Найденов Илья
05.04.2021

1 Реализация задачи кредитного скоринга. Описание основных этапов.

Оборудование: `python 3.8`, `pandas 1.1.3`, `scikit-learn 0.24.1`, `numpy 1.19.2`.
Код с пояснениями содержится в файле `credit_scoring.py`.

1.1 Предварительный анализ данных

Шаги приводятся в хронологическом порядке, после считывания таблицы в pandas.DataFrame:

- Удаление дублирующихся строк и строк, содержащих пустые значения, из датафрейма;
- Исследование колонки `order_id`. Установлено, что одному значению `order_id` может соответствовать 2 строки.
  Лишние строки удалены, так как большинство параметров у копий совпадает.
- Исследование колонки `client_id`. Установлено, что один клиент может подать заявку на несколько кредитов. Не противоречит здравому смыслу.
- Исследование колонки `date`.
  Принято решение не включать дату в матрицу признаков, потому что временной интервал в 3 месяца в 2017 году неактуален на сегодняшний день.
- Исследование колонки `region`. Составлен словарь с количеством включений в массив каждого региона.
  Принято решение преобразовать признак в бинарный („0“ – свой регион, „не 0“ – чужой).
- Исследование колонки `age`.
  Принято решение отбросить кредитуемых старше 85 лет.
- Совместное исследование `month_income, expert`, `closed_cred_sum`.
  Принято решение удалить данные, где безработным клиентам без кредитной истории присвоено $expert = 1$.
- Исследование колонки `active_cred_day_overdue`.
  Принято решение устранить пики в данных с помощью 0.999-квантиля. Этим пикам соответствуют большие значения суммарной просрочки в днях.
- Исследование колонки `expert`. Результаты показывают неравномощность классов (ответ „1“ преобладает, соотношение 1 к 0 примерно 70/30). Учтено в выборе метрики оценивания модели.

1.2 Фильтрация

В ходе предварительного анализа было принято решение не включать в обучающую выборку следующие данные:

- Исключаются кредитованные безработные без истории;
- Исключаются пожилые люди (возраст больше 85);
- Устраняются выбросы данных в колонке, содержащей суммарную просрочку выплат в днях;
- Удаляются строки, в которых сумма просроченных выплат превышает сумму активных кредитов на данный момент. Такие показатели не имеют физического смысла.

1.3 Преобразование переменных

В ходе преобразования переменных каждая из колонок была шкалирована, интервал возможных значений приведен к [0,1], либо к [-1, 1].

- Переменные, подвергшиеся шкалированию по min/max: `active_cred_day_overdue, month_income, closed_creds, active_cred_sum, active_cred_max_overdue, age`;
- Переменная `gender` не подвергалась преобразованиям;
- Переменная `region` преобразована в бинарный признак (см. 1.1). Свой регион – регион, в котором находится банк.
- Переменная `closed_cred_sum` преобразована по формуле:

$$\text{closed\_cred\_sum} = \frac{\text{closed\_cred\_sum}}{\text{closed\_creds}},$$

  учитывая количество отданных кредитов. Таким образом, получаем что-то вроде средней суммы выплачиваемого кредита. Данные впоследствии шкалируются по min/max.

- Переменные `month_income, loan_cost, first_days` преобразуются в переменную `credit_potential`, смысл которой – отразить кредитный потенциал клиента. Рассчитывается по формуле:

$$\text{credit\_potential} = \text{month\_income} - \text{loan\_cost} \cdot \frac{30}{\text{first\_days}},$$

  после чего шкалируется по abs-max в силу того, что может принимать отрицательные значения. Переменная – разность между месячным доходом и суммой выплат, которую в среднем клиент отдает за 1 месяц.

- Переменные `active_cred_sum_overdue` и `active_cred_sum` преобразуются в переменную `reliability`, смысл которой – оценить количественную долю выплат, отданную клиентом позже срока. Формула для подсчета:

$$\text{reliability} = 1 - \frac{\text{active\_cred\_sum\_overdue}}{\text{active\_cred\_sum}}$$

Для клиентов, у которых нет кредитной истории, надежность по умолчанию принимает значение своего матожидания от выборки клиентов с имеющимися активными кредитами:

$$\text{mean\_reliability} = \text{E(reliability)},\ \text{active\_cred\_sum} \neq 0.$$

Переменная не нуждается в шкалировании, поскольку отображает процентное соотношение.

Впоследствии была составлена корреляционная матрица признаков. Признак `month_income` удален, поскольку сильно ($r^2 > 0.9$) коррелировал с `credit_potential`.

1.4 Моделирование

Было проведено сравнительное исследование эффективности трех предлагаемых моделей (параметры подобраны эмпирически с целью максимизации *auc* и обеспечения вычислительной емкости).
Выборка предварительно разделяется на обучающую и тестовую методом `train_test_split` случайным образом. Эффективность алгоритма определяется на тестовой выборке.

1. Линейный *SGD-классификатор* с логарифмической функцией потерь. Максимальное количество итераций – 20.

2. Ансамбль решающих деревьев – *случайный лес*, состоящий из 200 деревьев с максимальной глубиной 5.

3. Ансамбль решающих деревьев, метод *градиентного бустинга*. Параметры идентичны п.2.

1.5 Оценка качества модели; ROC, AUC

Определим предварительно матрицу ошибок (confusion matrix):

$$conf = \begin{pmatrix} TP & FP \\ FN & TN \end{pmatrix},$$

где TP – количество верных положительных ответов, FP – количество ложных срабатываний, FN – количество ложных пропусков, TN – количество верных отрицательных ответов.
Построение roc-кривой производится следующим образом:

1. Вектор предсказанных вероятностей принадлежности объекта к тому или иному классу, полученный после применения обученной модели, сортируется по убыванию. В идеальном случае вектор правильных ответов, содержащий метки класса (0 или 1), сортируется таким же образом, по убыванию. Однако в общем случае распределение нулей и единиц в отсортированном по вероятности массиве более сложное.

2. Возьмем область на координатной плоскости $(x, y) \in [0, 1] \times [0, 1]$. Разделим единичный интервал оси абсцисс на m равноудаленных участков, ординат аналогично на n участков. m – число объектов с меткой „1“, n – число объектов с меткой „0“. При этом m, n берутся из вектора правильных ответов, а не из предсказываемого моделью.

3. Построение кривой происходит, если начать с точки (0,0), итерироваться по массиву целевой переменной и двигаться:

   - Вверх на 1/n на каждой итерации, если метка класса в данной строке – 1;
   - Вправо на 1/m на каждой итерации, если метка класса в данной строке – 0.

Физический смысл: сумма под кривой ошибок (AUC) есть мера того, насколько отсортированное распределение меток класса tar_i соответствует отсортированным значениям вероятностей p_i, указывающих на метки класса. Чем меньше площадь, тем более вразнобой расположены нули и единицы, тогда как вероятности, предсказанные алгоритмом, идут строго в порядке убывания. Соответственно, тем меньше здравого смысла в работе такого алгоритма.

1.6 Результаты

Для алгоритма, обучающегося решению задачи кредитного скоринга, была использована база данных клиентов, состоящая из 18 колонок и 50000 строк.

В ходе обработки данных, их анализа и фильтрации, получили число строк, наиболее информативных и пригодных для включения в матрицу признаков. Было использовано 10 колонок, при этом в обучающую выборку вошла 39561 строка, что составляет чуть меньше 80% исходных данных.

Было проведено сравнительное исследование трех различных моделей (одной линейной и двух ансамблей решающих деревьев). Качество классификации измерялось величиной площади под кривой ошибок (AUC), подсчитываемой методом `roc_auc_score`. Ниже представлен график, изображающий три кривые ошибок, соответствующие каждой из моделей.

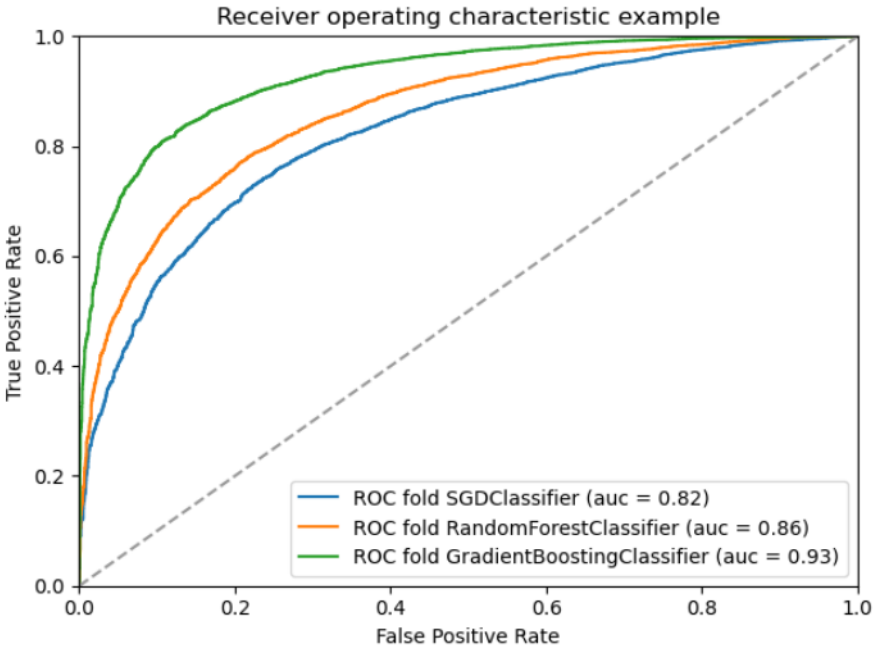

График 1: Построение ROC для предсказаний каждой из исследуемых моделей.

Величины AUC:

$$AUC\_Gradient\_Boosting = 0.927$$

$$AUC\_Random\_Forest = 0.863$$

$$AUC\_SGD\_log\_classifier = 0.825$$

Исходя из полученных результатов, ансамблевые методы, объединяющие решающие деревья, обладают лучшим качеством классификации по отношению, по крайней мере, к линейному методу градиентного спуска.

Наиболее качественно тестовую выборку классифицирует алгоритм градиентного бустинга, что соответствует данным в открытом доступе.

Приведу ниже результаты работы алгоритма на обучающей выборке, чтобы убедиться в отсутствии переобучения:

$$AUC\_Gradient\_Boosting = 0.961$$

$$AUC\_Random\_Forest = 0.868$$

$$AUC\_SGD\_log\_classifier = 0.831$$

AUC-величины на обучающей и тестовой выборках различаются на несколько процентов, что свидетельствует о низкой вероятности переобучения.