

Московский государственный технический университет имени Н. Э. Баумана
(национальный исследовательский университет)

Обнаружение дефектов программного обеспечения с использованием алгоритмов машинного обучения

Студент: Климов Илья Сергеевич, ИУ7-82Б

Научный руководитель: Вишневская Татьяна Ивановна

Москва, 2023

Цель и задачи

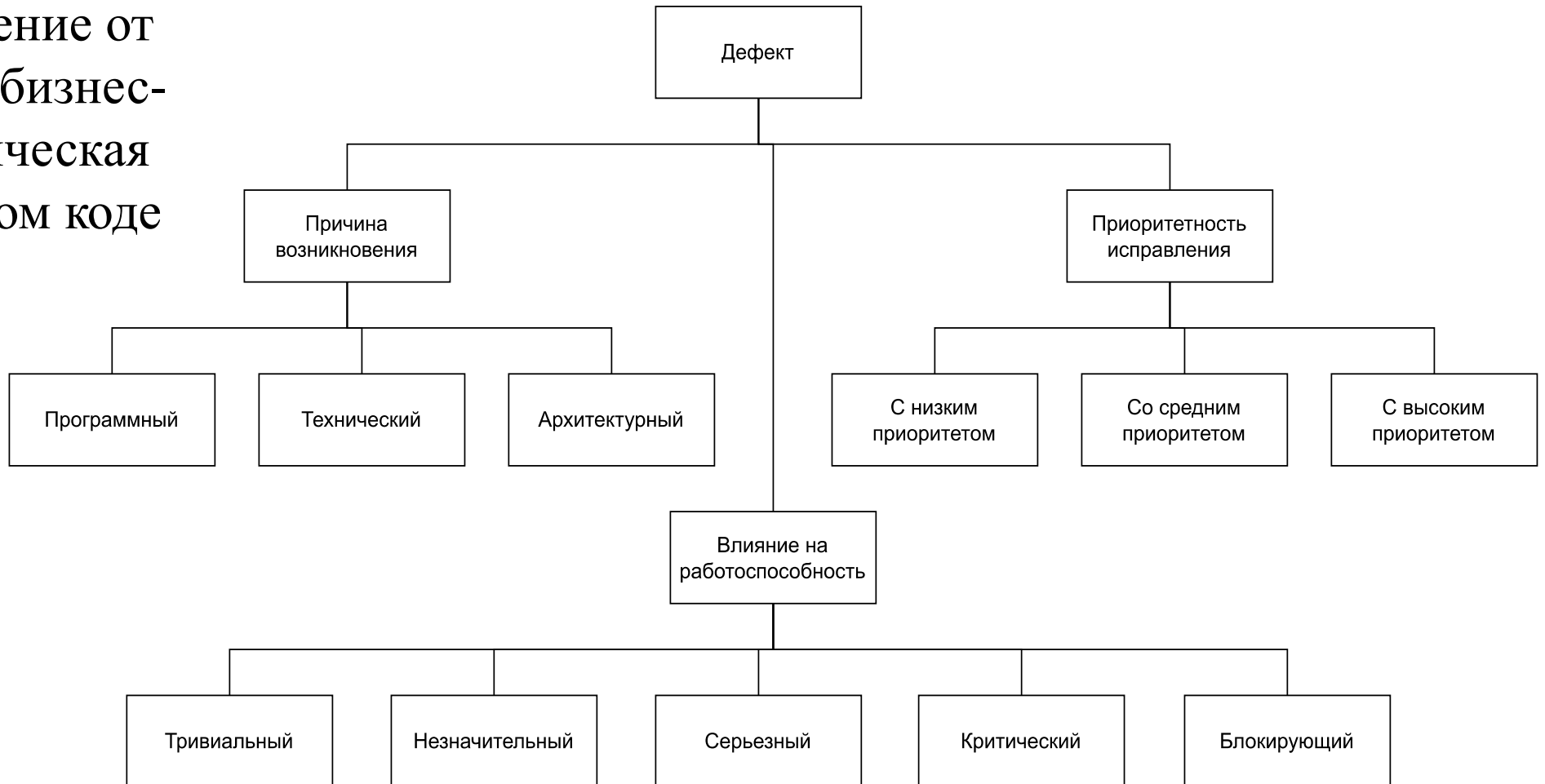
Цель: разработка и программная реализация метода обнаружения дефектов ПО с использованием алгоритмов машинного обучения.

Задачи:

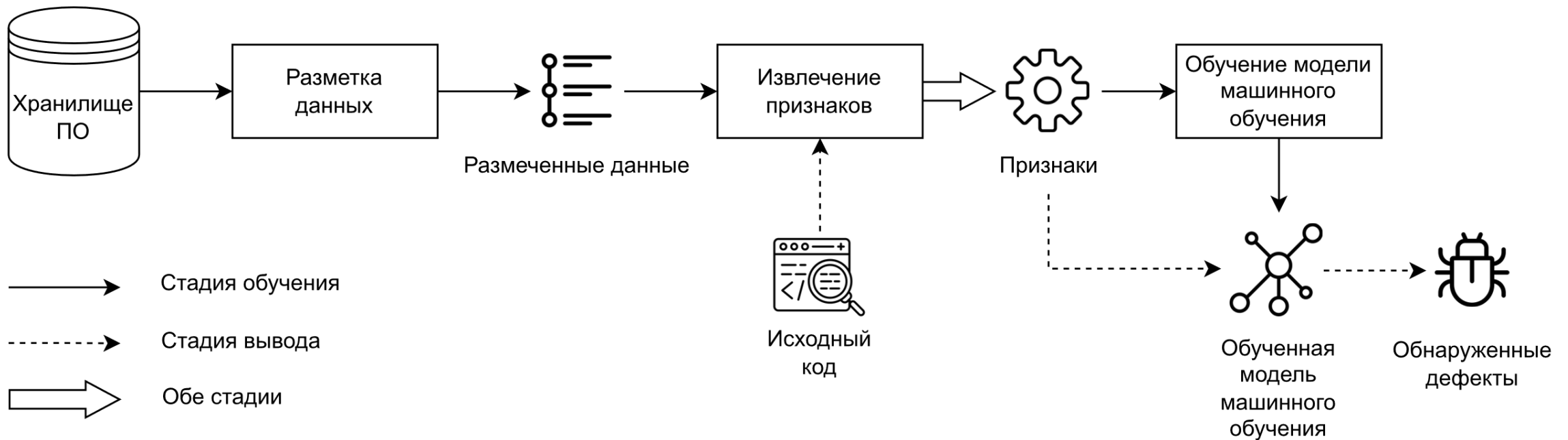
- 1) проанализировать и сравнить существующие методы машинного обучения для обнаружения дефектов ПО;
- 2) разработать метод обнаружения дефектов ПО с применением ансамбля деревьев решений (градиентного бустинга);
- 3) разработать программное обеспечение, реализующее метод обнаружения дефектов ПО;
- 4) провести исследование эффективности разработанного метода и сравнение его с существующими реализациями.

Дефекты разрабатываемого ПО

Дефект – отклонение от первоначальных бизнес-требований, логическая ошибка в исходном коде программы.



Процесс обучения модели обнаружения дефектов ПО



Метрики для сравнения алгоритмов

$$\text{accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

$$\text{recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

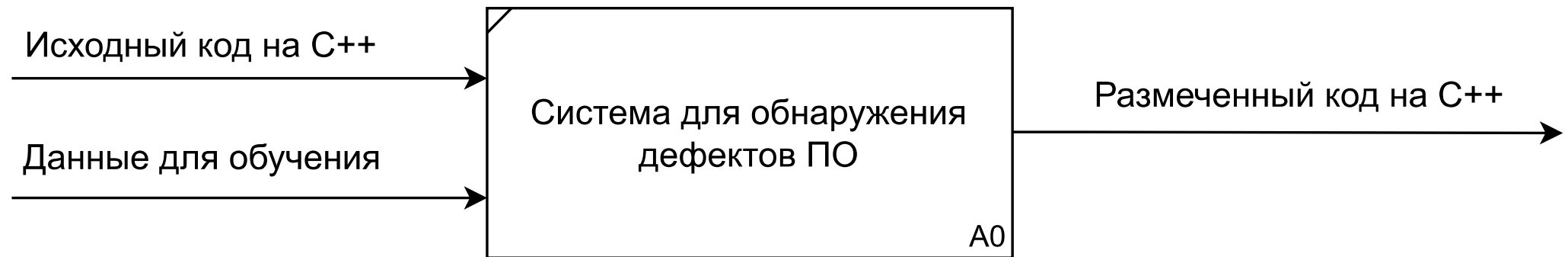
$$\text{F - measure} = \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

TP – объекты, которые были верно классифицированы как положительные;
TN – объекты, которые были верно классифицированы как отрицательные;
FP – объекты, которые были ложно классифицированы как положительные;
FN – объекты, которые были ложно классифицированы как отрицательные.

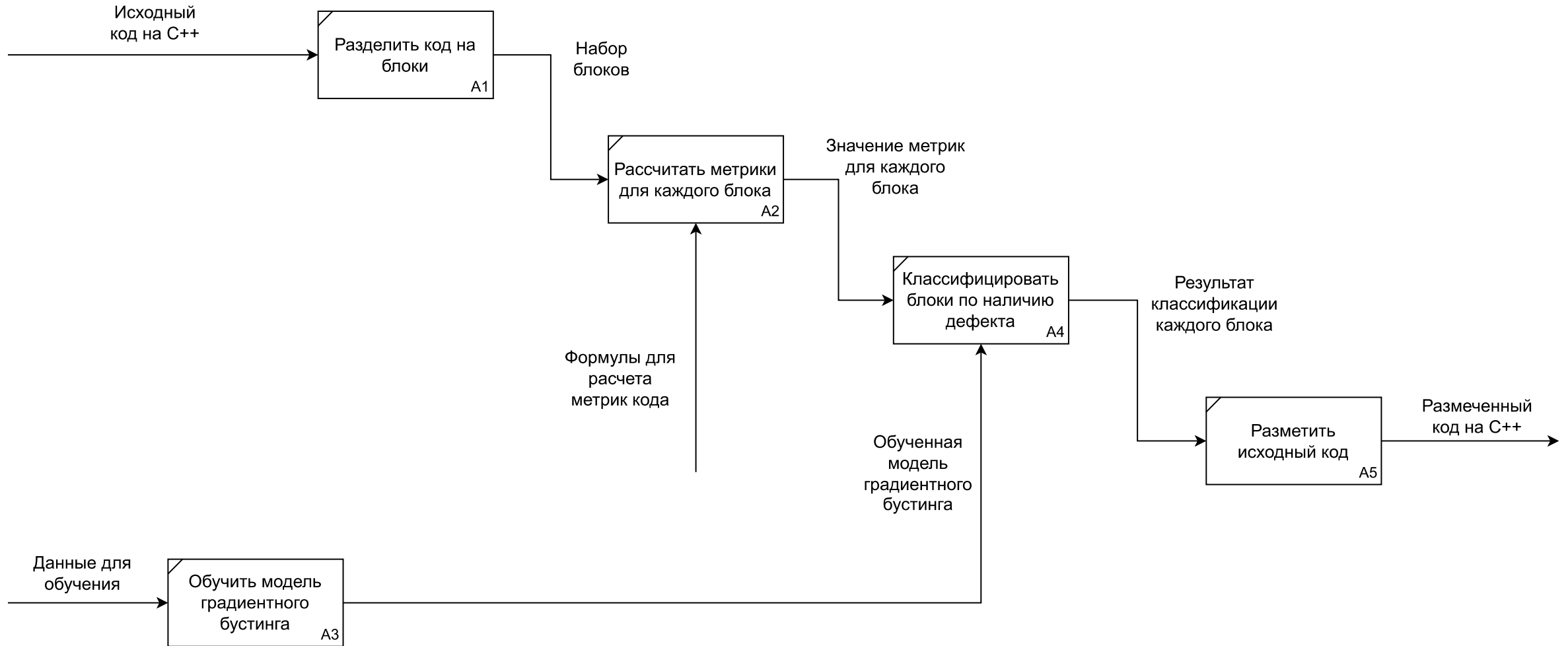
Сравнительная таблица результатов работы алгоритмов

Метрики Алгоритмы	Accuracy (точность)	Precision (точность)	Recall (полнота)	F-measure (F-мера)
Наивный байесовский классификатор	0.795	0.845	0.803	0.849
Метод опорных векторов	0.841	0.901	0.879	0.902
Дерево решений	0.823	0.845	0.878	0.889
Алгоритм случайного леса	0.845	0.859	0.863	0.890
Градиентный бустинг	0.847	0.903	0.883	0.903
Адаптивный бустинг	0.835	0.858	0.861	0.889

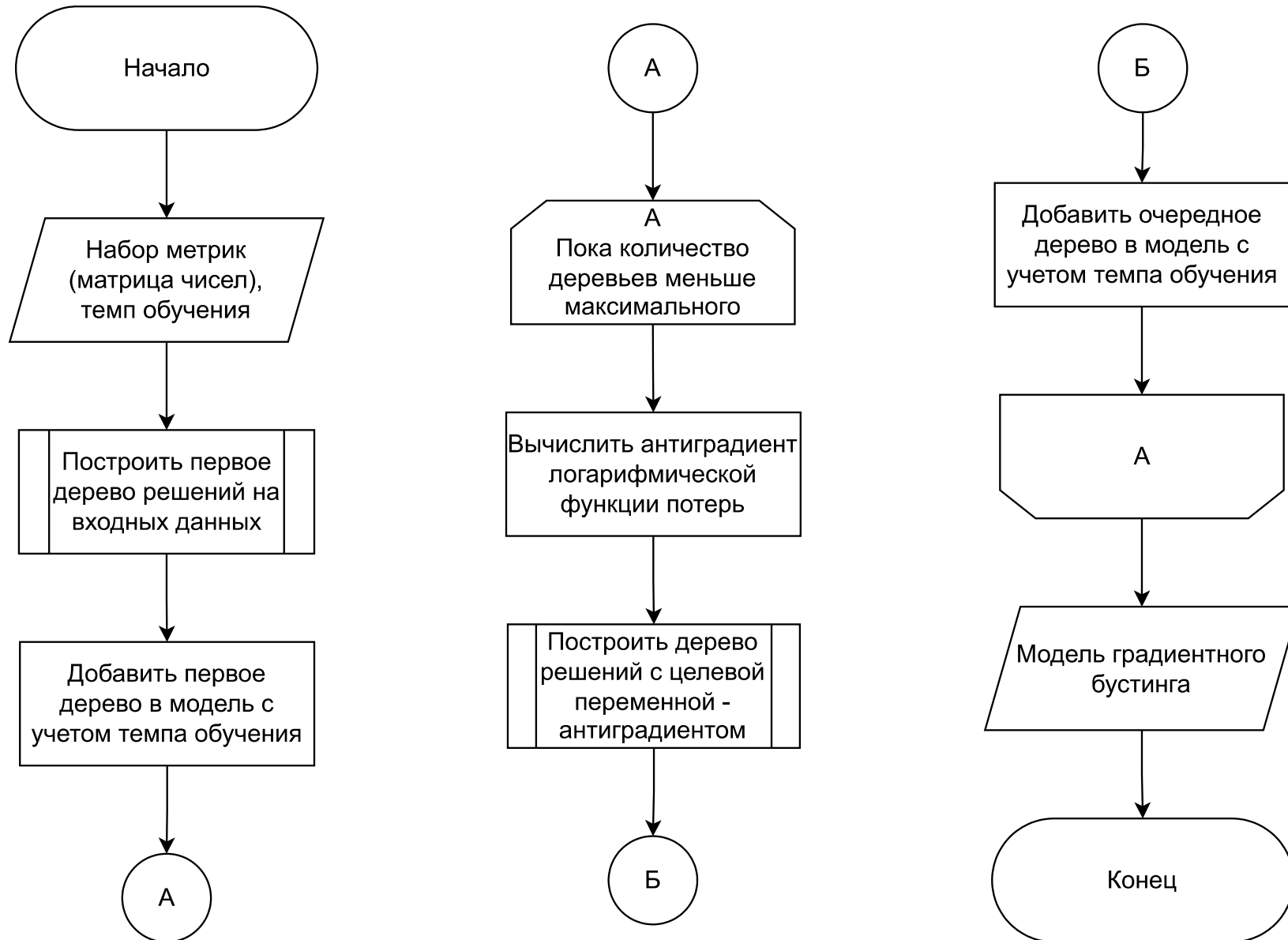
Формализованная постановка задачи



Метод обнаружения дефектов ПО



Обучение модели градиентного бустинга



Обучение модели градиентного бустинга

Логарифмическая функция потерь

$$\mathcal{L}(y, p) = -(y \cdot \log(p) + (1 - y) \cdot \log(1 - p))$$

Антиградиент

$$g_i^k = -\frac{a_k(x_i) - y_i}{a_k(x_i)(a_k(x_i) - 1)}$$

y — действительное значение целевой переменной;

p — предсказанное значение целевой переменной;

y_i — действительное значение целевой переменной для i -го объекта;

$a_k(x_i)$ — предсказанное значение целевой переменной для i -го объекта;

x_i — значения метрик для i -го объекта.

Обучение модели градиентного бустинга

$$a_k(x) = \eta b_1(x, y_1) + \eta b_2(x, y_2) + \dots + \eta b_k(x, y_k)$$

k — количество деревьев, участвующих при построении модели;

η — темп обучения, $\eta \in (0, 1]$;

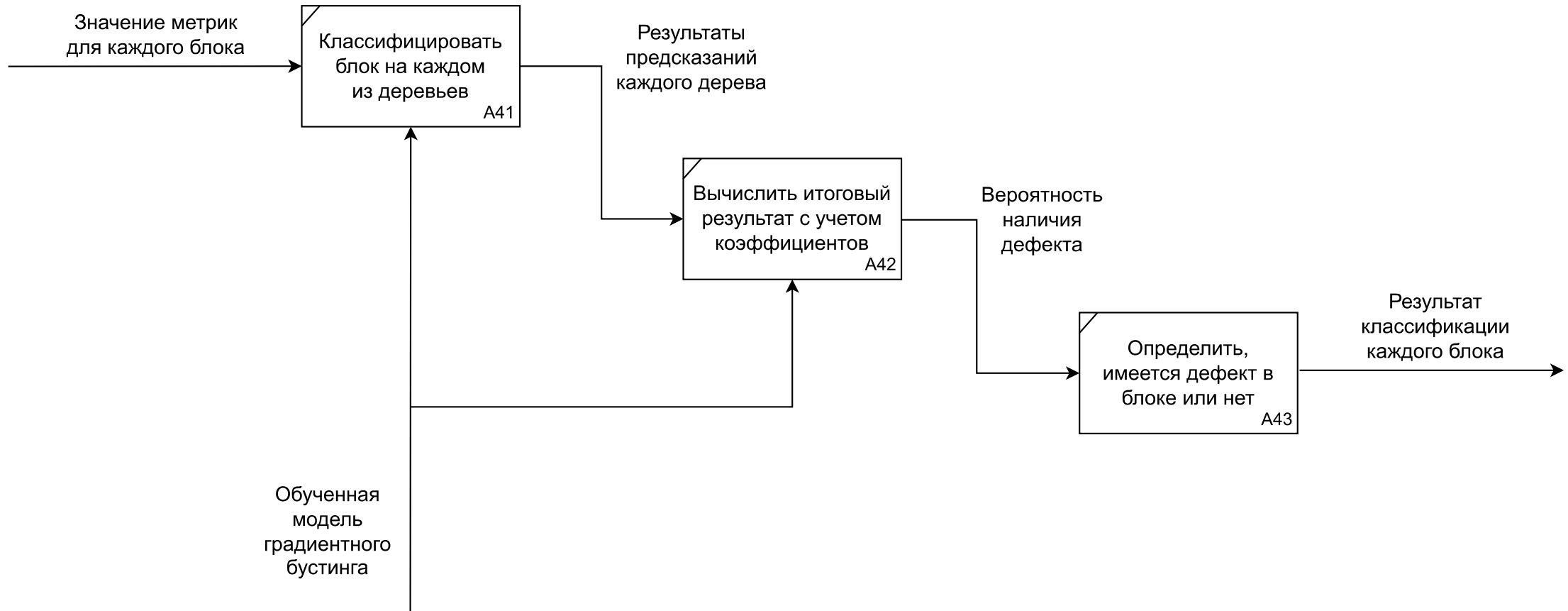
$b_j(x, y_j)$ — результат классификации j -го дерева, $j \in \overline{1; k}$;

x — значения метрик для каждого объекта;

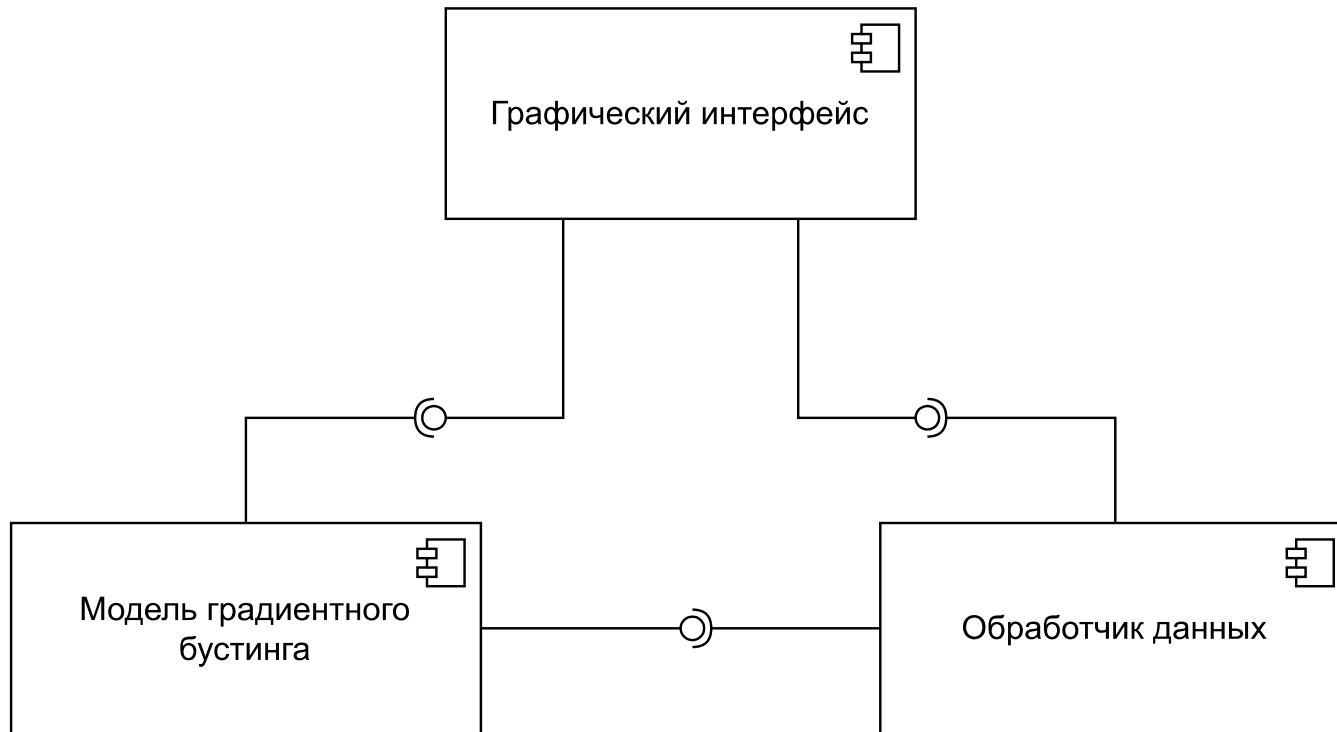
y_j — значения целевой переменной для каждого объекта.

Модель обучается на наборе данных JM1, который представляет собой матрицу метрик, вычисленных для модулей на C++.

Классификация кода по наличию дефекта



Программная реализация разработанного метода



Язык программирования – Python.

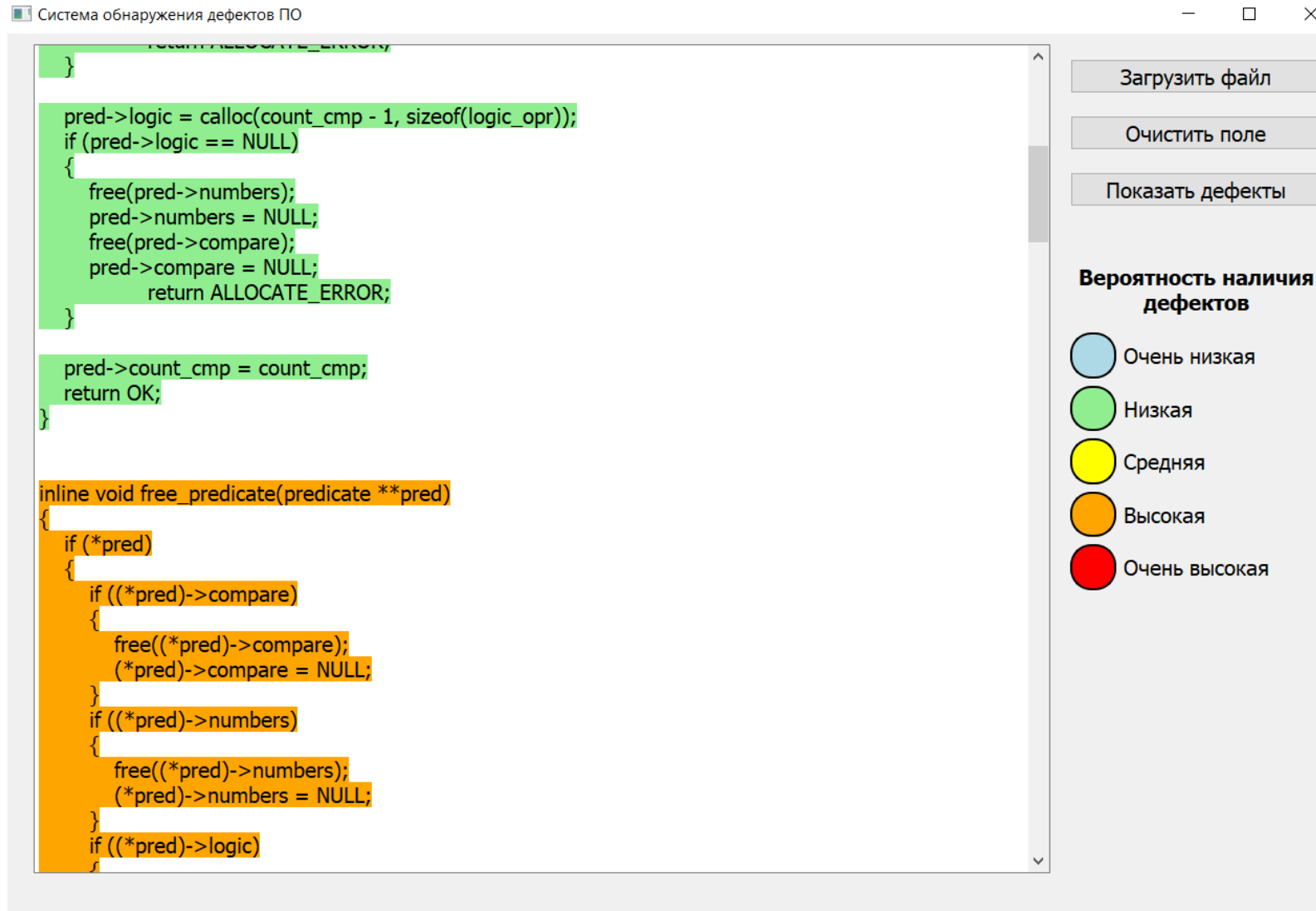
Библиотека для предобработки данных – scikit-learn.

Библиотека для обучения модели методом градиентного бустинга – XGBoost.

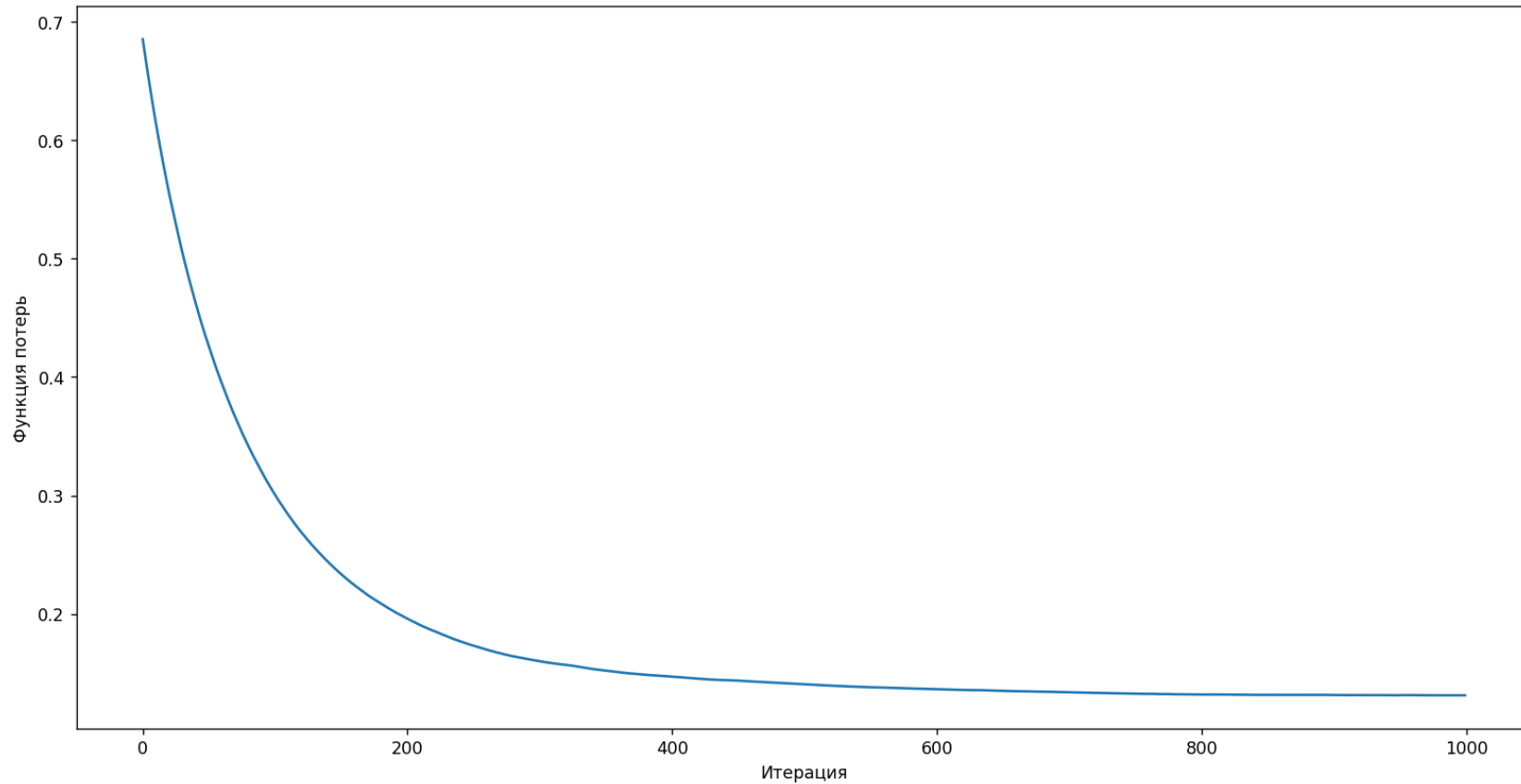
Фреймворк для разработки приложения – PyQt5.

Среда разработки – PyCharm.

Пример работы программы



Анализ точности метода



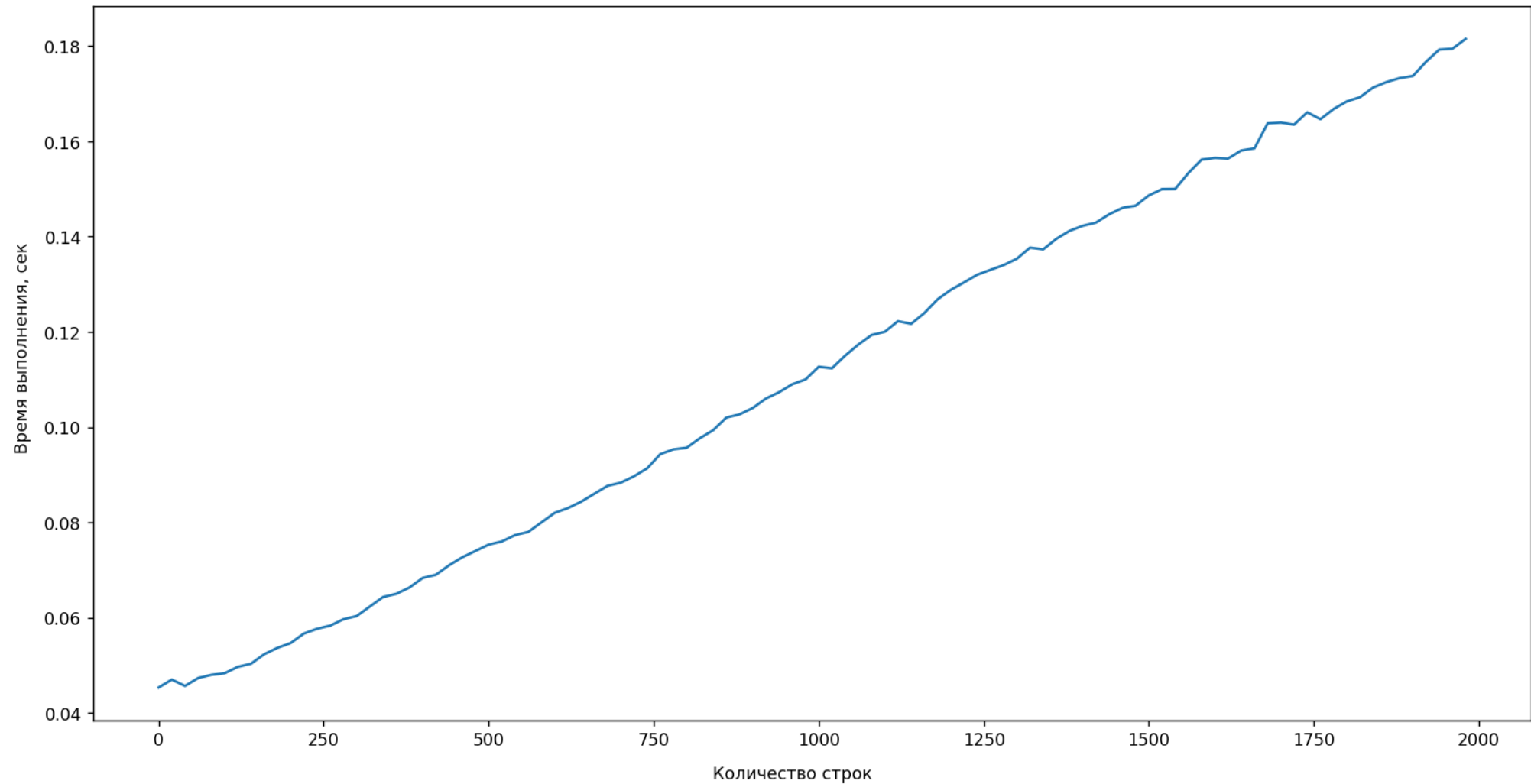
Accuracy = 0.941

Precision = 0.818

Recall = 0.936

F-measure = 0.873

Оценка времени выполнения программы



Заключение

В результате выполнения работы достигнута цель и решены все поставленные задачи:

- 1) проанализированы и сравнены существующие методы машинного обучения для обнаружения дефектов ПО;
- 2) разработан метод обнаружения дефектов ПО с применением ансамбля деревьев решений (градиентного бустинга);
- 3) разработано программное обеспечение, реализующее метод обнаружения дефектов ПО;
- 4) проведено исследование эффективности разработанного метода и сравнение его с существующими реализациями.