Московский государственный университет имени М.В. Ломоносова

Направление 38.03.01 Экономика

Программа «Национальная экономика»

Прогнозирование рисков банковского кредитования с использованием технологий искусственного интеллекта

Выпускная квалификационная работа

Студент

Касенова Асем Ардаковна

Научный руководитель:

к.э.н., к.ф.-м.н., к.ю.н., доцент

Сидоренко Владимир Николаевич

Астана

2026

ОГЛАВЛЕНИЕ

ГЛАВА 1. ТЕСТИРОВАНИЕ АЛГОРИТМОВ МАШИННОГО И ГЛУБОКОГО ОБУЧЕНИЯ НА ДАННЫХ ПО КРЕДИТНОМУ СКОРИНГУ

В предыдущей главе было выяснено, что алгоритмы градиентного бустинга и нейронных сетей не имеют эффектов переобучения и недообучения. В случае градиентного бустинга используются гиперпараметры по умолчанию, результаты метрик которых не ниже метрик других алгоритмов. Это означает, что не требуется поиск гиперпараметров для данного алгоритма и можно остановиться на гиперпараметрах по умолчанию. Идентичная ситуация возникает у нейронной сети с отличием в том, что у нее метрики точности немного хуже, чем у алгоритма градиентного бустинга. Поэтому в этой главе будут подобраны гиперпараметры для алгоритмов деревьев решений и случайного леса, поскольку они обладают эффектом переобучения. Основным инструментом для поиска оптимальных гиперпараметров используются инструменты *GridSearchCV*[1] и *RandomizedSearchCV*[2]. Также в конце главы будут проанализированы прибыль и убыток банка на основании предсказаний четырех рассмотренных алгоритмов.

1.1 Подбор гиперпараметров для алгоритмов машинного обучения при оценке возврата кредита

Гиперпараметры – это такие параметры модели, которые задаются до начала работы модели. Правильно подобранные параметры позволяют улучшить качество предсказаний моделей. Каждому алгоритму МО присуще собственные наборы гиперпараметров, которые будут рассматриваться ниже. Существуют два метода подбора гиперпараметров:

1. *GridSearchCV* – метод подбора оптимальных гиперпараметров для модели с помощью перебора всех возможных комбинаций из заданного набора, что является ресурсозатратным способом.

2. *RandomizedSearchCV* – метод, позволяющий выбирать количество случайных комбинаций из заданного набора гиперпараметров. Является наиболее эффективным методом по времени при работе с большими данными.

[1] GridSearchCV [Электронный ресурс] / Python-библиотека для машинного обучения. URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html (дата обращения: 25.11.2025).

[2] **RandomizedSearchC** (дата обращения: 25.11.2025).

1.1.1 Деревья решений

Для алгоритма классификации решающих деревьев (*Decision Tree Classifier*) основными гиперпараметрами являются:

1. *criterion* – критерий качества разбиения, определяющая показатель, по которому алгоритм лучше классифицирует клиентов.

   a) *gini* – индекс Джини[3], измеряющий неоднородность двух классов (0 и 1) внутри узла. Если индекс равен 0, значит в узле содержатся объекты одного класса. Чем индекс ближе к 0, тем лучше алгоритм определяет класс клиента. Индекс Джини в алгоритме *Decision Tree* используется по умолчанию, поскольку быстрее вычисляется;

   b) *entropy* – энтропия Шеннона[4], измеряющий степень смешанности классов. Высокая энтропия свидетельствует о высокой неоднородности классов, чем ниже энтропия, тем узел более однородный. Алгоритм выбирает разбиения с наиболее низкой энтропией, т.е. содержатся большинство объектов одного класса;

   c) *log_loss* – логарифмическая функция потерь. В данном случае алгоритм не просто выбирает класс, а оценивает их вероятности. За каждый неверный прогноз вводит штраф.

1.1.2 Случайный лес

1.1.3 Градиентный бустинг

1.1.4 Нейронная сеть

1.2 Ограничения и возможности моделей

[3]Джини К. (1884-1965 гг.) – итальянский статистик, экономист, социолог и демограф.

[4]Шеннон К.Э. (1916-2001 гг.) – американский инженер, криптоаналитик и математик, заложил основы теории информации.

СПИСОК ЛИТЕРАТУРЫ

1. GridSearchCV [Электронный ресурс] / Python-библиотека для машинного обучения. — URL: https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html. — Загл. с экрана.

Приложение 1. Предобработка данных

Листинг 1 – Формирование единого датасета

```
import pandas as pd
import os
import pyarrow.parquet as pq

path = "train_data"
for i,file in enumerate(os.listdir(path)):
    file = 'train_data_' + str(i) + '.pq'
    dataset = pq.ParquetDataset(os.path.join(path,file))
    df = dataset.read(use_threads=True).to_pandas()
    df_gr = df.groupby('id').agg('mean')
    file_csv = file.replace('.pq', '.csv')
    df_gr.to_csv(os.path.join('train_data_csv_all',file_csv))
os.listdir(path)

path = 'train_data_csv_all'
frames = []
for file_csv in os.listdir(path):
    if file_csv.endswith('.csv'):
        df = pd.read_csv(os.path.join(path, file_csv))
        frames.append(df)

result = pd.concat(frames)
file_csv_all = '1_data_csv_all.csv'
result.to_csv(file_csv_all)
df_all = pd.read_csv(file_csv_all)
```

Листинг 2 – Датасет, содержащий 41 признак

```
from sklearn.decomposition import PCA

file_csv = 'data_csv_small.csv'
df = pd.read_csv(file_csv)

columns = ['pre_pterm', 'pre_fterm',
           'pre_loans_next_pay_summ','pre_loans_outstanding',
           'pre_loans_total_overdue', 'pre_loans_max_overdue_sum',
           'pre_loans_credit_cost_rate', 'is_zero_loans5',
           'is_zero_loans530', 'is_zero_loans3060',
           'is_zero_loans6090', 'is_zero_loans90',
           'pre_util','pre_maxover2limit',
           'is_zero_util', 'is_zero_over2limit',
           'enc_paym_0', 'enc_paym_1', 'enc_paym_2',
           'enc_paym_3', 'enc_paym_4', 'enc_paym_5',
           'enc_paym_6', 'enc_paym_7', 'enc_paym_8',
           'enc_paym_9', 'enc_paym_10', 'enc_paym_11',
           'enc_paym_12', 'enc_paym_13', 'enc_paym_14',
           'enc_paym_15', 'enc_paym_16', 'enc_paym_17',
           'enc_paym_18', 'enc_paym_19', 'enc_paym_20',
           'enc_paym_21', 'enc_paym_22',
           'enc_paym_23', 'enc_paym_24']

X = df.loc[:, columns].copy()
pca = PCA()
pca.fit(X)

100*pca.explained_variance_ratio_.round(3)
```

Листинг 3 – Датасет, содержащий 24 признака

```
file_csv = 'data_csv_small.csv'
df = pd.read_csv(file_csv)

columns_pay = ['enc_paym_0', 'enc_paym_1', 'enc_paym_2',
               'enc_paym_3', 'enc_paym_4', 'enc_paym_5',
               'enc_paym_6', 'enc_paym_7', 'enc_paym_8',
               'enc_paym_9', 'enc_paym_10', 'enc_paym_11',
               'enc_paym_12', 'enc_paym_13', 'enc_paym_14',
               'enc_paym_15', 'enc_paym_16', 'enc_paym_17',
               'enc_paym_18', 'enc_paym_19', 'enc_paym_20',
               'enc_paym_21', 'enc_paym_22', 'enc_paym_23',
               'enc_paym_24']

X_pay = df.loc[:, columns_pay].copy()
pca = PCA()
pca.fit(X_pay)

100*pca.explained_variance_ratio_.round(3)
```

Листинг 4 – Алгоритм выявления аномальных клиентов

```
df_y = pd.read_csv('train_target.csv')
y  = df_y.flag.values[:1000000].copy()
y_ = df_y.flag.values[:1000000].copy()

counter_norm, counter_anom = 0, 0
ind_norm, ind_anom = [], []
for i,x_pay in enumerate(np.array(X_pay)):
    if y_[i]==0 and min(x_pay)>0:
        counter_anom +=1
        ind_anom.append(i)
        y_[i]=1
    else:
        counter_norm +=1
        ind_norm.append(i)
```

Листинг 5 – Критерий Колмогорова-Смирнова

```
col = 'enc_paym_15'
T = X_pay.enc_paym_15
x00, x01 = T[(y==0) & (y_== 0)], T[ind_anom]
x11, x01 = T[(y==1)], T[ind_anom]
ind00 = (x00!=0) & (x00!=1) & (x00!=2) & (x00!=3) & (x00!=4) & (x00
   !=5)
ind01 = (x01!=0) & (x01!=1) & (x01!=2) & (x01!=3) & (x01!=4) & (x01
   !=5)
ind11 = (x11!=0) & (x11!=1) & (x11!=2) & (x11!=3) & (x11!=4) & (x11
   !=5)
x00, x11, x01 = x00[ind00], x11[ind11], x01[ind01]

def my_cdf(x1, x2, step):
    tm = max(max(x1), max(x2))
    F1,F2 = [], []
    n1, n2 = x1[x1>0].shape[0], x2[x2>0].shape[0]
    for t in np.arange(0, tm+0.1, step):
        F1.append(x1[(x1>0)&(x1<t)].shape[0])
        F2.append(x2[(x2>0)&(x2<t)].shape[0])
    F1, F2 = np.array(F1)/n1, np.array(F2)/n2
    K = max(abs(F1-F2))*np.sqrt( n1*n2/(n1+n2))
    return F1, F2, K, tm
step = 0.01
F01,  F11, K,  tm = my_cdf(x01, x11, step)
F01_, F00, K_, tm = my_cdf(x01, x00, step)
K, K_
```

Листинг 6 – Тест Фишера на равенство коэффициентов корреляции выборочной и генеральной совокупностей

```
def transform_f(r, R, n):
    if n > 3:
        return 0.5*np.log(((1+r)/(1-r))*((1-R)/(1+R)))*np.sqrt(n-3)
    else:
        return 0

mth = 17
n_all, n_nod, n_def, n_ano = X_pay.shape[0], len(ind_nod), len(
   ind_def), len(ind_ano)
R_all, r_nod, r_def, r_ano = all_c[mth], nod_c[mth], def_c[mth],
   ano_c[mth]
transform_f(r_nod, R_all, n_nod), transform_f(r_def, R_all, n_def ),
   transform_f(r_ano, R_all, n_ano )
```

Приложение 2