

**РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**  
***К НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ***  
***НА ТЕМУ:***

## *Разработка и оценка моделей машинного обучения*

Студент	<u>ИУ5-63Б</u>	<u>(подпись, дата)</u>	<u>Л.А. Ювенский</u>
	(группа)		(И.О. Фамилия)
Руководитель НИР		<u>(подпись, дата)</u>	<u>Ю.Е. Гапанюк</u>
			(И.О. Фамилия)

2025 г.

Министерство науки и высшего образования Российской Федерации  
Федеральное государственное бюджетное образовательное учреждение  
высшего образования  
«Московский государственный технический университет имени Н.Э. Баумана  
(национальный исследовательский университет)»  
(МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ

Заведующий кафедрой

ИУ5

(индекс)

В.И. Терехов

(И.О. Фамилия)

(подпись)

(дата)

**ЗАДАНИЕ**  
**на выполнение научно-исследовательской работы**

по теме Разработка и оценка моделей машинного обучения

Студент группы ИУ5-63Б

Ювенский Лев Александрович

Направленность НИР (учебная, исследовательская, практическая, производственная, др.)

**ИССЛЕДОВАТЕЛЬСКАЯ**

Источник тематики (кафедра, предприятие, НИР) КАФЕДРА

График выполнения НИР:

25% к \_\_\_\_\_ нед., 50% к \_\_\_\_\_ нед., 75% к \_\_\_\_\_ нед., 75% к \_\_\_\_\_ нед

**Техническое задание:** решение задачи машинного обучения на основе материалов

дисциплины. Выбор датасета, первичный анализ, выбор метрик для оценки качества моделей, построение базового решения, оценка качества, подбор гиперпараметров.

**Оформление научно-исследовательской работы:** \_\_\_\_\_

Расчетно-пояснительная записка на \_\_\_\_\_ листах формата А4.

Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)

Дата выдачи задания «07» февраля 2025 г.

Руководитель НИР

(подпись, дата)

**Ю.Е. Гапанюк**

(И.О. Фамилия)

Студент

(подпись, дата)

**Л.А. Ювенский**

(И.О. Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

# СОДЕРЖАНИЕ

ВВЕДЕНИЕ .....	4
1. ПОСТАНОВКА ЗАДАЧИ.....	5
2. АНАЛИЗ ДАТАСЕТА .....	6
3. ВЫБОР МОДЕЛЕЙ И МЕТРИК ДЛЯ ОЦЕНКИ КАЧЕСТВА .....	8
4. ПОСТРОЕНИЕ БАЗОВОГО РЕШЕНИЯ.....	9
5. ПОДБОР ГИПЕРПАРАМЕТРОВ.....	10
6. ОЦЕНКА КАЧЕСТВА МОДЕЛЕЙ .....	12
7. ВЕБ-ПРИЛОЖЕНИЕ .....	13
ЗАКЛЮЧЕНИЕ .....	15
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ .....	16

## **ВВЕДЕНИЕ**

В современном мире вопросы оценки кредитоспособности заемщиков становятся все более актуальными для финансовых организаций, и понимание факторов риска, влияющих на вероятность невозврата кредита, является ключевым для разработки эффективных стратегий управления кредитными рисками. В рамках представленной работы мной было проведено исследование, целью которого был анализ различных моделей машинного обучения, которые могут быть использованы для оценки вероятности невозврата кредитных средств.

Исследование было сосредоточено на анализе набора данных, содержащего информацию о финансовых и социально-демографических характеристиках заемщиков, а также об их кредитной истории. Целью работы стало построение и оценка моделей машинного обучения для прогнозирования вероятности дефолта клиента на основе имеющихся данных.

Модели машинного обучения представляют собой мощный инструмент для анализа сложных взаимосвязей между различными признаками и их влиянием на кредитное поведение заемщиков. Представленное исследование включает в себя всесторонний анализ данных, включая предварительное исследовательское изучение, выбор и обработку признаков, корреляционный анализ, а также подбор гиперпараметров для моделей машинного обучения.

Результаты работы могут быть использованы в качестве основы для дальнейших исследований в области оценки кредитных рисков, а также при разработке надежных и эффективных систем кредитного скоринга.

# 1. ПОСТАНОВКА ЗАДАЧИ

Целью научно-исследовательской работы является разработка и оценка моделей машинного обучения для **прогнозирования кредитного риска** на основе набора данных о заёмщиках. В качестве целевой переменной в задаче **классификации** использована переменная, отражающая высокую, низкую или среднюю вероятность дефолта заемщика.

Датасет включает в себя следующие признаки:

Поле	Описание	Тип данных
age	Возраст человека	int64
annual_income	Годовой доход	float64
num_bank_acc	Количество банковских счетов	int64
num_credit_card	Количество кредитных карт	int64
interest_rate	Процентная ставка по кредитной карте	float64
delay_from_due_date	Количество дней просрочки по платежу	int64
outstanding_debt	Сумма непогашенного долга	float64
credit_history_age	Возраст кредитной истории	float64
payment_of_min_amount	Указание, был ли внесён минимальный платёж	bool
installment_per_month	Сумма ежемесячного платежа	float64
monthly_balance	Ежемесячный баланс	float64

## 2. АНАЛИЗ ДАТАСЕТА

В результате анализа **пропуски** в данных **не обнаружены**:

age	0
annual_income	0
num_bank_acc	0
num_credit_card	0
interest_rate	0
delay_from_due_date	0
outstanding_debt	0
credit_history_age	0
installment_per_month	0
monthly_balance	0
payment_of_min_amount_yes	0
credit_score	0

Рисунок 1 – Результаты анализа пропусков в данных.

Построим **диаграммы типа boxplot** для иллюстрации распределения значений признаков.

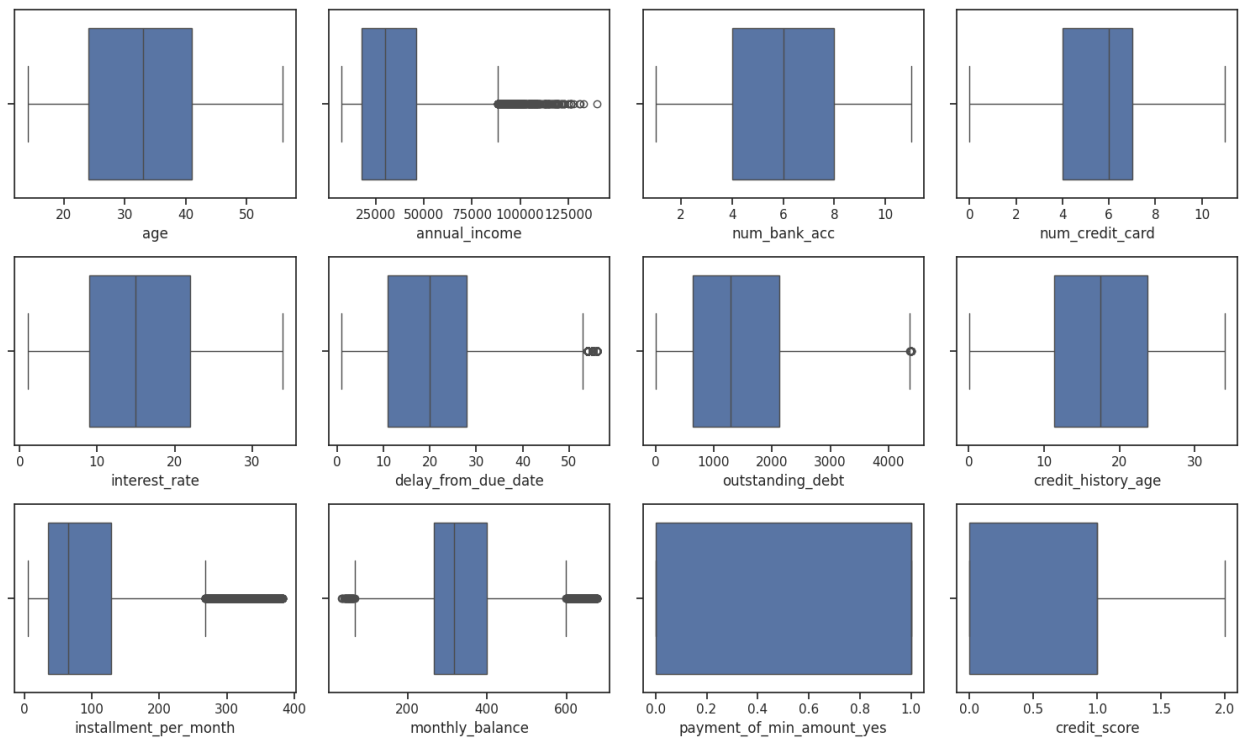


Рисунок 2 – Распределение значений признаков.

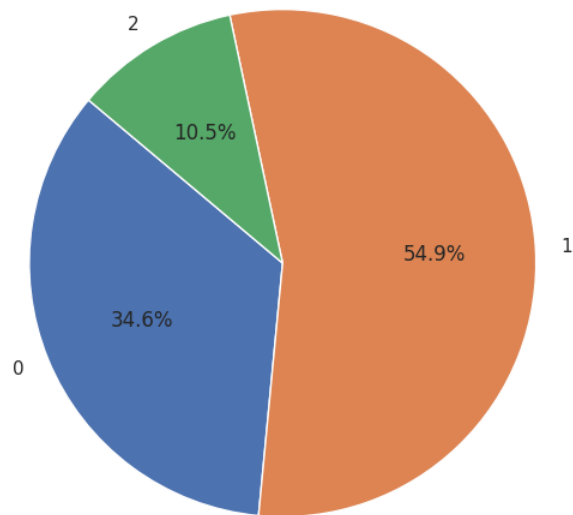


Рисунок 3 – Распределение данных по целевому признаку.

Построим корреляционную матрицу признаков:

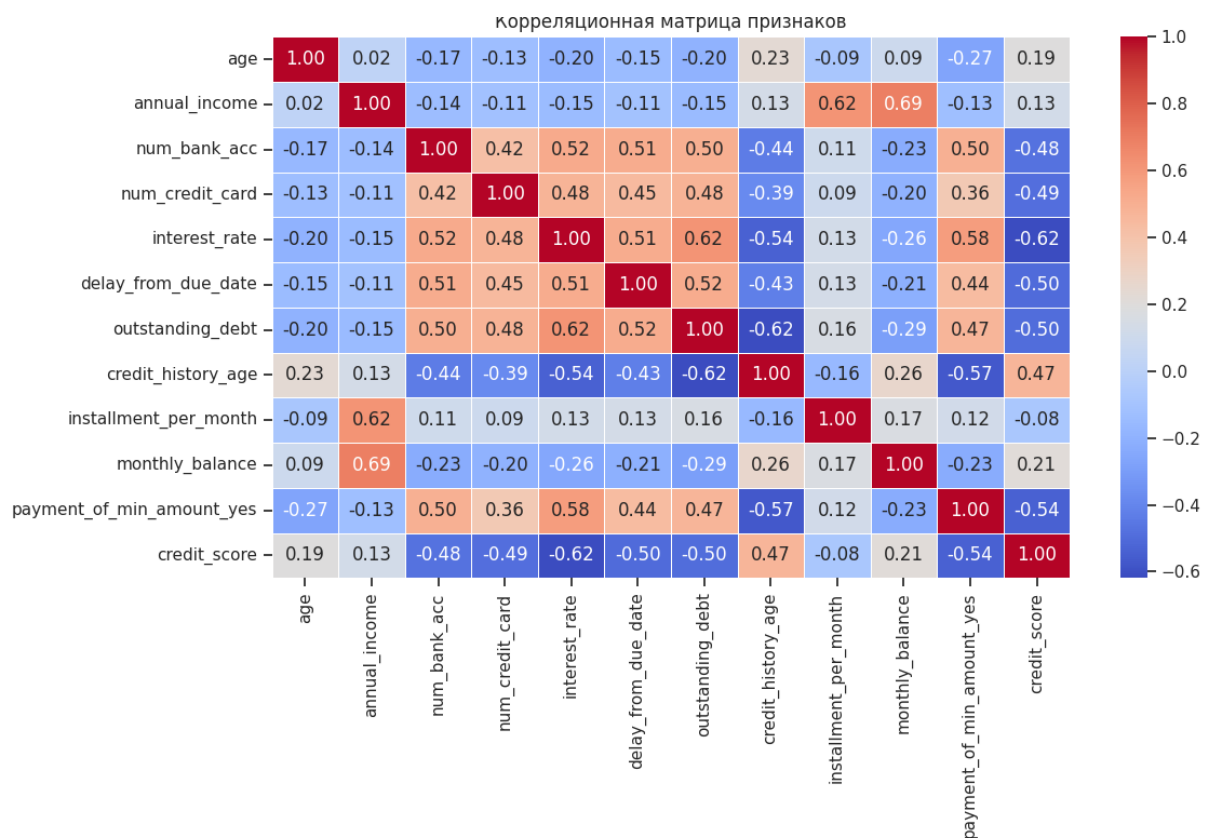


Рисунок 4 – Корреляционная матрица признаков.

Большая часть признаков имеют сильную корреляцию с целевым признаком. Признаки не имеют сильной корреляции между собой, следовательно, датасет не содержит избыточных признаков.

### 3. ВЫБОР МОДЕЛЕЙ И МЕТРИК ДЛЯ ОЦЕНКИ КАЧЕСТВА

Для оценки качества моделей выберем следующие метрики:

- **Accuracy** (Точность общего предсказания). Доля правильно предсказанных примеров от общего числа
- **Precision**. Доля правильных положительных предсказаний среди всех предсказанных положительных классов
- **Weighted F1-score**. Среднее значение F1-метрики с учётом частоты классов.
- **Recall**. Показывает, насколько хорошо модель находит все объекты, которые действительно принадлежат классу.

С учетом следующих факторов:

- Требование задания НИРС: не менее 5 моделей, не менее 2 ансамблевых моделей;
- Решение задачи многоклассовой классификации;
- Датасет содержит числовые признаки.

выберем следующие модели:

- Логистическая регрессия;
- k-ближайших соседей;
- Дерево решений;
- Случайный лес;
- Градиентный бустинг.




## 4. ПОСТРОЕНИЕ БАЗОВОГО РЕШЕНИЯ

Построим базовое решение для каждой из моделей:

### ✓ Логистическая регрессия

```
[ ] log_reg = LogisticRegression(max_iter=1000)
log_reg.fit(xTrain, yTrain)
y_pred_log_reg = log_reg.predict(xTest)
```

 /usr/local/lib/python3.11/dist-packages/sklearn/linear\_model/\_logistic.py:465: ConvergenceWarning: lbfgs failed to converge (status=1):  
STOP: TOTAL NO. OF ITERATIONS REACHED LIMIT.

Increase the number of iterations (max\_iter) or scale the data as shown in:

<https://scikit-learn.org/stable/modules/preprocessing.html>

Please also refer to the documentation for alternative solver options:

[https://scikit-learn.org/stable/modules/linear\\_model.html#logistic-regression](https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression)

n\_iter\_i = \_check\_optimize\_result(

### ✓ k-ближайших соседей

```
[ ] knn = KNeighborsClassifier()
knn.fit(xTrain, yTrain)
y_pred_knn = knn.predict(xTest)
```

### ✓ Дерево решений

```
[ ] decision_tree = DecisionTreeClassifier(random_state=1)
decision_tree.fit(xTrain, yTrain)
y_pred_tree = decision_tree.predict(xTest)
```

### ✓ Случайный лес

```
[ ] random_forest = RandomForestClassifier(random_state=1)
random_forest.fit(xTrain, yTrain)
y_pred_forest = random_forest.predict(xTest)
```

### ✓ Градиентный бустинг

```
[ ] gradient_boosting = GradientBoostingClassifier(random_state=1)
gradient_boosting.fit(xTrain, yTrain)
y_pred_gb = gradient_boosting.predict(xTest)
```

Рисунок 5 – Базовое решение.

Метрика \ Модель	Логистическая регрессия	k-ближайших соседей	Дерево решений	Случайный лес	Градиентный бустинг
Accuracy	0.7424	0.7471	0.8878	0.9290	0.8418
Precision	0.7331	0.7307	0.8873	0.9290	0.8415
Recall	0.7424	0.7471	0.8878	0.9290	0.8418
Weighted F1-score	0.7324	0.7366	0.8875	0.9290	0.8416

Рисунок 6 – Оценка точности базового решения.

## 5. ПОДБОР ГИПЕРПАРАМЕТРОВ

Для повышения точности моделей произведём подбор гиперпараметров моделей с использованием функции **GridSearchCV**.

### ✓ Логистическая регрессия

```
[ ] log_reg = LogisticRegression(max_iter=5000)

    log_reg_param_grid = {
        'solver': ['lbfgs', 'liblinear']
    }
    log_reg_grid = GridSearchCV(log_reg, log_reg_param_grid, cv=5, scoring='accuracy')
    log_reg_grid.fit(xTrain, yTrain)
    y_pred_log_reg_tuned = log_reg_grid.best_estimator_.predict(xTest)
```

### ✓ k-ближайших соседей

```
[ ] knn = KNeighborsClassifier()

    knn_param_grid = {
        'n_neighbors': [5, 3, 7, 9], # 5 — дефолт
        'weights': ['uniform', 'distance'], # uniform — дефолт
        'metric': ['minkowski', 'euclidean'], # minkowski — дефолт
    }
    knn_grid = GridSearchCV(knn, knn_param_grid, cv=5, scoring='accuracy')
    knn_grid.fit(xTrain, yTrain)
    y_pred_knn_tuned = knn_grid.best_estimator_.predict(xTest)
```

### ✓ Дерево решений

```
[ ] decision_tree = DecisionTreeClassifier(random_state=1)

    tree_param_grid = {
        'criterion': ['gini', 'entropy'], # gini — дефолт
        'min_samples_split': [2, 5], # 2 — дефолт
        'min_samples_leaf': [1, 2], # 1 — дефолт
    }
    tree_grid = GridSearchCV(
        estimator=decision_tree,
        param_grid=tree_param_grid,
        cv=5,
        scoring='accuracy'
    )

    tree_grid.fit(xTrain, yTrain)
    y_pred_tree_tuned = tree_grid.best_estimator_.predict(xTest)
```

### ✓ Случайный лес

```
[ ] random_forest = RandomForestClassifier(random_state=1)

random_forest_param_grid = {
    'n_estimators': [100, 50, 200], # 100 – дефолт
    'min_samples_split': [2, 5], # 2 – дефолт
}
random_forest_grid = GridSearchCV(random_forest, random_forest_param_grid, cv=5, scoring='accuracy')
random_forest_grid.fit(xTrain, yTrain)
y_pred_forest_tuned = random_forest_grid.best_estimator_.predict(xTest)
```

### ✓ Градиентный бустинг

```
[ ] gb = GradientBoostingClassifier(random_state=1)

gb_param_grid = {
    'max_depth': [3, 7], # 3 – дефолт
}
gb_grid = GridSearchCV(gb, gb_param_grid, cv=5, scoring='accuracy')
gb_grid.fit(xTrain, yTrain)
y_pred_gb_tuned = gb_grid.best_estimator_.predict(xTest)
```

Рисунок 7 – Подбор гиперпараметров.

Метрика \ Модель	Логистическая регрессия	к-ближайших соседей	Дерево решений	Случайный лес	Градиентный бустинг
Accuracy	0.7539	0.8354	0.8891	0.9314	0.9145
Precision	0.7533	0.8312	0.8892	0.9314	0.9141
Recall	0.7539	0.8354	0.8891	0.9314	0.9145
Weighted F1-score	0.7425	0.8328	0.8891	0.9314	0.9143

Рисунок 8 – Оценка точности моделей после подбора гиперпараметров.

## 6. ОЦЕНКА КАЧЕСТВА МОДЕЛЕЙ

Построим диаграммы, отображающие оценки качества исследуемых моделей до и после подбора гиперпараметров:

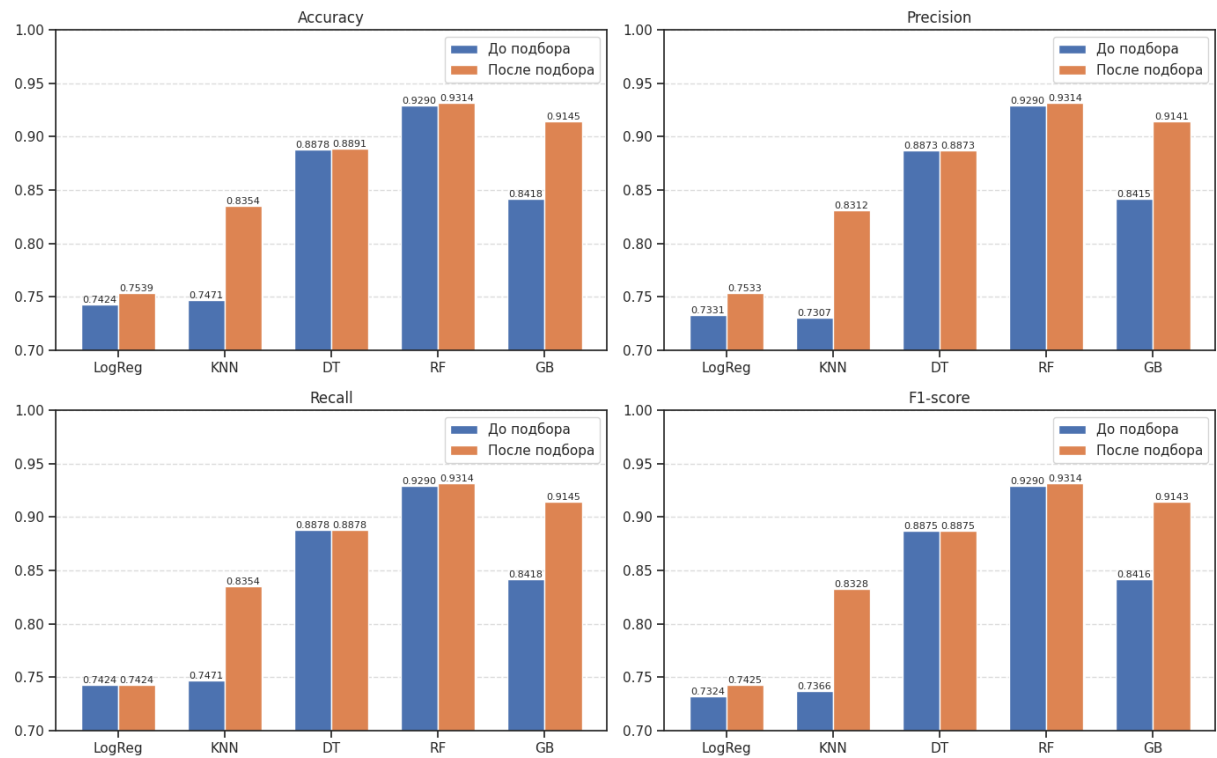


Рисунок 9 – Оценка точности моделей.

В ходе исследования наибольшую точность (93,14%) по всем анализируемым метрикам показала модель **Случайный лес (Random Forest)**.

## 7. ВЕБ-ПРИЛОЖЕНИЕ

Реализуем веб-приложение для демонстрации влияния гиперпараметров на точность модели **Случайный лес**. Используем фреймворк **Streamlit**.

```
random-forest-streamlit.py

import streamlit as st
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, precision_score, recall_score, f1_score

def load_data():
    return pd.read_csv("../dataset.csv")

def model_training_page():
    st.title("Обучение модели RandomForest для определения кредитного рейтинга")

    df = load_data()

    X = df.drop(columns=["credit_score"])
    y = df["credit_score"]

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=1)

    st.subheader("Параметры модели")

    col1, col2 = st.columns(2)
    with col1:
        n_estimators = st.slider("Количество деревьев (n_estimators)", 10, 1000, 100, step=10)
    with col2:
        max_depth = st.selectbox("Максимальная глубина дерева (max_depth)", [None, 1, 10, 100, 1000])

    model = RandomForestClassifier(n_estimators=n_estimators, max_depth=max_depth, random_state=1)

    model.fit(X_train, y_train)

    y_pred = model.predict(X_test)

    accuracy = accuracy_score(y_test, y_pred)

    precision = precision_score(y_test, y_pred, average='weighted', zero_division=0)
    recall = recall_score(y_test, y_pred, average='weighted', zero_division=0)
    f1 = f1_score(y_test, y_pred, average='weighted', zero_division=0)

    metrics_data = {
        "Метрика": ["Accuracy", "Precision", "Recall", "F1-Score"],
        "Значение": [f"{accuracy * 100:.2f}%", f"{precision * 100:.2f}%", f"{recall * 100:.2f}%", f"{f1 * 100:.2f}%"]
    }
    metrics_df = pd.DataFrame(metrics_data)

    st.subheader("Метрики модели")
    st.table(metrics_df.set_index("Метрика"))

    cm = confusion_matrix(y_test, y_pred)
    st.subheader("Матрица ошибок")
    fig, ax = plt.subplots(figsize=(8, 6))
    sns.heatmap(cm, annot=True, fmt="d", cmap="Blues", xticklabels=["Poor", "Standard", "Good"], yticklabels=["Poor", "Standard", "Good"])
    ax.set_xlabel("Предсказано")
    ax.set_ylabel("Реальное значение")
    st.pyplot(fig)

def main():
    model_training_page()

if __name__ == "__main__":
    main()
```

Рисунок 10 – Реализация веб-приложения для демонстрации модели.

# Обучение модели RandomForest для определения кредитного рейтинга

## Параметры модели

Количество деревьев (n\_estimators)  
100  
101080

Максимальная глубина дерева (max\_depth)  
None

## Метрики модели

Метрика	Значение
Accuracy	93.00%
Precision	93.00%
Recall	93.00%
F1-Score	93.00%

## Матрица ошибок

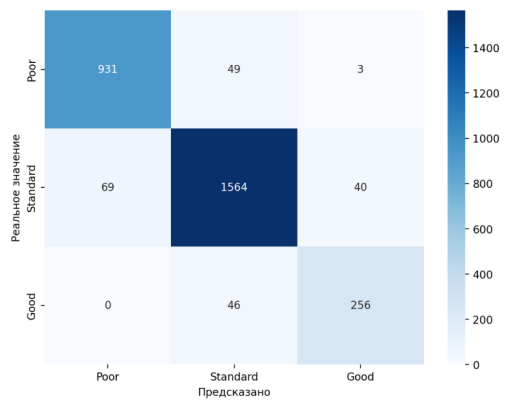


Рисунок 11 – Веб-приложение.

## ЗАКЛЮЧЕНИЕ

В ходе выполнения научно-исследовательской было проведено исследование задачи определения кредитного рейтинга. Были проанализированы различные модели машинного обучения, в том числе ансамблиевые, а также их оптимизация путём подбора гиперпараметров.

Наиболее высокую точность по исследуемым метрикам показала модель **Случайный лес** после подбора гиперпараметров.

Для демонстрации влияния гиперпараметров на точность модели было разработано веб-приложение.

Таким образом, исследуемые модели машинного обучения и проведенный анализ позволяют не только прогнозировать вероятность дефолта заемщика, но и выявлять ключевые факторы, влияющие на кредитное поведение клиентов. Результаты исследования могут служить основой для дальнейших научных и прикладных работ в области финансовой аналитики и управления рисками.

## СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Credit Score Prediction [Электронный ресурс] // github.com. URL: <https://github.com/ongaunjie1/credit-score-prediction> (дата обращения: 02.05.2025);
2. Документация Streamlit [Электронный ресурс] // streamlit.io URL: <https://streamlit.io/> (дата обращения: 01.05.2025);
3. «Python Data Science Handbook» Джейк Вандер-Плас [Электронный ресурс] // jakevdp.github.io. URL: <https://jakevdp.github.io/PythonDataScienceHandbook/> (дата обращения: 02.05.2025);
4. Документация по Python [Электронный ресурс] // Python. URL: <https://docs.python.org/3/index.html/> (дата обращения: 01.05.2025);
5. Методические указания НИРС по дисциплине «Технологии машинного обучения».