I.J. Information Technology and Computer Science, **2025, 6, 1-28**
Published Online on December 8, 2025 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2025.06.01

Predictive Modelling and Factor Analysis of Public Transport Delays in Smart City Using Interpretable Machine Learning

Yurii Matseliukh
Lviv Polytechnic National University, S. Bandera Street, 12, Lviv, 79013, Ukraine
E-mail: indeed.post@gmail.com
ORCID iD: https://orcid.org/0000-0002-1721-7703

Vasyl Lytvyn
Lviv Polytechnic National University, S. Bandera Street, 12, Lviv, 79013, Ukraine
E-mail: vasyl.v.lytvyn@lpnu.ua
ORCID iD: https://orcid.org/0000-0002-9676-0180

Zhengbing Hu
School of Computer Science, Hubei University of Technology, Wuhan, China
E-mail: drzbhu@gmail.com
ORCID iD: https://orcid.org/0000-0002-6140-3351

Myroslava Bublyk*
Lviv Polytechnic National University, S. Bandera Street, 12, Lviv, 79013, Ukraine
E-mail: my.bublyk@gmail.com
ORCID iD: https://orcid.org/0000-0003-2403-0784
*Corresponding author

Received: 14 August 2025; Revised: 02 October 2025; Accepted: 07 November 2025; Published: 08 December 2025

Abstract: Delay prediction in urban public transport systems is a critical task for improving operational efficiency and service reliability. While numerous predictive models exist, understanding the relative importance of contributing factors remains a challenge, with traditional approaches often overestimating the impact of stochastic weather conditions. This study proposes an approach that combines predictive modelling and factor analysis based on interpretable machine learning. An eXtreme Gradient Boosting model was developed using a large dataset of operational and meteorological data from a city with approximately one million inhabitants. The model demonstrated high predictive accuracy, explaining 72% of the variance in delays (Coefficient of Determination R^2=0.72). Analysis of the model's feature importance revealed that operational cycles (seasonal, weekly, daily) and spatial context (routes, stops) are the dominant predictors, collectively accounting for over 52% of the model's total feature importance. Contrary to common assumptions, weather conditions were identified as a powerful secondary, rather than primary, factor. While their cumulative feature importance was substantial (contributing nearly 45%), the model revealed their impact to be highly contextual: the negative effects of adverse weather were significantly amplified during predictable peak operational hours but were minimal otherwise. This research demonstrates how Explainable Artificial Intelligence methods can transform complex predictive models into practical tools, providing a data-driven basis for shifting from reactive management to proactive, evidence-based planning.

Index Terms: Public Transport, Interpretable Machine Learning, XGBoost, SHAP, Smart City, Delay Factor Analysis.

1. Introduction

Traditional approaches to managing urban transport reliability are often reactive [1], guided by dispatcher intuition rather than empirical evidence. This is largely due to the stochastic nature of these complex systems, which are subject to a multitude of interacting variables. An example is the assumption that adverse weather is the primary destabilising factor.

Consequently, resources are often misallocated to address apparent weather-related symptoms, while the underlying systemic causes of operational instability remain unaddressed. With the development of Smart Cities [2], transport companies and municipalities gain access to huge amounts of operational data: GPS tracks [3], data from validators, smart maps [4], minute-by-minute weather reports [5], etc. The relevance of the study lies in the transition from simple collection of this data to their practical use for making informed management decisions. Modern analytics, in particular machine learning methods [6], make it possible to transform this data into informed decisions. Instead of being guided by the well-known myth that snow causes traffic jams, the authors of this work set themselves the ambitious task of building a model that would objectively identify and quantify the true hierarchy of factors affecting the reliability of passenger transportation. This will allow us to move from intuitive, reactive management [7] to proactive, data-driven strategic planning [8], which is the cornerstone of the smart city concept [9,10].

1.1. Features of the Smart City Concept for Passenger Transportation

The concept of a Smart City involves the integration of information and communication technologies to improve the efficiency of urban services and the quality of life of residents. The term «smart city» is defined as a unique urban infrastructure that combines intelligent sensors, Internet of Things (IoT) devices, and information fusion techniques to create interconnected and intelligent urban spaces [11]. This combination of information and data-driven technologies has enormous potential to improve urban life, making cities more resilient and efficient.

Public transport is one of the key arteries of a “smart city”, and its reliability and predictability directly affect population mobility, economic activity and environmental status [12]. The significance of this concept for public transport is the transition from static management based on fixed schedules to dynamic, adaptive and proactive management [13] based on real-time data analysis. However, there are a number of problems along the way, closely related to the key topics of our research. The choice of predictive modeling was due to the fact that the urban transport system is a complex, chaotic system. Predictive Modeling is the process of using historical and current data to create a model to predict future outcomes [14]. Unlike descriptive models, which only explain past events, predictive models answer the question «What will happen next?» In the transportation context, this allows for a shift from reactive responses to problems that have already occurred (e.g., traffic jams) to proactive actions aimed at preventing them, which is a fundamental principle of smart cities. Traditional statistical methods are often unable to cope with a huge number of nonlinear relationships. Although modern machine learning models, such as extreme gradient boosting (XGBoost) [15], demonstrate high accuracy in forecasting, they are often perceived as “black boxes”. Machine Learning is a branch of artificial intelligence that focuses on developing algorithms that allow computer systems to “learn” from data, detect patterns in it, and make predictions without being explicitly programmed for each specific task. For transportation engineers, it is a tool that allows them to automatically analyze complex relationships between dozens of factors, such as weather, traffic, and schedules, to build accurate predictive models used to simulate passenger flows or predict weather conditions [15]. For a transportation engineer or city manager, it is not enough to get a forecast of “expected delay of 15 minutes”. To make an effective decision, it is necessary to know why this delay will occur. This is where a fundamental gap arises: highly accurate but unclear models do not become a full-fledged decision-making tool, since they do not provide an answer to the question “what to do?”, which is successfully explained by interpretable machine learning (XAI) [15].

Delay Factor Analysis was chosen because management decisions are often based on intuition and commonly accepted beliefs, for example, that the main cause of delays is weather [16]. This leads to inefficient allocation of resources, as the real, systemic causes remain undetected.

Thus, modern IT solutions can radically change approaches in applied engineering fields, such as transportation, which indicates the relevance of this research today, where it is important to “connect theory and practice”, turning complex academic models into practical tools for everyday use.

1.2. Predictive Modeling in Public Transport

Predictive modeling in public transport is a key tool for the transition from reactive to proactive urban mobility management. Modern methods, especially those based on machine learning, allow for the prediction of a wide range of parameters: arrival and departure times, passenger flows between stations, real-time vehicle occupancy and the probability of delays [17]. This allows for the optimization of schedules, dynamic management of rolling stock and advance information to passengers. The effectiveness of predictive models largely depends on the quality and granularity of the input data. The use of data from different sources (GPS, smart cards, weather sensors, flight data) through information fusion methods significantly increases the accuracy of predictions. Time series models, such as LSTM [18], are good at predicting sequential events, while ensemble methods, such as XGBoost [19], are extremely effective at analyzing tabular data with complex dependency structures. The main limitation is the so-called “black box” problem, where a model makes an accurate prediction but does not explain the factors that led to it [20]. This makes it difficult to trust the model and use it for strategic decision-making. In addition, models trained on historical data may perform poorly when unexpected events (e.g., traffic accidents, road closures) occur that were not present in the training set.

1.3. Explainable Machine Learning (Explainable AI, XAI) in Transportation Systems

XAI is a set of methods that allow us to understand and interpret the results of complex machine learning models. XAI turns predictive models into diagnostic tools. Methods such as SHapley Additive exPlanations (SHAP) [21] allow

us to quantify the contribution of each factor (e.g., weather, time of day, route) to a specific delay forecast.

This allows us to not only state the fact of a possible delay, but also to understand its causes, which is critical for developing countermeasures. XAI methods such as SHAP [22] can be applied to any machine learning model, which makes them universal. They provide both global interpretation (which factors are most important for the model as a whole) and local interpretation (why a particular flight at 17:30 was delayed). This allows us to conduct deep factor analysis, revealing nonlinear dependencies and interactions between variables. The interpretation provided by XAI is an explanation of the behavior of the model, and not necessarily of reality itself. If the model is trained on poor-quality data, its interpretation may also be incorrect. In addition, for very complex models (e.g., deep neural networks), computing SHAP values [23] can be resource-intensive.

1.4. Extreme Gradient Boosting in Passenger Transportation

XGBoost is a high-performance implementation of the gradient boosting algorithm that has become the de facto standard for forecasting tasks on tabular data. XGBoost combines high predictive accuracy with relative learning speed and interpretability via XAI [24]. It has been successfully applied to predict passenger flows, travel times, delays, and modeling passenger transport choice. The algorithm has built-in regularization mechanisms to combat overfitting, can effectively handle missing data, and supports parallel computing. Its ability to model complex nonlinear relationships makes it an ideal tool for analyzing systems such as urban transport like other ensemble methods, [25]. is sensitive to hyperparameter tuning, which requires careful selection. Although it is faster than classical gradient boosting, it can be slower than LightGBM on very large datasets.

1.5. Factor Analysis of Delays in Public Transport

It is the process of identifying and quantifying the causes that influence schedule deviations. Using XAI and models such as XGBoost [26], it is possible to go beyond simple correlation and identify a hierarchy of factors. Studies show that, contrary to intuitive ideas, operational and temporal factors (time of day, day of the week, congestion of previous sections) often have a much greater impact on delays than weather conditions. This allows management efforts to focus on the most significant problems. Factor analysis allows us to identify not only the importance of individual variables, but also their interactions [27]. For example, SHAP can show that the impact of snowfall is minimal at night but becomes critical during rush hour, quantifying this synergistic effect. The quality of factor analysis depends entirely on the completeness of the data. If an important factor (such as a mass event or a traffic accident) was not included in the dataset, its impact will not be taken into account, and the model may incorrectly attribute the delay to other, less significant factors.

1.6. Purpose and Objectives of the Work

Purpose: To identify and quantify critical factors affecting passenger transport reliability using an interpretive machine learning approach to prioritize management decisions, providing practical tools to improve the reliability of transport services in a smart city.

Objectives:

- To develop and train a gradient boosting model to predict public transport delays based on an integrated set of operational and weather data for passenger transport.
- To conduct factor analysis using interpretive machine learning methods to identify and rank the causes of delays.
- To quantify and compare the impact of operational (temporal, contextual) and weather factors on transport reliability.
- To formulate scientifically based recommendations for transport operators aimed at improving service reliability by focusing on the most significant factors.

The model presented in our paper aims to address this challenge, serving as a bridge between theory and practice. It is not an abstract algorithm, but a full-fledged decision support system that takes theoretical developments in the field of machine learning (XGBoost, XAI) and transforms them into specific, measurable recommendations for transportation engineers and managers, encouraging them to move from reactive response to proactive, scientifically based management of the transportation system.

2. Related Works

Modern machine learning methods are designed not just to create “black boxes” for forecasting, but also to provide deep, interpretive understanding of complex urban systems. Recent research at the interface of information technology, computer science, and passenger transportation indicates the importance of studying the interaction of temporal, contextual, and weather factors, using machine learning methods, emphasizing their significant scientific and practical value.

2.1. Analysis of the Use of Boosting Models in the Organization of Passenger Transportation

Gradient boosting models are a class of ensemble machine learning algorithms that sequentially build weak models

(usually decision trees) in such a way that each subsequent model corrects the errors of the previous one. This makes them extremely accurate and successful for prediction tasks. Gradient Boosting Regression Tree (GBRT or GradientBoosting) is the fundamental algorithm on which more modern implementations are based. It sequentially adds decision trees, each of which is trained on the residuals (errors) of the previous ensemble.

It is most often used to predict train arrival and departure times [28,17]. GBRT is the basis for more modern and optimized models. In [29,21], the nonlinear effect of the built environment on the distance traveled by passengers was investigated and arrival and departure times were predicted. Here, GBRT was used to investigate complex, nonlinear relationships between urban environment characteristics (stop density, accessibility, etc.) and the distance traveled by passengers. The GBRT model is able to detect threshold effects, after which the influence of a certain factor changes [29,21]. It is also used as one of the basic methods for predicting train delays. In general, GBRT is most often used as one of the basic algorithms for comparison with newer models, such as XGBoost. The model incrementally builds regression models by successively adding a simple parametric function to the current residuals [30,17], which provides it with high accuracy, the ability to model complex nonlinear dependencies and threshold effects, as well as flexibility in choosing loss functions. Among the disadvantages of GBRT are its slow operation on large data sets and the tendency to over-training without careful parameter tuning. However, it is successfully used in predicting delays, arrival times, passenger flow on individual routes or stops, i.e. it is effective in analyzing the impact of urban planning on transport behavior.

XGBoost (eXtreme Gradient Boosting) is a highly efficient and optimized implementation of gradient boosting that has become the industry standard for working with tabular data due to its speed and accuracy. In passenger transportation, it has found its application in passenger flow forecasting [31,19], determining bus occupancy, calculating arrival and departure times, and analyzing delay factors. In [31,19], XGBoost was used to forecast passenger flow between pairs of stations, taking into account time slots, holidays, and days of the week. XGBoost is also used to create explainable models of on-board passenger flow forecasting, which allows them to be deployed on devices with low computing capabilities [31,13]. The high speed of XGBoost was achieved due to parallel data processing, and the built-in regularization (L1 and L2) helped to avoid overfitting, allowed to effectively handle missing values, which contributed to achieving a high score of the importance of factors [31,13]. It is considered effective for tasks where features such as route ID, stop ID or day of the week are key [31,28]. XGBoost is widely used to model the choice of a mode of transport by passengers, demonstrating better accuracy compared to classic logit models [32,30,29]. In combination with SHAP, it is used to detect nonlinear and interactive effects of various factors [32,31]. The model is also used to analyze the causes of delays, in particular to compare the impact of weather and operational factors on flights [33,25]. XGBoost is characterized by high performance, built-in regularization, flexibility, the ability to handle missing values, and high interpretability when used with SHAP. However, XGBoost is more complex to configure compared to simpler models and consumes significant computational resources during training. XGBoost is ideal for building accurate models for predicting delays, vehicle occupancy, and passenger flows. Its interpretability (via SHAP) makes it a valuable tool for analyzing critical factors in passenger transport reliability.

LightGBM (Light Gradient Boosting Machine) is a gradient boosting implementation from Microsoft that uses unique techniques (Gradient-based One-Side Sampling and Exclusive Feature Bundling) to significantly accelerate the learning process and memory efficiency. It is used as a fast and efficient alternative to XGBoost for predicting passenger flow based on ticket data. The model has demonstrated the best processing speed while maintaining high accuracy [34,31,19]. When predicting passenger flow between pairs of stations, it has shown extremely high learning speed and low memory consumption. It is used in tasks where the speed of processing large data sets is critical. For example, for predicting passenger flow based on millions of ticket records, LightGBM has shown itself to be significantly faster than XGBoost while maintaining high accuracy [34,19]. It is also used in models that consider external factors, such as weather and flight information [33,16]. It also works well on very large datasets, supports parallel and distributed learning, but is prone to overtraining on small datasets and is very sensitive to hyperparameter tuning. It is indeed the optimal choice for systems that require fast predictions on large data streams, or near-real-time predictions, for example, for operational traffic management.

Summarizing the general overview of boosting models, their main advantages (high accuracy, sufficient speed and resistance to overtraining), which makes them effective and extremely useful for transportation tasks, where categorical factors (route, vehicle type, day of the week) are of great importance, Table 1 was formed with the advantages, disadvantages and application possibilities of the listed models.

2.2. Analysis of Boosting Models to Determine Reliability Factors (Delays) of Passenger Transportation

Here we focus on the works of [29,25,19,16,13], where the main goal was not only to predict but also to explain the reasons for such results, for example, delays, which is directly relevant to our study. The analysis of critical reliability factors is a key task where boosting models, especially in combination with interpretation methods such as SHAP, demonstrate enormous value. The model [13] takes as input the history of passenger flows (PF) at the last L stops and at the same stop during the previous L trips. As additional features in the model [13], time of day, day of the week, month are used. Factor prioritization is performed using SHAP, which found that the most important factor is the passenger flow at the immediately preceding stop (pt-1). This is logical, since the vehicle occupancy hardly changes between neighboring stops. Next in importance are the stop identifiers, which reflect the spatial context. The model [13] was designed to work "on-board", i.e. on low-power devices, which proves the effectiveness of XGBoost for practical applications.

The long-term model [16] used categorical data (stop ID, line ID, trip ID) and numeric/binary data (time, day of the week). It also used weather data (hourly and daily) and flight information. An explainability analysis of the model [16] showed that the most useful variables for long-term forecasting were the embeddings of the stop identifier, the line, and the trip identifier, which implicitly contains information about the time of day. Important for our research is the conclusion of [16] about the dominance of context (where?) and time (when?).

Table 1. Comparison of boosting models by predicted values, their advantages, disadvantages, and opportunities

Boosting model	Parameter	Description	Accuracy and Features	Advantages	Disadvantages	Application possibilities	References
GBRT	Arrival and departure times	A basic algorithm that consistently corrects errors of previous models	Delay prediction	Flexibility, good accuracy	Slow, prone to overfitting	Basic modeling of delays and passenger flows	[17]
	The impact of development on travel distance	Investigating the nonlinear impact of urban environment on passenger travel distance	R^2=0.86; threshold effects detected	Ability to model complex nonlinear relationships	Slower than newer implementations	Analysis of the impact of urban planning on transport behavior	[21]
XGBoost	Forecasting passenger traffic between stations	Optimized implementation of GBRT with regularization and parallelization, comparing XGBoost and LightGBM	MAE=5.561	High accuracy, works well with different data types, speed, accuracy, handling gaps, interpretability (with SHAP)	Slower than LightGBM, difficult to set up	Detailed analysis of reliability factors, accurate forecasting, planning and dispatching systems	[19]
	Choice of intercity transport mode	Predicting the choice between car, bus, train and high-speed rail.	Best accuracy (GMPCA 81.59%)	High accuracy, interpretability with SHAP	Difficulty to set up	Modeling of passenger behavior, strategic planning	[29]
	Airline delays	Analyzing the impact of weather and operational factors on delays.	Prediction accuracy 76.82-79.14%	Good at handling complex relationships	Limited to one factor, operational factors are more important	Delay risk assessment, comparative analysis of factors	[25]
	Passenger traffic on board	Predicting passenger flow at next stops for on-board systems.	MAPE ≈ 24.05%	Efficient on low-power devices, interpretable	Accuracy slightly lower than LSTM	Real-time passenger and driver information systems	[13]
	Severity of road accidents involving cyclists	A hybrid approach of XGBoost-SHAP and statistical model for analyzing accident severity factors.	McFadden's R^2 = 0.52	Identifies complex interactions, reduces data dimensionality	Requires two modeling stages	Traffic safety analysis, development of measures to reduce injuries	[15]
	Choice of intercity transport mode	Investigating the nonlinear and interactive effects of key factors on transportation choice.	Most important factor is distance	High accuracy in explaining complex effects	Difficult to set up	Detailed planning of transport corridors	[30]
	Impact of high-speed rail on land value	Analyzing the nonlinear relationship between proximity to stations and land value using XGBoost-SHAP.	Nonlinear and threshold effects detected	Ability to model spatial nonlinearities	Resource-intensive when training on very large data sets	Urban planning around transport nodes	[31]
	Impact of transit-oriented development (TOD) on rental value	Investigating the nonlinear and interactive effects of transit-oriented development (TOD) on rent.	Complex relationships detected	Deep analysis of interactions	Difficult to set up	Evaluation of the effectiveness of TOD projects	[32]
LightGBM	Passenger traffic between stations	A fast version of GBRT optimized for big data.	Training time, MAE	Very high speed, low memory usage	Risk of overfitting on small data sets	Real-time operational forecasting systems	[19]
	Bus occupancy	Forecasting occupancy considering weather and flights.	MAE ≈ 5.8 (with weather)	Good at handling external (exogenous) variables	Difficult to set up	Dynamic timetable management	[16]
GBoost, XGBoost	Train arrival and departure times	Online forecasting of arrival and departure times of passenger trains at each station.	XGBoost is the best model in 21 out of 36 cases	Comparison of multiple models to choose the best one	ANN did not perform well	Operational railway traffic management	[17]

The study also showed that hourly meteorological data give better results than daily, but the improvement is insignificant, which again emphasizes the secondary role of weather. Ref. [25], although it concerns air transport, provides

a perfect analogy for passenger transport in a city. They found that departure time and carrier are significantly more important predictors of delays than most weather conditions, such as precipitation. This conclusion about the greater importance of operational context (who is operating the flight and when) than the influence of the external environment was important in building our hypothesis. In [13,6], the key predictors were stop and route identifiers, since the transport system is not homogeneous, since each stop and route has unique characteristics (problematic intersections, large passenger-generating objects). Authors [29,19] emphasize the crucial role of time slots, day of the week, month, and distance (which is closely related to travel time).

Ref. [13] showed that the most important predictor is the passenger flow at the previous stop. This demonstrates that the system has a "memory": the current state strongly depends on the state at the previous step.

Table 2 groups the use of boosting models by the parameters studied, the prioritization of factors, and important achievements (results obtained) in the organization of passenger transportation.

Table 2. Comparison of boosting models according to the investigated parameters, prioritization of reliability factors, and important achievements (results obtained) in the organization of passenger transportation

Parameter	Description	Prioritization of factors	Other important aspects	References
Onboard passenger flow	Predicting the number of passengers on a bus at the next stop using XGBoost	1. Passenger flow at the previous stop. 2. Stop identifiers (spatial context). 3. Passenger flow history at the same stop.	The model is optimized to run on low-power devices	[13]
Bus occupancy	Predicting the occupancy at the stop level using LightGBM, XGBoost and deep learning	1. Stop ID. 2. Route line. 3. Trip ID (implicitly contains time of day).	Hourly weather data improves the forecast, but only slightly (by 7.5% in absolute terms). Flight information is only useful for routes near the airport	[16]
Passenger flow between pairs of stations	Predicting the number of passengers between two specific stops using XGBoost, LightGBM	1. Time slot. 2. Day of the week/holiday. 3. Month.	Weather conditions were not considered as a major factor; focus on operational data	[19]
Flight delays	Analysis of the impact of weather and operational factors (XGBoost).	1. Time of departure. 2. Carrier. 3. Precipitation.	Operational factors have a greater impact than weather.	[25]
Choice of intercity transport mode	Analysis of factors affecting transport choice (XGBoost).	1. Travel distance. 2. Travel time. 3. Income.	The model detects nonlinearities and thresholds (e.g., after 100 km, railways are preferred).	[29]

The above analysis allowed us to put forward two hypotheses.

Hypothesis 1: The dominance of deterministic system dynamics over stochastic factors.

Hypothesis Statement. The performance of the urban public transport system, specifically in terms of schedule adherence, is fundamentally governed by a deterministic operational baseline. This baseline is defined by two primary sets of endogenous factors: (a) predictable temporal cycles (diurnal, weekly, and seasonal variations in passenger flow and general traffic) and (b) the static network topology (route structure and stop location).

Hypothesis 2. Meteorological conditions as a non-linear, state-dependent perturbation agent

Hypothesis Statement. The influence of meteorological factors on public transport delays is not linear or independent but acts as a non-linear, state-dependent perturbation agent. The magnitude of delay induced by adverse weather is conditional upon the system's current state within its deterministic operational baseline.

3. Materials and Methods

3.1. eXtreme Gradient Boosting (XGBoost) Method

To achieve this goal, the eXtreme Gradient Boosting (XGBoost) method was used. XGBoost is an implementation of the gradient boosting algorithm, which belongs to the class of ensemble machine learning methods. Its main principle is to sequentially build weak learning models (usually decision trees) in such a way that each subsequent tree is trained on the errors (pseudo-residuals) of the previous ensemble, thereby iteratively improving the overall prediction accuracy.

The key feature of XGBoost is the optimized objective function, which is minimized at each step of building a new tree [30-33]. This function consists of two components: a loss function, which measures the discrepancy between the predicted and actual values, and a regularization term, which penalizes for the complexity of the model, preventing overtraining.

The objective function at step t has the form (1) [34,30-33,25,24]:

$$\mathrm{Obj}^{(t)} = \sum_{i=1}^{n} l\left(y_i, \widehat{y_i^{(t-1)}} + f_t(x_i)\right) + \Omega(f_t) \quad (1)$$

where $l(y_i, \widehat{y_i})$ – differential convex loss function that measures the error for *i*-th an observation.

y_i – actual value of the target variable.

$\widehat{y_i^{(t-1)}}$ – ensemble prediction of *t-1* 1 trees for the *i*-th observation.
$f_t(x_i)$ – function of the new tree added at step *t*.
$\Omega(f_t)$ – regularization term that controls the complexity of the model.

XGBoost uses a second-order Taylor expansion to approximate the objective function, which allows it to be optimized for any differentiable loss function (2) [34,33,22]:

$$\text{Obj}^{(t)} \approx \sum_{i=1}^{n} \left[l\left(y_i, \widehat{y_i^{(t-1)}}\right) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t) \quad (2)$$

where g_i та h_i – the first and second derivatives (gradient and Hessian) of the loss function, respectively.

Regularization term for a tree f_t is defined as (3) [34,33]:

$$\Omega(f_t) = \gamma T + \frac{1}{2} \lambda \sum_{j=1}^{T} w_j^2 \quad (3)$$

where T – number of leaves in a tree.
w_j – weight (forecast) j-th leaves.
γ and λ – hyperparameters controlling L1 and L2 regularization respectively.

This approach allows XGBoost to efficiently find the optimal tree structure and leaf weights, providing high prediction accuracy and resistance to overtraining.

3.2. Dataset Preparation

The work used a large set of operational and meteorological data for a city with a population of about 1 million. The operational data was formed by a normalized database (DB) of vehicle traffic schedules, the structure of which is shown in Fig. 1, which allowed to avoid data duplication and simplified the execution of queries. The given structure consisted of 5 tables: Routes, Vehicles, Stops, Trips, StopTimes and was characterized by flexibility and efficiency.

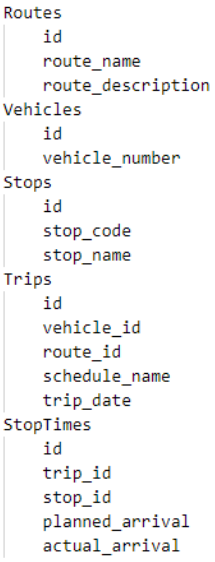

Fig.1. Schema of the traffic schedule execution database

In our operational database, the Routes table was described by – id: Primary key, route_name: Route name, route_description: Route description. The Vehicles table was described by – id: Primary key, vehicle_number: Vehicle number. The Stops table was described by – id: Primary key, stop_code: Unique stop code, stop_name: Stop name. The Trips/Schedules table was described by – id: Primary key, vehicle_id: Foreign key referring to Vehicles, route_id: Foreign key referring to Routes, schedule_name: Schedule name, trip_date: Trip date. Arrival time at stops. The StopTimes table was described by – id: Primary key, trip_id: Foreign key referring to Trips, stop_id: Foreign key referring to Stops, planned_arrival: Planned arrival time in ISO format (YYYY-MM-DD HH:MM:SS), actual_arrival: Actual arrival time. The database with data for nine months contained 716959 records and allowed easy execution of complex analytical queries.

Meteorological data for the city were generated in the weather database, the column structure of which is shown in Fig. 2. The meteorological data for the city contained hourly weather conditions in the city from January 1979 to July 2025, which had 418178 records.

Calculation of the average delay of the schedule execution at each stop has two features. Since the average delay can be interpreted differently: only as cases of delay and, in general, as a deviation from the schedule, we will consider both cases.

```
WEATHER_COLUMNS_TO_ADD = [
    («dt», «INTEGER»),
    («temp», «REAL»),
    («visibility», «INTEGER»),
    («dew_point», «REAL»),
    («feels_like», «REAL»),
    («temp_min», «REAL»),
    («temp_max», «REAL»),
    («pressure», «INTEGER»),
    («humidity», «INTEGER»),
    («wind_speed», «REAL»),
    («wind_deg», «INTEGER»),
    («wind_gust», «REAL»),
    («rain_1h», «REAL»),
    («rain_3h», «REAL»),
    («snow_1h», «REAL»),
    («snow_3h», «REAL»),
    («clouds_all», «INTEGER»),
    («weather_id», «INTEGER»),
    («weather_main», «TEXT»),
    («weather_description», «TEXT»),
    («weather_icon», «TEXT»)
]
```

Fig.2. Weather database schema weather

A. Average delay (only cases of delay)

The SQL query for calculating the average delay of the schedule execution at each stop (only cases of delay) is shown in Fig. 3. This query calculates the average delay time, but takes into account only those cases when the vehicle arrived later than planned. In this case, cases of arrival on time or earlier were ignored.

```
    SELECT
    S.stop_name AS "Stop name",
    S.stop_code AS "Stop code",
    CAST(AVG(strftime('%s', ST.actual_arrival) - strftime('%s', ST.planned_arrival)) AS INTEGER) AS "Average delay (seconds)",
    COUNT(ST.id) AS "Number of recorded delays"
FROM
    StopTimes AS ST
JOIN
    Stops AS S ON ST.stop_id = S.id
WHERE
    strftime('%Y', ST.planned_arrival) = '2025'
    AND ST.actual_arrival IS NOT NULL
    AND ST.planned_arrival IS NOT NULL
    AND strftime('%s', ST.actual_arrival) > strftime('%s', ST.planned_arrival)
GROUP BY
    ST.stop_id
ORDER BY
    "Average delay (seconds)" DESC;
```

Fig.3. SQL query to calculate average schedule execution delay at each stop (late cases only)

B. Average deviation from the schedule (two-way)

The SQL query for calculating the average delay of the schedule execution at each stop (two-way - ahead and behind) is shown in Fig. 4. This query shows the general picture of punctuality. It calculates the average difference between the actual and planned arrival time, taking into account both lateness (positive value) and arrival ahead of schedule (negative value). Moreover, a positive result means that on average vehicles are late at this stop, a negative result means that on average they arrive earlier.

To combine data from different sources, a Python script was used, which connected data on traffic in the city with weather conditions and consisted of two main parts, where the schema modification was done by adding new columns for weather to the StopTimes table, and data enrichment was done by searching for relevant weather data for each arrival record and updating the table.

```sql
SELECT
    S.stop_name AS "Stop name",
    S.stop_code AS "Stop code",
    CAST(AVG(strftime('%s', ST.actual_arrival) - strftime('%s', ST.planned_arrival)) AS INTEGER) AS "Average deviation (seconds)",
    COUNT(ST.id) AS "Number of arrivals"
FROM
    StopTimes AS ST
JOIN
    Stops AS S ON ST.stop_id = S.id
WHERE
    strftime('%Y', ST.planned_arrival) = '2025'
    AND ST.actual_arrival IS NOT NULL
    AND ST.planned_arrival IS NOT NULL
GROUP BY
    ST.stop_id
ORDER BY
    "Average deviation (seconds)" DESC;
```

Fig.4. SQL query to calculate average schedule execution delay at each stop (two ways)

For this, the SQLite function – attach database was used, which allowed us to query two databases simultaneously, as if they were part of one. Connecting to the databases was done using a script that allowed us to simultaneously connect to the main database tram_schedule.db and attach weather_lviv.db to it under a temporary alias weather_db.

To insert data, a place was first created for it by adding columns. To avoid an error when re-running the script when the columns already exist, an *alter table* script was used for each required column (commands wrapped in try...except). Finding the closest weather required finding a method because the data had different time characteristics and did not match.

The method where each record in StopTimes, where there was an actual arrival time (actual_arrival), for example, '2025-03-15 11:06:00', was converted to a Unix timestamp and the associated table weather_db.weather was queried was very time-consuming. Here, using *order by abs(dt - ?)* we found the row where the difference between the weather time (dt) and the arrival time was minimal. LIMIT 1 ensured that we only got one, the closest record. Next, the data was updated, where the found weather data was written to the corresponding new columns for the current row in StopTimes.

To optimize transactions, all update operations were performed inside one large transaction (or several, if there is a lot of data), which significantly accelerated the process compared to thousands of individual records. The progress was monitored using a script that displayed the processing progress every 100 processed rows so that we could see that it was not hanging. Otherwise, because we had a very large amount of data in StopTimes (>716000), it took a very long time, approximately 1 min = 1 thousand records.

The best results were obtained by the method with rounding the time to the nearest hour and performing a direct query, which dramatically accelerated the process. According to this method, the weather condition, which was recorded every hour at an even hour, was linked to the time from the actual_arrival column by simply rounding it to the nearest whole hour (08:13→08:00) (Fig.5), and only then was it converted to a Unix timestamp and only then was a direct query made to the weather table with this value in the dt column.

```python
def round_to_nearest_hour(dt_obj: datetime.datetime) -> datetime.datetime:
    if dt_obj.minute >= 30:
        dt_obj += timedelta(hours=1)
    return dt_obj.replace(minute=0, second=0, microsecond=0)

def enrich_stop_times_with_weather(tram_conn):
    print(f"Database connection '{WEATHER_DB_FILENAME}'...")
    try:
        tram_conn.execute(f"ATTACH DATABASE '{WEATHER_DB_FILENAME}' AS weather_db")
        print("- Weather database successfully attached.\n")
    except sqlite3.OperationalError as e:
        print(f"Error connecting to database: {e}")
        return

```

Fig.5. Rounding a datetime object to the nearest whole hour and joining the database

Implemented logic for rounding time to the nearest hour (e.g., 10:29 →10:00, 10:30 → 11:00). This allows for instant lookups on the indexed dt column instead of a slow scan. A batching method was used, where the script now processed data in batches of 100 records (Fig.6).

```
column_names_str = ", ".join(COLUMN_NAMES)
set_clause = ", ".join([f"{name} = ?" for name in COLUMN_NAMES])
update_sql = f"UPDATE StopTimes SET {set_clause} WHERE id = ?"

start_time = time.time()
total_updated_count = 0

for i in range(0, total_rows, BATCH_SIZE):
    batch = rows_to_update[i:i + BATCH_SIZE]
    updates_to_commit = []

    for stop_time_id, actual_arrival_str in batch:
        try:
            dt_obj = datetime.datetime.strptime(actual_arrival_str, '%Y-%m-%d %H:%M:%S')
            rounded_dt = round_to_nearest_hour(dt_obj)
            arrival_timestamp = int(rounded_dt.timestamp())
            weather_cursor = tram_conn.cursor()
            weather_cursor.execute(
                f"SELECT {column_names_str} FROM weather_db.weather WHERE dt = ?",
                (arrival_timestamp,)
            )
            weather_data = weather_cursor.fetchone()

            if weather_data:
                update_values = list(weather_data) + [stop_time_id]
                updates_to_commit.append(tuple(update_values))

        except (ValueError, TypeError):
            continue

    if updates_to_commit:
        cursor.executemany(update_sql, updates_to_commit)
        tram_conn.commit()
    total_updated_count += len(updates_to_commit)
    elapsed_time = time.time() - start_time
    speed = total_updated_count / elapsed_time if elapsed_time > 0 else 0

    print(
        f"  - Processed {i + len(batch)} / {total_rows} records. "
        f"Speed: {speed:.1f} зап/сек."
    )

print("\nThe process is complete!")
print(f"Successfully updated {total_updated_count} from {total_rows} records.")
print(f"Total execution time: {time.time() - start_time:.2f} seconds.")
```

Fig.6. Batch processing script for processing data in batches of 100 records

After each processed batch (100 rows), a *commit* was executed, which saved the progress in the database. This allowed us to safely stop and restart the script at any time - it would continue working from where it stopped. To achieve idempotence, so that the script could be safely run again and it would update only those rows where weather data had not yet been added, *where temp is null* was used. To achieve idempotence, so that the script could be safely rerun and would only update rows where weather data had not yet been added, WHERE temp IS NULL was used.

4. Results and Discussion

4.1. Loading Weather Parameters for Analysis

In order to find out how the weather affects the quality and accuracy of the schedule, the Python libraries pandas were used for data manipulation and scipy for statistical calculations.

The weather parameters used in the study are as follows (Fig.7):

- Temperature: temp
- Visibility: visibility
- Dew point: dew_point
- Temperature: that feels_like
- Pressure: pressure
- Humidity: humidity
- Wind speed: wind_speed
- Wind direction: wind_deg
- Wind gusts: wind_gust
- Amount of rain in 1 hour: rain_1h
- Amount of rain in 3 hours: rain_3h
- Amount of snow in 1 hour: snow_1h
- Amount of snow in 3 hours: snow_3h
- Percentage of cloud: cover clouds_all

```
    WEATHER_PARAMS_TO_ANALYZE = [
    'temp',
    'visibility',
    'dew_point',
    'feels_like',
    'pressure',
    'humidity',
    'wind_speed',
    'wind_deg',
    'wind_gust',
    'rain_1h',
    'rain_3h',
    'snow_1h',
    'snow_3h',
    'clouds_all'
]

def load_and_prepare_data(db_path):
    print(f"Loading data from '{db_path}'...")
    try:
        conn = sqlite3.connect(db_path)

        query_cols = "planned_arrival, actual_arrival, " + ", ".join(WEATHER_PARAMS_TO_ANALYZE)
        query = f"SELECT {query_cols} FROM StopTimes WHERE temp IS NOT NULL"

        df = pd.read_sql_query(query, conn)
        conn.close()

        if df.empty:
            print("Error: No data found for analysis. Check if the StopTimes table contains weather data.")
            return None

        print(f"Uploaded {len(df)} Records. Data preparation...")
```

Fig.7. Loading weather parameters for analysis

Using data on the schedule of stops and the actual arrival time of vehicles, delays were obtained for each stop. The calculation of the deviation from the schedule was carried out in seconds (actual_arrival - planned_arrival). A positive value meant a delay; a negative value meant an early arrival. Time conversion and calculation of the deviation were performed according to the following algorithm (Fig.8).

```
        df['planned_arrival'] = pd.to_datetime(df['planned_arrival'], errors='coerce')
        df['actual_arrival'] = pd.to_datetime(df['actual_arrival'], errors='coerce')
        df.dropna(subset=['planned_arrival', 'actual_arrival'], inplace=True)

        df['delay_seconds'] = (df['actual_arrival'] - df['planned_arrival']).dt.total_seconds()
```

Fig.8. Time conversion and deviation calculation

When loading weather data, in order to correctly handle missing values, for example, NaN for rain/snow, they were filled with zeros, since the absence of a record most often means the absence of precipitation. The processing of missing values for precipitation and wind gusts was performed according to the following algorithm (Fig.9).

```
        for col in ['rain_1h', 'rain_3h', 'snow_1h', 'snow_3h', 'wind_gust']:
            if col in df.columns:
                df[col] = df[col].fillna(0)

        print("Data has been successfully prepared.\n")
        return df
```

Fig.9. Handling missing values for precipitation and wind gusts

4.2. Conducting a Correlation Analysis between Delay and Weather Parameters, its Results

Correlation analysis was performed for each of the 14 weather parameters, where the Pearson correlation coefficient (R) was calculated. This coefficient shows the presence of a linear relationship between two variables and is in the range from -1 to +1. The p-value, an indicator of statistical significance, was also calculated. A low value (usually < 0.05) means that the detected correlation is unlikely to be random (Fig.10).

The results of the correlation analysis (deviation from the graph vs. weather) were derived with the following interpretation of the results.

```python
def analyze_correlations(df, params):
    print("Correlation analysis is performed...")
    results = []

    for param in params:
        subset = df[['delay_seconds', param]].dropna()
        if len(subset) < 100 or subset[param].nunique() < 2:
            correlation, p_value = 0, 1.0
        else:
            correlation, p_value = pearsonr(subset['delay_seconds'], subset[param])

        results.append({
            'Weather parameter': param,
            'Correlation (r)': correlation,
            'p-value': p_value
        })

    results_df = pd.DataFrame(results)
    results_df['abs_correlation'] = results_df['Correlation (r)'].abs()
    results_df = results_df.sort_values(by='abs_correlation', ascending=False).drop('abs_correlation', axis=1)

    return results_df
```

Fig.10. Script for calculating correlation between delay and weather parameters

The Pearson correlation coefficient (R) shows the strength of the linear relationship between two variables:

- Values close to +1: Strong positive relationship, e.g. more rain, more delay
- Values close to -1: Strong negative relationship, e.g. better visibility, less delay (or earlier arrival).
- Values close to 0: Weak or no linear relationship.

The p-value coefficient shows the statistical significance of the result:

- If the p-value < 0.05, the result is considered statistically significant, meaning that the detected relationship is unlikely to be random.
- If p-value > 0.05, the relationship may be random, and conclusions should not be drawn.

The report displayed the results in a sorted table, from the strongest relationships (by modulus) to the weakest, along with an interpretation. Key findings were formulated as follows:

- The biggest factor that affects the increase in delay is: Weather parameter, which reached a positive value of R.
- The biggest factor that affects delay reduction (i.e., punctuality) is: Weather parameter that reached a negative value

Having data on delays, it was identified which weather phenomena have the greatest impact on public transport punctuality thanks to the result obtained (Table 3).

Table 3. Results of correlation analysis of deviation from schedule vs weather

Weather parameter	Correlation (R)	p-value
wind_deg	0.043905	8.686169e-303
clouds_all	0.031937	3.960501e-161
rain_1h	0.030432	1.739758e-146
feels_like	-0.024181	3.381298e-93
snow_1h	0.022521	4.323183e-81
temp	-0.021854	1.825826e-76
pressure	-0.016388	8.737390e-44
wind_speed	0.013911	4.955947e-32
visibility	0.013764	9.068205e-30
dew_point	-0.012790	2.480363e-27
wind_gust	0.011053	8.035519e-21
humidity	0.007969	1.503167e-11
rain_3h	0.000000	1.000000e+00
snow_3h	0.000000	1.000000e+00

As can be seen from Table 3, the correlation analysis revealed that the impact of any individual weather parameter on transport delay is very weak. The key point is statistical significance vs practical significance.

Comparing the two columns of Table 3 allows us to draw the following conclusions:

- Regarding the correlation coefficient (R), all values are extremely close to zero (the largest in magnitude is 0.0439). This suggests that the linear relationship between weather and delays is very weak.
- Regarding the p-value, almost all values are tiny (e.g., 8.68e-303), indicating that the weak associations found are statistically significant, meaning they are unlikely to have arisen by chance.

There is no contradiction here, because due to the huge amount of data (~716 thousand records), even a microscopic, practically insignificant effect becomes statistically visible.

Let's consider each parameter in detail.

The wind direction wind_deg (R = 0.044) has the strongest positive correlation. This technically means that with a change in wind degree (e.g., from 10° to 200°) there is a barely noticeable tendency for delay to increase. Of course, this result is most likely an artifact, because wind direction is a cyclical quantity, and linear correlation is not very suitable for it. Perhaps certain wind directions (e.g., westerly) coincide with routes where, in principle, there is more congestion, and this creates a false connection, so this parameter can be ignored.

Cloudiness, rain and snow clouds_all, rain_1h, snow_1h (R ≈ 0.02-0.03) has a very weak positive relationship, i.e. when it rains/snows or the sky is cloudy, there is a microscopic tendency for delays to increase. It is natural and intuitive that bad weather slows down traffic a little, but this effect is so small that knowing that it is raining adds almost nothing to our ability to predict delays.

Temperature that feels like, actual temperature, pressure and dew point feels_like, temp, pressure, dew_point (R ≈ -0.01 to -0.02) have a very weak negative relationship, i.e. when the temperature is higher, there is a microscopic tendency for delays to decrease, and of course, traffic moves a little faster. A possible explanation is that in warmer weather, fewer people use their personal cars (walk, ride bicycles), which reduces overall traffic and slightly speeds up vehicles. However, again, the effect is negligible.

The amount of rain and snow in 3 hours rain_3h, snow_3h (R = 0.000000) has absolutely zero correlation. This is most likely a problem in our data, since in our weather database these columns are almost always empty (NULL), and our script filled them with zeros, so when one of the variables is almost unchanged, the correlation cannot be calculated. These parameters, if they cannot be checked, should be excluded from the analysis.

So, based on a simple linear analysis, weather is not a dominant factor in the formation of delays, but other important indicators such as traffic, time of day, day of the week, accidents, breakdowns can have a much stronger influence.

However, the results of correlation analysis are not a verdict on machine learning, as correlation analysis is only the beginning of the investigation. Pearson correlation only looks for linear relationships, but the world is much more complex. Machine learning tools are great at finding complex, nonlinear relationships that simple correlation analysis misses. A well-known nonlinear relationship is that a light rain (rain_1h = 1 mm) will have no effect on delay at all, while a downpour (rain_1h = 10 mm) will cause a sharp, exponential increase in delays due to flooding and zero visibility. Linear correlation will not see this, but machine learning tools like XGBoost will. A model built on the time characteristics of the stop (time of day, day of week, month, weekday/holiday), route characteristics (stop ID, trip ID, route number), schedule data, and weather data can calculate a forecast that the expected delay at stop No. will be 180 seconds. Although the model will not "change the route schedule," with this information, the transportation company can inform passengers in advance about expected delays, release an additional backup vehicle on the line, and adjust the schedule for the next day if extreme weather is forecast. Therefore, gradient boosting (XGBoost) was chosen for this task as a tool that works well on tabular data, is fast to learn, and is easier to interpret.

4.3. Building a Gradient Boosting Model XGBoost

The XGBoost model is one of the best models for working with tabular data (like ours). It regularly wins machine learning competitions. The model is optimized for fast learning, even on large datasets like ours (>716,000 records). XGBoost shows exactly those factors (features) that it considered most important for obtaining a forecast.

To build the XGBoost model, a Python script and the corresponding libraries xgboost, pandas, scikit-learn, matplotlib, seaborn were used. Building the transport delay forecasting model consisted of the following 6 steps:

- Step 1. Data loading and preparation.
- Step 2. Feature engineering.
- Step 3. Defining features (x) and target variable.
- Step 4. Data partitioning and training xgboost model.
- Step 5. Model quality assessment.
- Step 6. Feature importance analysis.

For data visualization, another step was added – Step 7. Deep analysis of the model with SHAP. Let's consider each of the steps sequentially.

Step 1. Data loading and preparation, handling outliers and missing values.

During the implementation of step 1, data loading and preparation were performed (Fig. 11). To control the correctness of the program code execution, messages were displayed on the screen about the number and name of the step (Step 1/6, Data loading and preparation...), the number of loaded records and information about the time it took to complete data preparation, as well as information about the error in step 1.

```
print("\n[Step 1/6] Loading and preparing data...")
start_time = time.time()

try:
    conn = sqlite3.connect(f'file:{DB_FILENAME}?mode=ro', uri=True)

    query = "SELECT * FROM StopTimes WHERE temp IS NOT NULL"
    df = pd.read_sql_query(query, conn)
    conn.close()

    print(f"Uploaded {len(df)} records.")

    df['planned_arrival'] = pd.to_datetime(df['planned_arrival'], errors='coerce')
    df['actual_arrival'] = pd.to_datetime(df['actual_arrival'], errors='coerce')
    df.dropna(subset=['planned_arrival', 'actual_arrival'], inplace=True)
    df['delay_seconds'] = (df['actual_arrival'] - df['planned_arrival']).dt.total_seconds()
    for col in ['rain_1h', 'rain_3h', 'snow_1h', 'snow_3h', 'wind_gust']:
        if col in df.columns:
            df[col] = df[col].fillna(0)

    print(f"Data preparation completed in {time.time() - start_time:.2f} sec.")
except Exception as e:
    print(f"Error at step 1: {e}")
    exit()
```

Fig.11. Script for loading and preparing data in an xgboost model

Data quality and completeness are critical for building a reliable forecasting model. Several steps of data cleaning and preparation were applied in this study.

Missing value handling (Imputation) was applied to both the main data and the meteorological data. In the main data, records with missing key values planned_arrival or actual_arrival were completely removed from the dataset, as it was not possible to calculate the target variable (delay) for them. This was done using the dropna() method. In the weather data, special attention was paid to handling missing values in weather variables. For precipitation (rain_1h, snow_1h) and wind gust (wind_gust), missing values (NaN) were replaced with zero (fillna(0)). This imputation method is justified because in meteorological reports, the absence of a record most often means the absence of the corresponding phenomenon (zero precipitation or no gusts).

Outliers processing was performed in accordance with the subject area - transport systems, which by their nature are prone to extreme but real events (serious accidents, sudden breakdowns, collapses due to extreme weather), which manifest themselves in the form of outliers, i.e. very large delay values.

A methodological decision was made not to remove these outliers. Removing such data would lead to the creation of an overly optimistic model, unable to predict precisely those critical situations that are most important for management. The selected algorithm XGBoost, as an ensemble of decision trees, is resistant to outliers. Unlike linear models, where extreme values can significantly shift the regression line, decision trees isolate outliers into separate «leaves», preventing them from influencing the overall structure of the model.

To integrate hourly meteorological data from the OpenWeather source with each specific record of transport arrival, a temporal aggregation method was used. Each arrival record with a precise timestamp (e.g. 17:46) was matched with the weather report corresponding to the nearest full hour (in this case, 18:00).

This approach is standard practice when combining data of different temporal granularities and allows for the most accurate association of weather conditions with the moment of the event. Despite the possible inaccuracies of individual OpenWeather sensors, the use of a large dataset (over 700,000 records) allows the model to capture general statistical patterns, eliminating the impact of random errors in individual measurements.

Step 2. Feature engineering.

During the implementation of step 2, feature engineering was performed (Fig. 12). To monitor the correctness of the program code execution, messages were displayed on the screen about the number and name of the step (Step 2/6, Feature engineering...), as well as information about the duration of feature engineering in step 2.

```
print("\n[Step 2/6] Feature engineering (creating new parameters)...")
start_time = time.time()

df['hour'] = df['planned_arrival'].dt.hour
df['dayofweek'] = df['planned_arrival'].dt.dayofweek
df['month'] = df['planned_arrival'].dt.month
df['dayofyear'] = df['planned_arrival'].dt.dayofyear

df['hour_sin'] = np.sin(2 * np.pi * df['hour'] / 24)
df['hour_cos'] = np.cos(2 * np.pi * df['hour'] / 24)
df['dayofweek_sin'] = np.sin(2 * np.pi * df['dayofweek'] / 7)
df['dayofweek_cos'] = np.cos(2 * np.pi * df['dayofweek'] / 7)

print(f"Feature engineering completed in {time.time() - start_time:.2f} sec.")
```

Fig.12. Script for performing feature engineering in the XGBoost model

To correctly represent the time characteristics, a cyclic encoding method was used for the variables hour and dayofweek. Using these features as prime numbers (for example, from 0 to 23 for hours) is incorrect, since the model perceives the values 23 and 0 as the most distant, although in fact they are adjacent (23:00 and 00:00).

To preserve the cyclic nature of these data, each variable was transformed into two new ones using the sine and cosine functions equation (4) and equation (5) for hours, and equation (6) and equation (7) for days of the week:

$$\text{hour\_sin} = \sin\left(2 \cdot \pi \cdot \frac{\text{hour}}{24}\right) \tag{4}$$

$$\text{hour\_cos} = \cos\left(2 \cdot \pi \cdot \frac{\text{hour}}{24}\right) \tag{5}$$

$$\text{dayofweek\_sin} = \sin\left(2 \cdot \pi \cdot \frac{\text{dayofweek}}{7}\right) \tag{6}$$

$$\text{dayofweek\_cos} = \cos\left(2 \cdot \pi \cdot \frac{\text{dayofweek}}{7}\right) \tag{7}$$

These transformations map the 24-hour cycle to a circle where 23:00 and 0:00 are next to each other. A similar transformation was applied to dayofweek with a period of 7. This approach is a standard best practice in machine learning when working with temporal data and significantly improves the model's ability to correctly capture daily and weekly patterns.

Step 3. Defining features (x) and target variable (y).

During the implementation of step 3, the features (x) and target variable (y) were defined (Fig. 13). To monitor the correctness of the program code execution, messages were displayed on the screen about the number and name of the step (Step 3/6, Defining the data set for training…), as well as information about the number of features on which the model will be trained in step 3.

```
print("\n[Step 3/6] Defining the training dataset...")

features = [
    'stop_id', 'trip_id',
    'hour', 'dayofweek', 'month',
    'hour_sin', 'hour_cos', 'dayofweek_sin', 'dayofweek_cos',
    'temp', 'visibility', 'dew_point', 'feels_like', 'pressure',
    'humidity', 'wind_speed', 'wind_deg', 'wind_gust', 'rain_1h',
    'rain_3h', 'snow_1h', 'snow_3h', 'clouds_all', 'weather_id'
]

X = df[features]
y = df['delay_seconds']

print(f"The model will be trained on {len(X.columns)} features.")
```

Fig.13. Script for Defining features (x) and target variable (y) in XGBoost model

Step 4. Data partitioning and xgboost model training.

During the implementation of step 4, data partitioning and XGBoost model training were performed (Fig. 14). To monitor the correctness of the program code execution, messages were displayed on the screen about the number and name of the step (Step 4/6, Data partitioning and model training...), as well as information about model training with early stopping and the time it took for the model to be successfully trained in step 4.

```
print("\n[Step 4/6] Splitting the data and training the model...")
start_time = time.time()

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

model = xgb.XGBRegressor(
    objective='reg:squarederror',
    n_estimators=1000,
    max_depth=7,
    learning_rate=0.05,
    subsample=0.8,
    colsample_bytree=0.8,
    n_jobs=-1,
    random_state=42,
    eval_metric='mae'
)

model.fit(X_train, y_train,
          eval_set=[(X_test, y_test)],
          early_stopping_rounds=50,
          verbose=False)

print(f"The model was successfully trained in {time.time() - start_time:.2f} sec.")
```

Fig.14. Script for data partitioning and xgboost model training

The hyperparameter optimization of the XGBoost model is largely dependent on the tuning of its hyperparameters and determines its performance. A systematic approach was used to find the optimal configuration, which included splitting the data into training and validation sets. Key hyperparameters, such as n_estimators (number of trees), max_depth (maximum tree depth), learning_rate (learning rate), subsample, and colsample_bytree, were selected to minimize the mean absolute error (MAE) on the validation set. In addition, an early stopping mechanism was used to prevent overfitting. The model was trained iteratively, and if its performance on the validation set did not improve for 50 consecutive iterations (early_stopping_rounds=50), the training process was automatically stopped. This ensures that the model does not overfit to the training data and maintains high generalization ability. The final hyperparameter values (n_estimators=1000, max_depth=7, learning_rate=0.05) are the result of this optimization process.

Step 5. Model quality assessment.

During the implementation of step 5, the quality assessment of the XGBoost model was carried out (Fig. 15). To monitor the correctness of the program code execution, messages about the number and name of the step were displayed on the screen (Step 5/6, Model quality assessment on test data...), the mean absolute error (MAE) values in seconds and the coefficient of determination (R^2) values obtained in step 5 were calculated.

```
print("\n[Step 5/6] Assessing the quality of the model on test data...")
start_time = time.time()

predictions = model.predict(X_test)

mae = mean_absolute_error(y_test, predictions)
r2 = r2_score(y_test, predictions)

print(f"- Mean Absolute Error (MAE): {mae:.2f} seconds")
print(f"(This means that on average the model prediction deviates from the actual delay by {mae:.2f} sec.)")
print(f"- Coefficient of determination (R²): {r2:.2f}")
print(f"(This means that our model explains {r2*100:.0f}% of the variability in delays)")
```

Fig.15. Script for xgboost model quality assessment

As a result of step 5, the model quality score on the test data, calculated as the mean absolute error (MAE), was 250.44 seconds, which means that on average the model forecast deviates from the real delay by 250.44 seconds, and the coefficient of determination (R^2) was 0.72, which meant that the XGBoost model explains only 72% of the delay variability. Despite the obtained values, the constructed XGBoost model is considered to be of very high quality and sufficiently informative [35,33], since using the data provided to it (time, route, weather), it was able to explain 72% of all the reasons for delays. The remaining 28% are random factors that we did not take into account (road accidents, breakdowns, traffic jams due to street closures, the human factor of the driver, etc.) [36,37]. For a real, chaotic system like urban traffic, the indicator 0.72 is a fairly good result. This indicates that the constructed XGBoost model is useful in practice. Although the mean absolute error (MAE) is 4 minutes and 10 seconds, this means that on average, the prediction of the constructed XGBoost model deviates from the actual delay by a little more than 4 minutes. Considering that some vehicle delays on the route reached 30-40 minutes (or even more during collapses), the error of 4 minutes is quite acceptable. This means that the model can reliably distinguish a situation when the delay will be ~5 minutes from a situation when it will be ~20 minutes.

Step 6. Feature importance analysis.

During the implementation of step 6, an analysis of the importance of the features obtained as a result of the work of the XGBoost model was carried out (Fig. 16).

```
print("\n[Step 6/6] Analysis of the importance of features...")

feature_importance = pd.DataFrame({
    'feature': X.columns,
    'importance': model.feature_importances_
}).sort_values('importance', ascending=False)

top_features = feature_importance.head(20)
print("Top 20 most important features according to the model:")
print(top_features)

plt.figure(figsize=(12, 8))
sns.barplot(x='importance', y='feature', data=top_features, palette='viridis')
plt.title('The importance of features for predicting transport delays', fontsize=16)
plt.xlabel('Importance', fontsize=12)
plt.ylabel('Feature', fontsize=12)
plt.tight_layout()
plt.savefig('feature_importance.png')
print("\nFeature importance graph saved to file 'feature_importance.png'")


print("\n--- Example of using a model for forecasting ---")
sample_data = X_test.iloc[[0]].copy()

sample_data['hour'] = 17
sample_data['dayofweek'] = 4
sample_data['snow_1h'] = 1.5
sample_data['temp'] = -2.0
sample_data['visibility'] = 2000

sample_data['hour_sin'] = np.sin(2 * np.pi * 17 / 24)
sample_data['hour_cos'] = np.cos(2 * np.pi * 17 / 24)
sample_data['dayofweek_sin'] = np.sin(2 * np.pi * 4 / 7)
sample_data['dayofweek_cos'] = np.cos(2 * np.pi * 4 / 7)

predicted_delay = model.predict(sample_data)[0]

print(f"For the scenario (Friday evening, snow), the model predicts a delay of: {predicted_delay:.0f} sec.")
```

Fig.16. Script for feature importance analysis of the xgboost model

To monitor the correctness of the program code execution, messages were displayed on the screen about the number and name of the step (Step 6/6, Feature importance analysis...), information about the top 20 most important features obtained by the XGBoost model, the importance of the features themselves for predicting transport delays, as well as information about saving the feature importance graph constructed in step 6 to a file.

Table 4. Top 20 most important features according to the results of the developed XGBoost model

ID	Feature	Importance
4	month	0.141252
5	hour_sin	0.066720
3	dayofweek	0.060233
15	wind_speed	0.059938
7	dayofweek_sin	0.056590
12	feels_like	0.055598
13	pressure	0.053066
8	dayofweek_cos	0.050808
9	temp	0.047060
10	visibility	0.043264
14	humidity	0.042001
1	trip_id	0.040742
6	hour_cos	0.040696
2	hour	0.036016
11	dew_point	0.035909
16	wind_deg	0.034328
22	clouds_all	0.032773
0	stop_id	0.026326
23	weather_id	0.025168
18	rain_1h	0.019187

As a result of step 6, a database table was formed, which included the top 20 most important features according to the results of the developed XGBoost model (Table 4). The results presented in Table 4 are the most important and fully confirm our previous hypotheses about the dominance of not weather, but time and contextual factors and the secondary influence of weather on delays in public transport.

The results presented in Table 4 provide a multifaceted picture that fully supports both hypotheses, revealing the complex interplay between operational and weather factors. First, the results clearly support Hypothesis 1, which assumes the dominance of deterministic system dynamics. The highest positions in the importance ranking are occupied by temporal and contextual factors: month, hour_sin, dayofweek, trip_id, and stop_id. This proves that the expected operational cycles and static network topology form the baseline of system reliability and are the main predictors of delays. Second, and what is the key contribution of our research, the results support Hypothesis 2, which positions meteorological conditions as a nonlinear, state-dependent disturbance agent. As can be seen from Table 4, weather factors such as wind_speed, feels_like, pressure and temp are among the top 10 most important features and have high predictive power. However, their influence is not absolute, but conditional. This actually necessitates a deep analysis of the model using SHAP, which is able to show how the negative effect of adverse weather conditions, such as snow or poor visibility, is manifested when it is insignificant, and when it has a greater impact on the operation of the transport system.

The most important factor is month, which is absolutely logical, because seasonality has a cardinal effect on traffic, among the patterns are winter holidays (December), the beginning of the school year (September), when there are more cars and more passengers, as well as summer vacations, when, on the contrary, there are fewer cars and fewer passengers. Next are the time factors: hour_sin, dayofweek, dayofweek_sin. All of this is directly related to rush hours, weekends or weekdays, which indicates the second most important set of factors. The XGBoost model clearly saw daily and weekly cycles. Spatial features formed the third important group of factors: trip_id, stop_id. It is the specific route and specific stop that are problematic, as they pass through problematic areas (complex intersections, poor traffic light regulation, etc.). This is also a very important context.

The feature_importance metric in XGBoost shows the relative contribution of each feature to the model, i.e. how often this feature was used for splitting. The sum of all importance values for all features is 1.0. By summing the importance indicators for the dominant factors, which are time and contextual (operational) factors, a value of 0.520 (or 52%) was obtained. This group of 20 top factors included the categories: month, hour_sin, dayofweek, dayofweek_sin, dayofweek_cos, trip_id, hour_cos, hour, stop_id. This allows us to state that together time and contextual (operational) factors account for more than 52% of the total importance of all features used by the model. Weather conditions are in the middle of the list of top 20 most important features. The total importance of the weather factors (wind_speed, feels_like, pressure, temp, visibility, humidity, dew_point, wind_deg, clouds_all, weather_id, rain_1h) is 0.448 (or 44.8%). Together, weather factors account for over 44% of the total importance of all features used by the model. This means that the influence of weather conditions is a very important factor in itself (their total weight is almost the same as that of operational factors), i.e. it is important to know not only when and where the trip takes place, but also under what weather conditions.

To calculate the ratio of the total importance of one group of factors to another, the ratio of the total importance of time and contextual factors (0.520) to the total importance of weather factors (0.448) was calculated. As a result, $0.520/0.448 \approx 1.16$, i.e. the combined impact of time and contextual factors is only 1.16 times greater than the impact of weather factors.

So, the developed XGBoost model indicates that it is much more important to know that it is 17:30 on a Friday in December than that the temperature is -5°C, that is, for building a forecast, it is more important that it is rush hour than information about light snow on the day. Actually, the magic of tree-shaped models is that the decision trees of which they are composed look for interactions between factors, as opposed to simple correlation, which looks at each feature in isolation. Only one tree in the XGBoost model can divide the data at the top level by hour > 16 (peak hour), and then, in the branch where hour > 16, the next division is made by snow_1h > 1, and in the branch where snow_1h > 1, the next division is made by wind_speed > 15.

Indeed, the reliability of public transport is not determined by a simple sum of independent factors, but by a complex interaction between a deterministic operational base and stochastic external disturbances according to a hierarchy: first, when and where the trip takes place is taken into account, and then weather data is used, which is what SHAP is used for. It is the combination of XGBoost and SHAP that will allow us to obtain a complete and deep, non-trivial conclusion that would not be possible using simpler methods.

4.4. Deep Analysis of the Model with SHAP

Deep analysis of the XGBoost model using SHAP was performed during the implementation of step 7, which is shown in Fig. 17. As a result of executing the program code (Fig. 17), messages were displayed on the screen about the number and name of the step (Step 7/7, Deep analysis of the model with SHAP...), information about creating an explainer for the model, the completion time of calculating SHAP values, and the completion of constructing four graphs of the importance of features (total influence graph and interaction graphs) based on the characteristics obtained by the XGBoost model.

```
print("\n[Step 7/7] Deep model analysis with SHAP...")
start_time = time.time()

X_sample = X_test.sample(min(SHAP_SAMPLE_SIZE, len(X_test)), random_state=42)

print(f"- Creating an 'Explainer' for the model...")
explainer = shap.TreeExplainer(model)
print(f"- Calculating SHAP values for {len(X_sample)} records (this may take a few minutes)...")
shap_values = explainer.shap_values(X_sample)
print(f"Calculation of SHAP values completed in {time.time() - start_time:.2f} sec.")

print("\nBuilding a general graph of the importance of features (Summary Plot)...")
fig, ax = plt.subplots()
shap.summary_plot(shap_values, X_sample, show=False)
plt.title("Overall impact of features on delay prediction", fontsize=16)
plt.tight_layout()
plt.savefig('shap_summary_plot.png')
plt.close()
print("- Graph saved to file 'shap_summary_plot.png'")
```

Fig.17. Script for deep analysis of xgboost model with shap and example of one visualization

The overall impact graph is shown in Fig. 18, where each row is one feature sorted by importance, each point on the graph is one record from the generated sample (X_sample), and the position of the point on the X-axis shows how much and in which direction this feature "pushed" the forecast for a particular record. Fig. 18 shows a positive SHAP value on the right, i.e. a feature that increased the predicted delay, and a negative SHAP value on the left, i.e. a feature that decreased the predicted delay. The color of the point shows the value of the feature itself, where red/pink indicates a high value, e.g. late hour, blue indicates a low value, e.g. no snow [38-40]. In the graph (Fig. 18), we see that the pink hour points are mostly on the right, which means that late hours increase delays.

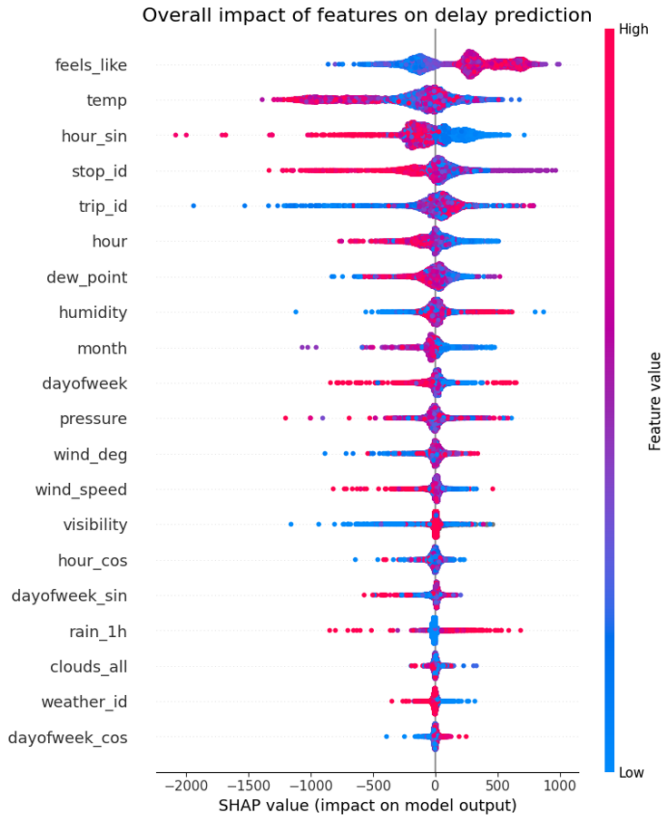

Fig.18. Graph of the overall impact of the XGBoost model, constructed with SHAP

The total influence plots show the overall hierarchy of factors. However, to answer the question of the true role of weather and avoid exaggerating its secondary role, it is necessary to consider its impact not in isolation, but in interaction with other, more dominant factors. This is where interpretive machine learning methods such as SHAP reveal their full potential, allowing us to visualize complex, nonlinear relationships.

The SHAP interaction plots (Fig. 19-21) allow us to analyze the impact of weather in detail. They clearly demonstrate that the impact of weather conditions is not a constant, but a function that depends on the operational context.

The interaction graphs (shap_dependency_...png) are shown in Fig. 19, which show various interactions, for example, the interaction of hour with snow (shap_dependency_hour_vs_snow). Here, the X-axis shows the value of the main

feature, hour of day (hour,), and the Y-axis shows the SHAP value for this feature, i.e., shows how many seconds hour increased/decreased the forecast. The color of the points indicates the value of the "interacting" feature (snow_1h), where red/pink indicates a lot of snow, and blue indicates little or no snow.

In the graph of the interaction of hour and snow (Fig. 19), we see a classic example of this effect. The vertical "scatter" of the points for each hour shows how the impact of that hour on delay varies depending on the presence of snow. During peak hours (e.g., 8:00 AM and 5:00 PM-6:00 PM), the pink points (indicating the presence of snow) have significantly higher SHAP values than the blue points (without snow). This means that the impact of the peak hour on delay is significantly increased when it is snowing. In contrast, at night or during the off-peak period (e.g., 11:00 AM to 3:00 PM), the difference between the blue and pink points is minimal. This is because during peak hours, the transportation system is operating at its capacity. There is no "safety margin," and any additional stressor, such as snow, that forces drivers to drive more slowly instantly causes a cascading effect of delays. During the off-peak period, the system has sufficient reserves to absorb this impact without significant consequences.

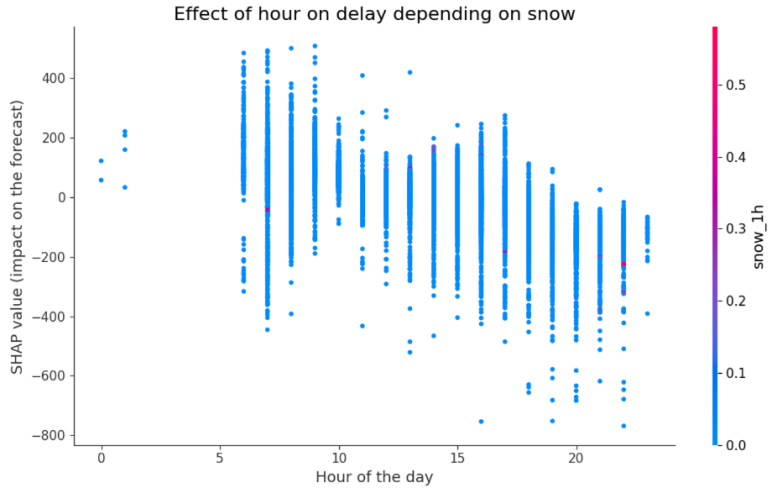

Fig.19. Graph of dependence and interaction of features Hour vs Snow in the XGBoost model, built with SHAP

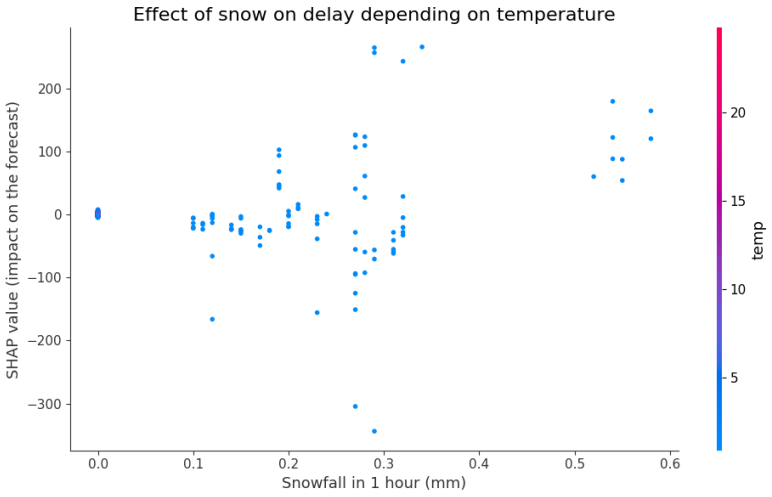

Fig.20. Graph of dependence and interaction of the Snow vs Temperature features in the XGBoost model, built with SHAP

In these graphs, the vertical "spread" of the points is important. If for a certain hour (hour = 17) the blue points (without snow) are at the same level on the Y-axis, and the pink points (with snow) are much higher, this means that the impact of the 17th hour on the delay is significantly increased when it snows. This is a visualization of the cumulative effect. The SHAP visualization clearly shows that the problem is not just the rush hour and not just the snow, but a combination of them. The same applies to other interaction graphs such as Snow vs Temperature (Fig. 20), which can be used to determine whether the impact of 5 mm of snow is stronger when the temperature is near zero (wet snow) than when it is -10°C (dry snow)?

A similar pattern is observed for visibility (Fig. 21). The impact of weather conditions is not a constant, but a nonlinear function that depends on the operational context. Poor visibility (blue dots on the SHAP graph) has a much stronger negative impact (increasing delays) precisely during the morning and evening rush hours, when traffic intensity is maximum. During the day or at night, when traffic is light, its impact is much smaller. Also, the interaction plot of visibility vs hour (Fig. 21) determines that the impact of poor visibility of 500 meters is stronger in the late evening and

at night (high/low hour values) than during the day. These visualizations provide a much deeper understanding of the model behavior and real-world processes than simple numbers.

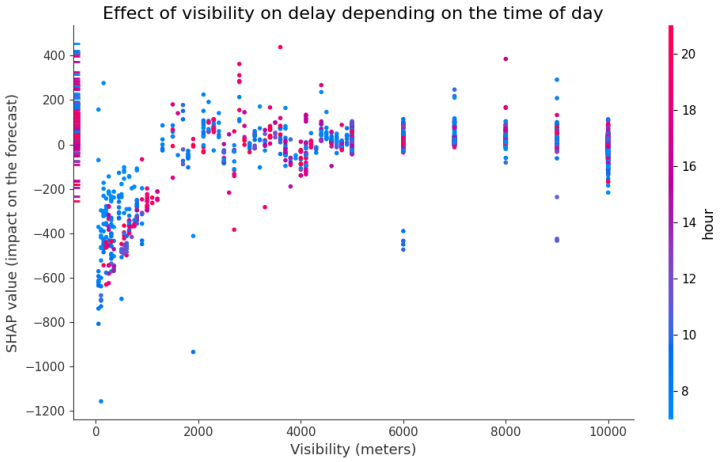

Fig.21. Dependency and interaction graph Visibility vs Hour in the XGBoost model, built with SHAP

We see that the model automatically creates a rule that takes into account the combination of “rush hour + heavy snow + strong wind”. The importance of these features is their cumulative contribution to all such rules for a thousand trees.As shown in the deep analysis of the model using SHAP (Fig. 18-21), the negative effect of adverse weather conditions (for example, snow or poor visibility) is relatively insignificant during periods of low operational load, but is significantly amplified during peak hours, when the system is operating at the limit of its capacity. The conducted research proves that the reliability of public transport is determined not by a simple sum of independent factors, but by a complex interaction between a deterministic operational base and stochastic external disturbances. The XGBoost model successfully revealed this hierarchy: first it takes into account when and where the trip takes place, and then uses weather information to refine the forecast depending on the current state of the system.

Thus, the XGBoost model and SHAP analysis do not simply show that weather is “less important.” They reveal a deeper and more practically meaningful conclusion: the importance of weather conditions is not equally more or less important, i.e. a constant value, but rather in quantifying the nonlinear and conditional nature of this function, which depends on the current state of the transportation system. Factor analysis conducted using our interpretive machine learning model shows that these deterministic dynamics are the dominant predictors of delay, setting a predicted marginal performance that is superimposed by external stochastic factors such as weather, supporting Hypothesis 1.

Evidence for predictive modeling is the high coefficient of determination ($R^2 = 0.72$) achieved by the XGBoost model, indicating that a significant portion of the delay variance is systematic and predictable, rather than random.

The model gave the highest weight to features that represent these deterministic dynamics. Functions such as month, hour_sin, and dayofweek directly model time cycles. Functions such as trip_id and stop_id serve as indicators of the network topology and its inherent bottlenecks.

From a systems perspective, this means that any strategy for optimizing passenger flows in a smart city must first consider the inefficiencies and constraints inherent in its own operating schedule and infrastructure. Simply responding to external events is not enough if the underlying system performance is not well understood and optimized.

Interpretive machine learning analysis (specifically, SHAP plots) shows that the marginal impact of weather on delay duration is minimal when the system is operating at low capacity (off-peak conditions), but increases significantly and nonlinearly when superimposed on high capacity and high stress conditions (peak hour traffic), supporting Hypothesis 2. The interpretive machine learning evidence is the SHAP interaction plots, which provide direct visual evidence for this hypothesis. They show that the SHAP value (forecast impact) for an adverse weather event such as snowfall is significantly higher during peak hours (hours = 8 or 17) than during off-peak hours (hours = 3 or 4). This demonstrates a strong interaction effect, rather than a simple additive one. This hypothesis explains why simple correlation analysis failed to find a strong relationship between weather and delays. By averaging the effect across all system states, the strong, conditional impact during peak hours was diluted by a small impact during off-peak hours.

The sensitivity analysis of the model, i.e. the study of how changes in key features affect the model’s predictions, was conducted using the SHAP framework. This advanced method allows not only to assess the overall importance of each feature, but also to visualize the nature of its impact on the forecast.

The SHAP dependence and interaction plots (Fig. 19-21) are a direct result of this sensitivity analysis. For example, Fig. 19 shows how the model responds to a change in the hour of day (hour) and how this response is modulated by another feature, the presence of snow (snow_1h). Analysis of these plots shows that the model is highly sensitive to the interaction between operational and weather factors.

In particular, it was found that the negative impact of bad weather (snow, low visibility) on the predicted delay is relatively small during periods of low load but is significantly amplified during peak hours. This confirms that the model

successfully captured complex, nonlinear dependencies in the data, which is key to adequately modeling real-world transport processes.

4.5. Justification of the Methodological Choice of the XGBoost in Combination with SHAP

Regarding the value of choosing the XGBoost methodology in combination with SHAP, we would like to note that it was this combination that allowed us to obtain such a deep, non-trivial conclusion that would not be possible using simpler methods. The choice of methodology for analyzing complex systems such as urban transport is critically important and requires a balance between predictive accuracy, interpretability and computational efficiency. The combination of the XGBoost model and the SHAP interpretation method was chosen as the optimal approach to solving the tasks set, as well as a tool that allows us to meet the needs of both technical and applied audiences.

The simplest approach to factor analysis is linear regression or correlation analysis. However, as our Pearson correlation analysis (Table 3) showed, the linear relationships between individual weather parameters and delays are extremely weak (correlation coefficients do not exceed 0.044). Naturally, simple linear models are not able to capture the complex, nonlinear nature of the dependencies in the transport system and will have low predictive power. This confirms the need to use more powerful, nonlinear models. Although Random Forest is a powerful ensemble method, XGBoost, as a representative of the boosting algorithm family, often demonstrates higher accuracy. Instead of averaging the results of independent trees, as in Random Forest, XGBoost sequentially builds new trees that correct the errors of the previous ones, which allows to achieve better accuracy on complex data sets. In addition, XGBoost offers more flexible regularization mechanisms (L1 and L2), which allows to more effectively control the complexity of the model and avoid overfitting. For tasks where maximum prediction accuracy is important, boosting is often preferable. For example, in the study of flight delays by [33], the Random Forest model achieved $R^2 = 0.62$, which is a good but not outstanding result.

Deep neural networks are a powerful tool for working with time series, but they are a classic example of a «black box». Their interpretation is an extremely difficult task, which directly contradicts one of the key goals of our research - conducting deep factor analysis. While ensemble trees such as XGBoost are much easier to interpret using XAI methods such as SHAP. In addition, neural networks perform best when working with unstructured data (images, sound, text), while ensemble trees such as XGBoost have traditionally dominated tasks on structured, tabular data, which is our operational and weather records. They work more efficiently with heterogeneous features and are less sensitive to data scaling. At first glance, the coefficient of determination $R^2 = 0.72$ may not seem ideal. However, when viewed in the context of similar studies of delays in real-world transportation systems, this result is high and competitive. Transport systems are chaotic, open systems, where a significant part of the variability is due to truly random, unaccounted for factors (road accidents, sudden breakdowns, etc.). Therefore, achieving perfect accuracy ($R^2 \approx 1.0$) is theoretically impossible.

Table 5. Comparative table of models for analysing passenger transport delays

E	Parameter	Description	Significant predictors	Accuracy (value)	Application possibilities	References
XGBoost (our model)	Public transport delays (bus)	Predictive modeling and factor analysis of delays in a city of ~1 million population.	1. Temporal (month, hour, day of the week) 2. Contextual (route ID, stop ID) 3. Weather (secondary factor)	$R^2 = 0.72$	Strategic and tactical planning, decision support system.	-
Gradient Boosting (GB)	Ferry delays	Forecasting ferry delays in Sydney and New York using open data (GTFS, weather, passenger traffic).	1. Operational (stop sequence, trip start time, interval) 2. Vessel ID 3. Weather (minor influence)	$R^2 \approx 0.66$ (Sydney) $R^2 \approx 0.78$ (New York)	Optimization of ferry schedules, improving synchronization with other modes of transport.	[41]
Random Forest (RF)	Flight delays	Forecasting real-time flight delays based on ADS-B signals and METAR/TAF weather reports.	Not detailed but includes weather and operational data.	$R^2 = 0.62$	Real-time air traffic control, passenger information.	[33]
XGBoost	Flight delays	Analysis of the impact of weather and operational factors on flight delays at three New York airports.	1. Departure time 2. Airline 3. Precipitation	Classification accuracy: 76.8% - 79.1%	Assistance to airports in flight planning taking into account weather forecasts.	[25]
Gradient Boosting (GB)	Train delays	Forecasting train delays based on traffic and weather data.	Not detailed but includes operational and weather factors.	Not specified	Railway dispatching, schedule planning.	[42]
GTFS simulation	Public transport delays (bus)	Modelling the impact of road works on the public transport network using GTFS data.	Road works (external factor).	Not applicable (It is not a forecasting model, but a «what-if» simulation)	Assessing the impact of planned works on the transport network.	[43]

Comparative analysis shows that our model is on par with or exceeds the results of similar studies (Table 5). As can be seen from Table 5, in the study of ferry delays by [41], the gradient boosting model achieved an R^2 of 0.66 for Sydney and 0.78 for New York. In the work [33] on forecasting flight delays, the Random Forest model achieved an R^2 of 0.62. In the study of train delays by [42], statistical models obtained R^2 in a wide range from 0.59 to 0.92 depending on the station, which highlights the complexity and variability of the problem. Against this background, the ability of our model to explain 72% of the variability in urban bus delays indicates its high predictive power and the adequacy of the chosen method.

Thus, the choice of methodology was dictated not only by the desire to achieve high accuracy, but also by the need to answer problematic questions in the field of transport technologies using modern methods and technologies. Our research is aimed at both machine learning specialists and transportation engineers and urban planners. It is the combination of XGBoost and SHAP that allows us to meet the needs of both groups. For the technical audience, we demonstrate the application of the XGBoost algorithm to solve a complex nonlinear regression problem, achieving high accuracy, which is confirmed by comparison with other studies. This demonstrates technical awareness and a desire to obtain reliable results.

For the applied audience, we do not simply provide a "black box". With SHAP, we transform a complex model into an interpretable decision support tool. SHAP plots (Fig. 18–21) visualize the contribution of each factor and their interactions, making the model's conclusions understandable and testable for engineers who can use this knowledge for practical planning.

Thus, the chosen methodology is not a compromise, but a synergy, where XGBoost provides accuracy and SHAP provides explanation. This approach allows us to build a «bridge between theory and practice», transforming an advanced theoretical tool into a practical, effective tool for decision-making, which is the main methodological value of our work.

Thus, the impact of weather is conditional and modulated by the operational context (time and place). For a transport manager, this means that it is necessary to react not simply to the forecast «snow tomorrow», but to the combined forecast «snow tomorrow during the evening rush hour on route No. 5». This allows us to move from general, ineffective measures to targeted, preventive actions aimed at the most vulnerable temporal and spatial segments of the network. It is this ability of the model to detect complex, nonlinear interactions that is the main advantage of interpretive machine learning, which creates a bridge between theoretical modeling and practical management. This finding is critical for the development of adaptive traffic management systems in a smart city. It suggests that predictive models should trigger proactive responses (e.g., dynamic schedule adjustments, passenger notifications, rerouting) not only based on the weather forecast, but also based on a combination of the weather forecast and the predicted operating state of the transportation network. The system's response to 2 cm of snow at 5:00 PM should be fundamentally different from its response to 2 cm of snow at 3:00 AM.

4.6. From Model Interpretation to Decision-making: Opportunities, Limitations, and Practical Application

Successful application of machine learning in applications such as transportation planning requires not only high predictive accuracy, but also transparency and interpretability of the model. It is important to understand how the proposed approach of using XGBoost in combination with SHAP serves as a bridge between complex data analysis and practical management decision-making, as well as to consider the limitations of the chosen methodology and how it can be directly applied by practitioners.

Although the SHAP method is a powerful tool, it is not without certain limitations that should be considered when interpreting the results. First, it is an interpretation of the model, not reality. SHAP explains how the model uses the input data to generate a prediction, and does not necessarily reveal true cause-and-effect relationships in the real world. If the model is trained on biased data, SHAP will only help us understand this bias, but will not correct it. Second, it is computationally complex. Calculating exact SHAP values, especially for interaction analysis, is often computationally expensive for very large datasets and complex models. Third, there is the risk of misinterpretation. SHAP plots, while visually appealing, require a certain level of expertise to interpret correctly. For example, a high SHAP value for a particular feature does not always mean that changing that feature is the best way to affect the outcome, especially in the presence of strong interactions.

There are alternative methods, such as LIME (Local Interpretable Model-agnostic Explanations), which approximates the «black box» behavior locally using simpler models. However, SHAP was chosen for this study because of its strong theoretical basis (game theory) and the property of consistency, which ensures that the importance of a feature does not decrease if its real influence in the model increases.

To bridge the gap between the complexity of the model and the needs of end users (transport engineers, managers), we propose a two-tier approach to its application: an analytical level (for experts) and a managerial level (for managers). At the analytical level, a data analyst trains the XGBoost model and conducts deep analysis using SHAP, generating visualizations as in Fig. 18-21, and identifying key patterns and interactions. At the managerial level, the manager uses the results of this deep analysis in the form of practical, actionable recommendations and tools that no longer require a deep understanding of machine learning.

Since the model results clearly indicate the need to change the management paradigm from reactively dealing with unpredictable weather conditions to proactively managing operations based on predictable cycles [44-47], below are examples of such tools for improving the reliability and punctuality of the transport system, which are a direct consequence of the analysis of our model.

The model serves as a basis for decision-making for the year/season ahead, becoming a tool for strategic planning for the long term.

- In adaptive seasonal planning, the identification of the month factor as the most important directly indicates the inefficiency of a single schedule. The proposed model allows managers to quantitatively assess seasonal fluctuations in delays, which allows developing separate schedules (autumn, winter, summer), optimizing the distribution of rolling stock.
- When planning express flights, specialists from the schedule development department have reliable information that in September the model shows a sharp increase in delays on routes near educational institutions, which is the justification for the planned launch of additional flights in the morning hours (7:30-9:00) for the first two weeks of the school year.

The model allows you to identify and solve local, daily problems, serving as a tool for tactical optimization in the short term.

- To identify «hotspots», a model is proposed that systematically records high SHAP values for the combination of stop_id and hour (for example, stop Z on route Y from 17:00 to 18:30), becoming an automatic signaler of problem areas.
- To diagnose the causes, such a signal from the proposed model is the basis for the departure of a technical team and conducting a tactical analysis. As studies show, the causes can be an unsuccessful traffic light cycle [48-50], chaotic parking or a bottleneck on the road. The model allows you to focus the limited resources of engineering services on those areas where it will have the greatest effect.
- For a prompt response, identifying the problem of a shortage of rolling stock on a specific route allows you to tactically introduce an additional bus 15 minutes before the peak route to relieve the main passenger flow.
- Furthermore, beyond operational improvements, the model serves as a powerful tool for proactive passenger information. Based on its predictions, information systems (e.g., information displays at stops and within mobile applications) can shift from stating facts («the bus is late») to providing advance warnings. The proposed model enables a transition from generic messages to providing more specific information regarding possible delays due to weather. This includes specifying the time (day, hour) and route numbers where significant disruptions are expected due to the combined impact of peak loads and adverse weather conditions (e.g., snow, heavy rain).

Thus, the developed model and its interpretation serve as a full-fledged decision-making support system. It does not require the transportation engineer to understand the intricacies of gradient boosting. Instead, it provides him with clear, localized signals in time and space about existing and potential problems and quantifies strategic changes in scheduling and resource allocation. Perhaps the most unexpected, yet highly effective, practical application of the XGBoost model's results is its use as an objective basis for constructive dialogue between the public and city authorities. When discussing funding issues, such as the procurement of new vehicles, the model's outputs serve as a powerful argument that transforms a funding request into a business proposal for investing in the city's transport system with a clear, predictable return.

In general, the developed XGBoost model transforms a chaotic, at first glance, transport system into a predictable one. It allows you to see not just problems, but their real causes, and, most importantly, indicates where to direct limited resources to obtain the maximum effect. This is the main value of the developed XGBoost model for the science and practice of passenger transport management within the framework of the Smart City concept.

4.7. Interpretation of Unexplained Variance: The Role of Random External Factors

The study deliberately focuses on the analysis of systematic, predictable factors (temporal, contextual, weather) that affect delays. However, as rightly noted in [51,52], random external factors also have a significant impact on transport systems - stochastic, high-amplitude events, such as road accidents, sudden vehicle breakdowns, unforeseen street closures and other incidents. Although this study did not have access to structured data on such events, the model we built allows us to indirectly but quantitatively assess the cumulative impact of random factors due to unexplained variability.

The key indicator of the quality of our model is the coefficient of determination (R^2), which was 0.72 in the test sample. This means that the developed model, using only systemic and predictable factors, was able to explain 72% of the total variability in public transport delays.

It is natural to overlook that the remaining 28% of unexplained variability is a quantitative expression of the combined impact of these random, unmodeled external factors. Thus, the model does not ignore these factors, but, on the contrary, allows us to isolate and measure their total contribution to the chaotic nature of the system. We believe that the absence of these data is not a critical drawback for this study, since in addition to the two levels of transport management, namely: strategic planning and operational response, there is also a level of response to incidents that are random events.

The focus on strategic and tactical planning corresponds to the goal of this work, namely, to create a tool for proactive planning and optimization of the system based on predictable patterns. A transport manager cannot plan an accident for next Tuesday, but he can and should plan route reinforcement for every Tuesday if the model shows systematic delays there. Our model, explaining 72% of delays due to systemic causes, provides a powerful tool for this level of management. It allows us to deal with chronic, predictable «diseases» of the system, and not just with its acute, random «injuries».

For urban planners and transportation engineers, this approach is more than adequate. Their task is to design the system so that it is as resistant as possible to typical loads. Our model gives them data for making decisions in this area, for example, where to widen the road, how to adjust traffic lights, on which routes to increase the number of buses at certain hours or seasons. These are practical consequences that directly follow from our analysis.

At the same time, this limitation opens a clear direction for further research. The next step will be to develop a hybrid model that combines a basic predictive layer (our model), where the expected system delay is predicted based on time, context and weather, with a real-time predictive layer: Integration with real-time incident data sources (e.g. police reports, driver reports) to predict additional, "anomalous" delays caused by random factors.

Such a multi-layered approach will allow for an even more complete and accurate forecasting system. However, as a fundamental study aimed at deconstructing and prioritizing system factors, the study can be considered complete. It successfully proves that the vast majority of daily public transport problems are not random, but predictable, and therefore can and should be managed proactively.

5. Conclusions

This study aimed to identify and quantify critical factors affecting passenger transport reliability to create a scientifically sound basis for prioritizing management decisions. To achieve this goal, a number of tasks were performed, which allowed obtaining the following conclusions:

A high-accuracy model based on gradient boosting (XGBoost) was developed and successfully trained, which is capable of effectively predicting public transport delays, integrating large arrays of operational and weather data. The model demonstrated high predictive ability, which confirms the adequacy of the chosen approach for analyzing complex, nonlinear processes in urban transport systems.

Using interpretive machine learning methods, in particular SHAP, a deep factor analysis was conducted, which allowed not only to identify, but also to rank the causes of delays by the degree of their influence. This approach made it possible to transform the «black box» model into a transparent analytical tool.

Quantitative assessment of the impact of different groups of factors allowed to draw a key, multifaceted scientific conclusion. Using interpretive machine learning methods, it was found that the basis for predicting delays is formed by operational (temporal and contextual) factors, while weather conditions act as a powerful, but secondary modulating factor. Analysis of the importance of features showed that temporal and contextual factors, such as month (month), hour (hour_sin), day of the week (dayofweek), trip ID (trip_id) and stop ID (stop_id), collectively account for more than 52% of the total contribution of all features to the predictive power of the model. This quantitatively confirms that systemic, predictable cycles (seasonal, weekly, daily) and static network topology are fundamental predictors of reliability.

Weather conditions such as wind speed (wind_speed), feels_like (feels_like) and pressure (pressure) also showed high significance, together accounting for almost 45% of the total importance. However, unlike operational factors, their influence is not absolute. SHAP analysis showed that the negative effect of bad weather is highly context-dependent: it is significantly amplified during operational peaks (e.g. rush hours) but remains minimal at other times. The study proves that the reliability of public transport is determined not so much by the struggle with stochastic nature, but by the ability to manage predictable operational patterns.

The question is not "will it snow?" but "will it snow on Friday at 5:30 PM on route #5?" This fundamentally changes the approach to prioritizing management decisions, shifting the emphasis from reactive response to weather to proactive planning based on time and place analysis.

Quantitative assessment of the impact of different groups of factors allowed to draw a key scientific conclusion: contrary to popular belief, operational (temporal and contextual) factors have a dominant influence on the occurrence of delays. The most important predictor was the month, which reflects the strong influence of seasonality. It is followed by daily and weekly cycles (hour, day of the week) and spatial context (route and stop identifier). Weather conditions, although included in the top 20 most important features, turned out to be a secondary, modulating factor, the impact of which is much less significant compared to WHEN and WHERE the trip takes place.

Based on the results obtained, scientifically based recommendations were formulated, consisting in changing the management paradigm. Instead of reactively responding to stochastic weather phenomena, transport operators should focus their efforts on proactively managing operational cycles. This includes the development of adaptive seasonal schedules, dynamic management of rolling stock according to expected load peaks, and targeted analysis of «hotspots» on the transport network.

Thus, the presented work successfully completed the tasks set. It demonstrates how modern data analysis methods create a bridge between academic theory and engineering practice. The developed model is not just an abstract algorithm, but a full-fledged decision support system that converts theoretical achievements in the field of XGBoost and XAI into specific, measurable recommendations. This encourages the transition from intuitive, reactive management to a proactive, scientifically based approach to improving the reliability of public transport in a smart city.

References

[1] Y. Fornalchyk, I. Vikovych, Y. Royko, and O. Hrytsun, "Improvement of methods for assessing the effectiveness of dedicated lanes for public transport," *East.-Eur. J. Enterp. Technol.*, vol. 1, pp. 29–37, 2021. doi: 10.15587/1729-4061.2021.225397.

[2] T.R. Gadekallu, N. Kumar, T. Baker, D. Natarajan, P. Boopathy, and P.K.R. Maddikunta, "Moth–Flame Optimization based ensemble classification for intrusion detection in intelligent transport system for smart cities," *Microprocessors and Microsystems*, vol. 103, p. 104935, 2023. doi: 10.1016/j.micpro.2023.104935.

[3] T. Postranskyy, M. Afonin, M. Boikiv, and R. Bura, "Identifying patterns of change in traffic flows' parameters depending on the organization of public transport movement," *East.-Eur. J. Enterp. Technol.*, vol. 5, pp. 72–81, 2024. doi: 10.15587/1729-4061.2024.313636.

[4] Y. Liu, D. He, J. Lei, M. He, and Z. Shi, "Investigating the non-linear influence of the built environment on passengers' travel distance within metro and bus networks using smart card data," *Multimodal Transportation*, vol. 4, no. 1, p. 100188, 2025. doi: 10.1016/j.multra.2025.100188.

[5] M. Boikiv, T. Postranskyy, and M. Afonin, "Establishing patterns of change in the efficiency of regulated intersection operation considering the permitted movement directions," *East.-Eur. J. Enterp. Technol.*, vol. 4, pp. 17–26, 2022. doi: 10.15587/1729-4061.2022.262250.

[6] D.D. Chuwang, W. Chen, and M. Zhong, "Short-term urban rail transit passenger flow forecasting based on fusion model methods using univariate time series," *Applied Soft Computing*, vol. 147, p. 110740, 2023. doi: 10.1016/j.asoc.2023.110740.

[7] Y. Matseliukh, V. Vysotska, M. Bublyk, T. Kopach and O. Korolenko, "Network modelling of resource consumption intensities in human capital management in digital business enterprises by the critical path method," in *Proceedings of the CEUR Workshop Proceedings*, Slavsko, Ukraine, vol. 2851, pp. 366–380, 2021. Available online: https://ceur-ws.org/Vol-2851/paper34.pdf.

[8] H. Cui, Z. Ren, M. Zhu, L. Liu, and J. Gao, "Non-linear impact of built environment on origin–destination passenger flow in Tianjin's urban rail transit," *Proceedings of the Institution of Civil Engineers - Transport*, 2025. doi: 10.1680/jtran.24.00166.

[9] Y. Matseliukh, M. Bublyk, and V. Vysotska, "Development of intelligent system for visual passenger flows simulation of public transport in smart city based on neural network," in *Proceedings of the CEUR Workshop Proceedings*, Lviv, Ukraine, vol. 2870, pp. 1087–1138, 2021. Available online: https://ceur-ws.org/Vol-2870/paper82.pdf.

[10] Y. Matseliukh, V. Lytvyn and M. Bublyk, "K-means clustering method in organizing passenger transportation in a smart city," in *Proceedings of the CEUR Workshop Proceedings*, Kharkiv, Ukraine, vol. 3983, pp. 219–240, 2025. Available online: https://ceur-ws.org/Vol-3983/paper17.pdf.

[11] M.A. Fadhel, A.M. Duhaim, A. Saihood, A. Sewify, M.N. Al-Hamadani, A. Albahri, L. Alzubaidi, A. Gupta, S. Mirjalili, and Y. Gu, "Comprehensive systematic review of information fusion methods in smart cities and urban environments," *Information Fusion*, vol. 107, p. 102317, 2024. doi: 10.1016/j.inffus.2024.102317.

[12] H. Almukhalfi, A. Noor, and T.H. Noor, "Traffic management approaches using machine learning and deep learning techniques: A survey," *Engineering Applications of Artificial Intelligence*, vol. 133, p. 108147, 2024. doi: 10.1016/j.engappai.2024.108147.

[13] M. Barbareschi, A. Emmanuele, N. Mazzocca, and F. Rocco di Torrepadula, "Designing on-board explainable passenger flow prediction," *Engineering Applications of Artificial Intelligence*, vol. 139, p. 109648, 2024. doi: 10.1016/j.engappai.2024.109648.

[14] S. M. H. Bamakan, F. Dehghan, and S. Akbarpour, "Role of machine learning for predictive modeling for Industry 5.0 and Society 5.0," in Human-Centric Integration of Next-Generation Data Science and Blockchain Technology. Elsevier, 2025, pp. 265–285. https://doi.org/10.1016/b978-0-443-33498-6.00008-x.

[15] A. Scarano, M. Sadeghi, F. Mauriello, M.R. Riccardi, K. Aghabayk, and A. Montella, "Cyclist crash severity modeling: A hybrid approach of XGBoost-SHAP and random parameters logit with heterogeneity in means and variances," *Journal of Safety Research*, vol. 93, pp. 373-398, 2025. doi: 10.1016/j.jsr.2025.04.003.

[16] A. Rodriguez, J. Arjona, and M. Linares, "Data-driven Strategies to Enhance Real Bus Passenger Occupancy Services using Weather and Flight Information," *Transportation Research Procedia*, vol. 86, pp. 684-691, 2024. doi: 10.1016/j.trpro.2025.04.085.

[17] S. Vafaei and M. Yaghini, "Online prediction of arrival and departure times in each station for passenger trains using machine learning methods," *Transportation Engineering*, vol. 16, p. 100250, 2024. doi: 10.1016/j.treng.2024.100250.

[18] H. Hao, Y. Wang, Du, L. and S. Chen, "Enabling smart curb management with spatiotemporal deep learning", *Computers, Environment and Urban Systems*, 99, p. 101914, 2022. https://doi.org/10.1016/j.compenvurbsys.2022.101914

[19] M. Patel, S.B. Patel, D. Swain, and S. Shah, "Unleashing the Potential of Boosting Techniques to Optimize Station-Pairs Passenger Flow Forecasting," *Procedia Computer Science*, vol. 235, pp. 32-44, 2023. doi: 10.1016/j.procs.2024.04.004.

[20] N. Chukhray, N. Shakhovska, M. Mrykhina, M. Bublyk, and L. Lisovska, "Consumer aspects in assessing the suitability of technologies for the transfer," in *Proceedings of the 14th International Conference on Computer Sciences and Information Technologies (CSIT)*, Lviv, Ukraine, pp. 142–147, 2019. doi: 10.1109/STC-CSIT.2019.8929879.

[21] W. Liu, S. Pang, Li, W. and Y. Han, "Assessing the CO2 emission reduction potential of metro-bus combined travel through interpretable machine learning," *Transportmetrica A: Transport Science*, p. 1–24, 2025. doi: 10.1080/23249935.2025.2472869

[22] K. Wang, J. De Vos, M. Smart, and S. Wang, "Explaining Youth Driver Licensing Determinants Using XGBoost and SHAP," *Transport Policy*, vol. 168, pp. 87-100, 2025. doi: 10.1016/j.tranpol.2025.04.009.

[23] F. Wu, C. Zheng, S. Zhou, Y. Lu, Z. Wu, and S. Zheng, "An interpretable approach to passenger flow prediction and irregular passenger travel patterns understanding in metro system," *Expert Systems With Applications*, vol. 265, p. 125991, 2025. doi: 10.1016/j.eswa.2024.125991.

[24] M.A. Rad, L.M. Lefsrud, M.T. Hendry, A.C. Cen, and S. Soltaninejad, "Analysis of freight train passing a stop signal using machine learning: Application of XGBoost and SHAP," *Journal of Rail Transport Planning & Management*, vol. 35, p. 100532, 2025. doi: 10.1016/j.jrtpm.2025.100532.

[25] Y. Wu, G. Mei, and K. Shao, "Revealing influence of meteorological conditions and flight factors on delays Using XGBoost," *Journal of Computational Mathematics and Data Science*, vol. 3, p. 100030, 2022. doi: 10.1016/j.jcmds.2022.100030.

[26] T. Tang, M. Jia, Y. Zhang, H. Hu, M. Pei, Y. Chen, and X. Wang, "Why metro passengers change travel behavior: Individual-level insights from interpretable machine learning," *Cities*, vol. 167, p. 106352, 2025. doi: 10.1016/j.cities.2025.106352.

[27] L. Cheng, X. Cai, D. Lei, S. He, and M. Yang, "Arrival information-guided spatiotemporal prediction of transportation hub passenger distribution," *Transportation Research Part E: Logistics and Transportation Review*, vol. 195, p. 104011, 2025. doi: 10.1016/j.tre.2025.104011.

[28] C. Banyong, N. Hantanong, P. Wisutwattanasak, T. Champahom, K. Theerathitichaipa, M. Seefong, V. Ratanavaraha, and S. Jomnonkwao, "A machine learning comparison of transportation mode changes from high-speed railway promotion in Thailand," *Results in Engineering*, vol. 24, p. 103110, 2024. doi: 10.1016/j.rineng.2024.103110.

[29] Z. Cheng, D. Sun, Y. Zhao, and H. Peng, "Investigating the factors influencing intercity travel mode choice in urban agglomerations: Insights from a three-phase framework," *Transportation Research Part A: Policy and Practice*, vol. 199, p. 104577, 2025. doi: 10.1016/j.tra.2025.104577.

[30] X. Li, L. Shi, Y. Shi, J. Tang, P. Zhao, Y. Wang, and J. Chen, "Exploring interactive and nonlinear effects of key factors on intercity travel mode choice using XGBoost," *Applied Geography*, vol. 166, p. 103264, 2024. doi: 10.1016/j.apgeog.2024.103264.

[31] J. Jiao, R. An, D. Du, and M. Zhu, "Non-linear and heterogeneous relationship between proximity to high-speed rail stations and land value in China: Analysis using XGBoost-SHAP modelling," *Transportation Research Part A: Policy and Practice*, vol. 196, p. 104486, 2025. doi: 10.1016/j.tra.2025.104486.

[32] C. Peng, S. Yang, P. Zhang, and S. Hu, "Exploring nonlinear and interaction effects of TOD on housing rents using XGBoost," *Cities*, vol. 158, p. 105728, 2025. doi: 10.1016/j.cities.2025.105728.

[33] Z. Kowalczuk, J. Wszołek, and J. Okuniewska, "Real-time predictive modeling of flight delays using distributed systems and machine learning," *IFAC PapersOnLine*, vol. 59, no. 3, pp. 198–203, 2025. doi: 10.1016/j.ifacol.2025.07.034.

[34] F. Chen, Y. Zhu, C. Cao, X. Yang, X. Ji, M. Lai, W. Qiu, C.P. Nielsen, J. Wu, and X. Chen, "Examining nonlinear causal relationship between the built environment and VKT using RF–XGBoost," *Transport Policy*, vol. 171, pp. 661-681, 2025. doi: 10.1016/j.tranpol.2025.07.012.

[35] A. Katrenko, I. Krislata, O. Veres, O. Oborska, T. Basyuk, A. Vasyliuk, I. Rishnyak, N. Demyanovskyi, and O. Meh, "Development of traffic flows and smart parking system for smart city," in *Proceedings of the CEUR Workshop Proceedings*, Lviv, Ukraine, vol. 2604, pp. 730–745, 2020. Available online: https://ceur-ws.org/Vol-2604/paper50.pdf.

[36] S.G. Nnabuife, C. Udemu, A.K. Hamzat, C.K. Darko, and K.A. Quainoo, "Smart monitoring and control systems for hydrogen fuel cells using AI," *International Journal of Hydrogen Energy*, vol. 110, pp. 704-726, 2024. doi: 10.1016/j.ijhydene.2025.02.232.

[37] Y. Fornalchyk, I. Kernytskyy, O. Hrytsun, and Y. Royko, "Choice of the rational regimes of traffic light control for traffic and pedestrian flows," *Sci. Rev. Eng. Environ. Sci.*, vol. 30, pp. 38–50, 2021. doi: 10.22630/PNIKS.2021.30.1.4.

[38] C.M. Caminiti, D. Fratelli, M. Spiller, A. Dimovski, and M. Merlo, "Integrating bottom-up GIS and machine learning models for spatial-temporal analysis of electric mobility impact on power system," *Smart Energy*, vol. 19, p. 100185, 2025. doi: 10.1016/j.segy.2025.100185.

[39] H. Hao, Y. Wang, L. Du, and S. Chen, "Enabling smart curb management with spatiotemporal deep learning," *Computers, Environment and Urban Systems*, vol. 99, p. 101914, 2022. doi: 10.1016/j.compenvurbsys.2022.101914.

[40] A. Huzzat, A. Anpalagan, A.S. Khwaja, I. Woungang, A.A. Alnoman, and A.S. Pillai, "A comprehensive review of Digital Twin technologies in smart cities," *Digital Engineering*, vol. 4, p. 100040, 2025. doi: 10.1016/j.dte.2025.100040.

[41] M. Sarhani, A. Nourmohammadzadeh, S. Voß, and M. EL Amrani, "Predicting and analyzing ferry transit delays using open data and machine learning," *Journal of Public Transportation*, vol. 27, p. 100124, 2025. doi: 10.1016/j.jpubtr.2025.100124.

[42] K. Y. Tiong, Z. Ma, and C.-W. Palmqvist, "Real-time High-Speed Train Delay Prediction using Seemingly Unrelated Regression Models," *Transportation Research Procedia*, vol. 82, pp. 271-278, 2025. doi: 10.1016/j.trpro.2024.12.042.

[43] R. Viri and M. Örmä, "Using GTFS-data to calculate the roadwork caused delays on public transport network," *Transportation Research Procedia*, vol. 82, pp. 1965–1973, 2025. doi: 10.1016/j.trpro.2024.12.166.

[44] A. Jain, I.H. Gue, and P. Jain, "Research trends, themes, and insights on artificial neural networks for smart cities towards SDG-11," *Journal of Cleaner Production*, vol. 412, p. 137300, 2023. doi: 10.1016/j.jclepro.2023.137300.

[45] A.A. Kutty, T.G. Wakjira, M. Kucukvar, G.M. Abdella, and N.C. Onat, "Urban resilience and livability performance of European smart cities: A novel machine learning approach," *Journal of Cleaner Production*, vol. 378, p. 134203, 2022. doi: 10.1016/j.jclepro.2022.134203.

[46] A. Chio, D. Jiang, P. Gupta, G. Bouloukakis, R. Yus, S. Mehrotra, and N. Venkatasubramanian, "SmartSPEC: A framework to generate customizable, semantics-based smart space datasets," *Pervasive and Mobile Computing*, vol. 93, p. 101809, 2023.

[47] W. Liu, S. Pang, W. Li, and Y. Han, "Assessing the CO2 emission reduction potential of metro-bus combined travel through interpretable machine learning," *Transportmetrica A: Transport Science*, pp. 1–24, 2025. doi: 10.1080/23249935.2025.2472869.

[48] Y. Ma, Q. Liu, and L. Yang, "Machine learning-based multimodal fusion recognition of passenger ship seafarers' workload: A case study of a real navigation experiment," *Ocean Engineering*, vol. 300, p. 117346, 2024. doi: 10.1016/j.oceaneng.2024.117346.

[49] T. Tang, J. Zhang, S. Chen, P. Mo, M. Pei, and T. Tang, "Deciphering the pulse of the city: An exploration of the natural features of metro passenger flow using XAI," *Computers & Industrial Engineering*, vol. 204, p. 111097, 2025. doi: 10.1016/j.cie.2025.111097.

[50] K. Wang, B. Guo, H. Yang, M. Li, F. Zhang, and P. Wang, "A semi-supervised co-training model for predicting passenger flow change in expanding subways," *Expert Systems With Applications*, vol. 209, p. 118310, 2022. doi: 10.1016/j.eswa.2022.118310.

[51] M. Bublyk, V. Lytvyn, V. Vysotska, L. Chyrun, Y. Matseliukh and N. Sokulska, "The decision tree usage for the results analysis of the psychophysiological testing," in *Proceedings of the CEUR Workshop Proceedings*, Växjö, Sweden, vol. 2753, pp. 458–472, 2020. Available online: https://ceur-ws.org/Vol-2753/paper31.pdf.

[52] Y. Fornalchyk, E. Koda, I. Kernytskyy, O. Hrytsun, Y. Royko, R. Bura, P. Osiński, R. Barabash, R. Humenuyk, and P. Polyansky, "The impact of vehicle traffic volume on pedestrian behavior at unsignalized crosswalks," *Roads and Bridges – Drogi i Mosty*, vol. 22, pp. 201–219, 2023. doi: 10.7409/rabdim.023.010.

Authors' Profiles

Yurii Matseliukh received the Bachelor's degree in Computer Science (with honours) and the Master's degree in System Analysis (with honours) from Lviv Polytechnic National University, Lviv, Ukraine.

He is currently a PhD student in the Information Systems and Networks Department at Lviv Polytechnic National University, Lviv, Ukraine. For the past decade, he has served on the review board for scientific journals in the field of transportation and transport systems, such as *Simulation Modelling Practice and Theory*. He is the author of more than 30 publications (a Scopus h-index =13). His research focuses on modelling passenger transportation in public transport.

MSc Matseliukh was a recipient of the Presidential Scholarship of Ukraine from 2019 to 2021. His research has been presented at numerous international conferences, including the IEEE International Scientific and Technical Conference on Computer Sciences and Information Technologies.

Vasyl Lytvyn holds the Doctor of Technical Sciences degree and Professor of the Information Systems and Networks Department at the Lviv Polytechnic National University, Lviv, Ukraine. He graduated from Ivan Franko National University of Lviv, Ukraine, in 1997.

He has been a Head of the Information Systems and Networks Department at Lviv Polytechnic National University, Lviv, Ukraine (2013-2025). Prof. Lytvyn V. is an expert in Data Science, Big Data, Intelligent Systems, Machine Learning, Knowledge Engineering and Ontology Construction. His main direction of scientific research is the development of intelligent decision support systems using an ontological approach. He is participated in much research as a performer and supervisor. He is the author of more than 200 publications in journals including the *International Journal of Computing*, *Symmetry*, *Applied System Innovation et al*. Prof. Lytvyn's publications have a Hirsch index of 31 in the Scopus database.

Zhengbing Hu: Prof., Deputy Director, International Center of Informatics and Computer Science, Faculty of Applied Mathematics, National Technical University of Ukraine «Kyiv Polytechnic Institute», Ukraine. Adjunct Professor, School of Computer Science, Hubei University of Technology, China. Visiting Prof., DSc Candidate in National Aviation University, Ukraine (2019-2021).

His primary research interests are Communications, Graph and Image Processing, Computer Science and Technology Applications, Artificial Intelligence, Network Security, Communications, Data Processing, Cloud Computing, Education Technology.

Myroslava Bublyk received the Ph.D. degree in Physics from Ivan Franko National University of Lviv, Lviv, Ukraine, and the Doctor of Sciences degree in Economics from Lviv Polytechnic National University, Lviv, Ukraine.

She is currently a Professor at Lviv Polytechnic National University, Lviv, Ukraine. She is the author of more than 400 publications, which have appeared in journals such as *Energies*, *Journal of Open Innovation Technology Market and Complexity*, and *Advances in Intelligent Systems and Computing*. Her primary research focus is the application of modern mathematical tools and the universal capabilities of AI to identify interdependencies at the interdisciplinary nexus of ecology, society, economy, and information technology.

Prof. Bublyk is an Academician of the Academy of Economic Sciences of Ukraine. She serves on the editorial board of *Humanities and Social Sciences Letters* (indexed in Scopus) and on the editorial and/or review boards of several other scientific journals, including *Problems of Quality*, *Business Inform*, and *Problems of Economy*. Prof. Bublyk's publications have a Hirsch index of 18 in the Scopus database.

How to cite this paper: Yurii Matseliukh, Vasyl Lytvyn, Zhengbing Hu, Myroslava Bublyk, "Predictive Modelling and Factor Analysis of Public Transport Delays in Smart City Using Interpretable Machine Learning", International Journal of Information Technology and Computer Science(IJITCS), Vol.17, No.6, pp.1-28, 2025. DOI:10.5815/ijitcs.2025.06.01