УДК 004.8

МОДИФИКАЦИЯ НЕЙРОННОЙ СЕТИ XGBOOST В ЗАДАЧИ ДЕТЕКЦИИ МОШЕННИЧЕСКИХ БАНКОВСКИХ ТРАНЗАКЦИЙ

Пылов П.А., студент гр. ИТм-201, 2 курс

Протодьяконов А. В., к.т.н., доцент

Кузбасский государственный технический университет
имени Т.Ф. Горбачева

Аннотация: Искусственный интеллект и машинное обучение, в частности, применяются во всё больших областях человеческой деятельности. Сфера финансов и транзакций денежных единиц всегда оставалась резонным поводом мошенничества, поэтому требует усиленного контроля со стороны закона. С наступлением эры электронного финансового потока правоохранительным органам становится всё сложнее оценивать законность совершения транзакций [1], поэтому внедрение технологий автоматического контроля на основе алгоритмов искусственного интеллекта позволяет специалистам существенно снизить риски незаконных операций. Однако, следует обратить внимание, что не все алгоритмы искусственного интеллекта ориентированы на решение поставленной задачей. В статье представлена модель, которая при всех сложностях обработки входных данных и поиска закономерностей в них, позволяет получить эффективное решение вышеописанной задачи.

С появлением электронных транзакций финансовой сфере требуются самые совершенные решения информационных технологий, иначе она становится особенно уязвимой для мошеннических операций [1].

Кроме защищённых каналов передачи информации, токенизации и других подходов, обеспечивающих информационную безопасность банковских операций, не следует упускать из виды системы неочевидной безопасности, то есть, когда злоумышленники могут завладеть данными карты клиента и расплачиваться ею от его имени. Именно эта подобласть и является большинством в мошеннических операциях.

При решении данной проблемы широко применяются технологии машинного и глубокого обучения. Практически весь спектр наиболее

распространенных алгоритмов машинного обучения был апробирован при решении данной задачи (логистическая регрессия, линейная регрессия, случайный лес, деревья решений), усложнением решения выступали и нейронные сети [2].

Поставленная задача сформулирована на основе датасета, в котором записаны данные транзакций, совершенных пользователями стран Европейского Союза за период одного месяца 2020 года[1].

При решении задачи прежде всего стоит обратить внимание на несбалансированность набора данных, представленного в проекте. Из этого следует, что выбор инструмента решения напрямую влияет на итоговый результат, поэтому следует либо видоизменять алгоритм к несбалансированному исходному набору, либо подбирать наиболее подходящую модель и совершенствовать её результаты [3].

В данной статье представлено решение задачи, основанное на использование алгоритма xgboost и языка программирования Python. Фундаментальная основа xgboost восходит к поиску на основе дерева, а в качестве внутренней модели оптимизации используется градиентный бустинг [4]. Такая комбинация полностью удовлетворяет требованиям поставленной задачи и, как следует логически предположить, позволит получить наилучшие результаты в прикладной области.

В качестве первого шага решения изучим баланс классов в данных (рисунок 1)

[1] Источник данных – Machine Learning Group. Ссылка проекта https://mlg.ulb.ac.be/wordpress/portfolio_page/defeatfraud-assessment-and-validation-of-deep-feature-engineering-and-learning-solutions-for-fraud-detection/

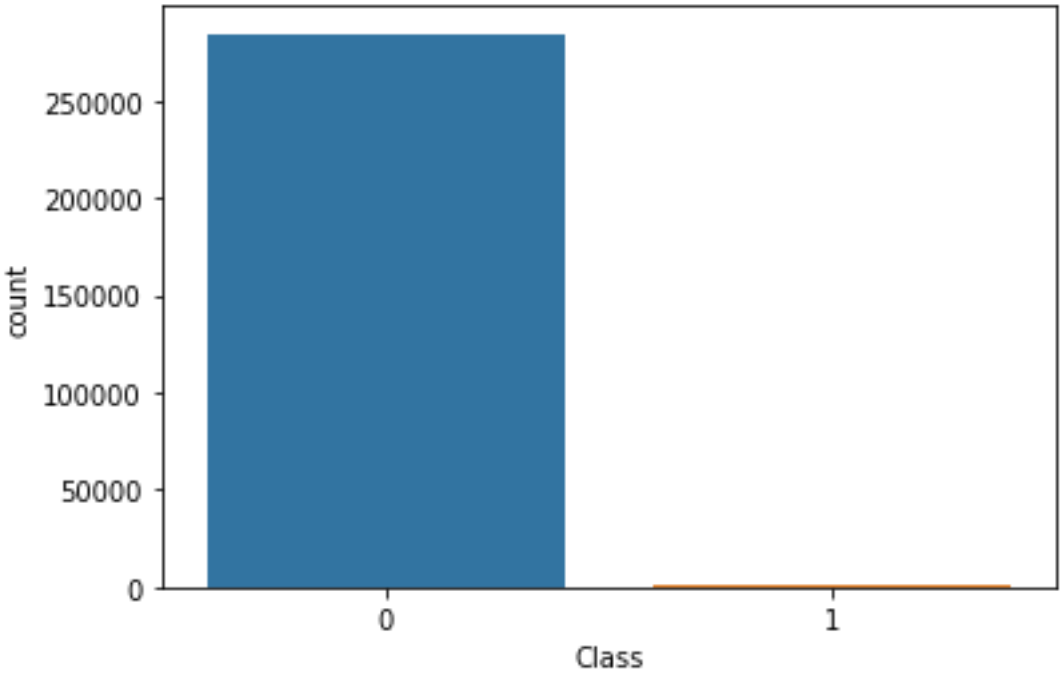

Рисунок 1 – Соотношение классов в исходных данных

Из рисунка 1 очевидным образом заметен высокий уровень дисбаланса данных. Весь представленный набор содержит в себе немногим более 1% мошеннических операций, что усложняет задачу обучения при разделении выборок на тренировочную и тестовую [6].

По этой причине многие исследователи данных остановили свой выбор на алгоритмах логистической и линейной регрессии, так как использование hold – out подхода работы с данными в этих алгоритмах сохраняют эффективность своего решения на значениях процентного соотношения разделения выборки на тренировочную и тестовую 90/10 [5].

Но если останавливать внимание только на сложности разделения выборки и недостаточно большем количестве экземпляров оцениваемого класса, то возрастает вероятность упустить наиболее подходящий алгоритм для решения задачи.

Предложенный в качестве решения алгоритм xgboost позволяет эффективно решить поставленную задачу определения мошеннических операций на основе высокой обобщающей способности (рисунок 2).

```
Ввод [15]: def prediction(model,X_train, X_test, y_train, y_test):
               model.fit(X_train,y_train)
               preds=model.predict(X_test)
               print(confusion_matrix(y_test,preds))
               print(classification_report(y_test,preds))
               return accuracy_score(y_test,preds)

Ввод [16]: prediction(XGBClassifier(),X_train,X_test, y_train, y_test)

c:\users\petr\appdata\local\programs\python\python39\lib\site-packages\xgboost\sklearn.py:1146: UserWarning: The use of label e
ncoder in XGBClassifier is deprecated and will be removed in a future release. To remove this warning, do the following: 1) Pas
s option use_label_encoder=False when constructing XGBClassifier object; and 2) Encode your labels (y) as integers starting wit
h 0, i.e. 0, 1, 2, ..., [num_class - 1].
  warnings.warn(label_encoder_deprecation_msg, UserWarning)

[23:48:14] WARNING: C:/Users/Administrator/workspace/xgboost-win64_release_1.4.0/src/learner.cc:1095: Starting in XGBoost 1.3.
0, the default evaluation metric used with the objective 'binary:logistic' was changed from 'error' to 'logloss'. Explicitly se
t eval_metric if you'd like to restore the old behavior.
[[56927    15]
 [    1 56783]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56942
           1       1.00      1.00      1.00     56784

    accuracy                           1.00    113726
   macro avg       1.00      1.00      1.00    113726
weighted avg       1.00      1.00      1.00    113726

Out[16]: 0.9998593109755025
```

Рисунок 2 – Алгоритм xgboost

На рисунке 2 представлены значения точности модели искусственного интеллекта на основе xgboost. Обобщенная точность на тестовых данных при отложенной тестовой выборке в соотношение 80/20 по методу hold – out составила 0,99859.

Теперь сравним алгоритм и его результаты при тех же начальных условиях с логистической регрессией (рисунок 3), которая успешно применяется при решении поставленной задачи.

```
Ввод [17]: prediction(LogisticRegression(), X_train, X_test, y_train, y_test)

[[55843  1099]
 [ 2063 54721]]
              precision    recall  f1-score   support

           0       0.96      0.98      0.97     56942
           1       0.98      0.96      0.97     56784

    accuracy                           0.97    113726
   macro avg       0.97      0.97      0.97    113726
weighted avg       0.97      0.97      0.97    113726

Out[17]: 0.9721963315336862
```

Рисунок 3 – Результаты точности логистической регрессии

Результирующее значение точности составляет 0,97220, что существенно меньше практической точности алгоритма xgboost.

Главным фактором практической точности является не модульная разница итоговых значений (равная в данной задаче 0,99859 – 0,97220 = 0,02659), а прикладное значение критерия, так как эта точность характеризует использование первоначального, «настоящего» набора исходных данных.

Например, если подвергнуть начальные данные методике увеличения числа примеров миноритарного класса (Synthetic Minority Oversampling Technique – SMOTE) [7, 8], то с её помощью точность логистической регрессии позволит превзойти алгоритм xgboost (рисунок 4).

```
Ввод [19]: prediction(LogisticRegression(),X_train,X_test, y_train, y_test)

[[56929    13]
 [    0 56784]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00     56942
           1       1.00      1.00      1.00     56784

    accuracy                           1.00    113726
   macro avg       1.00      1.00      1.00    113726
weighted avg       1.00      1.00      1.00    113726


Out[19]: 0.9998856901675958
```

Рисунок 4 – Логистическая регрессия на видоизмененных данных

Как следует из рисунка 4, точность логистической регрессии заметно повышается на «новом» наборе данных. Эту точность нельзя рассматривать как основополагающую, так как данные были подвергнуты изменению и теперь уже не соответствуют в полной мере действительным показателям [8].

Подводя итог, необходимо отметить, что алгоритм xgboost по своей структуре отлично подходит для решения поставленной задачи, что было подтверждено его практической эффективностью. По этой причине при выборе алгоритма искусственного интеллекта следует больше внимания уделять структурной соответственности, чем внешним факторам задачи (например, таким как несбалансированность классов и сложность разделения выборок)

Список литературы:

1. Информационная безопасность в банковской сфере [Электронный ресурс]. – Режим доступа: https://arinteg.ru/articles/informatsionnaya-bezopasnost-bankov-26722.html#:~:text=%D0%98%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F%20%D0%B1%D0%B5%D0%B7%D0%BE%D0%BF%D0%B0%D1%81%D0%BD%D0%BE%D1%81%D1%82%D1%8C%20%D0%B1%D0%B0%D0%BD%D0%BA%D0%B0%20%E2%80%93%20%D1%8D%D1%82%D0%BE%20%D1%81%D0%BE%D1%81%D1%82%D0%BE%D1%8F%D0%BD%D0%B8%D0%B5%20%D0%B7%D0%B0%D1%89%D0%B8%D1%89%D0%B5%D0%BD%D0%BD%D0%BE%D1%81%D1%82%D0%B8%20%D0%B2%D1%81%D0%B5%D1%85%20%D0%B5%D0%B3%D0%BE%20%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D1%85%20%D0%B0%D0%BA%D1%82%D0%B8%D0%B2%D0%BE%D0%B2.&text=%E2%80%93%20%D1%80%D0%B8%D1%81%D0%BA%20%D1%80%D0%B0%D1%81%D0%BF%D1%80%D0%BE%D1%81%D1%82%D1%80%D0%B0%D0%BD%D0%B5%D0%BD%D0%B8%D1%8F%20%D0%B2%D0%BE%20%D0%B2%D0%BD%D0%B5%D1%88%D0%BD%D0%B5%D0%B9%20%D1%81%D1%80%D0%B5%D0%B4%D0%B5%20%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%B8%2C%20%D1%83%D0%B3%D1%80%D0%BE%D0%B6%D0%B0%D1%8E%D1%89%D0%B5%D0%B9%20%D1%80%D0%B5%D0%BF%D1%83%D1%82%D0%B0%D1%86%D0%B8%D0%B8%20%D0%B1%D0%B0%D0%BD%D0%BA%D0%B0.&text=%E2%80%93%20%D0%BF%D1%80%D0%B8%D1%80%D0%BE%D0%B4%D0%BD%D1%8B%D0%B5%20%D0%B8%20%D1%82%D0%B5%D1%85%D0%BD%D0%BE%D0%B3%D0%B5%D0%BD%D0%BD%D1%8B%D0%B5%20%D0%BA%D0%B0%D1%82%D0%B0%D1%81%D1%82%D1%80%D0%BE%D1%84%D1%8B%2C%20%D0%BD%D0%B0%D1%80%D1%83%D1%88%D0%B0%D1%8E%D1%89%D0%B8%D0%B5%20%D0%BD%D0%BE%D1%80%D0%BC%D0%B0%D0%BB%D1%8C%D0%BD%D1%8B%D0%B9%20%D1%80%D0%B5%D0%B6%D0%B8%D0%BC%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D1%8B%20%D0%B8%D0%BD%D1%84%D0%BE%D1%80%D0%BC%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D1%8B%D1%85%20%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC

2. Machine Learning Group [Электронный ресурс]. – Режим доступа: https://mlg.ulb.ac.be/wordpress/
3. Serge Kruk. Practical Python Al Projects. – Apress. 2018. – 271 с.

4. Samir Madhavan. Mastering Python for Data Science – Packt Publishing. 2018. – 276 с.
5. Yves Hilpisch. Derivatives Analytics with Python. – Willey Press. 2015. – 347 с.
6. Ivan Idris. Python Data Analysis. – Packt Publishing. 2014. – 319 с.
7. Wiley Brand. Python for Data Science for Dummies – John Willey & Sonc. 2015. – 407 с.
8. Уэс Маккинни. Python и анализ данных. – O'Reilly Media. 2015. – 466 с.