing (Berlin et al., 2009; Hill, 2010; Jeffery et al., 2001; López-Martín and Abran, 2015; Mittas et al., 2015) agreed upon among researchers.

For this study, only projects from the last decade were used, in order to reflect more recent advancements in software development. In the process of reviewing and selecting data for modelling, firstly dependent variables were chosen. For effort, it was decided to use *Normalised Work Effort* which presents the total effort required to perform an initiative. In terms of duration, the real elapsed time was obtained by subtracting two variables: *Project Elapsed Time* and *Project Inactive Time* (International Software Benchmarking Standards Group, 2013). Next, records which were classified by ISBSG as unreliable, with low quality were removed (cat. C and D from *Data Quality Rating*). From 125 variables, grouped into 15 categories, only a subset of them was chosen that may influence prediction of effort and duration of software projects at the early stage of lifecycle. As was mentioned in the previous paragraph ISBSG contain a large amount of missing values. Due to a significant number of observations available in the dataset and primarily to not yield biased estimates of coefficients, it was decided to apply listwise deletion.

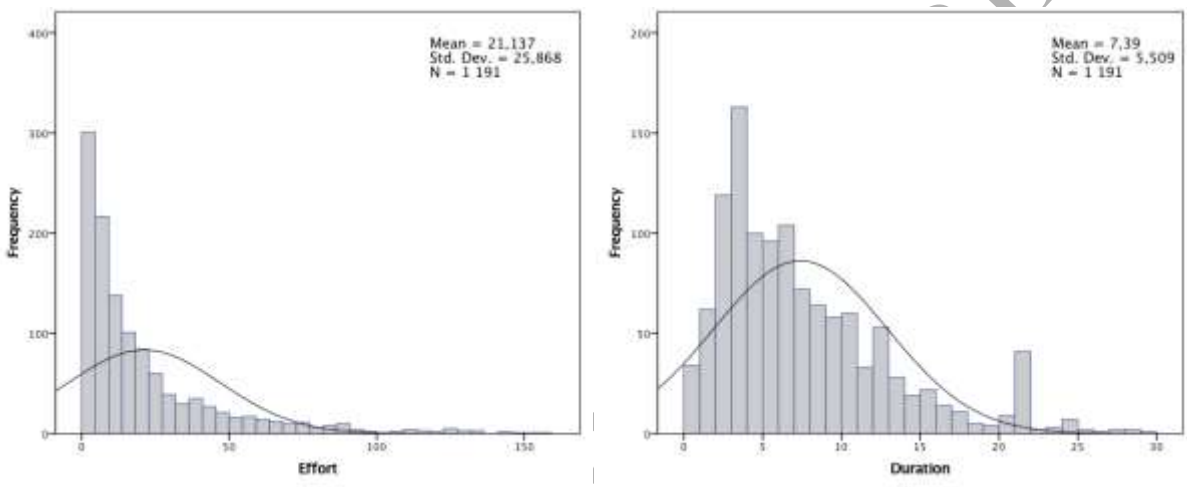

Fig. 1. Distribution of Effort (man-months) and Duration (months).

Qualitative variables categories were aggregated and balanced to decrease the number of classes and furthermore, those with less than five observations were removed. Moreover, they were binary coded that tend to boost ML prediction capability and performance (Singh and Misra, 2012) were binary coded. In relation to numerical variables, the outliners were cleaned using three standard deviations from mean criterion. In the ISBSG dataset the duration of initiatives is measured in man-months (MM) and effort man-hours (MH). For standardisation and interpretability reasons the effort was transformed to MM using Boehm's approach applied for the COCOMO model (Boehm et al., 2000) where 1 person-month corresponds to 152 person-hours (38-hour working week).

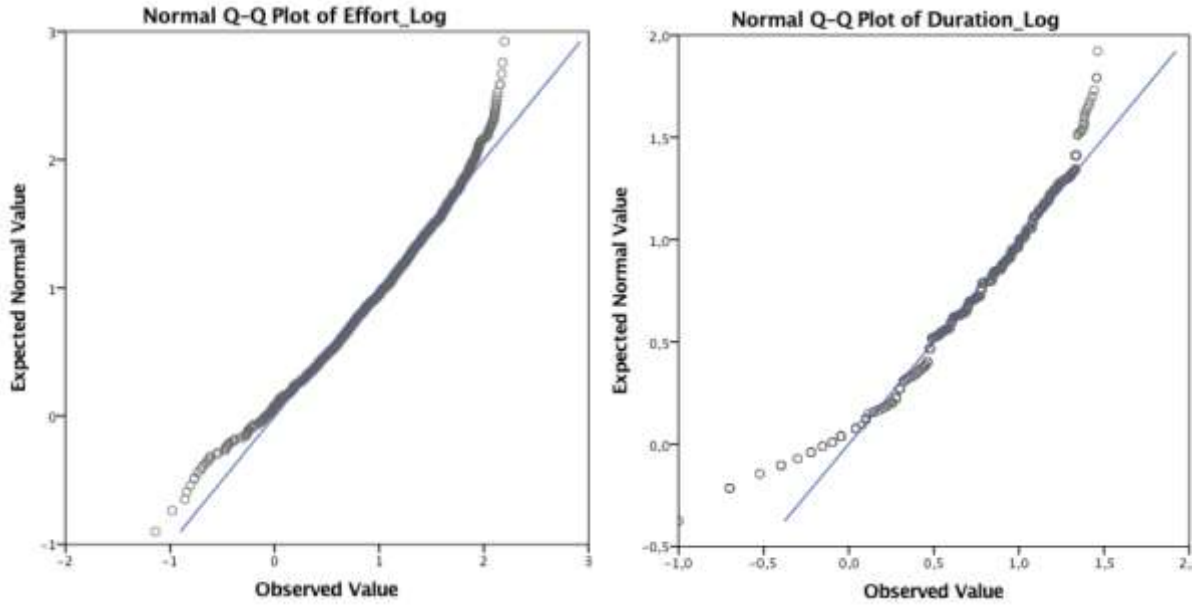

Fig. 2. Normal Q-Q plot for log-transformed dependent variables.

Both output variables had positive skewness (Fig. 1) due to a majority of small to medium projects included in the dataset. Additionally, in order to confirm abnormal distribution of effort and duration, continuous variables Kolmogorov–Smirnov and Shapiro-Wilk tests were conducted, which both rejected null hypothesis of depended variables' normal distribution. Despite ML algorithms being able to handle distributions other than normal, they generate more accurate predictions if output variables are symmetrically concentrated around the mean (de Barcelos Tronto et al., 2008; Keung et al., 2013). Therefore, effort and duration were log transformed (Fig. 2) and again tests K-S and S-W were performed and indicated near-normal distribution which is acceptable considering the large dataset.

Table 2.
Selected variables for effort and duration estimation.

Variable	Description	Type	Categories	Role
Industry Sector	Organisation type	Nominal	14	Input
Application Type	Type of application being addressed	Nominal	16	Input
Development Type	New development, enhancement or re-development	Nominal	3	Input
Development Platform	PC, Mid Range, Main Frame or Multiplatform	Nominal	4	Input
Language Type	Programming language (2GL, 3GL, 4GL)	Nominal	3	Input
Package customization	Indicates whether the project was a package customisation	Nominal	3	Input
Relative Size	Function points grouped into categories	Nominal	7	Input
Architecture	System architecture	Nominal	6	Input
Agile	Agile used?	Flag	2	Input
Used Methodology	Development methodology used?	Nominal	3	Input
Resource Level	Development team effort, development team support, computer operations involvement, end users or clients	Nominal	4	Input
Effort	Total project effort in work months, log-transformed	Continuous	−	Target
Duration	Total project elapsed time in months, log-	Continuous	−	Target

transformed

The obtained dataset (Table 2) presents the process of data selection, cleaning and transformation and consisted of 1192 projects, 11 independent and 2 target variables. Next, Pearson correlation was applied on data for exploring relationships between variables and their influence on effort and duration. The obtained coefficients were of a low level mostly due to the large dataset. The dependency between target variables was relatively strong (0.47). In respect to independent variables for effort, the most significant relationship was software size (0.672), other variables were substantially less but still significantly influenced the mentioned target variable. In relation to duration, the correlation coefficients were more equally distributed, with advantage of software size (0.256), used methodology (0.217) and required package customization (0.215).

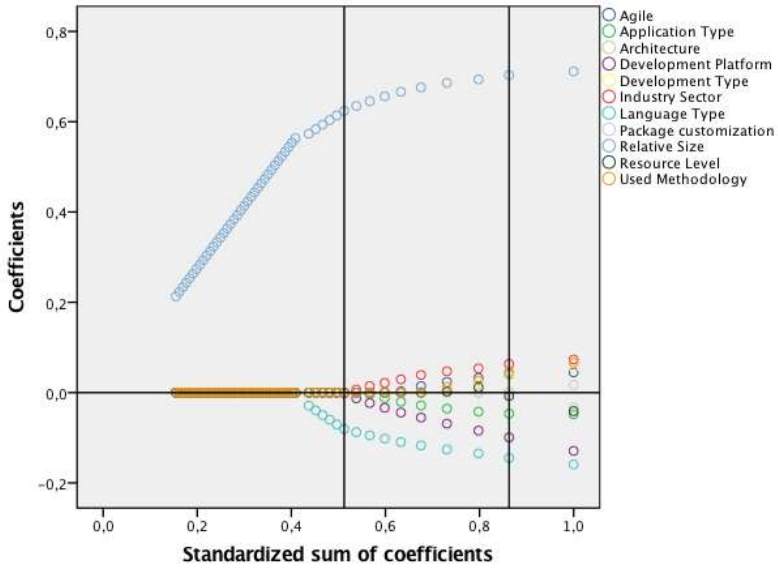

Fig. 3. Lasso paths for Effort.

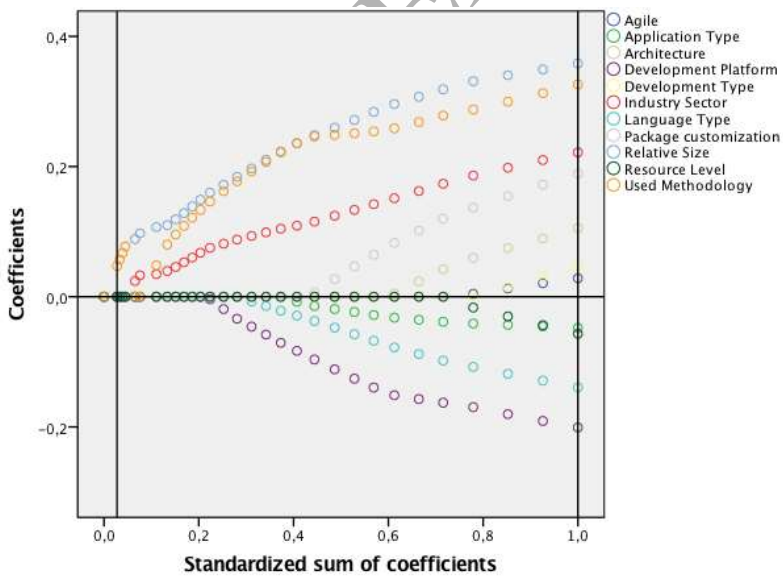

Fig. 4. Lasso paths for Duration.

Machine learning algorithms' performance may suffer from including insignificant variables. Therefore, as an addition to Pearson correlation, least absolute shrinkage and

selection operator (LASSO) and stepwise regression was preformed to exclude those with limited influence on target variables. The results indicated that all of them have capability to influence prediction of effort and duration. Therefore, the complete set of 11 independent variables, selected in the process of data preparation were further used for building models. Moreover, each of those applied in the next section algorithms for predicting effort and duration has their own approach for feature selection, hence in case of inconclusive results it could be more reasonable not to exclude any variable that could potentially improve forecast quality.

4.2 Ensemble models for effort and duration estimation

Figure 5 presents the approach taken for building models. As a result of preliminary modelling and the literature review conducted, the three best performing modern machine learning algorithms were chosen for building effort and duration estimation: Support Vector Machines (SVM), Multi-Layer Perceptron Artificial Neural Network (MLP-ANN) and Generalized Linear Models (GLM). Depending on input data, each of them may generate distinct results, hence in order to prevent overfitting it was decided to apply three different predictive supervised algorithms.

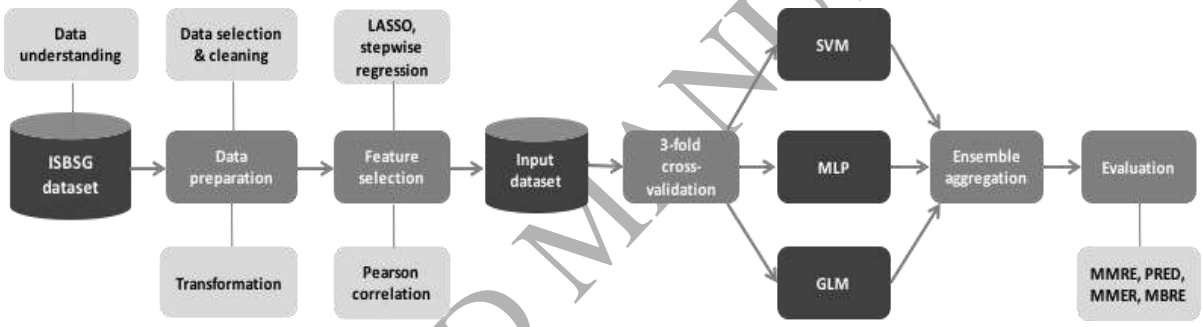

Fig. 5. Process of building effort and duration models.

For predicting effort and duration two separate models were built, each with utilization of three distinct algorithms: SVM, MLP and GLM. To maintain simplicity, the configuration of algorithms was the same for effort and duration models and are presented as follows:

a) Support Vector Machines – kernel function: radial basis (RBF), stopping criteria: 0.001, regression precision: 0.1;
b) Multi-Layer Perceptron Artificial Neural Network - one hidden layer, activation function: hyperbolic tangent;
c) General Linear Models - normal distribution, link function: identity, scale parameter method: Pearson Chi-square.

The obtained predictions were merged using ensemble averaging to reduce bias and variance error. This approach prevents both over and underfitting of the models which could be caused by noises and outliners in the training dataset. Furthermore, the resulting ensemble tends to generate more accurate predictions over a single algorithm (Krogh, Anders Jesper, 1995; Minku and Yao, 2013; Pai et al., 2013), especially on small heterogeneous datasets. Hence, the achieved models are more stable and generate more accurate results. It should be added that there are numerous effective ensemble approaches such as boosting and bagging (Breiman, 1996; Freund and Schapire, 1996). Nevertheless, due to simplicity and performance reasons an averaging approach tends to be more suitable for effort and duration estimation.

Additionally, 3-fold cross validation (Krogh, Anders Jesper, 1995), was applied to verify accuracy of the models obtained. This procedure ensures that models are not overfitted, increases prediction robustness and allows for their proper assessment. The dataset was first randomly divided into three folds, then into a set of three algorithms (SVM, MLP and GLM) trained using k-1 folds, and tested on left-out fold. For both dependent variables, the process was repeated 3 times until each fold was used for testing. As a result, each observation is used for both training and validation, with an equal weight of k-1 during training and each observation is used once for validation (Barrow and Crone, 2016). Despite that, 10-fold cross-validation is the most common approach. For the purpose of this article it was sufficient to apply a 3-fold one because additional folds did not add any value.

4.3 Evaluation criteria

Apart from the standard error measures such as ME, MAE, MSE and RMSE, for evaluation of obtained models, two criteria were applied which are widely used in software estimation: mean magnitude relative error (MMRE) and percentage relative error deviation (PRED). They are scale and unit independent and foremost allow comparison of results between different prediction models and datasets. Both are based on magnitude relative error (MRE) which measures the difference between actuals and estimates and is defined as follows:

$$MRE = \frac{|y_t - y_t^p|}{y_t} \tag{1}$$

$$MMRE = \frac{1}{n}\sum_{t=1}^{n} MRE \tag{2}$$

$$PRED(x) = \frac{1}{n}\sum_{t=1}^{n} \begin{cases} 1 & if\ MRE \le x \\ 0 & otherwise \end{cases} \tag{3}$$

where
y_t - actual value
y_t^p - estimated value
x - percentage, typically 25% or 30%
n – number of samples

According to Conte et al. (1986) a software estimation model is considered accurate when MMRE ≤ .25 and PRED (.25) ≥ .75. In reality, both levels are rarely achieved and for PRED a 30% MRE threshold – PRED (.3) is more often applied (Jorgensen and Shepperd, 2007). Despite their wide application there is an ongoing debate concerning the usefulness of these measures (Dave and Dutta, 2011; Foss et al., 2003), especially concerning their outliers' sensitivity (Port and Korte, 2008) and the fact that they are 'estimators' of a function of the parameters related to the distribution of the MRE values. Therefore, additional measures were used for evaluation of effort and duration models, which have been most often used in recent years in software estimation studies: mean magnitude relative error to estimate (MMER) and mean of balanced relative error (MBRE). Especially MBRE is a practical evaluation criterion, because as a balanced symmetric error metric it penalizes underestimation and overestimation on the same level and better handles outliners (Huang et al., 2008).

$$MER = \frac{|y_t - y_t^p|}{y_t^p} \tag{4}$$

$$MMER = \frac{1}{n}\sum_{t=1}^{n} MER \quad (5)$$

$$BRE = \frac{|\, y_t - y_t^p \,|}{\min(y_t^p - y_t)} \quad (6)$$

$$MBRE = \frac{1}{n}\sum_{t=1}^{n} BRE \quad (7)$$

5. Results and discussion

This section evaluates individual and ensemble models obtained for effort and duration estimation, referencing other publications that utilized ISBSG and ML algorithms for software estimation for results comparison. Additionally, it provides guidelines for their deployment and maintenance in practice.

5.1 Performance of models

For the outcome of approach taken for building models presented in section 4.2, two ensemble models were created separately for predicting effort and duration. Each of them utilizes three machine learning algorithms: SVM, ML and GLM. The most informative performance and error measures for individual and ensemble models are presented in Table 3.

Table 3.
Evaluation measures for effort and duration models.

	SVM	MLP	GLM	Ensemble	SVM	MLP	GLM	Ensemble
	Effort				***Duration***			
ME	0.02	0.02	0.01	0.02	0.01	0.01	0.01	0.01
MAE	0.19	0.26	0.27	0.23	0.14	0.19	0,19	0.17
MSE	0.07	0.12	0.13	0.1	0.04	0.06	0.07	0.05
RMSE	0.27	0.34	0.35	0.31	0.2	0.25	0.26	0.23
MMRE	0.13	0.21	0.18	0.17	0.15	0.24	0.26	0.21
PRED(0.25)	76.91%	64.65%	61.96%	69.44%	75.65%	64.82%	62.55%	69.02%
PRED(0.3)	81.19%	71.37%	67.93%	74.73%	81.61%	72.96%	71.12%	76.49%
MMER	0.47	0.45	0.57	0.42	0.26	0.31	0.34	0.29
MBRE	0.16	0.22	0.23	0.2	0.17	0.15	0.25	0.22

For both effort and duration, all models generated substantially precise estimates. The best performing individual algorithm was SVM, for MLP and GLM, error measures were almost on the same level. Ensemble models were slightly less accurate than SVM, nevertheless, considering their robustness for handling outliners, noises within the new input data and their ability to prevent model's overfitting, applying the combined approach is a more stable and practical procedure. Effort ensemble model characterized MMRE below 0.2 and PRED(0.25) at a 70% level. Considering that PRED(0.3) is used mostly for interpretation because PRED(0.25)>75% is hardly achievable and the model meets Conte criterion as a good estimation model. The difference between MAE and RMSE may indicate that a small portion of large errors could occur which can also be concluded from Fig. 6 where MRE and PRED

distribution is presented. The small value of MSE may suggest that there was a limited number of outliners within the real values of effort due to the data preparation performed.

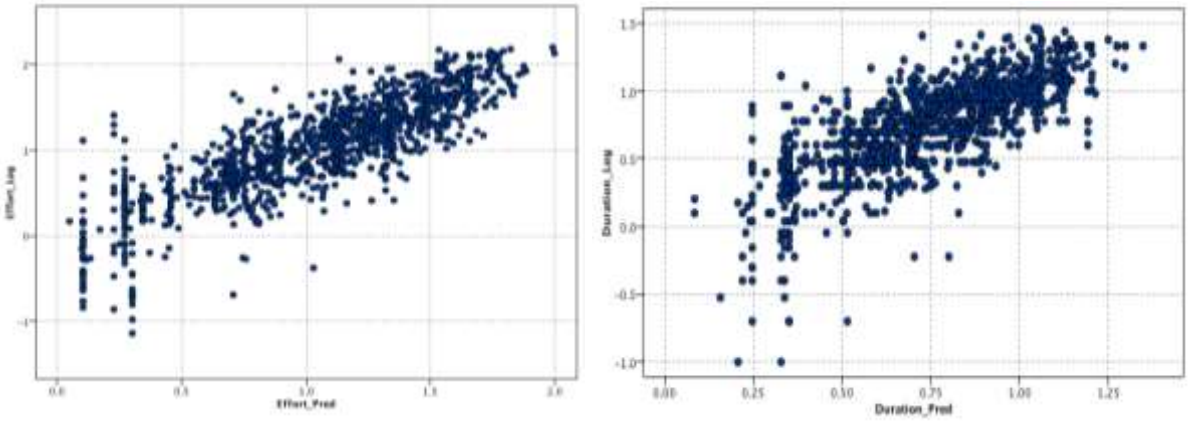

Fig. 6. Scatter plot of projects for effort and duration ensemble models.

As studies based on ISBSG dataset present (Berlin et al., 2009; Lopez-Martín and Abran, 2015), the duration models should be less accurate than those built for predicting effort because of less correlation with software size, which has the highest influence on predicting effort. Therefore unexpectedly, for duration estimation of both individual and ensemble models performed almost comparably to effort related ones, with MMRE=0.21 and PREDs at 70% and 76% for average approach. RMSE, MAE and distribution of predictions connote that there was a limited existence of large errors. Additionally, a very small SME indicates lack of outliners within actuals of duration. Similarly, for effort, MBRE for duration was close to MMRE values which resulted from an absence of skewness for predictions and almost equal occurrence of over and underestimations.

When comparing performance of individual algorithms, a significant advantage of SVM over MLP and GLM can be noticed for both effort and duration, with MMRE below 0.15 and PREDs reaching 81%. Other algorithms generated accurate estimations as well, but were characterized by a marginally higher error measures. The reason behind this may lay in the outstanding ability of SVM to handle complex dependencies within the heterogeneous data. Although in practice, if applied to a new dataset specific to a chosen company, other algorithms could be superior and estimate effort and duration more precisely. Therefore, the ensemble approach is more feasible and provides a stable prediction approach.

From the results, it can be concluded that the ensemble models obtained for effort and duration estimation generate remarkably accurate predictions due to: the applied smart data preprocessing, three effective ML algorithms and cross-validation method for preventing overfitting occurrence. Moreover, models were built on a heterogeneous ISBSG dataset, which characterizes a high variety and large amount of noises and missing values within it. However, both effort and duration were log transformed for normalization purposes, therefore it should be noted that if estimates are converted back to real estimates the accuracy of models would moderately decrease. Nevertheless, they are aimed to be deployed in practice on a homogeneous dataset, hence the drop of precision should be marginally impacted.

5.2 Comparison of results

Referring the results to other studies which addressed ML algorithms for effort and duration estimation, there is a noticeable advantage from the approach presented, especially for predicting timeframe of software projects. From Wen et al.'s (2012) comprehensive

comparison of 143 effort related publications, similarly to the results obtained, the superiority of SVM over any other algorithm can be concluded, with an average MMRE=0.34 and PRED(.25)= 72%. Other ML algorithms' accuracy such as ANN was lower (MMRE=0.37, PRED(.25)= 64%). However, most of those models were built on homogeneous small legacy datasets such as COCOMO, Desharnais or Albrecht which may lead to under or overfitting.

Berlin et al. (2009) followed a similar ensemble approach (ANN and LR), based on the ISBSG dataset with log transformation for both effort and duration estimation. According to error measures for effort, their results were comparable to those achieved in this study: MMRE=0.22, PRED(.25)=81% and MBRE=0.29. However, model for duration prediction characterized with poor accuracy: MMRE=0.88, PRED(.25)=27% and MBRE=1. For timeframe estimation, López-Martín and Abran (2015) achieved better results. In their recent study, they compared different ANN algorithms with application of the ISBSG dataset, log transformation of dependent variable and cross-validation. The accuracy of the MLP model was vastly superior to any other ANN approach: MMRE=0.18 and PRED(.25)=80%, even better than the MLP obtained in this study, but worse than the best preforming algorithm (SVM).

Furthermore, the obtained results should be compared to ensemble models for effort and duration developed by Pospieszny et al. (2015) with application of ISBSG dataset and GLM, MLP and CHAID decision threes algorithms. MMREs (0.19 and 0.21 respectively) are on a similar level to those presented in Table 3. However, a difference in PREDs (.25): 62% and 66% indicate that the approach taken in this study for data preprocessing, including SVM in the ensemble model, and using cross-validation for results' evaluation, ensured that the error dispersion within predictions is lesser, therefore enabling the achievement of more accurate estimates. It should be noted, that as presented above, even if the similar database and algorithms are applied, the prediction accuracy can differ, depending mostly on quality of data preparation and methods utilized for that purpose.

5.3 Implementation guidelines

For accurate estimation of effort and duration at the initial project stage the proposed ensemble models are required to be specific to the organization of the database of completed projects in which they will be applied. The process of learning the algorithms' prediction capability is adapted to the organizational culture, project and software development methodology in place, soft and technical skills of project teams, communication processes and other factors which are distinct to each organization and severely influence the budget and timeframe of initiatives. The quality of the database is an important factor and except the approach proposed in this article for data preparation the homogeneous input data should be processed through data cleansing to validate completed projects and ensure records' completeness. This can be sourced from Enterprise Project Management software (EPM) which are in place in most organizations and maintained by PMOs for project portfolios monitoring. The size of database can vary depending on the maturity of the organization. Nevertheless, for accurate estimation with machine learning algorithms it should contain dozens or preferably hundreds of diverse projects within it. Nevertheless, due to the proposed ensemble approach of algorithms a database of even 40-60 completed projects should be sufficient.

In relation to input variables, they may deviate from those used for the purpose of this article due to their availability in the organizational project database, EPM software functionalities and project portfolio processes in place. Hence their selection, cleaning and transforming should be handled during deployment of the models within the organization.

Fig. 7. Effort and duration deployment and maintenance process.

The implementation process should follow the widely applied CRISP-DM methodology (Pete et al., 2000), although it should go beyond deployment and the process should be extended to additional steps that involve integration, monitoring and optimization (see Fig. 7) of ensemble models for effort and duration estimation. The steps mentioned should ensure models' incorporation within existing organizational processes and adoption to changes that frequently occur within a project management environment which result from technological innovations, soft and hard skills of employees and altered business processes. For building models in this article IBM SPSS Modeler was used, although for that purpose open-source software or programming language such as R, Python or WEKA can also be applied. These can be deployed within the organization by integrating them with EPM software through implementing necessary functional changes that would allow project practitioners to input features of new initiatives to obtain effort and duration estimation. Another approach is to build the GUI on top of models and use the application as standalone software. One other important aspect is monitoring their performance during business as usual activities. At least once per month models' estimation accuracy should be evaluated and algorithms retrained utilizing new completed initiatives in order to retain prediction capability and adapt them to changing factors that may take place within an organization and impact deployed models. The above activities should be managed by PMO and require relevant adjustments within existing project management processes.

The approach proposed in this article for building and deploying models for effort and duration estimation at the early stages of project lifecycle should provide a framework that requires adaptation for organizational needs, EPM software in place and availability of high quality data. The purpose of the models is to serve as a decision support tool for project practitioners and may be used in conjunction with traditional or parametric software estimation techniques. The latter, can be used especially for assessing the size of a project which has the most significant impact especially on estimating effort of initiatives. The most feasible approach would be to use function points for software sizing which tend to provide the most accurate approximation, and machine learning algorithms for effort and duration estimation (Pospieszny et al., 2015a). Nevertheless, alternatively, for sizing purposes other widely applied organization techniques such as estimation by analogy or expert judgment can be used.

Additionally, as mentioned earlier, implementation of models in practice require a change of processes and procedures in place. Moreover, this entails organizational cultural change, therefore there is a necessity to engage end-users early on during the deployment process to effectively communicate benefits to promote engagement and provide necessary training for project practitioners. Considering the above, those most suitable for deployment, but not limited to, are organizations which have their project and software processes assessed at least at level 3 CMMI.

Implementation of models, their daily use and maintenance, is beyond doubt associated with cost that decreases with the scale of the company. An additional important factor is availability of completed projects, which can be used for training purposes. The ensemble approach prevents the possibility of models' overfitting; however, the sample should be of a reasonable level (at least 40-60) in order to retain accuracy. Another aspect for consideration is typical project size and its duration. If most projects within an organization are small, and their duration is short, it may not be feasible to apply the models since traditional estimation methods could deliver comparable accuracy. Hence, prior to any implementation, a comprehensive business case should be prepared and cost benefit analysis preformed. In conclusion, the proposed approach for effort and duration estimation is most suitable for medium and large, mature organizations, with a significant volume of projects, and where inaccurate effort and duration estimation may lead to negative implications such as project overrun or termination.

6. Threats to validity

Regarding the internal validity of the study, there are several aspects that should be discussed. Firstly, as mentioned in the data preparation, the most influential variable in relation to effort and duration, is software size, which in the ISBSG dataset was calculated using functional size measurement methods, mostly IFPUG and reflected by function points. However, impact of other various sizing methods, including traditional ones, based on expert knowledge and on effort and duration estimation using machine learning algorithms should be researched. Nevertheless, this experiment may be difficult to set up since there is limited availability of reliable and good quality project data.

Another important factor that needs to be highlighted is, performance of individual algorithms in comparison to the ensemble approach. In this study, SVM accuracy is superior to ensemble models that consist of three algorithms, both for effort and duration. This could be a result of SVM's ability to handle complex and diverse data, but also could be caused by algorithm's overfitting, and if applied to new data, the performance of individual ones could be dissimilar. Hence, the ensemble approach is applied, which averages predictions of three robust machine learning algorithms, stabilize the model, reduce influence of noise within data and the impact of algorithms' abnormal behavior.

In section 5.2, the results achieved in this study referred to other articles that utilized ML algorithms and preferably an ISBSG dataset for building effort and duration estimation models. Although a similar approach has been applied in numerous studies (Berlin et al., 2009; López-Martín and Abran, 2015; Pospieszny et al., 2015), those obtained in this research were more accurate if compared with MMRE and PRED. However, there is limited knowledge regarding other error measures and detailed data preparation approaches applied that would allow a comparison of models using a more holistic framework. The differences between algorithms' accuracy indicate that the most influential factor could be the approach applied for ISBSG data selection, cleansing and transformation. The superiority of the achieved results in this study indicate the effectiveness of methods utilized for ISBSG data preparation.

In relation to external validity, the research was based on a heterogeneous ISBSG dataset which provides data of completed projects from the last decade taken from diverse organisations around the world. Reliability, quality and update of data is maintained by the ISBSG organisation, which is unique in relation to other available datasets of completed software projects. Therefore, ISBSG is considered as the most comprehensive source of project data. Nevertheless, the proposed models and the deployment framework should be further validated through proof-of-concept conduct for 2-3 organisations by utilizing their distinctive data.

The implementation of machine learning models for software effort and duration estimation requires numerous prerequisites, mentioned in section 5.3. The most significant, is certainly the availability of completed projects that are used for the models' training. Depending on diversity of initiatives, their size and nature, the number of required completed projects may vary, but 40-60 should be sufficient. This may not be achievable for small organisations, therefore the models are more intended for medium and large size organisations where incorrect estimation can lead to significant implications. Additionally, prior to implementation, a cost benefit analysis should be performed, which for companies that conduct mostly small projects may have negative results, therefore, traditional methods based on expert knowledge could be sufficient.

7. Conclusions

In recent decades, end-users' requirements entailed tremendous advancements in software development that impacted every aspect of it, from distributed architecture and development teams, codeless programming to agile methodologies. Nevertheless, this had a limited effect on software estimation techniques, which despite developments in parametric approaches still depended excessively on expert judgment, required complex calculations and oftentimes lead to overoptimistic assumptions. Additionally, considering modern development and project management methodologies which rely on rapid time-to-market, iterative development and vague initial requirements, the traditional and parametric estimation techniques may be inefficient and impractical. Despite extensive research conducted in recent years on implementation of state of art machine learning algorithms for effort and duration estimation, which aim to tackle the mentioned deficiencies of existing approaches, few if any implementations can be found within organisations, potentially due to an emphasis on the effectiveness of individual algorithms and over complexity of building and implementation procedures. Therefore, the aim of this article was to present a holistic and practical approach for effort and duration estimation models with utilization of smart data preparation, three effective ML algorithms (SVM, MLP, GLM), ensemble averaging and proposed approach for model deployment and maintenance.

The results of ensemble models built for effort and duration estimation at the initial project lifecycle indicate that they are very accurate compared to other approaches taken by various researchers and are suitable for deployment in practice. For the learning process of ML algorithms, a highly diverse ISBSG dataset of software projects from various companies was utilized, hence for homogeneous data specific to the chosen organisation in which they will be deployed, the prediction accuracy of models is expected to be even greater. However, their performance in practice is dependent on the quality of input data, therefore significant attention should be put on data cleaning and preparation. In relation to deployment tools, the models can be built using commercial or open source software but it is valuable to incorporate them into EPM software existing within the organisation and foremost within software

development and project management processes in place. This will ensure models' usability and achieve cultural change that is required to occur for their successful implementation.

The proposed effort and duration estimation models are intended to serve as a decision support tool for any organisation developing and implementing software systems regardless of the industry sector where incorrect estimation may lead to negative implications. However, due to data availability and cost benefit analysis that may penalize small companies, it could be more feasible to apply them only for medium and large companies which are more mature from a project management perspective and have a large volume of completed projects. Their application is most beneficial at the initial stages of the project lifecycle when uncertainty regarding a final product and which initiative should be delivered is the highest. At that point the models can be used with waterfall and even agile methodologies because early estimation of budget and timeframe is often required for sponsors' approval, defining the project business case and establishing boundaries in general. Moreover, accurate estimation of effort and duration at that stage is often the most challenging activity for every project practitioner upon which successful project completion is dependent. Therefore, the proposed ensemble approach of machine learning algorithms, smart data preparation and outlined guidelines for their deployment and maintenance can certainly contribute to their application in practice within organisations, thereby boosting estimation accuracy, automation of the process and ultimately increase the project success rate.

In conclusion, directions of future work should be outlined. The models were built by utilization of a cross industry ISBSG dataset which contained up to date software projects which occurred in the last decade within multiple geographically distributed organisations. Moreover, it is the most comprehensive database available and reflects real initiatives that are currently performed by companies. Further research should focus on verifying the proposed approach through proof-of concept with 2-3 organisations to validate the model's accuracy and adjust the deployment and maintenance framework. Another limitation which should be addressed is the impact of software sizing especially on effort estimation. As an input variable, it has the most significant impact on forecasting the mentioned output parameter. In the ISBSG dataset, size was calculated using various FSM techniques, mostly IFPUG. For that purpose, other techniques can be applied, including traditional ones based on expert judgment and decomposition. Therefore, additional research should be performed on impact of various methods, especially the different FSM techniques that are considered as the most preferable sizing approach for accuracy of machine learning effort and duration estimation models. Additionally, future work could explore new trends in business analytics, such as prescriptive analytics (Pospieszny, 2017), which enable not only insights into the future but also allow scenario planning and smart foresight. Considering the current dynamic software development environment with agility, prototyping and rapid delivery, software simulation using machine learning algorithms could further enhance project estimation methods and contribute to better resource allocation and utilization.

Bibliography

Alami, A., 2016. Why Do Information Technology Projects Fail? Procedia Comput. Sci. 100, 62–71.

Azhar, D., Riddle, P., Mendes, E., Mittas, N., Angelis, L., 2013. Using Ensembles for Web Effort Estimation, in: 2013 ACM / IEEE International Symposium on Empirical Software Engineering and Measurement. IEEE, pp. 173–182.

Azzeh, M., Cowling, P.I., Neagu, D., 2010. Software stage-effort estimation based on association rule mining and Fuzzy set theory, in: Proceedings - 10th IEEE International Conference on Computer and Information Technology, CIT-2010, 7th IEEE International Conference on Embedded Software and Systems, ICESS-

2010, ScalCom-2010. pp. 249–256.
Balsera, J.V., Montequin, V.R., Fernandez, F.O., González-Fanjul, C.A., 2012. Data Mining Applied to the Improvement of Project Management. InTech.
Barrow, D.K., Crone, S.F., 2016. Cross-validation aggregation for combining autoregressive neural network forecasts. Int. J. Forecast. 32, 1120–1137.
Ben-David, S., Shalev-Shwartz, S., 2014. Understanding Machine Learning: From Theory to Algorithms, Understanding Machine Learning: From Theory to Algorithms. Cambridge University Press, New York.
Berlin, S., Raz, T., Glezer, C., Zviran, M., 2009. Comparison of estimation methods of cost and duration in IT projects. Inf. Softw. Technol. 51, 738–748.
Boehm, B.W., 1981. Software Engineering Economics. Prentice Hall. 10, 4–21.
Boehm, B.W., Abts, C., Brown, A.W., Chulani, S., Clark, B.K., Horowitz, E., Madachy, R., Reifer, D.J., Steece, B., 2000. Software Cost Estimation with Cocomo II. Prentice Hall PTR.
Breiman, L., 1996. Bagging predictors. Mach. Learn. 24, 123–140.
Carvalho, M.M. de, Patah, L.A., de Souza Bido, D., 2015. Project management and its effects on project success: Cross-country and cross-industry comparisons. Int. J. Proj. Manag. 33, 1509–1522.
Cios, K., Pedrycz, W., Swiniarski, R., Kurgan, L., 2007. Data Mining A Knowledge Discovery Apporach. Springer Science, New York, New York, USA.
Clarke, B., Fokoue, E., Zhang, H.H., 2009. Principles and Theory for Data Mining and Machine Learning. Springer Science, New York, New York, USA.
Conte, S.D., Dunsmore, H.E., Shen, V.Y., 1986. Software engineering metrics and models. Benjamin/Cummings Pub. Co.
Dave, V.S., Dutta, K., 2011. Neural network based software effort estimation & evaluation criterion MMRE, in: Computer and Communication Technology (ICCCT), 2011 2nd International Conference on. pp. 347–351.
de Barcelos Tronto, I.F., da Silva, J.D.S., Sant'Anna, N., 2008. An investigation of artificial neural networks based prediction systems in software project management. J. Syst. Softw. 81, 356–367.
Eveleens, J., Verhoef, C., 2009. The rise and fall of the chaos report figures. IEEE Softw. 1, 30–36.
Fayyad, U., Piatetsky-Shapiro, G., Smyth, P., 1996. From data mining to knowledge discovery in databases. AI Mag. 37–54.
Fernández-Diego, M., González-Ladrón-De-Guevara, F., 2014. Potential and limitations of the ISBSG dataset in enhancing software engineering research: A mapping review. Inf. Softw. Technol. 56, 527–544.
Flyvbjerg, B., 2006. From Nobel Prize To Project Management: Getting Risks Right. Proj. Manag. J. 37, 5.
Foss, T., Stensrud, E., Kitchenham, B., Myrtveit, I., 2003. A simulation study of the model evaluation criterion MMRE. Softw. Eng. IEEE Trans. 29, 985–995.
Freund, Y., Schapire, R.R.E., 1996. Experiments with a New Boosting Algorithm, in: International Conference on Machine Learning. pp. 148–156.
Galorath, D., Evans, M., 2006. Software Sizing, Estimation, and Risk Management. Auerbach Publications.
García, S., Luengo, J., Herrera, F., 2016. Tutorial on practical tips of the most influential data preprocessing algorithms in data mining. Knowledge-Based Syst. 98, 1–29.
Gasik, S., 2011. A model of project knowledge management. Proj. Manag. J. 42, 23–44.
Glass, R.L., 2006. The Standish report: Does it really describe a software crisis? Commun. ACM 49, 15–16.
González-Ladrón-de-Guevara, F., Fernández-Diego, M., Lokan, C., 2016. The usage of ISBSG data fields in software effort estimation: A systematic mapping study. J. Syst. Softw. 113, 188–215.
Han, J., Kamber, M., Pei, J., 2006. Data Mining: Concepts and Techniques. Morgan Kaufmann.
Hill, P., 2010. Practical Software Project Estimation: A Toolkit for Estimating Software Development Effort & Duration. McGraw Hill Professional.
Ho, T.K., 1998. The random subspace method for constructing decision forests. IEEE Trans. Pattern Anal. Mach. Intell. 20, 832–844.
Huang, J.., Li, Y.-F.., Xie, M.., 2015. An empirical analysis of data preprocessing for machine learning-based software cost estimation. Inf. Softw. Technol. 67, 108–127.

Huang, J., Li, Y.-F., Xie, M., 2015. An empirical analysis of data preprocessing for machine learning-based software cost estimation. Inf. Softw. Technol. 67.

Huang, S.-J., Chiu, N.-H., Chen, L.-W., 2008. Integration of the grey relational analysis with genetic algorithm for software effort estimation. Eur. J. Oper. Res. 188, 898–909.

Idri, A., Hosni, M., Abran, A., 2016. Improved Estimation of Software Development Effort Using Classical and Fuzzy Analogy Ensembles. Appl. Soft Comput. 49, 990–1019.

International Software Benchmarking Standards Group, 2013. ISBSG Repository Data Release 12 - Field Descriptions.

Iranmanesh, S.H., Mokhtari, Z., 2008. Application of data mining tools to predicate completion time of a project. Proceeding world Acad. Sci. Eng. Technol. 32, 234–240.

Jeffery, R., Ruhe, M., Wieczorek, I., 2001. Using public domain metrics to estimate software development effort, in: Proceedings Seventh International Software Metrics Symposium. IEEE Comput. Soc, pp. 16–27.

Jorgensen, M., Shepperd, M., 2007. A Systematic Review of Software Development Cost Estimation Studies. IEEE Trans. Softw. Eng. 33, 33–53.

Kemerer, C., 1993. Reliability of function points measurement: a field experiment. Commun. ACM 36, 85–97.

Keung, J., Kocaguneli, E., Menzies, T., 2013. Finding conclusion stability for selecting the best effort predictor in software effort estimation. Autom. Softw. Eng. 20, 543–567.

Kobyliński, A., Pospieszny, P., 2015. Zastosowanie technik eksploracji danych do estymacji pracochłonności projektów informatycznych, in: Studia i Materiały Polskiego Stowarzyszenia Zarządzania Wiedzą. Bydgoszcz, pp. 67–82.

Kocaguneli, E., Menzies, T., Keung, J.W., 2012. On the value of ensemble effort estimation. IEEE Trans. Softw. Eng. 38, 1403–1416.

Krogh, Anders Jesper, V., 1995. Neural Network Ensembles , Cross Validation , and Active Learning, in: Advances in Neural Information Processing Systems 7. pp. 231–238.

Laird, L.M., Brennan, M.C., 2006. Software Measurement and Estimation: A Practical Approach. John Wiley & Sons.

Larose, D.T., 2007. Data Mining Methods and Models. John Wiley & Sons.

Larose, D.T., Larose, C.D., 2015. Data Mining and Predictive Analytics. John Wiley & Sons, New Jersey.

Lavazza, L., 2015. Automated function points: Critical evaluation and discussion, in: International Workshop on Emerging Trends in Software Metrics, WETSoM. pp. 35–43.

Lehtinen, T.O.A., Mäntylä, M. V., Vanhanen, J., Itkonen, J., Lassenius, C., 2014. Perceived causes of software project failures – An analysis of their relationships. Inf. Softw. Technol. 56, 623–643.

Linoff, G.S., Berry, M.J.A., 2011. Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management. John Wiley & Sons.

Lopez-Martin, C., Isaza, C., Chavoya, A., 2012. Software development effort prediction of industrial projects applying a general regression neural network. Empir. Softw. Eng. 17, 738–756.

López-Martín, C., 2015. Predictive accuracy comparison between neural networks and statistical regression for development effort of software projects. Appl. Soft Comput. 27, 434–449.

López-Martín, C., Abran, A., 2015. Neural networks for predicting the duration of new software projects. J. Syst. Softw. 101, 127–135.

McConnell, S., 2009. Software Estimation: Demystifying the Black Art: Demystifying the Black Art. Microsoft Press.

Mieritz, L., 2012. Survey Shows Why Projects Fail, Gartner Research Report.

Minku, L.L., Yao, X., 2013. Ensembles and locality: Insight on improving software effort estimation, in: Information and Software Technology. pp. 1512–1528.

Mitchell, T.M., 1997. Machine Learning, McGraw-Hill.

Mittas, N., Mamalikidis, I., Angelis, L., 2015. A framework for comparing multiple cost estimation methods using an automated visualization toolkit. Inf. Softw. Technol. 57, 310–328.

Moeyersoms, J., Junqué De Fortuny, E., Dejaeger, K., Baesens, B., Martens, D., 2015. Comprehensible software fault and effort prediction: A data mining approach. J. Syst. Softw. 100, 80–90.

Myrtveit, I., Stensrud, E., 2012. Validity and reliability of evaluation procedures in comparative studies of effort prediction models. Empir. Softw. Eng. 17, 23–33.

Nagwani, N.K., Bhansali, A., 2010. A data mining model to predict software bug complexity using bug estimation and clustering, in: ITC 2010 - 2010 International Conference on Recent Trends in Information, Telecommunication, and Computing. pp. 13–17.

Pai, D.R., McFall, K.S., Subramanian, G.H., 2013. Software effort estimation using a neural network ensemble. J. Comput. Inf. Syst. 53, 49–58.

Pete, C., Julian, C., Randy, K., Thomas, K., Thomas, R., Colin, S., Wirth, R., 2000. CRISP-DM 1.0, CRISP-DM Consortium.

Port, D., Korte, M., 2008. Comparative Studies of the Model Evaluation Criterions Mmre and Pred in Software Cost Estimation Research, in: Proceedings of the Second ACM-IEEE International Symposium on Empirical Software Engineering and Measurement. pp. 51–60.

Pospieszny, P., 2017. Software estimation: towards prescriptive analytics, in: IWSM Mensura '17. ACM, Gothenburg, pp. 221–226.

Pospieszny, P., Czarnacka-Chrobot, B., Kobyliński, A., 2015a. Application of Function Points and Data Mining Techniques for Software Estimation - A Combined Approach, in: 25th International Workshop on Software Measurement and 10th International Conference on Software Process and Product Measurement. Springer, pp. 96–113.

Pospieszny, P., Czarnacka-Chrobot, B., Kobyliński, A., 2015b. Application of function points and data mining techniques for software estimation - a combined approach, in: Lecture Notes in Business Information Processing.

Project Management Institute, 2013. A Guide to the Project Management Body of Knowledge - PMBOK Guide, Fifth Edit. ed, PMI Book. Project Management Institute.

PROMISE Software Engineering Repository [WWW Document], http://promise.site.uottawa.ca/SERepository/ (accessed 5.18.17).

Reifer, D.J., Boehm, B.W., Chulani, S., 1998. The Rosetta stone: Making COCOMO 81 Files Work With COCOMO II. Univ. South Calif. 1–10.

Richman, L., 2011. Successful Project Management, 3rd ed. AMACOM Div American Mgmt Assn, New York, New York, USA.

Ruan, D., Chen, G., Kerre, E.E., 2005. Intelligent data mining: techniques and applications, 5th ed. Springer Science & Business Media.

Savolainen, P., Ahonen, J.J., Richardson, I., 2012. Software development project success and failure from the supplier's perspective: A systematic literature review. Int. J. Proj. Manag. 30, 458–469.

Schwalbe, K., 2014. Information Technology Project Management, Technology. Course Technology, Boston.

Sehra, S.K., Brar, Y.S., Kaur, N., Sehra, S.S., 2017. Research patterns and trends in software effort estimation. Inf. Softw. Technol.

Shukla, R., Shukla, M., Misra, A.K., Marwala, T., Clarke, W.A., 2012. Dynamic software maintenance effort estimation modeling using neural network, rule engine and multi-regression approach, in: Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). pp. 157–169.

Singh, B.K., Misra, A.K., 2012. Software Effort Estimation by Genetic Algorithm Tuned Parameters of Modified Constructive Cost Model for NASA Software Projects. Int. J. Comput. Appl. 59, 975–8887.

SourceForge [WWW Document], http://sourceforge.net (accessed 4.18.17).

Spalek, S.J., 2005. Critical Success Factors in Project Management -- To Fail or Not To Fail, That is the Question!, in: PMI Global Congress Proceedings. pp. 1–7.

Standish Group, 2015. Standish Group 2015 Chaos Report [WWW Document].

Strike, K., El Emam, K., Madhavji, N., 2001. Software cost estimation with incomplete data. IEEE Trans. Softw. Eng. 27, 890–908.

Tan, S., 2011. How to Increase Your IT Project Success Rate. Gart. Res. Rep.

Todorović, M.L., Petrović, D.Č., Mihić, M.M., Obradović, V.L., Bushuyev, S.D., 2015. Project success analysis

framework: A knowledge-based approach in project management. Int. J. Proj. Manag. 33, 772–783.
Trendowicz, A., 2014. Software Project Effort Estimation: Foundations and Best Practice Guidelines for Success.
Trendowicz, A., Münch, J., Jeffery, R., 2011. State of the practice in software effort estimation: a survey and literature review. Softw. Eng. Tech. 232–245.
Wen, J., Li, S., Lin, Z., Hu, Y., Huang, C., 2012. Systematic literature review of machine learning based software development effort estimation models. Inf. Softw. Technol. 54, 41–59.
Wilkie, F.G., McChesney, I.R., Morrow, P., Tuxworth, C., Lester, N.G., 2011. The value of software sizing. Inf. Softw. Technol. 53, 1236–1249.
Wysocki, R.K., 2014. Effective Project Management: Traditional, Agile, Extreme, Industry Week. John Wiley & Sons.
Xu, L., Krzyzak, A., Suen, C.Y., 1992. Methods of combining multiple classifiers and their applications to handwriting recognition. IEEE Trans. Syst. Man. Cybern. 22, 418–435.

Biography - Przemyslaw Pospieszny

Dr. Przemyslaw Pospieszny is a Research Scientist in a field of data science, project management and smart cities. He holds MSc in Business Computing from Poznan University of Economics, PgD in Knowledge Management from Dublin Institute of Technology and PhD from Warsaw School of Economics. His academic development he is combining with 10 years of professional experience working as a business analyst and project manager within major international financial institutions. Currently he is leading a Data Lab for a commercial real estate company (JLL).