CHAPTER 3

Method of automatic assignment of tasks to IT project performers

Tetiana Borysenko
Maksym Yevlanov
Konstantin Petrov
Viktor Borysenko

Abstract

The aim of the study is to develop a method for solving the problem of assigning IT project tasks to its performers.

The object of the study is the process of planning an IT project.

During the study, the problem of assigning IT project tasks to its performers was solved. It is shown that in recent years heuristic solutions to this problem have gained popularity. The main requirements for such a solution are determined. The results of the analysis of modern research confirm the relevance of scientific and applied works devoted to solving the problem of assigning IT project tasks to its performers.

According to the results of the study, it is proposed to represent the problem of assigning IT project tasks to its performers as a type of classification problem. This representation allowed using a polynomial Bayesian classifier to solve this problem. In the process of the study, the classification rule with a minimum error and calculations of the main elements of this rule were adapted to the specifics of the problem. An additional classification condition was established that prevents the maximum load of the IT project performer from being exceeded. Based on the results of the adaptation, the algorithm for solving the classification problem, which uses this adapted classifier, was modified.

A general description of the method for solving the problem of assigning IT project tasks to its performers was developed. An algorithm for implementing this method was developed for a detailed description of the content of individual stages. The scheme of this algorithm and the proposed descriptions of the main steps of the algorithm determine the features of its implementation both as a methodology

 DOI: 10.21303/978-9908-9706-8-4.ch3

for applying the obtained solutions in the current management of an IT project and as a specialized information technology.

To verify the operability of the obtained results, an experimental test of the method and its implementation algorithm was conducted during the management of one of the IT projects of an outsourcing IT company. The test results indicate the feasibility of using the developed method to solve the problem of assigning IT project tasks to its performers. The developed method contributes to better project planning, minimizes the administrative burden and helps to avoid delays and errors in project implementation.

Keywords

IT project, task, performer, polynomial Bayesian classifier, effort, team, sprint.

3.1 Introduction

The main difference between IT project management and project management in any other field of human activity is the recognition of personnel as the single most used resource of the project. The emergence and widespread implementation of Agile project management methodologies in the IT industry have not led to significant changes in the existing attitude towards IT project personnel. As early as 2007, it was established that direct costs for an IT project are almost exclusively determined by labor costs [1]. Therefore, personnel management problems remain one of the main problems facing IT project management.

Among these problems, it is worth highlighting the problems that arise during IT project planning. In general, IT project planning includes:

– definition of tasks;

– assessment of task duration;

– placement of tasks on a timeline that visually reflects their sequence, duration, start and end dates;

– according to the results of the placement – assigning resources to each task from available jobs, divided by skill set [2].

Assigning tasks to IT project performers is a complex and time-consuming activity, especially in large and complex projects. During the implementation of this activity, important issues arise regarding the correct distribution of tasks between performers, ensuring the optimization of human resources for task performance, and maximizing the productivity of performers.

It should be noted that modern IT companies use a significant number of various information technologies (IT) to support the IT project planning process. However,

the use of such IT is severely limited by the fact that the personnel of the majority of IT projects is a limited resource. Recognition of this fact led to the formulation of the Resource Constrained Project Scheduling Problem (RCPSP), which belongs to NP-hard problems [3]. Such problems for large and medium-sized IT projects are very often intractable due to the excessive growth of the volume of required calculations. Therefore, basic research suggests two main ways to solve this problem [4]:

– using models and methods that provide accurate solutions to the project planning problem taking into account scarce constraints;

– using heuristic solutions.

As a result, the most common IT planning of IT projects mostly leaves the implementation of the activity of distributing project tasks between performers to the discretion of project management specialists. But this approach leads to subjective errors, irrational use of resources and overloading of individual performers. This often happens in large projects, where the complexity of coordination increases in proportion to the number of tasks and performers. Therefore, the automation of the activity of assigning tasks to performers during the planning and implementation of an IT project remains an extremely important and relevant scientific and applied task. Solving this task will undoubtedly contribute to the optimization of planning processes, avoid excessive administrative burden, ensure more accurate and efficient distribution of work, and also reduce the risk of errors or delays in an IT project.

3.2 Overview of the current state of solving the problem of automated planning of IT projects

The existence of RCPSP as an NP-hard problem has been confirmed in recent years at the level of a number of recent versions of PMBOK [5, 6]. Therefore, a significant amount of modern research is focused on the search and development of models, methods and technologies that allow either to obtain an exact solution or to develop a heuristic solution to this problem.

An example of research aimed at finding an exact solution to RCPSP is the work [2]. The goal of this research is to develop software that creates an optimal schedule that takes into account both technical and resource constraints of the project in order to prevent resource reallocation. In this case, [2] identifies three main factors that can affect the allocation of resources during the project life cycle:

– an increase in the scope of the project, which will require greater efforts to complete it;

– possible unavailability of resources during periods in which they were initially considered available;

– an existence of competition for the same resources between several projects that can be performed simultaneously.

As a result of the research in [2], the creation of software is claimed, which should help the project manager in creating the optimal schedule of this project. This software provides an accurate RCPSP solution in a reasonable period of time in the conditions of performing within the IT company of several projects simultaneously. At the same time, the developed system takes into account the distribution of IT project performers according to their skill set for a better assessment of the duration of project tasks [2]. However, the research result given in [2] is not free from shortcomings that complicate the practical application of the developed software. The main of these shortcomings should be recognized as the need to recalculate all IT project schedules as a result of any change in the availability of IT company personnel. This need leads to an increase in the number of computational operations performed during planning or replanning of IT projects. In addition, this need contradicts the principle of the oncoming wave, according to which detailed planning of the IT project schedule until its completion is considered impractical [5].

Using heuristic solutions to solve the RCPSP requires dividing this problem into a sequence of separate IT project planning management tasks and solving each task separately using the appropriate heuristic. An example of such an approach is research [7]. In this work, it is proposed to develop task and developer profiles to solve the problem of assigning each project task to the most suitable developer. After that, the task is solved by finding the best match of these profiles for assigning tasks to developers. A comparative analysis showed that the Sokal and Snit method [7] should be considered the best for such a search. It should be noted that one of the authors of this study also tried to solve this problem in a similar way. In [8], he proposed a solution to create task and project developer profiles based on a data-logical description of the project work packages. The use of these profiles allowed [8] to propose a method for solving the problem of assigning enterprise employees to tasks of a new IT project, the basis of which is the apparatus of clustering and classification tasks. This method made it possible to assign to the task of the new project an employee whose value of the integral indicator of the quality of work performance was maximum.

But this method of solving the problem of assigning each project task to the most suitable developer is not free from some shortcomings. First, the solutions proposed in [7] and [8] require the creation and constant maintenance of data warehouses in which it is necessary to store and update historical data on the profiles of completed

IT project tasks and project performers. Second, the use of complex data logical descriptions to build profiles leads to significant expenditure of money and time on additional work on the creation, support and maintenance of such profiles during the planning and performing of an IT project. Third, this method leaves open the question of the optimal distribution of tasks among project personnel from the point of view of maximizing and equalizing the employment of project performers.

It must be recognized that the approach to profiling project tasks or work packages and storing the corresponding historical information is characteristic not only of IT projects. In [9], a similar approach is considered for creating a hierarchy of construction project task organization. This study is interesting in that it recognizes the need to create project task profiles not based on special complex datalogical descriptions, but on the results of pre-processing text descriptions of project work packages. Although such a solution complicates the work of processing information about project work packages, it significantly simplifies the work of forming, maintaining and maintaining similar profiles of these work packages.

An attempt to find an exact RCPSP solution using heuristics is considered in [10]. This study proposes to use a genetic algorithm to form a schedule that provides maximum profit from the implementation of an IT project. The following assumptions are taken into account [10]:

– capacity constraints can often be systematically changed by temporarily assigning expensive additional production resources or using overtime;

– project revenue decreases with increasing duration of its implementation.

But such a solution to RCPSP is not free from the shortcomings identified for the software for finding the exact solution to RCPSP proposed in [2]. In addition to these shortcomings, the accuracy of the obtained result is affected by the shortcomings caused by the use of the genetic algorithm, of which it is worth noting in particular [11]:

– the tendency to converge to a local optimum instead of searching for a global optimum;

– the difficulties that arise when formulating the stopping condition of the genetic algorithm;

– the poor scalability of genetic algorithms to the complexity of the problem being solved (which is of particular importance for creating schedules for large complex IT projects).

To eliminate the last of the listed shortcomings, in [12] it was proposed to use a genetic algorithm to solve the problem of assigning IT project tasks to members of the development team within one iteration or sprint of the project. At the same time, when solving this problem, the experience of each specific performer is taken into account. As descriptions of IT project tasks, user stories that the development

team must implement in a planned iteration or sprint are considered [12]. Such a solution to the problem of assigning tasks to members of the IT project development team partially eliminates the above-discussed disadvantages of the stopping complexity and poor scalability of genetic algorithms, but remains quite complex to implement and use.

In a later study [13], it was proposed to consider the RCPSP problem for many skills as a task of assigning tasks to employees, taking into account predecessor tasks and constraints for many skills to create an appropriate schedule with the shortest performing time. Differential evolution with multidimensional real-valued functions was proposed to solve this problem. In [13], for this variant of the evolutionary algorithm, its modification was proposed by improving the mutation method, which is performed twice per generation, in order to increase the diversity of the population and obtain better results. Thus, the authors of [13] proposed to eliminate the tendency of genetic and evolutionary algorithms to converge to a local optimum. However, the solution based on the use of evolutionary algorithms is also quite complicated to implement and use. It should be recognized that the task of comparative analysis of the economic and technological efficiency of the operation of various artificial intelligence tools in the conditions of IT project management requires additional research.

In general, for the first half of the 2020s, it was considered appropriate to consider the problem of assigning project tasks to its performers as a multi-criteria optimization problem. An example of this point of view is the conceptual model proposed in [14] for studying the problem of selecting partners for the implementation of research or educational projects. The main components of this model are determining the criteria for selecting potential partners, applying the multi-objective optimization method, expert assessment, forming the preferences of partners or project performers, and making a decision on assigning project tasks to partners that correspond to the corresponding work package [14]. However, the use of expert assessments does not give this conceptual model an advantage over existing methods for solving the problem under consideration for Agile IT projects. The use of experts when solving the problem of assigning project tasks to its performers significantly reduces the objectivity of the resulting solution.

In [15], it is stated that the criteria by which the problem of assigning IT project tasks to its performers should be solved should be:

– qualification of the team of performers;
– competences of the team of performers;
– seriousness of individual tasks;
– priority of individual tasks.

To solve this problem, a mathematical model of labor resource distribution was developed in [15]. This model is based on a probabilistic analysis of the importance and priorities of tasks, ensures more efficient use of personnel and allows for timely completion of the most important tasks of an IT project.

The solution proposed in [15] to the problem of assigning IT project tasks to its performers is quite interesting for automated planning of the activities of the IT project team within one iteration (or sprint). But it is not without some shortcomings, among which it should be noted the impossibility of taking into account the experience of individual performers in solving individual types of IT project tasks. Such experience, in particular, takes into account the performer's knowledge of the subject area and the performer's specialization in the successful implementation of specific types of functions (services, etc.) of the created IT product.

Unlike [15], in [16] it is proposed to use a machine learning-based decision support system that works with data in real time to solve the problem of assigning IT project tasks to its performers. This system analyzes the description of new requested tasks using text mining and machine learning approaches, and then predicts the optimal available personnel that meet the needs of the project task. Personnel qualifications are iteratively updated by the system after each completed task, which provides up-to-date information about personnel capabilities. The basis of this system is the mathematical apparatus of vector descriptions of unstructured texts and structured documents. As a measure of comparison of vector descriptions of new project tasks and performers in the system, cosine similarity is used. The system is built on the basis of microservice architecture, which facilitates its integration into existing IT project planning systems [16]. But the use of vector representations of texts and documents is a rather significant drawback of the proposed solution to the problem of assigning IT project tasks to its performers. Such representations require quite large expenditures of computing resources to create and maintain such descriptions.

The analysis of the considered scientific and applied research in the field of automated solution of RCPSP and its individual tasks for IT projects allowed to draw the following conclusions.

Firstly, none of the considered RCPSP solutions and, in particular, the problem of assigning IT project tasks to its performers is free from shortcomings that significantly limit their practical use in applied IT project planning activities.

Secondly, the considered research results indicate the principle possibility and practical feasibility of solving such problems under the following conditions:

– it is advisable to consider the RCPSP solution as a set of solutions to individual IT project planning problems, in particular – the problem of assigning IT project tasks to its performers;

– it is advisable to consider the problem of assigning IT project tasks to its performers as a multi-criteria optimization problem, the criteria of which can be both the characteristics of individual tasks (seriousness, priority, etc.) and the characteristics of individual IT project performers (competences, specialization, previous work experience, etc.);

– the expected result of solving the problem of assigning IT project tasks to its performers should form a solution to this problem with minimal or no human operator participation (operate in automatic mode);

– the expected result of solving the problem of assigning IT project tasks to its performers should not require significant time and computing resources even when planning large complex projects;

– the expected result of solving the problem of assigning IT project tasks to its performers should be easily integrated into existing application systems for automated planning of IT projects and their iterations.

These conditions can be met if the solution to the problem of assigning IT project tasks to its performers is based on the information that can be obtained with minimal costs from existing IT planning and management of IT projects. Such information, in particular, includes text names and descriptions of project tasks that should be distributed between its performers. Therefore, it will be advisable to consider in the future the task of assigning IT project tasks to its performers as a special case of the text information classification problem. However, unlike the solution proposed in [16], it is necessary to use not vector, but scalar characteristics of project tasks and individual performers to solve such a problem. This makes it possible to use methods that have long been known and proven in various industries to solve the text classification problem.

A comprehensive and detailed analysis of existing text classification methods is given in [17, 18]. In addition, in [18], methods of preprocessing text before classification and indicators of the quality assessment of classification algorithms are described. In [19], the authors use empirical studies to establish a relationship between the size of the training data set and the amount of data required for testing.

The naive Bayesian classifier (NB) [17, 18] has proven itself well in solving such classification problems. This method is based on Bayes' theorem and uses a probabilistic approach to object classification. Confirmation of the NB use can be found in [20, 21], where the performance of different text classifiers is compared in the context of specific classification problems. NB, together with several other classifiers, has demonstrated quite high accuracy.

However, there are certain problems associated with the NB use, as described in [22], and solutions have been proposed to overcome these problems and improve the performance of the algorithms.

Various algorithms have been proposed for the NB implementation, which are used to solve classification problems. These algorithms differ in the requirements for input data and mathematical models that they use to predict object classes [18].

Thus, the aim of this study is to develop a method for solving the problem of assigning IT project tasks to its performers, which must meet the above conditions. Using this method when distributing IT project tasks between members of the team of performers will reduce time costs and increase the objectivity of project iteration (sprint) planning.

To achieve this aim, the following research objectives were set:

– adaptation of polynomial NB to the features of the problem of assigning IT project tasks to its performers;

– development of an algorithm for implementing the developed method;

– experimental verification of the operability of the obtained results.

3.3 Materials and methods

The object of the study is the IT project planning process, in particular, the activity of planning project and technical management [23]. It is within the framework of this activity that the vast majority of IT projects for the creation, modernization or development of IT products should solve the problem of assigning IT project tasks to its performers.

The main hypothesis of this study is the hypothesis of the possibility of automated solution of the problem of assigning IT project tasks to its performers using the methods and models of NB text classification or its varieties.

The study of the features of solving the problem of assigning IT project tasks to its performers is considered in the context of teams working on an IT project using the Agile methodology and, in particular, the Scrum process framework, which provides for an iterative development process. The principles, values, processes, artifacts (Product Backlog, Sprint Backlog, Increment, etc.) and the roles of the Scrum process framework are described in [24, 25]. The practice and problems of prioritizing requirements (and, accordingly, project tasks) in industrial Agile projects, which plays an important role in sprint planning, are highlighted in [26].

As an object of automation in this study, it is proposed to consider an IT company specializing in the release of several products. The term "product" in this study denotes a named set of business opportunities that are valuable for a certain segment of the IT company's customers. Each product of such a company can be the result of the implementation of several projects.

To initiate a new project or implement a certain activity in an existing project, initiatives are created in the company with the participation of product managers and product owners. According to project management standards, the term "initiative" can have several definitions depending on the context. However, in this study, an initiative is an initial action or direction that is chosen to launch a new project or implement a certain activity. An initiative can arise for a variety of reasons, from ideas and strategic goals, to changes in the business environment or consumer needs.

In the process of forming an initiative, high-level tasks of the company are created with the participation of top management, which are agreed upon with the relevant stakeholders of the project. These tasks (another name is topics) are large groups of values for customers, reflected in a set of requirements or user stories related to common functionality, data source or level of security. The created initiative to achieve the results planned by the company involves the further involvement of one or another team or a set of teams of performers. Such involvement occurs with the participation of product managers and product owners. Product Owner is a role in the product development team, which is responsible for managing the product backlog to achieve the desired result that the product development team seeks to achieve [27]. Next, at the project planning stage, the decomposition of the specified initiative into more structured tasks – epics occurs. An Epic is a large, interconnected set of work designed to hierarchically organize a set of requirements and deliver specific business outcomes [6]. Epics divide the initiative's topics into separate functional elements – a set of interconnected functional requirements. Each requirement is described by its own user story. Epics can be created repeatedly in the future, based on the need to create new or additional epics to better perform the initiative. It is important to note that already during the formation of epics, the product owner receives information about the preliminary (possible) assignment of the project team. This information can be based on the results of previous IT projects in which these teams have already been involved in working on similar tasks, or be formed through preliminary conversations and agreements with the managers (leaders) of these teams.

For each of the formed epics, its performing priority is determined. As a result of this determination, the following are formed:

– product roadmap: a publicly available live document that describes the type and direction of product development throughout its life cycle [28];

– product backlog: a prioritized list of user stories for the development team, compiled on the basis of the product roadmap and its requirements [29].

Next, epics are decomposed into individual tasks. This allows to detail the requirements and estimate the scope of work on their implementation in order to

further determine the specific steps of the performers to achieve the desired results. This decomposition occurs at the following events:

– sprint planning (an event in Scrum that launches a sprint);

– Backlog Grooming (a process of regular analysis, refinement, and prioritization of tasks in the product backlog).

The result of this decomposition is the division of epics into separate prioritized user stories and the addition of epics with additional tasks of a technical nature (Tasks). These Tasks may arise from requests to change existing requirements, feedback from IT product users, technical needs that arise during the project, or as a result of an initiative by a team of performers. The results of the decomposition are elements of the product backlog that describe the content of each of the project epics in as much detail as possible. These elements are tasks that must then be distributed among individual performers of one or more teams participating in the implementation of a specific initiative or epic.

A diagram showing an example of possible relationships between initiatives, epics, and tasks of an IT project is shown in **Fig. 3.1**.

Thus, the solution of the problem of assigning IT project tasks to its performers in this study is considered as starting after the decomposition of initiatives into epics, and epics into individual IT project tasks. These decompositions form the product backlog as a set of epics and individual IT project tasks, which belong either to the corresponding epics or to individual initiatives.

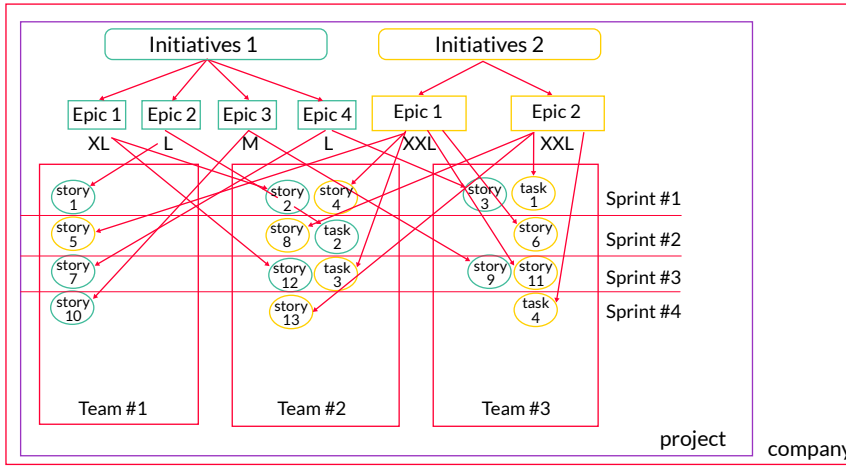

Fig. 3.1 Scheme of relationships between initiatives, epics and tasks of an IT project

To solve the problem of assigning IT project tasks to its performers, the following criteria are proposed to be used:

– the priority of performing that each task has;

– the possible belonging of the task to an epic that is assigned to one or several teams of performers;

– the total efforts for one sprint of each performer (depending on the position), which should not be exceeded when distributing tasks;

– the possible gradation of performers by positions (competences) in the team.

Of all the NB options, the multinomial naive Bayes classifier (MNB) was proposed to be used by the IT project task assignment task assigners. The choice of this classifier was due to the following considerations:

– attributes of IT project task descriptions as classification objects contain discrete features corresponding to a polynomial distribution;

– MNB is easy to implement and allows for fast processing of large amounts of data.

The MNB effectiveness has been confirmed by studies [30, 31]. In particular, [30] describes the results of a comparison of the Bernoulli-Naive Bayes algorithm and MNB and concludes that the polynomial model generally outperforms the Bernoulli model. [31] considers the application of seven different algorithms for the classification and analysis of tweets related to real natural disasters. MNB, in comparison with the Bernoulli-Naive Bayes algorithm and logistic regression, showed the best results in classifying tweets as concise texts with media elements.

Let the object of classification X be described by a set of features $(X_1, X_2,..., X_n)$. Then the posterior probability of belonging of the object X to the class y_k from the set of classes Y should be calculated by the formula

$$P(y_k \mid X_1, X_2, ..., X_n) = \frac{P(y_k)\prod_{i=1}^{n} P(X_i \mid y_k)}{P(X_1, X_2, ..., X_n)}, \tag{3.1}$$

where $P(y_k|X_1, X_2,..., X_n)$ – the posterior probability of belonging of a classification object with a set of features $(X_1, X_2,..., X_n)$ to the class y_k; $P(y_k)$ – the prior probability of the existence of the class y_k; $P(X_i|y_k)$ – the probability that a classification object that has a feature X_i with some fixed set of values will be classified as belonging to the class y_k; $P(X_1, X_2,..., X_n)$ – the probability that a classification object X will be described by a set of features $(X_1, X_2,..., X_n)$.

For the case when the prior probabilities $P(y_k)$ for each element y_k of the set of classes Y are known, the classification rule with minimum error [22] is used, which selects the class with the highest posterior probability

$$P(y_k \mid X_1, X_2, ..., X_n) \propto \text{argmax}_k P(y_k) \prod_{i=1}^{n} P(X_i \mid y_k), \tag{3.2}$$

where $\propto$ – the proportionality sign.

In many cases, the classification rule (3.2) is translated into logarithmic space. Then it takes the form [22]:

$$\log P(y_k \mid X_1, X_2, ..., X_n) \propto \text{argmax}_k \log\left(P(y_k) \prod_{i=1}^{n} P(X_i \mid y_k) \right), \tag{3.3}$$

$$\log P(y_k \mid X_1, X_2, ..., X_n) = \text{argmax}_k \left(\log P(y_k) + \sum_{i=1}^{n} X_i \log P(X_i \mid y_k) \right), \tag{3.4}$$

where X_i in (3.4) is the frequency of occurrence of the feature values in the set of studied data (training set), on the basis of which the prior probabilities $P(y_k)$ are calculated.

To estimate the parameters of each class, in [22] it was proposed to use a smoothed version of the maximum likelihood estimation

$$P(X_i \mid y_k) = \frac{N_{y_k i} + \alpha}{N_{y_k} + \alpha n}, \tag{3.5}$$

where $N_{y_k i}$ – the number of times the feature X_i occurs in the objects of classification X belonging to the class y_k; N_{y_k} – the total number of times all features X_i occur in the objects of classification X belonging to the class y_k; n – the number of features describing the objects of classification in the training set; α – the smoothing parameter.

Although the value of the parameter α may be different for each feature, in [22] it is recommended to follow the generally accepted practice, setting $\alpha = 1$ for all features.

The estimate of the probability value $P(y_k)$ is proposed to be calculated as follows

$$P(y_k) = \frac{O_{y_k}}{O}, \tag{3.6}$$

where O_{y_k} – the number of objects of classification present in the training set and belonging to the class y_k; O – the total number of objects of classification present in the training set.

The UML activity diagram describing the algorithm for solving the classification problem using MNB is shown in **Fig. 3.2**.

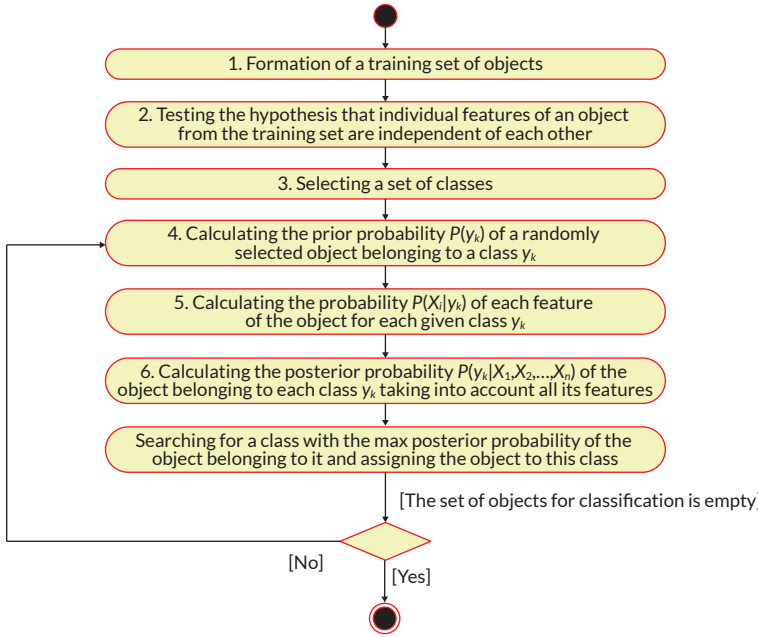

Fig. 3.2 UML activity diagram describing the algorithm for solving the classification problem using a polynomial Bayesian classifier

The use of MNB for solving applied classification problems is associated with the emergence of a number of problems [20]. Among these problems, it is necessary to highlight:

– the problem of reducing the MNB score for a class with a smaller number of training samples in the case when the training set has more training samples for one class than for another;

– the problem of the possible appearance of a class that lacks training sets and, as a result, cannot participate in the classification process;

– the problem of checking the independence of the features of the classification objects in the training set and those that need to be classified from each other.

These problems can be solved in the process of preparing the training set by appropriately selecting the training objects for classification and checking the training set for the independence of the features of these objects.

Another problem arises when the classification object, the class of which should be determined in the process of solving the problem, has a feature that has never

appeared in the training set [20]. This problem can be solved using the smoothing parameter α, $0 < \alpha \leq 1$. The choice of the value of this parameter can significantly affect the results of solving the classification problem.

For solving the problem of assigning IT project tasks to its performers, the description of the algorithm for solving the classification problem using MNB given in **Fig. 3.2** is not entirely suitable due to the following features:

– in this algorithm, the classification process continues until there are no objects left that have not yet been classified;

– the number of objects that can belong to a class is not limited in any way.

In the problem of assigning IT project tasks to its performers, the assignment process must continue until each performer accumulates the necessary total efforts for its position. Therefore, there is a need to modify the method for solving the classification problem using MNB, taking into account this feature.

3.4 Results of developing a method for solving the problem of assigning IT project tasks to its performers

3.4.1 Results of adapting the polynomial naive Bayes classifier to the specifics of the problem of assigning IT project tasks to its performers

The process of adapting MNB to the specifics of the task of assigning IT project tasks to its performers is proposed to be divided into two main stages:

– adaptation of the classification rule with minimal error (3.2) to the specifics of the task of assigning IT project tasks to its performers;

– modification of the algorithm for solving the classification problem using the adapted MNB.

During the adaptation of the classification rule with minimal error (3.2) to the specifics of the task of assigning IT project tasks to its performers, the main elements of this rule were considered and their definitions and calculation rules were clarified. As objects of the MNB classification X as a result of the adaptation, IT project tasks presented in the form of User Story and Task should be considered. The features $(X_1, X_2, ..., X_n)$ of the classification object X, which are taken into account in rule (3.2), should be the words from which the names of IT project tasks are composed. The names of tasks should reflect the characteristic features of the problem area or the performed part of the IT project. Classes y_k, by which the classification objects X are distributed, should describe individual performers who are part of one

or more teams of IT project performers. The training set should contain data on already completed tasks (in particular, their names) and performers of these tasks.

Then the classification rule with minimal error (3.2) should be formulated as a rule for selecting the performer e_k with maximum posterior probability

$$e_k \propto \text{argmax}_k P(e_k) \prod_{i=1}^{n} P(\omega_i | e_k), \tag{3.7}$$

where $P(e_k)$ – the prior probability that the IT project task belongs to the performer e_k; $P(\omega_i|e_k)$ – the probability that the word ω_i from the IT project task title will be classified as being distributed to the performer e_k; n – the number of words in the IT project task title.

The prior $P(e_k)$ should be calculated by the formula

$$P(y_k) = \frac{T_{e_k}}{T}, \tag{3.8}$$

where T_{e_k} – the number of tasks in the training set of tasks performed by the performer e_k; T – the total number of tasks in the training set.

The probability $P(\omega_i|e_k)$ should be calculated using formula (3.3)

$$P(\omega_i | e_k) = \frac{N_{e_k w_i} + \alpha}{N_{e_k} + \alpha V}, \tag{3.9}$$

where $N_{e_k w_i}$ – the number of occurrences of the word ω_i in the names of the tasks of the training set performed by the performer e_k; N_{e_k} – the total number of words in the names of the tasks of the training set performed by the performer e_k; V – the size of the vocabulary (the total number of words in the training set); α – the smoothing parameter (allows to solve the problem of the appearance in the names of tasks of such words that have never been encountered in the training set). In our case, it is possible to assume that $\alpha = 1$.

But the considered results of adapting the classification rule (3.2) to the specifics of the task of assigning IT project tasks to its performers do not take into account the specifics of the existence of individual performers within the IT project performed by the company. In particular, the adapted classification rule (3.7) does not take into account:

– the condition of no excess of the total efforts of each performer e_k for one sprint;

– the condition of the performer e_k being in the team in a specific position during the planning and performing of the sprint, which can be determined by the competencies of this performer.

To take these conditions into account, an assumption is introduced according to which each individual performer e_k can be characterized by the parameter "Maximum effort of the performer in one sprint". This parameter can be formally represented as the indicator "Maximum allowable weight of class e_k" V_{maxk}. The value of the indicator V_{maxk} is measured in story points (Story Points) – conventional units of measurement for expressing the assessment of the total effort required for the full implementation of a product backlog element or any other part of the work [32]. For each performer e_k, the value of the indicator V_{maxk} is determined based on the position held by the performer e_k in the team according to their own competencies. Similarly, to describe each individual task of an IT project, the parameter "Task completion effort" has been introduced, which can be formally represented by the indicator "Task weight" v_i. The value of the indicator v_i is also measured in Story Points.

Then the classification rule (3.7) when assigning each j-th task of an IT project will work only if the condition is met

$$S_{vk} = \sum_{i=1}^{j} v_i \le V_{maxk}, \tag{3.10}$$

where S_{vk} – the total weight of the class e_k accumulated as a result of the successful assignment of the previous j – 1 tasks and the current j-th task of the IT project.

If this condition is violated, then the posterior probability of the IT project task belonging to the performer e_k should be considered equal to zero. The search for the performer e_k with the maximum posterior probability of the task belonging to it in this case should be continued.

The introduction of condition (3.10) led to the emergence of a new variant of solving the classification problem. In the derived variant, this problem was solved until all available classification objects were classified. In our case, an additional variant of solving the classification problem should be considered the variant in which the value of the total weight S_{vk} of each of the classes approached the value of the indicator V_{maxk} as much as possible. This condition means that each performer received the maximum possible number of IT project tasks to perform, which it can perform during the planned sprint.

Based on the considered results of adapting MNB to the specifics of the task of assigning IT project tasks to its performers, the algorithm for solving the classification problem using MNB was modified and shown in **Fig. 3.2**. The UML activity diagram describing the modified algorithm for solving the classification problem using the adapted MNB is shown in **Fig. 3.3**.

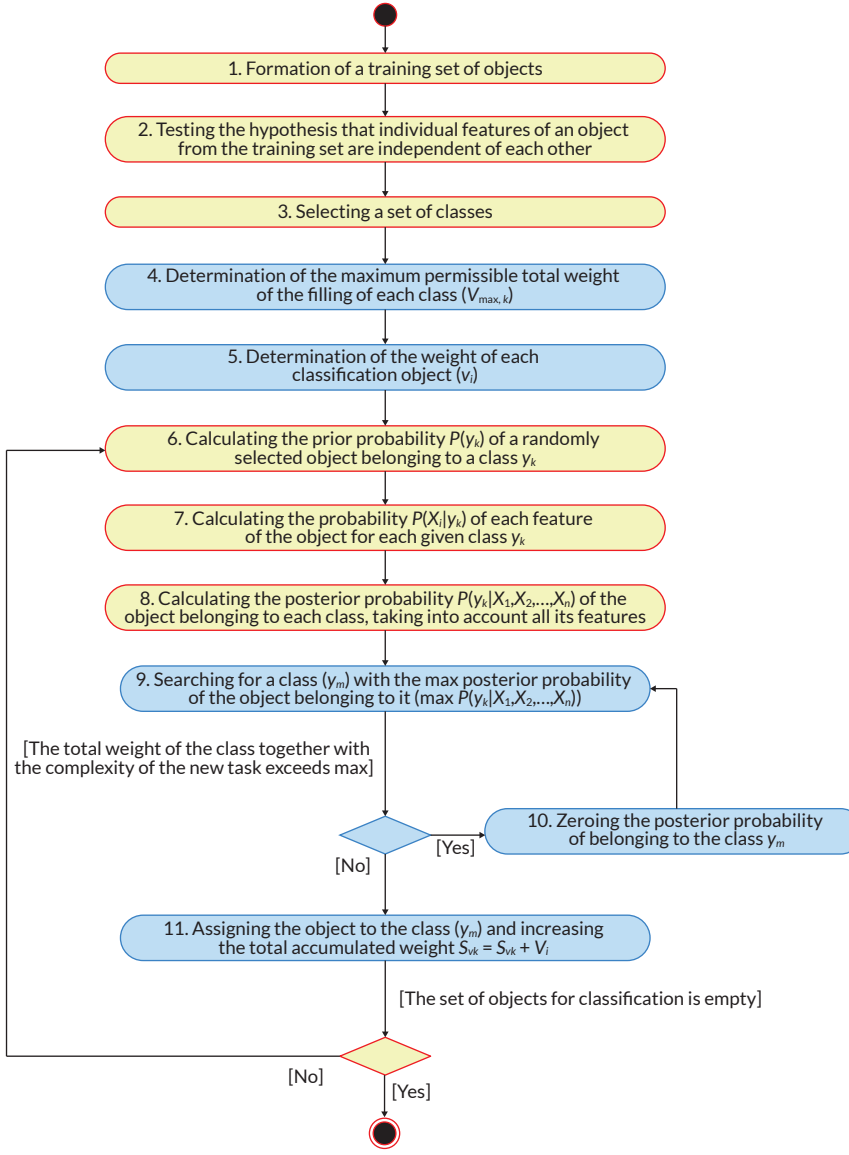

Fig. 3.3 UML activity diagram describing a modified algorithm for solving a classification problem using an adapted polynomial Bayesian classifier

3.4.2 Description of the results of developing an algorithm for implementing the developed method

The use of the modified algorithm allowed to develop a method for solving the problem of assigning IT project tasks to its performers as a sequence of such stages:

Stage 1 is the preparatory stage.

Stage 2 is the formation of the "Product Backlog" artifact as a set of IT project tasks.

Stage 3 is the formation of detailed descriptions of IT project tasks, the descriptions of which are included in the "Product Backlog" artifact.

Stage 4 is the assignment of IT project tasks from the "Product Backlog" artifact to performers. Formation of the "Sprint Backlog" artifact. Completion of the method.

But this representation of the developed method is too generalized. Therefore, for the successful implementation of this method, an appropriate algorithm was developed. The UML activity diagram, which describes the scheme of this algorithm, is shown in **Fig. 3.4**.

The use of the developed algorithm allowed to describe in detail the features of the implementation of each of the stages of the method as a sequence of steps of this algorithm. Let's consider these sequences of steps in more detail.

The implementation of Stage 1 of the developed method consists of a sequence of the following algorithm steps:

Step 1. Formation of the training set.

The implementation of Stage 2 of the developed method consists of a sequence of the following algorithm steps.

Step 2. Dividing the initiatives of the current IT project into epics.

Step 3. Distribution of epics between teams.

Step 4. Dividing epics into User Stories and Tasks.

Step 5. Adding Tasks to the task pool outside the epics.

The implementation of Stage 3 of the developed method consists of a sequence of the following algorithm steps.

Step 6. Determining the values of the attributes of User Stories and Tasks.

Step 7. Determining the maximum efforts of each performer.

Step 8. Sorting tasks by priority.

The implementation of Stage 4 of the developed method consists of a sequence of the following algorithm steps.

Step 9. If the "Product Backlog" artifact does not contain unassigned tasks, go to Step 17. Otherwise, select the next task from the "Product Backlog" artifact. If this task has the attribute of belonging to an epic, go to Step 10. Otherwise, go to Step 11.

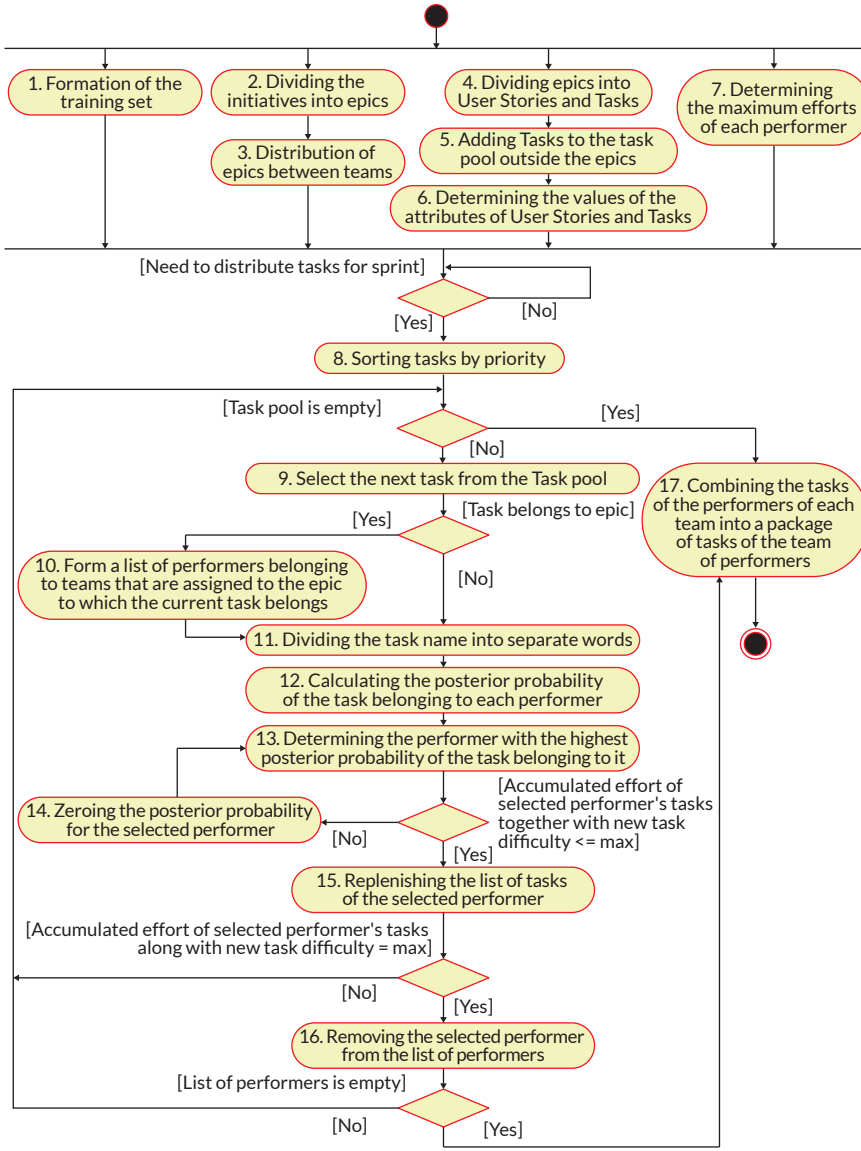

Fig. 3.4 UML activity diagram describing the algorithm for solving the problem of assigning IT project tasks to its performers

Step 10. Form a list of performers belonging to teams that are assigned to the epic to which the current task belongs.

Step 11. Dividing the task name into separate words.

Step 12. Calculating the posterior probability of the task belonging to each performer.

Step 13. Determining the performer with the highest posterior probability of the task belonging to it. Check the accumulated total weight of the performer S_{vk}. If this total weight does not exceed the value V_{maxk}, go to Step 15.

Step 14. Zeroing the posterior probability for the selected performer. Go to Step 13.

Step 15. Replenishing the list of tasks of the selected performer. Check the accumulated total weight of the performer S_{vk}. If this total weight is still less than V_{maxk}, go to Step 9.

Step 16. Removing the selected performer from the list of performers. If the list of performers of the planned sprint is not yet empty, go to Step 9. Otherwise, end the method.

Step 17. Combining the tasks of the performers of each team into a package of tasks of the team of performers. Form the artifact "Sprint Backlog".

End the method.

In Step 1 of the developed algorithm, data collection is first performed. It involves the accumulation of data on past tasks of those IT projects that are being performed, as well as on previously completed IT projects of the company. This data should include data on the performers to whom the tasks were assigned, and on the results of the work of these performers. This data must be prepared for further analysis and training of the adapted MNB before the first application of the developed method. When performing data collection, it is important to check the condition under which the number of tasks performed by different performers should be approximately the same. In the event that the use of the developed method is recognized as successful, this data can be updated after the completion of each individual IT project performed by the IT company.

After collecting historical data, it is necessary to test the hypothesis of the independence of individual words in the names of tasks. For this, it is proposed to use pairwise correlation analysis of each combination of two words of the name. Such an analysis can be done, for example, by counting how many times each pair of words occurs in the collected task names and comparing this number with the total number of word pairs in the collected task names. If the correlation percentage is lower than a priori set threshold, the correlation between these words can be considered absent.

For preprocessing of the collected data, it is necessary to remove stop words [18]. Stop words are words that do not add meaning to the phrase on their own, such as prepositions, articles, conjunctions, pronouns, etc.

After preprocessing, a training set is formed from the collected data. The size of the training set affects the accuracy of classification. To achieve the best results, the volume of test data should be 20–30%, and the remaining 70–80% should be training data [19]. However, it is advisable to determine the optimal size of the training set for a specific IT company and IT projects experimentally after obtaining several results using the developed method and algorithm. If among the performers of the IT project there is a new employee who has not yet worked in an IT company, then when forming the training set for this performer, it is advisable to add the names of tasks that correspond to its qualifications and specialization. These names can be obtained, for example, during the interview with such an employee or as a result of analyzing its resume.

The results of Step 1 of the algorithm are a repository of historical data and a training set prepared for preliminary training of the adapted MNB.

At Step 2, the following data must be collected:

– designation and/or name of the IT project that is planned for implementation;

– description of the set of initiatives of this IT project and each individual initiative from this set;

– description of the results of the decomposition of the IT project initiative into separate epics;

– description of each of the epics that are the result of this decomposition.

The division of initiatives into epics is usually carried out by project managers with the participation of product owners and the scrum master, who are responsible for planning and managing the IT project.

When forming epics, the project manager is guided by the principle of logical coherence, the size and complexity of the epic, as well as the relationships between the tasks of a single epic. Logical coherence aims to combine tasks that belong to the same functional area or have similar goals. This helps to avoid dispersion of efforts and ensures a more coherent implementation of the project. In terms of size and complexity, epics should be large enough to make their separation meaningful, but not so large that they become unmanageable.

The result of Step 2 of the algorithm is a data array that allows to form an information artifact "Product Backlog" as a description of the IT project initiative that is planned and/or being performed, and the epics to which this initiative is distributed.

In Step 3, it is necessary to collect the following data:

– a description of the epics planned for current performing;

– descriptions of the teams of potential IT project performers.

The distribution of epics is usually carried out by the product owner, with the participation of the scrum master, and in some cases (depending on the specifics of the IT project) and with the involvement of the teams of potential performers.

Epics are distributed in such a way that they correspond to the specialization and competencies of the team. When distributing epics, it is also necessary to take into account the dependencies between tasks in order to avoid situations where one team cannot complete its task due to dependence on the task of another team. An epic can be distributed to several teams at the same time if it contains tasks that require different competencies. In this case, the epic tasks are divided into separate subtasks, which are then distributed between the appropriate teams.

The result of Step 3 of the algorithm is a data array that describes the decision to involve individual teams of IT project performers in the performing of epics planned for current performing. The basis of this data array is the information artifact "Product Backlog", supplemented by descriptions of the teams of performers assigned to perform each epic and the individual performers that make up these teams.

At Step 4, it is necessary to collect the following data:

– a description of the epics planned for current performing;

– a set of descriptions of individual tasks that make up each planned epic;

– a description of the results of the decomposition of the IT project initiative into individual epics;

– an information artifact "Product Backlog" as a description of the set of IT project tasks planned for performing.

The breakdown of epics into individual IT project tasks (User Stories and Tasks) and a preliminary assessment of the efforts of these tasks are usually carried out by product owners with the participation of the scrum master, in close cooperation with the teams of performers. This step of the algorithm occurs during the planning of individual sprints or before the start of a new project phase.

The breakdown of epics into User Stories and Tasks is carried out according to the principle of logical sequence, effort and complexity. Epics are decomposed into individual tasks that are logically interconnected and can be performed independently of each other. This facilitates task management and their subsequent integration. When decomposing epics into individual tasks, it is necessary to constantly check whether the efforts of these tasks correspond to the planned maximum sprint effort. This allows to more accurately estimate the time and resources required to complete each task. The frequency of breaking epics into individual tasks of an IT project depends on the pace and specifics of this project. This breakdown can occur at regular planning meetings (for example, at the beginning of each sprint) or in cases where new epics appear that require distribution. This approach to breaking epics provides detailed work planning, facilitates the assessment of the efforts of individual tasks and allows to effectively manage the project.

The result of Step 4 of the algorithm is a data array that allows to form a "Product Backlog" information artifact as a detailed description of the IT project initiative, which contains:

– a description of the current IT project initiative that is planned and/or being performed;

– a description of the epics planned for current performing, to which this initiative is distributed;

– description of individual IT project tasks (User Stories and Tasks), into which epics planned for current performing are decomposed;

– descriptions of IT project teams involved in the performing of epics planned for current performing.

At Step 5, the following data must be collected:

– description of epics planned for current performing;

– a set of descriptions of individual tasks that make up each planned epic;

– a description of the results of the decomposition of the IT project initiative into individual epics;

– a set of descriptions of individual tasks that are not part of any epic;

– the "Product Backlog" information artifact as a description of the set of IT project tasks planned for performing.

Adding tasks that are not part of any epic to the "Product Backlog" artifact is usually done by the product owner, scrum masters or other responsible persons (the establishment of these persons depends on the team structure and IT project) mostly after discussing and agreeing on these tasks at the team level. Such tasks may appear, for example, as a result of detecting errors in already completed IT projects or if it is necessary to make changes to them at the request of the customer. This step is performed at planning meetings or planning of individual sprints in cases where new tasks appear that do not belong to existing epics.

Adding tasks that are not part of any epic allows not to lose important, but less global tasks that still need to be completed. Adding such tasks to the "Product Backlog" artifact ensures their inclusion in the general distribution and performing within the project.

The result of Step 5 of the algorithm is a data array that allows to specify the content of the "Product Backlog" information artifact as a detailed description of the IT project initiative, which includes:

– a description of the current IT project initiative that is planned and/or being implemented;

– a description of the epics planned for current performing, into which this initiative is distributed;

– a description of individual IT project tasks (User Stories and Tasks), into which the epics planned for current performing are decomposed;

– a description of individual IT project tasks (usually Tasks) that require current performing, but are not included in any of the epics planned for current performing;

– descriptions of the IT project teams involved in the performing of the epics planned for current performing.

At Step 6, the following data must be collected:

– a description of individual IT project tasks (User Stories and Tasks) planned for current performing;

– a set of tuples of attribute values that characterize each IT project task scheduled for current performing.

The determination of task attribute values can begin if there is a need for this, as well as on the eve of sprint planning. This step of the algorithm is performed by the product owner and scrum masters in interaction with the teams of performers assigned to perform the epics that include these tasks.

The attributes that characterize IT project tasks are the name, priority, evaluation of the performing effort, and the epic to which the task belongs. The values of these attributes significantly affect the performing of the modified classification task algorithm using the adapted MNB. Task priorities can be set based on their impact on the IT project, performing deadlines, and importance to the customer. These priorities are represented by numbers. The smaller the number, the higher the priority of the task.

Task effort is assessed in Story Points. Effort can range from minimum (1 Story Point) to maximum (13 Story Points). Typically, the following Story Points values are used for evaluation: 1, 3, 5, 8, and 13 [33].

The result of Step 6 of the algorithm is a data array that allows to specify the content of the "Product Backlog" information artifact as a detailed description of the IT project initiative, which includes:

– a description of individual IT project tasks (User Stories and Tasks) planned for current performing;

– a list of attribute values of each IT project task planned for current performing;

– descriptions of the teams of IT project performers involved in the performing of epics planned for current performing.

At Step 7, the following data must be collected for each performer:

– designation of the team to which this performer belongs;

– the level of the performer's competencies and the position that it occupies in the team according to these competencies;

– the standard of effort (in story points) for the planned sprint for the performer in the selected position.

This data can be collected by the scrum master for each performer of a subordinate team immediately before the start of a new iteration in the form of a sprint.

When setting this data, it is highly desirable to take into account such input data as the total number of sprint days, the number of performers, the number of holidays, the number of Story Points that were worked on during previous sprints. This additional data helps to plan the desired number of Story Points for each performer and thereby further plan the load of teams with tasks from the "Product Backlog" artifact in such a way as to maximally complete the tasks that have been taken into work.

The result of Step 7 of the algorithm is an array of normative data that allows to set the value of the V_{maxk} indicator for each performer of the IT project that is planned and performed.

At Step 8, it is necessary to collect the following data:

– a description of individual IT project tasks (User Stories and Tasks) planned for current performing;

– a list of priority values of each IT project task planned for current performing.

Typically, the sorting of IT project tasks by priority is carried out by the product owner, placing tasks in the "Product Backlog" artifact by their priorities from the highest (minimum number) to the lowest (maximum number) value of the "Task Priority" attribute.

Sorting tasks in the "Product Backlog" artifact by priorities allows to arrange these tasks in such a way that the most important of them are planned for performing first. This helps to optimize resource costs and increase the efficiency of IT project performing.

The result of performing Step 8 of the algorithm is a data array that contains the "Product Backlog" information artifact, ordered by task priority values, as a list of individual IT project tasks scheduled for current performing.

In fact, Steps 9–17 of the algorithm for implementing the method for solving the problem of assigning IT project tasks to their performers repeat the steps of the modified algorithm for solving the classification problem using the adapted MNB shown in **Fig. 3.3**. Therefore, these features of the performing of these steps are not considered in detail here.

3.5 Experimental verification of the improved method

It was decided to conduct an experimental verification of the obtained results in one of the outsourcing IT companies of Ukraine. The selected IT company is characterized by a significant number of implemented IT projects and a variety of their

tasks (from software development to cybersecurity). The company has multidisciplinary teams consisting of specialists from different areas: developers (Frontend, Backend), DevOps engineers, QA engineers, BI engineers, etc. These teams can be either permanent or formed for the duration of specific projects. The company also has an accumulated database of previous IT projects, tasks and results of their implementation, which is important for training and tuning the modified algorithm for solving the classification problem.

For experimental verification of the obtained results, two teams of performers (symbols "A" and "B") were selected in the selected IT company. Team A consists of two people, team B consists of three people. The competencies of these individuals are characterized by one of three possible levels of gradation: junior, middle and senior. A description of these teams is given in **Table 3.1**.

Table 3.1 Characteristics of performers

Team	Performer	Gradation level	Max effort (SP)
A	Ivan Petrov	Junior	20
	Oleksiy Kovalek	Middle	35
B	Maria Ivanova	Senior	50
	Gana Sergienko	Middle	35
	Dmytro Orlov	Junior	20

Table 3.1 shows the level of gradation and maximum effort standard, measured in Story Points, for each performer.

During Stage 1 of the developed method, a training set was formed for training the adapted MNB based on historical data on successfully completed tasks.

Information on successfully completed tasks, on the basis of which the training set was built, is given in **Table 3.2**.

From **Table 3.2** it can be seen that the total number of tasks in the training set $T = 20$.

The names of the completed IT project tasks given in **Table 3.2** were tokenized – divided into separate words (tokens) with the exception of conjunctions and prepositions. The number of occurrences of individual tokens in the names of the tasks of each of the performers is given in **Table 3.3**.

As a result of the implementation of Stage 2 and Stage 3 of the developed method, the "Product Backlog" information artifact was formed as a detailed list of IT project tasks, ordered by priority values, from epics, for the implementation of which teams A and B were assigned. The description of this artifact is given in **Table 3.4**.

Table 3.2 List of completed tasks and their performers

No.	Task	Task performer (gradation level)
1	Frontend Design Update	Ivan Petrov (Junior)
2	UX/UI Improvement	Ivan Petrov (Junior)
3	Frontend Framework Update	Ivan Petrov (Junior)
4	Backend API Integration	Oleksiy Kovalek (Middle)
5	Web Page Load Optimization	Oleksiy Kovalek (Middle)
6	Business Intelligence Report Creation	Oleksiy Kovalek (Middle)
7	Backend Data Processing Optimization	Oleksiy Kovalek (Middle)
8	Real-time Data Analytics Dashboard	Oleksiy Kovalek (Middle)
9	Database Performance Tuning	Maria Ivanova (Senior)
10	User Authentication Implementation	Maria Ivanova (Senior)
11	Data Encryption Setup	Maria Ivanova (Senior)
12	Data Migration from Legacy Systems	Maria Ivanova (Senior)
13	DevOps Pipeline Automation	Maria Ivanova (Senior)
14	Security Vulnerability Assessment	Gana Sergienko (Middle)
15	Cloud Resource Allocation	Gana Sergienko (Middle)
16	Continuous Deployment Setup	Gana Sergienko (Middle)
17	Network Configuration Management	Gana Sergienko (Middle)
18	Mobile App Bug Fixing	Dmytro Orlov (Junior)
19	Automated Testing Script Development	Dmytro Orlov (Junior)
20	System Backup Configuration	Dmytro Orlov (Junior)

Table 3.3 Number of occurrences of tokens in the names of the tasks of each of the performers

No.	Word (token)	Ivan Petrov (Junior)	Oleksiy Kovalek (Middle)	Maria Ivanova (Senior)	Gana Sergienko (Middle)	Dmytro Orlov (Junior)
1	2	3	4	5	6	7
1	Allocation	0	0	0	1	0
2	Analytics	0	1	0	0	0
3	API	0	1	0	0	0
4	App	0	0	0	0	1
5	Assessment	0	0	0	1	0
6	Authentication	0	0	1	0	0
7	Automated	0	0	0	0	1
8	Automation	0	0	1	0	0
9	Backend	0	2	0	0	0

Continuation of Table 3.3

1	2	3	4	5	6	7
10	Backup	0	0	0	0	1
11	Bug	0	0	0	0	1
12	Business	0	1	0	0	0
13	Cloud	0	0	0	1	0
14	Configuration	0	0	0	1	1
15	Continuous	0	0	0	1	0
16	Creation	0	1	0	0	0
17	Dashboard	0	1	0	0	0
18	Data	0	2	2	0	0
19	Database	0	0	1	0	0
20	Deployment	0	0	0	1	0
21	Design	1	0	0	0	0
22	Development	0	0	0	0	1
23	DevOps	0	0	1	0	0
24	Encryption	0	0	1	0	0
25	Fixing	0	0	0	0	1
26	Framework	1	0	0	0	0
27	Frontend	2	0	0	0	0
28	Implementation	0	0	1	0	0
29	Improvement	1	0	0	0	0
30	Integration	0	1	0	0	0
31	Intelligence	0	1	0	0	0
32	Legacy	0	0	1	0	0
33	Load	0	1	0	0	0
34	Management	0	0	0	1	0
35	Migration	0	0	1	0	0
36	Mobile	0	0	0	0	1
37	Network	0	0	0	1	0
38	Optimization	0	2	0	0	0
39	Page	0	1	0	0	0
40	Performance	0	0	1	0	0
41	Pipeline	0	0	1	0	0
42	Processing	0	1	0	0	0
43	Real-time	0	1	0	0	0
44	Report	0	1	0	0	0

Continuation of Table 3.3

1	2	3	4	5	6	7
45	Resource	0	0	0	1	0
46	Script	0	0	0	0	1
47	Security	0	0	0	1	0
48	Setup	0	0	1	1	0
49	System	0	0	0	0	1
50	Systems	0	0	1	0	0
51	Testing	0	0	0	0	1
52	Tuning	0	0	1	0	0
53	Update	2	0	0	0	0
54	User	0	0	1	0	0
55	UX/UI	1	0	0	0	0
56	Vulnerability	0	0	0	1	0
57	Web	0	1	0	0	0

Table 3.4 List of IT project tasks, ordered by priority values, planned for current performing

No.	Task name	Priority	Effort (SP)	Epic ID
1	Backend API performance tuning	1	8	E2
2	Mobile app security audit	1	8	E3
3	Web page speed optimization	1	8	E5
4	Cloud infrastructure setup	1	13	E6
5	Network configuration automation	1	8	E9
6	Business intelligence dashboard creation	2	13	E8
7	Backend service integration	2	8	E2
8	Continuous integration pipeline setup	2	5	E7
9	System backup strategy implementation	2	5	E10
10	Automated testing suite development	2	8	–
11	Frontend Feature Extension	2	8	E1
12	Frontend design enhancement	2	5	E1
13	Database schema optimization	2	5	E4
14	Real-time data analysis implementation	2	5	–
15	Mobile app testing	2	5	E3
16	Data encryption security setup	3	3	E11
17	User interface design update	3	3	E1
18	Cloud service migration	3	13	E6
19	Machine learning model training	3	13	E12

From **Table 3.4** it can be seen that the total number of IT project tasks planned for the current performing is 19, of which two tasks (No. 10 and No. 14) are tasks that do not belong to any of the epics and are assigned additionally.

The progress of Stage 4 of the developed method (the actual solution of the problem of assigning IT project tasks to their performers) was proposed to be considered using the example of its iteration for the task "Frontend design enhancement" and Ivan Petrov (Junior) as its possible performer. Before the start of this iteration, the names of the IT project tasks given in **Table 3.4** were also tokenized.

The parameters for calculating the value of the classification rule (3.7), which establishes the posterior probability of the performance of the task "Frontend design enhancement" by the performer Ivan Petrov, are given in **Table 3.5**.

Table 3.5 Parameters for calculating the value of the classification rule (3.7)

Parameter	Value
General parameters	
T_{e_k}	3
T	20
N_{e_k}	8
α	1
V	57
"Frontend" token	
$N_{e_k w_l}$	2
"Design" token	
$N_{e_k w_l}$	1
"Enhancement" token	
$N_{e_k w_l}$	0 (missing from the training set)

Then the value of the prior probability Pe_k according to formula (3.8) is

$$P(\text{"}Ivan\,Petrov\text{"}) = \frac{3}{20} = 0.15.$$

The value of the probability $P(\text{"}Frontend\text{"}|\text{"}Ivan\ Petrov\text{"})$ according to formula (3.9) is

$$P(\text{"}Frontend\text{"}|\text{"}Ivan\,Petrov\text{"}) = \frac{2+1}{8+1\times 57} = \frac{3}{65} = 0.046154.$$

The value of the probability *P*("*Design*"|"*Ivan Petrov*") according to formula (3.9) is

$$P(\text{"Design"}|\text{"Ivan Petrov"}) = \frac{1+1}{8+1\times 57} = \frac{2}{65} = 0.03077.$$

The value of the probability *P*("*Enhancement*"|"*Ivan Petrov*") according to formula (3.9) is

$$P(\text{"Enhancement"}|\text{"Ivan Petrov"}) = \frac{0+1}{8+1\times 51} = \frac{3}{65} = 0.015385.$$

The value of the posterior probability according to formula (3.7) is

$$P(e_k)\prod_{i=1}^{n} P(\omega_i|e_k) = 0.15\times 0.046154\times 0.03077\times 0.015385 =$$

$$= 0.000003277370962995 \approx 3.3\times 10^{-6}.$$

The results of the calculations of the posterior probability for each task and performer are given in **Table 3.6**.

Table 3.6 Results of the calculations of the posterior probability for each task and performer

Task No.	Ivan Petrov (Junior)	Oleksiy Kovalek (Middle)	Maria Ivanova (Senior)	Gana Sergienko (Middle)	Dmytro Orlov (Junior)
1	2	3	4	5	6
1	8.4×10^{-9}	3.48×10^{-8}	3.52×10^{-8}	7.44×10^{-9}	6.62×10^{-9}
2	8.4×10^{-9}	5.81×10^{-9}	8.8×10^{-9}	1.49×10^{-8}	2.65×10^{-8}
3	8.4×10^{-9}	6.97×10^{-8}	8.8×10^{-9}	7.44×10^{-9}	6.62×10^{-9}
4	5.46×10^{-7}	4.7×10^{-7}	1.29×10^{-6}	2.14×10^{-6}	4.57×10^{-7}
5	5.46×10^{-7}	4.7×10^{-7}	1.29×10^{-6}	2.14×10^{-6}	9.13×10^{-7}
6	8.4×10^{-9}	9.29×10^{-8}	8.8×10^{-9}	7.44×10^{-9}	6.62×10^{-9}
7	5.46×10^{-7}	2.82×10^{-6}	6.43×10^{-7}	5.36×10^{-7}	4.57×10^{-7}
8	8.4×10^{-9}	1.16×10^{-8}	3.52×10^{-8}	2.98×10^{-8}	6.62×10^{-9}
9	8.4×10^{-9}	5.81×10^{-9}	1.76×10^{-8}	7.44×10^{-9}	2.65×10^{-8}
10	8.4×10^{-9}	5.81×10^{-9}	8.8×10^{-9}	7.44×10^{-9}	5.29×10^{-8}
11	1.64×10^{-6}	4.7×10^{-7}	6.43×10^{-7}	5.36×10^{-7}	4.57×10^{-7}
12	3.3×10^{-6}	4.7×10^{-7}	6.43×10^{-7}	5.36×10^{-7}	4.57×10^{-7}
13	5.46×10^{-7}	1.41×10^{-6}	1.29×10^{-6}	5.36×10^{-7}	4.57×10^{-7}

Continuation of Table 3.6

1	2	3	4	5	6
14	8.4×10^{-9}	3.48×10^{-8}	5.28×10^{-8}	7.44×10^{-9}	6.62×10^{-9}
15	5.46×10^{-7}	4.7×10^{-7}	6.43×10^{-7}	5.36×10^{-7}	3.65×10^{-6}
16	8.4×10^{-9}	1.74×10^{-8}	1.06×10^{-7}	2.98×10^{-8}	6.62×10^{-9}
17	5.04×10^{-8}	5.81×10^{-9}	1.76×10^{-8}	7.44×10^{-9}	6.62×10^{-9}
18	5.46×10^{-7}	4.7×10^{-7}	1.29×10^{-6}	1.07×10^{-6}	4.57×10^{-7}
19	8.4×10^{-9}	5.81×10^{-9}	8.8×10^{-9}	7.44×10^{-9}	6.62×10^{-9}

The results of applying logarithmic transformation (based on the decimal logarithm) for better perception of the classification results according to formula (3.3) are given in **Table 3.7**.

Table 3.7 Results of applying logarithmic transformation of posterior probability calculations for each task and performer

Task No.	Ivan Petrov (Junior)	Oleksiy Kovalek (Middle)	Maria Ivanova (Senior)	Gana Sergienko (Middle)	Dmytro Orlov (Junior)
1	−8.076	−7.46	−7.45	−8.13	−8.18
2	−8.076	−8.24	−8.06	−7.83	−7.58
3	−8.076	−7.16	−8.06	−8.13	−8.18
4	−6.26	−6.33	−5.89	−5.67	−6.34
5	−6.26	−6.33	−5.89	−5.67	−6.04
6	−8.076	−7.03	−8.06	−8.13	−8.18
7	−6.26	−5.55	−6.19	−6.27	−6.34
8	−8.076	−7.94	−7.45	−7.53	−8.18
9	−8.076	−8.24	−7.75	−8.13	−7.58
10	−8.076	−8.24	−8.06	−8.13	−7.28
11	−5.79	−6.33	−6.19	−6.27	−6.34
12	−5.48	−6.33	−6.19	−6.27	−6.34
13	−6.26	−5.85	−5.89	−6.27	−6.34
14	−8.076	−7.46	−7.28	−8.13	−8.18
15	−6.26	−6.33	−6.19	−6.27	−5.44
16	−8.076	−7.76	−6.98	−7.53	−8.18
17	−7.3	−8.24	−7.75	−8.13	−8.18
18	−6.26	−6.33	−5.89	−5.97	−6.34
19	−8.076	−8.24	−8.06	−8.13	−8.18

The results of the IT project task distribution for the performers from teams A and B are presented in **Table 3.8**.

They show that none of the performers reached their maximum workload. This is probably due to the lack of tasks with a complexity that would exactly match the remaining maximum workload capabilities of the performers.

Table 3.8 Task distribution results

User Stories & Task	The complexity (SP)	Performer	Max effort (SP)	Remain-der SP
11. Frontend Feature Extension	8	Ivan Petrov (Junior)	20	4
12. Frontend design enhancement	5			
17. User interface design update	3			
3. Web page speed optimization	8	Oleksiy Kovalek (Middle)	35	1
6. Business intelligence dashboard creation	13			
7. Backend service integration	8			
13. Database schema optimization	5			
1. Backend API performance tuning	8	Maria Ivanova (Senior)	50	8
8. Continuous integration pipeline setup	5			
10. Automated testing suite development	5			
14. Real-time data analysis implementation	5			
16. Data encryption security setup	3			
18. Cloud service migration	13			
4. Cloud infrastructure setup	13	Gana Sergienko (Middle)	35	1
5. Network configuration automation	8			
19. Machine learning model training	13			
2. Mobile app security audit	8	Dmytro Orlov (Junior)	20	2
9. System backup strategy implementation	5			
15. Mobile app testing	5			

In the case of assignment of task No. 10, the total weight of assigned tasks for the performer Dmytro Orlov exceeded his maximum allowable weight. Therefore, it was decided to transfer this task to the next performer with the highest posterior probability. This performer turned out to be Maria Ivanova.

In the case of assignment of task No. 19, the total weight of assigned tasks for the performer Maria Ivanova exceeded her maximum allowable weight. Therefore, it was decided to transfer this task to the next performer with the highest posterior

probability. This performer turned out to be Ivan Petrov. But in the case of assignment of task No. 19 to him, the total weight of assigned tasks also exceeded his maximum allowable weight. Therefore, it was decided to transfer task No. 19 to the next performer. This performer turned out to be Gana Sergienko.

After distributing tasks among the performers of the IT project, task packages were formed for each team of performers by combining tasks that were distributed to performers of the same team. The formed task packages of each team are given in **Table 3.9**.

Table 3.9 Results of using the developed method

Team	Task
A	3. Web page speed optimization
	6. Business intelligence dashboard creation
	7. Backend service integration
	11. Frontend Feature Extension
	12. Frontend design enhancement
	13. Database schema optimization
	17. User interface design update
B	1. Backend API performance tuning
	2. Mobile app security audit
	4. Cloud infrastructure setup
	5. Network configuration automation
	8. Continuous integration pipeline setup
	9. System backup strategy implementation
	10. Automated testing suite development
	14. Real-time data analysis implementation
	15. Mobile app testing
	16. Data encryption security setup
	18. Cloud service migration
	19. Machine learning model training

3.6 Discussion of the research results

As a result of the study, a method was developed for automatically solving the problem of assigning IT project tasks to its performers. The developed method is based on the presentation of the problem of assigning IT project tasks to its

performers as a type of classification problem. This presentation made it possible to propose using MNB to solve this problem. This version of the classifier is quite simple to implement and allows for good processing of sets of attribute data, which are descriptions of IT project tasks.

During the study, the existing version of MNB was adapted to the features of the problem of assigning IT project tasks to its performers. In particular, the classification rule with minimal error (3.2) as a result of adaptation to the features of this problem took the form (3.7). The results of adapting the methods for calculating the values of the elements of the classification rule (3.7) are given in the form of expressions (3.8) and (3.9). To take into account the limitations that arise as a result of the IT project performer being in the team in a specific position, it was proposed to add condition (3.10) to the classification rule (3.7) of the possibility of assigning a task to a specific performer.

The results of adapting MNB to the specifics of the task of assigning IT project tasks to its performers were used as the basis for modifying the algorithm for solving the classification problem. The result of this modification in the form of an activity diagram in the UML language is shown in **Fig. 3.3**.

Based on the results obtained, a method was developed for automatically solving the problem of assigning IT project tasks to its performers. Using this method makes it possible to exclude human participation in the process of directly solving the problem of assigning IT project tasks to its performers (in the process of performing Stage 4 of the method). This makes it possible to obtain a solution to this problem in a sufficiently short period of time and reduce the time spent on planning individual sprints (iterations) of the IT project.

Another significant advantage of the developed method is a fairly high level of objectivity of the results of task distribution. The use of a modified algorithm for solving the classification problem based on the adapted MNB allows making decisions based on statistical data and historical information, which minimizes the possibility of subjective errors and bias. This ensures transparency of the task distribution process, which in turn increases trust on the part of the performers and helps to increase their motivation.

Unlike the solution to this problem described in [15], the proposed method allows at least partially to take into account the experience and competence of the performers of the IT project. The experience of the performers when solving the problem is taken into account by selecting for a specific performer those tasks whose names largely coincide with the names of tasks successfully completed by this performer earlier. The competence of the performers when solving the problem is taken into account by assigning to each specific performer their own value of the

indicator "Maximum permissible weight of class e_k" V_{maxk}. This value is set based on the position held by the e_k performer in the team based on their own competencies.

The use of MNB to solve the problem of assigning IT project tasks to its performers made it possible, in contrast to the solution described in [16], to abandon vector descriptions of IT project tasks and the calculation of the cosine similarity measure. The calculation of the scalar elements of the classification rule (3.7) and condition (3.10) requires less computational resources and requires less time to calculate the result of solving the problem.

The main limitation of the application of the developed method is the constant need to maintain the historical data repository, which is used to train the adapted MNB, in an up-to-date state. This data should be collected by analyzing previous IT projects, assessing the success of individual tasks, and providing feedback from the teams of performers. Regular updating and validation of historical data are critically important for ensuring the accuracy and relevance of the results of solving the problem of assigning IT project tasks to its performers.

Among the limitations and disadvantages of the application of the algorithm for implementing the developed method, it is also worth noting some difficulties with the classification of tasks that have similar names, but differ in content. This can lead to erroneous distribution of tasks between individual performers.

To improve the accuracy of classification, it is necessary to use a larger amount of historical data for training the model. Regular updating and expansion of the keyword dictionary will help the algorithm better understand the context of the tasks. Therefore, work on the creation and development of a specialized thesaurus, on the basis of which the training set for MNB should be formed, is one of the promising areas of further research on this issue.

Another area of further research into solving the problem is the use of modern artificial intelligence tools to increase the accuracy and objectivity of solving the problem of assigning IT project tasks to its performers. In particular, it is about the possibility and feasibility of using lemmatization for pre-processing text names of individual IT project tasks. Although one of the authors of this study has conducted work in this direction [34], it requires further development for successful use in the field of ongoing management of IT projects and their performers.

3.7 Conclusions

As a result of the research, a method for solving the problem of assigning IT project tasks to its performers was developed. The use of this method requires human

participation only to perform operations in preparation for the distribution of IT project tasks among the members of the team of performers. The actual solution to the problem of assigning IT project tasks to its performers using the developed method is carried out without the participation of stakeholders of the IT project that is planned and/or being performed.

During the development of the method, MNB was adapted to the specifics of the problem of assigning IT project tasks to its performers. As a result of this adaptation, the classification rule with a minimum error (3.7) was clarified by introducing an additional condition (3.10). This condition allows taking into account the level of competence of the IT project performer when solving the classification problem as the maximum permissible efforts of the tasks assigned to it for performing. The features of calculating the elements of the classification rule (expressions (3.8) and (3.9)) were also clarified. This made it possible to adapt the classification rule, proven in practice, to the task of assigning IT project tasks to its performers and to modify the algorithm for solving the classification problem accordingly.

Based on the results obtained, a general description of the method for solving the task of assigning IT project tasks to its performers was developed. For a detailed description of the content of individual stages, an algorithm for implementing this method was developed. The diagram of this algorithm shown in **Fig. 3.3** and the proposed descriptions of the main steps of this algorithm determine the features of its implementation both as a methodology for applying the obtained solutions in the current management of an IT project and as a specialized information technology.

To verify the operability of the obtained results, an experimental test of the method and its implementation algorithm was conducted during the management of one of the IT projects of an outsourcing IT company. The course of using the method and the main results of the implementation of its individual stages are presented. These results include:

– determination of teams of IT project performers and the composition of these teams, taking into account the level of gradation of individual performers;

– formed training set and results of its tokenization;

– "Product Backlog" information artifact as a detailed list of IT project tasks, ordered by the values of their performing priorities;

– example of iterations of Stage 4 of the developed method;

– results of calculating the values of the classification rule for all tasks and IT project performers;

– results of distributing IT project tasks to performers from selected teams;

– "Sprint Backlog" information artifacts as task packages for each team of performers.

The results of experimental verification indicate the feasibility of using the developed method to solve the problem of assigning IT project tasks to its performers. The developed method contributes to better project planning, minimizes administrative burden and helps to avoid delays and errors in project performing. The method is easily adapted to different projects and teams due to the ability to adjust the algorithm parameters in accordance with specific features.

The authors consider it advisable to continue the research to conduct a comparative analysis of the results of using different text classification algorithms to solve the problem of assigning IT project tasks to its performers.

Conflict of interest statement

The authors declare that there is no conflict of interest in relation to this paper, as well as the published research results, including the financial aspects of conducting the research, obtaining and using its results, as well as any non-financial personal relationships.

Use of artificial intelligence statement

The authors declare that they did not use artificial intelligence tools in preparing this manuscript.

References

1. Agrawal, M., Chari, K. (2007). Software Effort, Quality, and Cycle Time: A Study of CMM Level 5 Projects. IEEE Transactions on Software Engineering, 33 (3), 145–156. https://doi.org/10.1109/tse.2007.29
2. Chilton, M. A. (2022). Resource allocation in IT projects: using schedule optimization. International Journal of Information Systems and Project Management, 2 (3), 47–59. https://doi.org/10.12821/ijispm020303
3. Herroelen, W., Leus, R. (2005). Identification and illumination of popular misconceptions about project scheduling and time buffering in a resource-constrained environment. Journal of the Operational Research Society, 56 (1), 102–109. https://doi.org/10.1057/palgrave.jors.2601813

4. Kolisch, R. (1996). Serial and parallel resource-constrained project scheduling methods revisited: Theory and computation. European Journal of Operational Research, 90 (2), 320–333. https://doi.org/10.1016/0377-2217(95)00357-6
5. A guide to the project management body of knowledge (PMBOK guide) (2017). Project Management Institute, Inc.
6. Nastanova do zvodu znan z upravlinnia proiektamy. Nastanova PMBOK (2021). Project Management Institute, Inc. Available at: https://learn.ztu.edu.ua/pluginfile.php/274061/mod_resource/content/1/PMBOK7_Ukr_ForPersonalUseOnly.pdf
7. Ruiz, S., Escudero, D., Cervantes, J., Trueba, A.; Figueroa-García, J., López-Santana, E., Villa-Ramírez, J., Ferro-Escobar, R. (Eds.) (2017). Assigning-Tasks Method for Developers in Software Projects Using up Similarity Coefficients. Applied Computer Sciences in Engineering. Springer, 119–128. https://doi.org/10.1007/978-3-319-66963-2_12
8. Ievlanov, M. V., Pogorelaya, N. I. (2012). Planning for the use of staff in the works of IT-project. Eastern-European Journal of Enterprise Technologies, 2 (4 (56), 22–26. https://doi.org/10.15587/1729-4061.2012.3706
9. Martínez-Rojas, M., Soto-Hidalgo, J. M., Marín, N., Vila, M. A. (2018). Using Classification Techniques for Assigning Work Descriptions to Task Groups on the Basis of Construction Vocabulary. Computer-Aided Civil and Infrastructure Engineering, 33 (11), 966–981. https://doi.org/10.1111/mice.12382
10. Schnabel, A., Kellenbrink, C., Helber, S. (2018). Profit-oriented scheduling of resource-constrained projects with flexible capacity constraints. Business Research, 11 (2), 329–356. https://doi.org/10.1007/s40685-018-0063-5
11. Skiena, S. S. (2020). The Algorithm Design Manual. Texts in Computer Science. Springer International Publishing. https://doi.org/10.1007/978-3-030-54256-6
12. Vera-Rivera, F. H., Barbosa-Mora, J. L., Gaona-Cuevas, C. M. (2020). Generación automática de la planificación de la entrega en desarrollo de software agil, asignación de historias de usuario a los desarrolladores usando algoritmos genéticos. AiBi Revista de Investigación, Administración e Ingeniería, 8 (2), 29–38. https://doi.org/10.15649/2346030x.735
13. Homwiseswongsa, A., Ratanavilisagul, C. (2023). Modified Differential Evolution Algorithm for Solving Multi-Skill Resource-Constrained Project Scheduling Problem. 2023 15th International Conference on Information Technology and Electrical Engineering (ICITEE). Chiang Mai, 1–6. https://doi.org/10.1109/icitee59582.2023.10317769

14. Xu, H., Kuchansky, A., Biloshchytska, S., Tsiutsiura, M. (2021). A Conceptual Research Model for the Partner Selection Problem. 2021 IEEE International Conference on Smart Information Systems and Technologies (SIST). Nur-Sultan, 1–6. https://doi.org/10.1109/sist50301.2021.9465931
15. Druzhynin, V., Gladka, M., Borysenko, I., Hladkyi, Ya., Lisnevskyi, R. (2024). A Model for Allocating Labor Resources to Project Work Based on Task Prioritization. Information Technology and Implementation Workshop 2024: IT Infrastructure and Applied Solutions, IT and I-WS 2024: ITIAS. CEUR Workshop Proceedings, 3955, 42–54.
16. Arslan, H., Işik, Y., Görmez, Y., Temiz, M. (2024). Machine learning and text mining based real-time semi-autonomous staff assignment system. Computer Science and Information Systems, 21 (1), 75–94. https://doi.org/10.2298/csis220922065a
17. Mitchell, T. M. (1997). Machine Learning. McGraw-Hill, 414.
18. Manning, C. D., Raghavan, P., Schuetze, H. (2008). Introduction to Information Retrieval. Cambridge University Press. https://doi.org/10.1017/cbo9780511809071
19. Gholamy, A., Kreinovich, V., Kosheleva, O. (2018). Why 70/30 or 80/20 Relation between Training and Testing Sets: A Pedagogical Explanation. International Journal of Intelligent Technologies and Applied Statistics, 11 (2), 105–111. https://doi.org/10.6148/IJITAS.201806_11(2).0003
20. Tan, H. (2021). Machine Learning Algorithm for Classification. Journal of Physics: Conference Series, 1994 (1), 012016. https://doi.org/10.1088/1742-6596/1994/1/012016
21. Kulkarni, A., Brown III, L. L. (2019). Phishing Websites Detection using Machine Learning. International Journal of Advanced Computer Science and Applications, 10 (7), 8–13. https://doi.org/10.14569/ijacsa.2019.0100702
22. Rennie, J. D., Shih, L., Teevan, J., Karger, D. R. (2003). Tackling the poor assumptions of naive bayes text classifiers. ICML'03: Proceedings of the Twentieth International Conference on International Conference on Machine Learning, 3, 616–623. Available at: https://dl.acm.org/doi/10.5555/3041838.3041916
23. ISO/IEC/IEEE Standard No 15288:2015 (2015). Systems and software engineering – System life cycle processes. ISO/IEC/IEEE International Standard. https://doi.org/10.1109/IEEESTD.2015.7106435
24. Schwaber, K., Sutherland, J. (2020). The Scrum Guide. The Definitive Guide to Scrum: The Rules of the Game. Available at: https://scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-US.pdf

25. Sutherland, J. (2014). Scrum: A revolutionary approach to building teams, beating deadlines and boosting productivity. Random House, 248.
26. Jarzębowicz, A., Sitko, N. (2020). Agile Requirements Prioritization in Practice: Results of an Industrial Survey. Procedia Computer Science, 176, 3446–3455. https://doi.org/10.1016/j.procs.2020.09.052
27. What is Product Owner? Agile Alliance. Available at: https://www.agilealliance.org/glossary/product-owner/
28. Yang, A. (2023). Guide to building a product roadmap (with template and examples). LogRocket. Available at: https://blog.logrocket.com/product-manage-m2ent/product-roadmap-template-examples/
29. Radigan, D. Product backlog: tips for creation and prioritization. Atlassian. Available at: https://www.atlassian.com/agile/scrum/backlogs
30. McCallum, A., Nigam, K. (1998). A Comparison of Event Models for Naive Bayes Text Classification. AAAI/ICML-98 Workshop on Learning for Text Categorization, p Technical Report WS-98-05. AAAI Press, 41–48.
31. Iparraguirre-Villanueva, O., Melgarejo-Graciano, M., Castro-Leon, G., Olaya-Cotera, S., Ruiz-Alvarado, J., Epifanía-Huerta, A. et al. (2023). Classification of Tweets Related to Natural Disasters Using Machine Learning Algorithms. International Journal of Interactive Mobile Technologies (IJIM), 17 (14), 144–162. https://doi.org/10.3991/ijim.v17i14.39907
32. Cohn, M. (2023). What Are Agile Story Points? Mountain Goat Software. Available at: https://www.mountaingoatsoftware.com/blog/what-are-story-points
33. Planning (Scrum) Poker (2023). QATestLab. Available at: https://training.qatestlab.com/blog/technical-articles/planning-poker/
34. Ievlanov, M. V., Moroz, B. I., Moroz, D. M., Luchytskyi, V. V. (2024). Information technology for identifying terms and project artifacts in the requirements for the information system. Management Information System and Devises, 182, 73–93. https://doi.org/10.30837/0135-1710.2024.182.073