Applying artificial intelligence for optimal production scheduling and phase design in open pit mining

V Loor University of Chile, AMTC, Chile

N Morales University of Chile, AMTC, Chile

Abstract

The open-pit mine production scheduling (OPPS) problem aims to determine the extraction sequence of mining blocks of an orebody. The OPPS presents several restrictions that create a combinatorial optimization problem classified as NP-hard. Generally, an optimal solution for OPPS cannot be obtained in an acceptable computation time using linear programming; therefore, approximation methods called heuristics have been used to solve it. In this paper, an artificial intelligence (AI) based methodology is proposed to obtain operative pushbacks in open-pit mines respecting operational and design constraints. This integrated approach is achieved through a Genetic Algorithm and a clustering algorithm (k-means). A Genetic Algorithm is a search heuristic inspired by Charles Darwin's theory of natural evolution and is used to solve NP-hard problems. This methodology has been tested in an iron mine and a gold mine and has been shown to be a practical, viable approach. Results show that pushbacks obtained respect the design and operational constraints of pit extraction, while also maximizing the net present value (NPV).

1   Introduction

As the life of open-pit mines can span several decades, the optimization of the design and production plan of a project is critical to maximizing its value. This strategic problem can be seen as composed of two components: a spatial component, which determines portions or volumes to organize extraction (namely phases); and a temporal component, which is related to when to extract these volumes or portions of them.

The most widely used approach to address this problem (in literature and software) is nested pits, which is in turn based on the ultimate pit problem (Lerchs & Grossmann 1965). This approach focuses on the spatial component, as it looks for pushbacks or pseudo phases (phases with block support) that comply with global slope angles, without taking opportunity costs into account. In this approach, production scheduling is performed in later stages and is subordinated to the spatial definitions given by the nested pits.

Another possible approach to the problem is direct block scheduling (DBS), proposed by Johnson (Johnson 1968). This approach considers global slope angles, but also production and transportation capacities over time and, therefore, the opportunity cost. DBS relies on mathematical programming to decide in terms of NPV what is the best timing for the extraction and processing of each block. As such, DBS is more focused on the temporal component, and the definition of pushbacks (and phases) is, therefore, subordinated to the period-by-period extraction.

Another relevant distinction between nested pits and DBS is the computational complexity. As previously mentioned, the nested-pit methodology relies on the ultimate pit problem, which can be solved efficiently using the Lerchs & Grossman algorithm or pseudo-flow algorithms (Chandran & Hochbaum 2009). On the contrary, DBS is computationally hard, and therefore finding good feasible solutions for it has been

the topic of many research efforts. For example, Jélvez et al. developed a new algorithm to find good feasible solutions efficiently for large instances of the problem (Jélvez et al. 2020). Similarly, Suárez and Alipour et al. applied a genetic algorithm to approach the problem (Suárez 2017; Alipour et al. 2020). In both cases, this approach was tested in a three-dimensional model and showed good results compared with the optimum.

Something common to these approaches is that they consider operational constraints only as global slope angles that provide stability, but they do not take into account the design of ramps or operational spaces required at the bottom of the pit. Figure 1(a) presents an example of pushback with a global angle of 45° in a conceptual 2-D block model. As the figure shows, the bottom of the pit may not provide enough space to operate the equipment. Conversely, Figure 1(b) presents an optimal pushback, but where the bottom of the pit is guaranteed to be at least 4 blocks wide and allows for shovel operation. Notice that truncating the pit on (a) to have a minimum pit bottom size is not optimal.
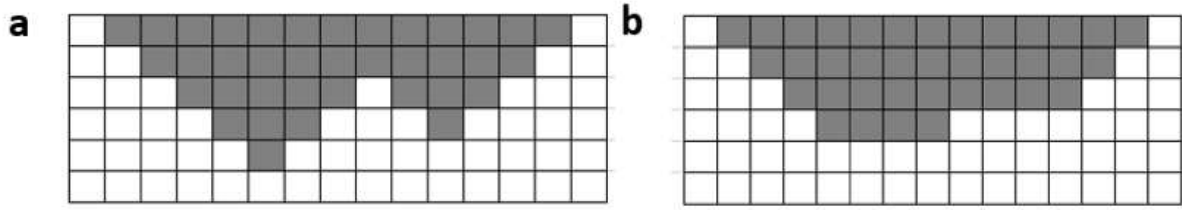

Figure 1   **Conceptual comparison between pushback section views (a) without considering minimum pit bottom dimensions and (b) considering a 4-block-bottom pit size**

The problem addressed in this work is how to plan and design an open-pit mine by considering the spatial and temporal components simultaneously, while also seeking operational geometries – like the one in Figure 1(b) to generate pushback designs that comply with minimum bottom widths – oriented to maximize the NPV. As the problem addressed is an extension of the existing methodologies, which are known to be either hard or incomplete, the resolution of the problem uses techniques from artificial intelligence (AI), namely Genetic Algorithms (GAs) and a clustering technique (in our case, K-Means).

2   State of the art

In this section, we briefly review the relevant aspects of DBS as well as the AI techniques utilized by the proposed methodology.

2.1   Open pit production scheduling considering capacities on resource consumption

The reference problem is the scheduling of open-pit production considering some constraints, which is known as the constrained pit limit problem (CPIT). It is a simplified version of the formulation presented by Johnson (Johnson, 1968), as it only considers capacities, but not multiple destinations or ore grade constraints. As originally presented, CPIT considers a set of blocks B, $t = 1,2,...,T$ periods of times, and the following parameters: an economic discounted value $\hat{p}_{bt}$ for each block b and period t; a set of predecessors $B_b \subset B$ for each block b (i.e., the blocks that must be extracted before b in order to comply with global slope angles); a set R of resources and q_{br} being the consumption of resource r by block b and R_{rt} the total availability of resource r at period t (Chicoisne et al. 2012).

Given the above, CPIT can formally be written as the following integer linear program (Espinoza et al. 2013), where the binary variable $\chi_{bt} = 1$ if and only if block b is extracted and processed at period t:

$$\max \sum_{b \in B} \sum_{t \in T} \hat{p}_{bt}\, \chi_{bt} \tag{1}$$

$$\text{subject to} \sum_{s \leq t} \chi_{b's} \leq \sum_{s \leq t} \chi_{b's} \qquad \forall b \in B, b' \in B_b, t \in T \qquad (2)$$

$$\sum_{t \leq T} \chi_{bt} \leq 1 \qquad \forall b \in B \qquad (3)$$

$$R_{rt} \leq \sum_{b \in B} q_{br}\chi_{bt} \leq R_{rt} \qquad \forall t \in T, r \in R \qquad (4)$$

$$\chi_{bt} \in \{0,1\} \qquad \forall b \in B, t \in T \qquad (5)$$

The goal function (1) corresponds to the NPV of the extracted blocks. Constraints (2) enforce the precedence due to global slope angles, constraints (3) that the blocks are extracted at most once, and constraints (4) that limit the utilization of available resources.

It is worth noting that the ultimate pit problem is a particular case of CPIT where $T = 1$ and $R = \emptyset$, i.e. there is only one period and no resource constraints. Also, it is relevant to mention that CPIT is an NP-hard problem.

2.1.1 *CPIT+ with minimum-size pit bottom*

As mentioned before, one limitation of CPIT (and other mathematical models) is that it considers geometric constraints only as global angles. In this article, we are interested in extending this to consider minimum operational space at the bottom of the pit. We call this extension CPIT+.

CPIT+ considers the same inputs to CPIT, i.e. a valuated block model, global angle precedence and resources, but adds a radius r, so that the bottom of each pushback has enough space for a circumference with a radius of at least r bocks (Navarro 2015).

2.2 The Genetic algorithm

The Genetic Algorithm (GA) is an optimization technique introduced by Holland, which is inspired by biological evolution (Holland 1975). It is considered a powerful tool for optimization in artificial intelligence as it allows high-quality solutions to be found for optimization problems within reasonable computation times.

The applicability of GAs is very broad. These algorithms can be applied to solve an optimization problem as:

$$\begin{array}{c} \max \ f(x) \\ x \in S \end{array} \qquad (6)$$

Where: f is the goal function and S is the set of feasible solutions. The main condition to use GA is that computation of $f(x)$ and verification of $x \in S$ can be done efficiently.

GAs use a population of potential solutions (individuals) for an optimization problem. In each iteration, the population is improved and updated by allowing individuals with high potential (fitness) to mix (cross) with each other, and by introducing small changes (mutations) to individuals. The algorithm is run until some criteria are achieved, for example, when good enough solutions are found or when the improvement stagnates.

The key elements of a GA are the following (Goldberg 1989):

1. Representation of individuals: Potential solutions to the optimization problem are encoded so the algorithm can process them by means of binary numbers, integers or decimal numbers, characters, etc. which in GA terminology are chromosomes within a gene.

2. Fitness Evaluation: The fitness of an individual emulates how well adapted it is for the conditions (constraints) of the problem. Usually, it is possible to use the evaluation of f for the individuals, but sometimes a corrected or adapted version $\tilde{f}$ is used. Also, this often requires that the encoding is first transformed into an actual solution.

3. Selection: This is the process of choosing what individuals will be utilized to generate the next generation of the population. The main principle is that individuals with higher fitness should have a higher probability of transferring their genes to future generations.

4. Genetic Operators: In order to produce the next population, a GA utilizes two types of genetic operators:

   a. Crossing. In this case, two or more individuals are combined in order to generate 1 or more descendants.

   b. Mutation. This operator changes one specific individual by changing some of its genes.

Some relevant parameters of a GA are the size of the populations, the number of generations, and the probability of crossing and mutation.

2.3   K-Means clustering algorithm

K-Means (MacQueen 1967) is a clustering technique commonly used to partition a set of data into K groups, where K is known in advance. The clustering is done by minimizing the sum of distances between each object and the centroid of its group. K-Means considers the following steps (Wagstaff et al. 2001):

1. Initialization: The algorithm starts with K centroids, randomly chosen.
5. Assignment of objects to centroids: Each object is assigned to its closest centroid.
6. Update of centroids: The centroid of each group is computed as the average of locations of objects assigned to that group.

The algorithm repeats these steps many times, until the position of centroids stabilizes.

3   Methodology

In this paper, we implement a methodology based on AI and compare it with the standard approach, which is presented in Figure 2 and described later in this section.
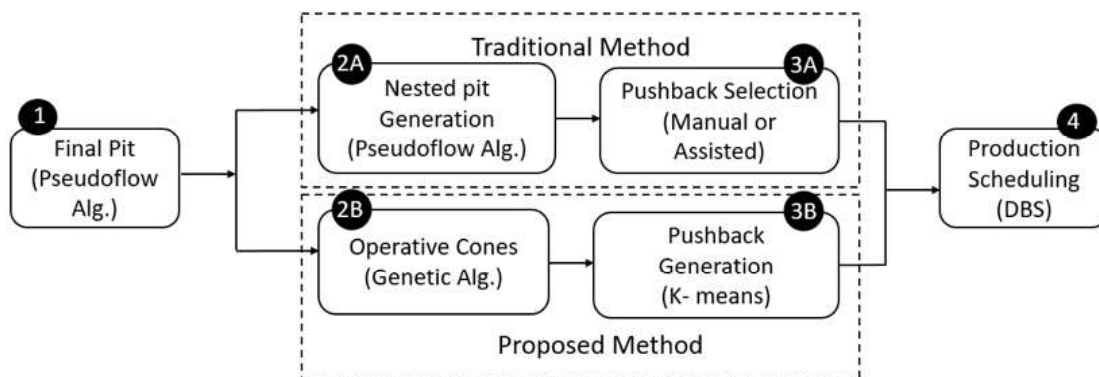

Figure 2   Comparison between the traditional method and AI-based method

3.1 Final Pit and production scheduling

The final pit computation and the production scheduling are common to both methodologies; therefore, we will describe them separately. The numbers correspond to those in Figure 2.

1   Final Pit computation — The economic block values (based on economic and operational parameters) are computed. The precedence arcs to represent the global angle precedence are calculated, and the ultimate pit is also computed utilizing the pseudo flow algorithm. Notice that the economic values and precedence are used in later stages in both methodologies.

4   Production Scheduling — The pushbacks are utilized to schedule the blocks for extraction, period by period. The scheduling considers total movement capacity as well as minimum and maximum distance between benches of consecutive phases (min/max lead parameters).

3.2 Traditional method

The traditional methodology that we use for comparison is based on nested pits. The steps considered are shown below. As before, the numbers correspond to Figure 2.

2A   Nested pit generation — In this stage, price is parametrized by a revenue factor from 0 to 1, to generate several nested pits. Price parametrization consists of scaling the metal price to create a sequence of N number of revenue factors. Each revenue factor produces a new pit, and all the pits are nested. The number of revenue factors depends on the initial value, the step, and the final value that the mine planning evaluator assigns (Jélvez et al. 2020).

3A   Pushback Selection — This is a manual and iterative process, in which some of the nested pits computed in the previous stage are selected to create pushbacks. For selecting pushbacks, some parameters were considered: stripping ratio (waste/ore) between pushbacks, tonnage distribution in each pushback, and minimum operational widths, among others

3.3 Method based on Artificial Intelligence

The two stages of the method based on AI are the following. Further detail can be found below as well.

2B   Operational cones — In this stage, a genetic algorithm is used to iterate over a population of individuals. An individual is represented as a set of operational cones.

3B   Pushback Generation — In this stage, the K-Means clustering algorithm is utilized to transform each individual generated by the GA, into a set of pushbacks that complies with the operational constraints.

3.3.1   GAs to generate optimize operational cones

In this section, we describe the main components of the genetic algorithm. To represent the algorithm solution satisfying the design constraints mentioned before in the CPIT+ problem, operational cones were used (Navarro 2015). The angle of the cone is the global slope angle. It is a truncated cone; therefore, its base has the operational space given by the radius r, which is an input (Figure 3(b)). Notice that given a block b, there is a unique operational cone that we denote as $C(b)$ so that b is the center of its base.

The algorithm can generate a cone from each block centroid, and the number of cones that we will get depends on the population size that we initially set for the GA. The population is randomly initialized.

3.3.1.1   Representation of individuals

As each operational cone is defined by the block centroid, its representation corresponds simply to the block ID number; the block ID is a unique integer calculated from the block model, and allows the spatial position of each block to be identified.

An individual in the GA is represented as a set of block centroids (bases of the cones). Figure 3(a) shows an individual constituted by four block centroids, $I = \{10, 51, 60, 90\}$. Each individual can have variable lengths.
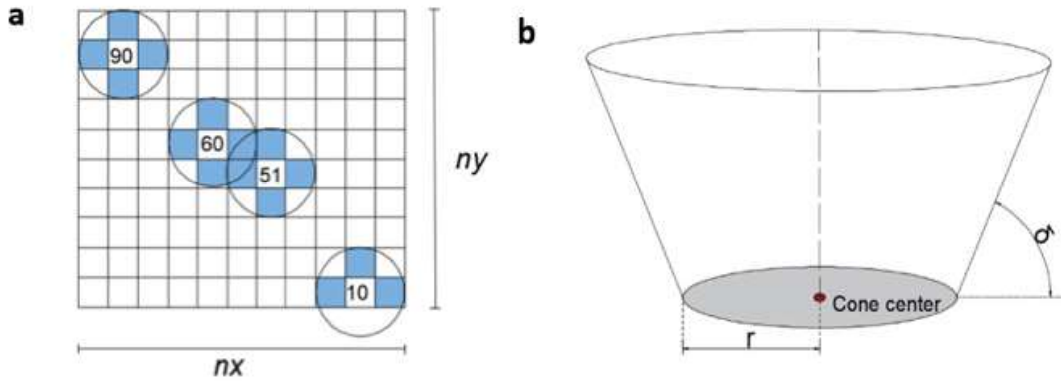

Figure 3   (a) Representation of an individual in the GA (plan view) (b) operational cone with a given cone center

3.3.1.2   Fitness function

Evaluating the objective function NPV would be very time consuming because it would require optimized scheduling of the blocks of an individual. Because of this, we used an approximation of the NPV that emulates the best case used in some planning software tools for open pit, as described next.

Given an individual $I = (i_1, i_2, i_3,...i_L)$, where i_p are IDs of the blocks, we computed the corresponding operational cones $C(i_1)$, $C(i_2)$, ..., $C(i_L)$, then the blocks present in the individual (bases of the cones) are clustered using K-means algorithm (more information in section 3.3.2). After that, increments $C_1 = C(i_1)$, $C_2 = C(i_2) - C(i_1)$, ..., $C_L = C(i_L) - C(i_{L-1})$ are computed.

We then assumed that the increments were extracted in order (i.e. C_p is finished before extraction of C_{p+1} begins). The extraction of blocks is done bench by bench, and the pass from one period to the next occurs when the capacity is reached, i.e., we keep track of the tonnage of blocks and consider that a period was ended when the capacities of that period are depleted.

The fitness of the individual is then the approximated NPV computed using the procedure described above. Notice that because of the increments, there is no double counting of blocks.

3.3.1.3 Selection

For this work, we implemented roulette selection, which works by assigning a probability to each individual in the population, which is proportional to its fitness. That is, if the population has fitness values ($g_1, g_2, ..., g_N$) (which are required to be non-negative), then the probability to pick one individual for crossing is:

$$q_i = \frac{g_i}{g_1 + g_2 + \cdots + g_N} \tag{7}$$

3.3.1.4 Genetic operators

- ☒ Crossover. This consists of selecting two individuals known as parents to exchange segments of their genetic code, producing offspring, which are combinations of their parents.

$$P1 = \{10, 51, 60, 90\} \qquad P2 = \{10, 25, 41, 62, 87, 90\}$$

$$h1 = \{10, 90\} \qquad h2 = \{25, 41, 51, 60, 62, 87\}$$

  In this case, the first offspring (h1) will have the common elements of both parents (P1) and (P2), while the second offspring (h2) will maintain elements that are not common between their parents.

- ☒ Mutation. This consists of adding or removing an element from the individuals. This process is carried out randomly. If crossover proceeds successfully, one or both offspring will mutate.

3.3.2 Using K-Means to generate pushbacks

As previously explained, each individual of the GA is a set of blocks, representing bases of operational cones. While it would be possible to interpret each cone as a phase, this is too constrained: there may be too many phases and there may be redundancies.

Because of the above, to generate more realistic pushbacks and to control their number, the methodology applies the K-Means method to cluster the blocks. The clustering process works in parallel with the GA, clustering the blocks of each individual in pushbacks. These may consist of one or more operational cones.

The output of the GA and the K-Means process will be the best individual with their respective pushback.

4 Numerical experiences

The methods presented above were tested in two case studies: Iron_bm and McLaughlin (MCL for short). Table 1 provides a summary of both cases.

Table 1   Summary of case studies

Case study	Ore	Number of blocks			Block size
		nx	ny	nz	(m)
Iron_bm	Fe	160	120	46	25x25x15
MCL	Au	139	295	67	7.62x7.62x6.1

4.1   Final Pit computation

The economic value is computed using the equation (8) for an ore block and equation (9) for a waste block, with y_b being the ore grade of block b, and ton_b the tonnage of the block.

$$Pv_b = (P - SC)Ry_b ton_b - (MC + PC)ton_b \tag{8}$$

$$Wv_b = -(MC)ton_b \tag{9}$$

Where:

R = the metallurgical recovery.

P = the metal price.

SC = the selling cost.

MC = the mining cost.

PC = the processing cost.

Table 2 shows some of the parameters used for the economic block evaluation and the final pit calculation. Results in terms of total perceived value and final pit size for both case studies are shown in Table 3.

Table 2   Parameters to generate nested pits from Iron_bm and MCL

Parameters		Slope angle	Metallurgical recovery	Metal price	Selling cost	Mining cost	Processing cost
Symbol		θ	R	P	SC	MC	PC
Value	(Iron_bm)	45°	0.92	118 (USD/ton)	29 (USD/ton)	2.70 (USD/ton)	10.85 (USD/ton)
Value	(MCL)	45°	0.76	1,100 (USD/ozt)	100 (USD/ozt)	1.50 (USD/ton)	8.20 (USD/ton)

Table 3   Final pit results for Iron_bm and MCL

Case study	Economic Value (MUSD)	Number of blocks
Iron_bm	21,000.21	167,612
MCL	3,043.32	254,398

In both cases, all blocks not included in the ultimate final pit were removed from the set and not considered in the subsequent steps.

4.2   Application of the traditional methodology

4.2.1   Nested Pits

Nested pits are generated by scaling the metal price of the block using a revenue factor RF, which takes values between 0 and 1, with a defined step. For the Iron_bm case, 52 nested pits were generated, with RF from 0.48 to 1.0 and a step of 0.01. Figure 4(a) shows a section view of the pits, and in Figure 4(b), the pit-by-pit graph plots the cumulative economic value and tonnage (ore and waste) for each pit.
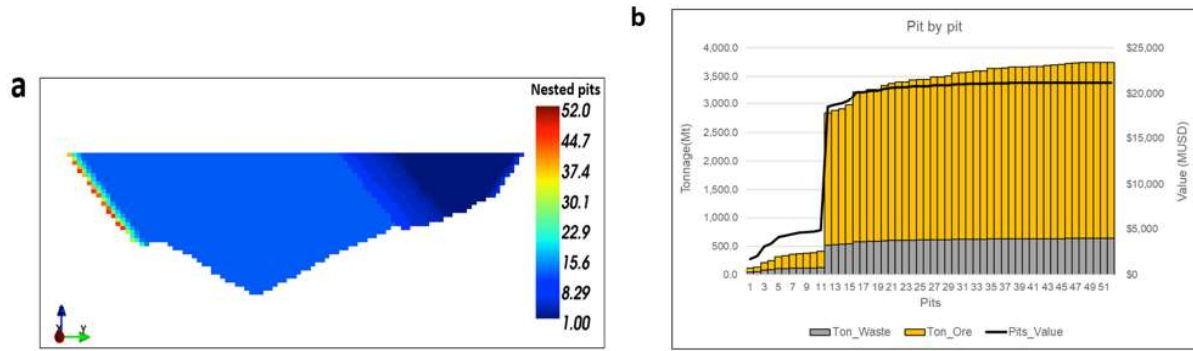
Figure 4   (a) Section view showing nested pits from Iron_bm model (b) pit-by-pit graph from Iron_bm model

For MCL, 87 pits were generated with RF from 0.13 to 1.0 and a step of 0.01. Figure 5(a) presents a section view of the pits, and Figure 5(b) shows the cumulative tonnage (ore and waste) and values for the pits.
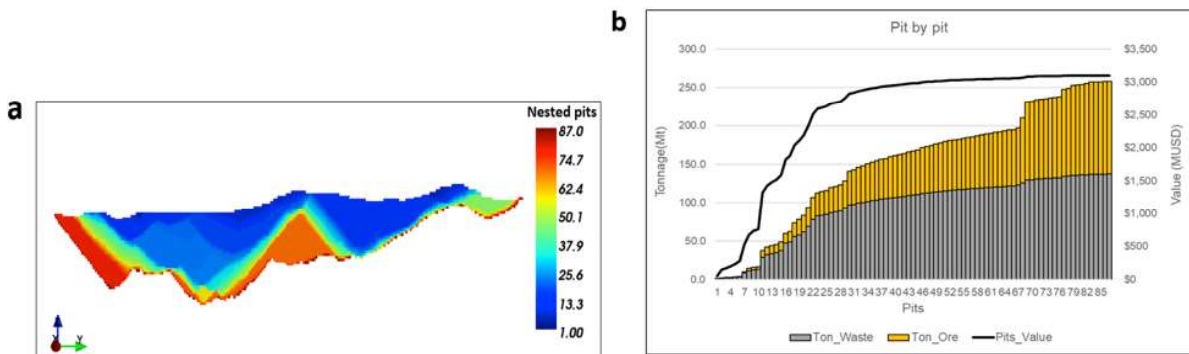

Figure 5   (a) Section view showing nested pits from MCL model (b) pit-by-pit graph from MCL model

4.2.2   Pushback selection

Once nested pits were generated, they are clustered to create pushbacks, which are used as a guide for the subsequent design. Pushbacks will be scheduled for their extraction over time.

Section views and a pit-by-pit graph are used as a reference to identify candidates for pushbacks. Selected pushbacks try to fulfill certain criteria:

1. Minimization of the difference in terms of tonnage and value between selected pushbacks in order to control the gap problem (Meagher et al. 2014).

2. Similar stripping ratio (waste/ore) between pushbacks, to meet plant constraints and requirements.

3. Pushbacks with a minimum mining width to allow access and mobility of mining equipment.

The number of pushbacks for both methodologies is the same to warrant a fair comparison. In Figure 6, the pit- by-pit graph shows each pushback selected, highlighted in a different color.
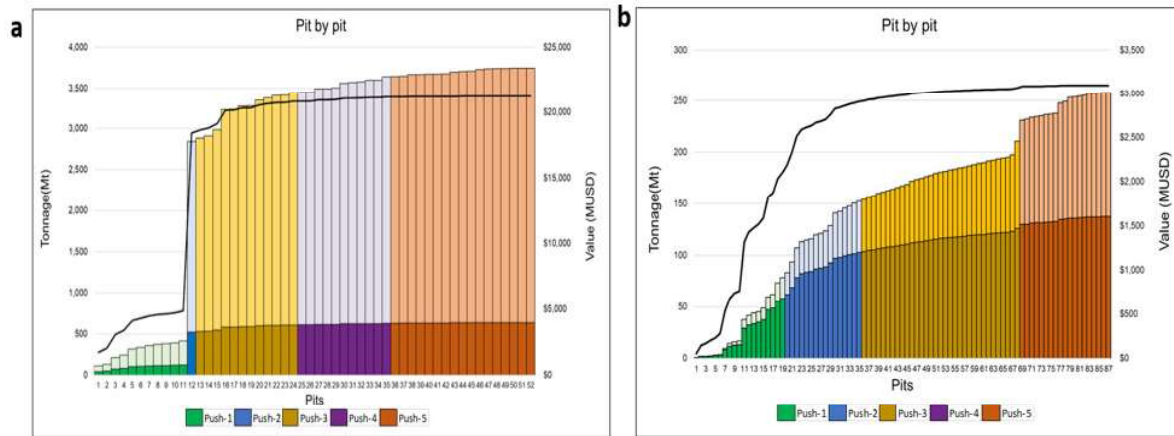
Figure 6 **(a) Pit-by-pit graph with each pushback selected from the Iron_bm model (b) pit-by-pit graph with each pushback selected from the MCL model**

4.2.3   Production scheduling (Traditional)

A production schedule known as phase-bench-destination was generated for each case study. Table 4 shows the implemented constraints for both cases: mine capacity, plant capacity, and min/max lead parameters. The min/max parameters refer to the minimum and the maximum difference between consecutive pushbacks. The objective function is to maximize NPV with a discount rate of 10% for Iron_bm and 15% for MCL.

Table 4   **Production scheduling Parameters**

Case study	Mining capacity (Mt)	Plant capacity (Mt)	Min Lead	Max Lead	GAP (%)
Iron_bm	150.00	25.00	3	5	5
MCL	9.00	4.00	3	5	5

The results obtained were for NVP = 3,976.45 MUSD for the Iron_bm model and NPV = 852.79 MUSD for the MCL model. The scheduling process was solved with a default optimality gap set to 5% for both cases.

4.3   Application of the AI methodology

4.3.1   GA for the generation of operational cones

The parameters used for AI methodology are shown in Table 5. The genetic algorithm process will start by choosing a population size. It represents a set of initial individuals and is randomly created. The generations correspond to the number of iterations of the GA process, and the MUTPB and CXPB represent the mutation and crossover probability, respectively. These parameters follow a series of sensitivity analyses that balanced computational time with the results obtained. The global angle comes from the final pit definitions.

In the case of the design parameters for the Iron_bm case, we considered a radius r = two blocks and for MCL a radius r = four blocks, which will correspond to the base size of each pushback. These parameters were chosen considering block sizes and potential equipment dimensions.

Table 5 **Parameters of the Genetic Algorithm**

	Algorithmic parameters				Design Parameters
Case study	Population Size	Generations	MUTPB	CXPB	Global Angle (°)
Iron_bm	100	200	0.2	0.6	45
MCL	100	150	0.2	0.7	45

4.3.2 Pushback generation using K-Means

The clustering process utilized the K-Means algorithm. This requires a pre-set "k" value for clustering the population into groups, which will represent the number of pushbacks. A number of iterations is also required to complete the clustering process and is equal 30 for both cases.

To determine the correct number of clusters "k", an analysis known as the elbow method was initially performed using the SPSS statistical software. We plotted a line chart of the error of squared sums (SSE) for each value of k (Figure 7(a) and Figure 8(a)). The elbow on the graph is the value of k that represents the best number. It was not necessary to select a larger K size because the difference in the decrease of the error is minimal and not considered significant (Purnima & Arvind 2014).

Additionally, a graphical analysis was performed (Figure 7(b) and Figure 8(b)) to plot the behavior of the NPV and each value of "k".
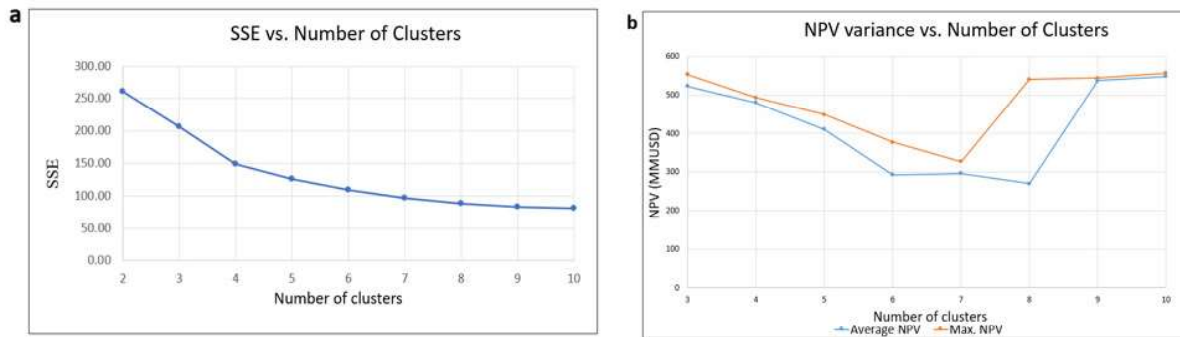

Figure 7 **(a) Elbow method for optimal cluster selection (b) NPV behavior based on the cluster number for the Iron_bm model**

As a result of these analyses, the number of pushbacks (clusters) was 5 for Iron_bm and 4 for MCL because these were the best options in terms of tonnage distribution, symmetry and value.
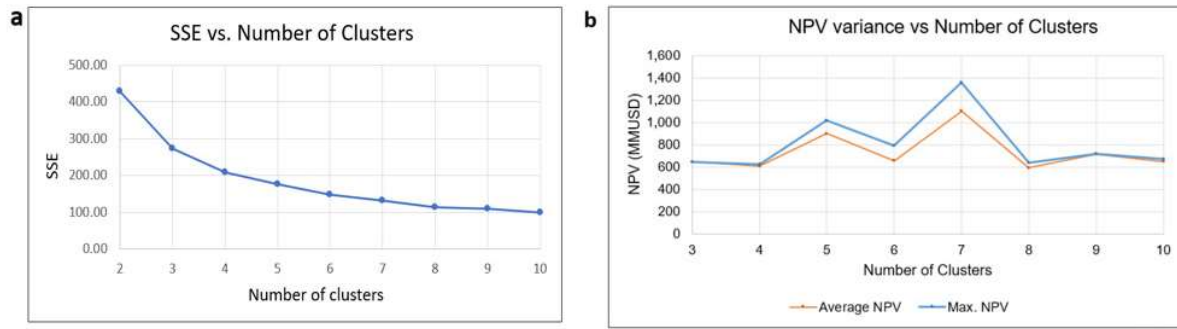
Figure 8   (a) Elbow method for optimal cluster selection (b) NPV behavior based on the cluster number for the MCL model

4.3.3   Production scheduling (AI)

For each case study, the phase-bench-destination production schedule was generated. The applied parameters were the same as the traditional methodology (Table 4) so that valid comparisons could be made between methodologies.

The results obtained were NVP = 856.34 MUSD for the Iron_bm model and NPV = 675.89 MUSD for the MCL model.

5   Analysis and discussion

5.1   Comparison of Pushbacks

The results for each methodology in the Iron_bm case are shown in Figure 9. The resulting pushbacks for each methodology look geometrically different. In the traditional methodology, there is a tendency to create large initial pushbacks and small final pushbacks. As this method does not consider design constraints, Figure 9(a) and Figure 10(a) show that the last three pushbacks do not have minimum operative width, and as a result, these pushbacks could be operationally unfeasible.
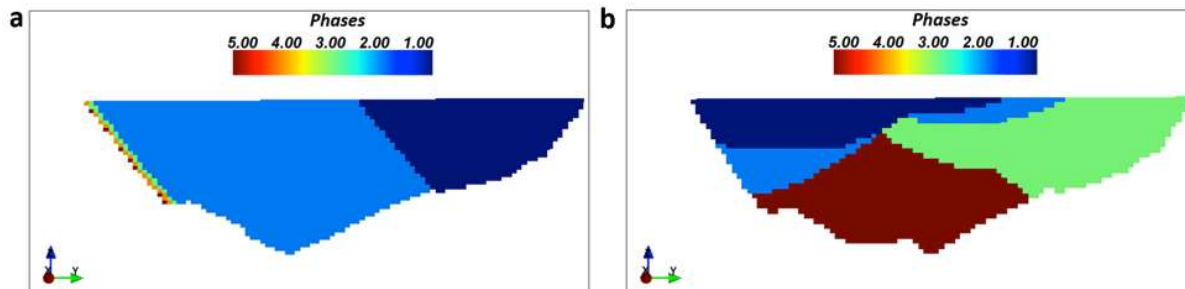

Figure 9   Section view showing resulting pushbacks for Iron_bm model from (a) the traditional methodology and (b) the AI methodology

For the traditional methodology, pushback 2 is bigger than the other pushbacks. When comparing pushback 2 with pushback 4 (the smallest one), the first is 93.24% larger and does not meet with the difference minimization requisite criteria in terms of tonnage between pushbacks.

Figure 6(a) shows a tonnage increment between successive pits. This increment represents a gap problem with significant volume and value increases. The traditional methodology did not allow us to solve the gap problem because it was not possible to find an RF number to generate intermediate pits. As a result, pushback 2 has a single nested pit.

In Figure 9(b) and Figure 10(b), the resulting pushbacks show that through AI methodology, it is possible to divide the biggest pushback (pushback 2) into smaller ones. It also shows better symmetry among the remaining pushbacks, and as far as design constraints, operational widths were also successfully achieved.
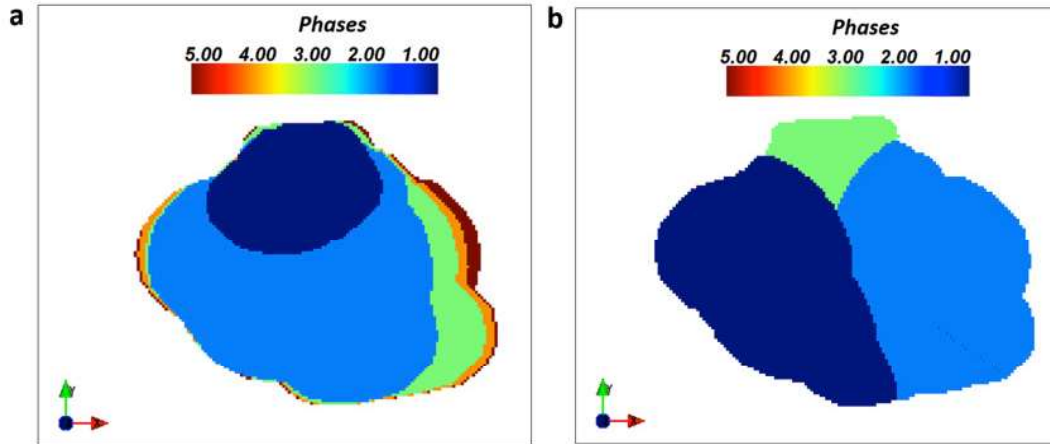

Figure 10   Plan view showing resulting pushbacks for Iron_bm model from (a) the traditional method and (b) the AI method

Table 6 shows the tonnage quantity of each pushback. In the particular case of the Iron_bm model, it has a higher percentage of waste because of a high stripping ratio in the first years. The amount of waste material is greater than the amount of ore, which will make it expensive to mine in the first years. With the traditional methodology, the waste tonnage represents 83.04% of the total, while with the AI methodology, it represents 83.85% of the total.

For the MCL case study, the resulting pushbacks are shown in Figure 11. In the traditional methodology (Figure 11(a)), some end areas do not have enough width to become operational; as early as the second pushback, the operational width decreases considerably. Due to the geometry of the deposit, it is difficult to achieve operational pushbacks using nested pits methodology. In the case of the proposed AI methodology, pushbacks look operationally feasible (Figure 11(b)). Both precedence constraints and minimum width constraints are achieved in each pushback.

Table 6   Results of tonnage in each pushback by methodology for Iron_bm case

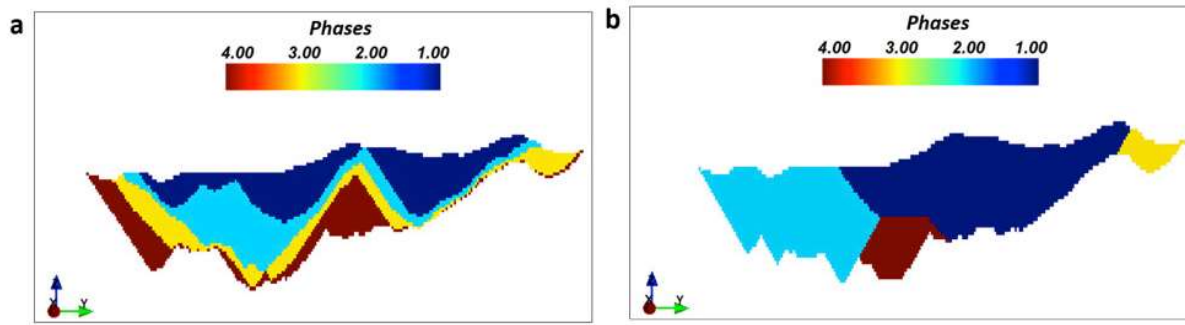| Methodology	Traditional				AI			
Pushback	Plant Tonnage (Mt)	Waste Tonnage (Mt)	Total Tonnage (Mt)	Average grade (%)	Plant Tonnage (Mt)	Waste Tonnage (Mt)	Total Tonnage (Mt)	Average grade (onz/t)
1	123.08	291.44	414.51	0.73	76.28	1,015.88	1,092.15	0.78
2	396.24	2,028.52	2,424.76	0.75	122.97	1,367.33	1,490.29	0.68
3	67.08	372.02	439.10	0.69	112.61	257.30	369.91	0.71
4	20.83	143.04	163.87	0.68	65.81	61.68	127.50	0.74
5	28.66	277.87	306.54	0.64	217.39	389.24	606.63	0.75
Total	635.89	3,112.89	3,748.77	0.70	595.05	3,091.42	3,686.47	0.73

Figure 11   Section view showing resulting pushbacks for MCL model from a) the traditional method and (b) the AI method

Table 7 shows the ore and waste tonnage distribution between the pushbacks. With the traditional methodology, 2.78% more ore tonnage is extracted than with the (AI) methodology. In both cases, distribution and symmetry look acceptable.

5.2   Comparison of production plans

Table 8 shows a summary of the NPV results obtained from both methodologies through scheduling by phase-bench-destination. In the first case study, NPV resulting from the (AI) methodology represents a far smaller percentage than NPV resulting from the traditional methodology.

Table 7   Results of tonnage in each pushback by methodology for MCL case

Methodology	Traditional				AI			
Pushbacks	Plant Tonnage (Mt)	Waste Tonnage (Mt)	Total Tonnage (Mt)	Average grade (%)	Plant Tonnage (Mt)	Waste Tonnage (Mt)	Total Tonnage (Mt)	Average grade (onz/t)
1	57.16	19.70	76.86	0.07	61.07	42.66	103.73	0.05
2	45.15	29.31	74.46	0.04	51.56	42.25	93.81	0.04
3	20.75	24.67	45.42	0.03	10.25	19.42	29.67	0.04
4	14.05	46.73	60.78	0.02	10.91	15.05	25.96	0.05
Total	137.11	120.41	257.52	0.04	133.79	119.38	253.17	0.05

In the second case study, MCL, NPV differences between both methodologies exist. The NPV obtained through AI is 20.74% lower than the result obtained from the traditional methodology. In both case studies, the NPV obtained from AI methodology was lower than the traditional one.

Table 8   Summary of the NPV results obtained through the different methodologies

Methodology	Iron_bm				MCL			
	NPV (MUSD)	Periods (Years)	Ore (Mt)	Waste (Mt)	NPV (MUSD)	Periods (Years)	Ore (Mt)	Waste (Mt)
Traditional	3,976.45	42	635.89	3,112.89	852.79	40	137.11	120.41
AI	856.34	38	595.05	3,091.42	675.89	34	133.79	119.38

The production plans are presented in Figure 12. For Iron_bm, the traditional method shows an early ore tonnage extraction starting in the second year (Figure 12(a)). This is not the case for the AI method, which prioritizes waste extraction in the early years. As we can see in Figure 12(c), ore extraction starts in the fifth year.

For the MCL model, the production plan obtained using the traditional method (Figure 12(b)) and AI method (Figure 12(d)) start the ore tonnage extraction in the first year. The results obtained with the AI method show uniform behavior in terms of value and ore tonnage, ensuring a continuous ore supply for the processing plant.
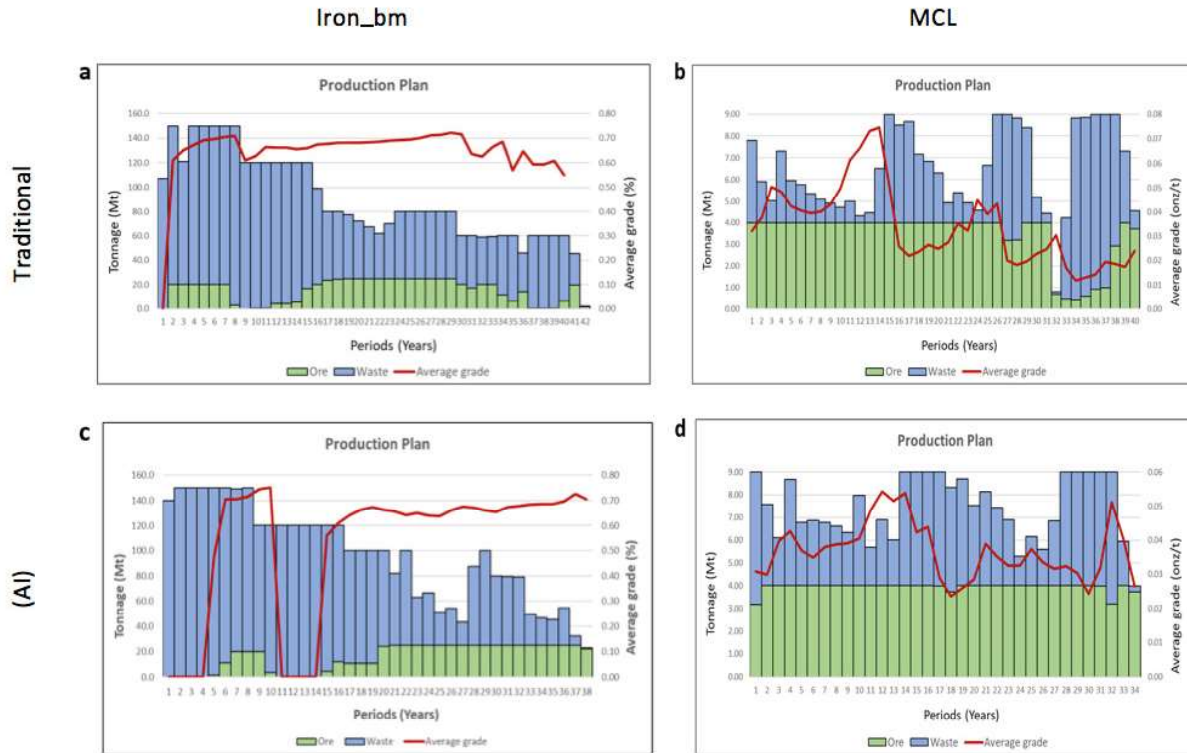

Figure 12   Comparison of production plans between (a) the traditional method for Iron_bm (b) the traditional method for MCL (c) AI method for Iron_bm (d) AI method for MCL

6   Conclusions

In this paper, AI was used to obtain operative pushbacks in open-pit mines, respecting operational and design constraints. The model works through a genetic algorithm and a clustering algorithm. These algorithms can solve combinatorial optimization problems. The proposed approach was applied in two large-scale mines, which are geometrically and mineralogically different. It proves to be applicable in both cases, and it was demonstrated that the size of the models is not a limitation.

A numerical study was performed to compare the solutions obtained between the traditional methodology and a methodology using AI. There are four main advantages of using AI. First, the pushbacks generated by AI do not require manual selection, as in traditional nested pit methodology. This is an advantage because the success of manual selection depends on the expertise of mine planner. In this way, AI can reduce process time in terms of generating pushbacks. Second, the AI method was able to produce smaller pushbacks and minimize the tonnage differences among the pushbacks, thus helping to avoid the gap problem. Third, the size of the block model is not a limitation. Finally, the pushbacks obtained using AI meet with minimum operational constraints, so in subsequent steps, the pushbacks will be operationally more feasible. On the other hand, pushbacks obtained through the traditional methodology show a more significant challenge to being operative.

One important difference in comparing the traditional and AI methodologies was found in comparing NPV Better results for NPV were achieved with the traditional method in both case studies; however, it is important to remember that the production plan of AI prioritizes waste extraction during early

periods. Then, during the first periods, a high percentage of the total tonnage mined is waste. In fact, NPV would be higher if ore blocks could be extracted during the early periods. Another reason for this variance in terms of NPV is that the AI methodology incorporates more constraints than the traditional one. Finally, without operative designs, the output is almost always overly optimistic: overestimating ore, and underestimating waste; these discrepancies should be reflected in a decreasing NPV in post-design production plans.

Acknowledgement

References

Alipour, A, Khodaiari, A, Jafari, A & Tavakkoli-Moghaddam, R 2020, 'Production scheduling of open-pit mines using genetic algorithm: a case study', International Journal of Management Science and Engineering Management, vol. 15, no. 3, pp. 176-183.

Chandran, B & Hochbaum, D 2009, 'A computational study of the pseudoflow and push-relabel algorithms for the maximum flow problem', Oper. Res, vol. 57, no. 2, pp. 358-376.

Chicoisne, R, Espinoza, D, Goycoolea, M, Moreno, E & Rubio, E 2012, 'A new algorithm for the open-pit mine production scheduling problem', Oper. Res, vol. 60, no. 3, pp. 517-528.

Espinoza D, Goycoolea M, Moreno E & Newman A 2013, 'Minelib 2011: A library of open pit production scheduling problems', Ann. Oper. Res, vol. 206, no. 1, pp. 91-114.

Goldberg, D 1989, 'Genetic Algorithms in Search, Optimization, and Machine Learning', Addison-Wesley, Boston, United States.

Holland, J 1975, 'Adaptation in Natural and Artificial Systems', University of Michigan Press, Ann Arbor, Republished by the MIT press, 1992.

Jélvez, E, Morales, N & Askari-Nasab, H 2020, 'A new model for automated pushback selection', Computers & Operations Research, vol. 115, 2020, 104456.

Jélvez, E, Morales, N, Nancel-Penard, P & Cornillier, F 2020, 'A new hybrid heuristic algorithm for the Precedence Constrained Production Scheduling Problem: A mining application,' Omega, vol.94, 2020, 102046.

Johnson, T 1968, 'Optimum open pit mine production scheduling', Doctoral dissertation, University of California, Berkeley.

Lerchs, H & Grossmann, I 1965, 'Optimum design of open-pit mines', Canadian Mining Metallurgical Bull, vol. 58, no. 633, pp. 47-54.

MacQueen, JB 1967, 'Some methods for classification and analysis of multivariate observations', proceedings of the Fifth Symposium on Math, Statistics, and Probability Berkeley, CA: University of California Press, pp. 281-297.

Meagher, C, Dimitrakopoulos, R & Avis, D 2014, 'Optimized open pit mine design, pushbacks and the gap problem: a review', Journal of Mining Science, vol. 50, no. 3, pp. 508-526.

Navarro, F 2015, 'A parallel and distributed genetic algorithm for scheduling blocks in open pit mines', Master's thesis, Universidad de Santiago de Chile, Santiago, Chile, in Spanish.

Purnima, B & Arvind, K 2014, 'A Clustering Technique based on Elbow Method and K-Means in WSN', International Journal of Computer Applications, vol. 105, no. 9, pp. 0975-8887.

Suárez, M 2017, 'Design of exploitation phases in open pit mines generated through a genetic algorithm', Master's thesis, Universidad de Chile, Santiago, Chile, in Spanish.

Wagstaff, K, Cardie C, Rogers, S & Schroed, S 2001, 'Constrained K-means Clustering with Background Knowledge', proceedings of the Eighteenth International Conference on Machine Learning, pp. 577-584.