MAE = 0,056

ИИ. Кредитная индукция.

Столбцы:

Итак, начнём с логики расставления столбцов. Столбец approve_dttm исчерпывающе вычисляется через approve_flg. Мало пустых полей - мало нулей в столбце H. Зависимость видна. По такой же логике коррелированы util_dttm и util_flg. Speciality **не нуждается в обьяснении** (далее ННВО). Semester_cost_amt проставлен из базовой логики и знаний о стоимости высшего образования за семестр 😊. Pid ННВО. Score был выставлен, опираясь на то что это число от 0 до 1. Semester_cnt получен опираясь на факт, что в столбце J числа кореллируются с суммой кредита: больше семестров учиться - на большую сумму взят кредит. Short_nm ННВО. Initial_approved_amt исчерпывающе вычисляется через reject_reason (где 0, там есть причина отказа). Reject_reason ННВО. Gender_cd определяется по буквам M – male, F – female. Initial_term вычисляется по логике больше семестров учиться - больше срок кредита. Name ННВО. Marketing_flag был выставлен последним (метод исключения). education_level_code ННВО. Age соответствует среднему возрасту студентов ВУЗов. Subside_rate приближено к трём: ставке субсидирования образовательных кредитов в России. Create_dttm - самая ранняя дата в таблице, следовательно это дата создания заявки.

Изучение библиотек и моделей:

Честно говоря, о python представление я имею весьма обобщённое, а о pandas узнал только в начале турнира. Начал я с изучения библиотек numpy, pandas, scikit-learn и моделей машинного обучения. Наиболее продуктивной в плане рассчёта Mean Absolute Error оказалась линейная регрессия. Для начала я отбросил все ненужные значения, разметил и обработал данные, стандартизировал их и “скормил” на обучение. Также я всячески менял коэффиценты для достижения

наименьшего mae. На тестовой выборке мой mae был равен 0.050. Весь код можно найти в репозитории: [imnotadatascientist/ at master · g1zman/imnotadatascientist] (конкретно о ступенях, связанных с машинным обучением в файле “ml part.py”. Все линейные модели давали примерно одинаковый mae, поэтому я решил остановиться на стандартной линейной регрессии, не прибегая к бустерам и нелинейным моделям (за сложностью “вычленения” из них итоговой формулы”). Параметры, которые я взял:

```
x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size= 0.01, random_state=42)
x =
df.drop(columns=["score","pid","util_dttm","create_dttm","
name","semester_cost_amt","util_flg","subside_rate","semes
ter_cnt","initial_approved_amt","approve_dtm"])
numfigures = ["approve_flg","age"]
catfigures =
["utm_source","short_nm","gender_cd","education_level_code
","speciality","initial_term"]
preprocessor = ColumnTransformer(
    transformers=[
        ("num", StandardScaler(), numfigures),
        ("cat", OneHotEncoder(), catfigures)
    ])
pipeline = Pipeline(steps=[("preprocessor", preprocessor)
,("model", model)])
pipeline.fit(x_train,y_train)
ypred = pipeline.predict(x_test)
mae = mean_absolute_error(y_test, ypred)
print("mae =", mae)
```

Создание формулы:

Создавая формулу, я решил попросту эмулировать линейную регрессию с небольшими изменениями. Так как данные реальные, а не тестовые, погрешность в mae составила +0,006. Код есть в формула.txt, получить его было несложно. Если вкратце, то можно разделить его на этапы: **Загрузка и подготовка данных - Обработка**

числовых признаков - Обработка категориальных признаков - Объединение данных - Разделение на обучающую и тестовую выборки - Вычисление наилучших коэффицентов (не ручное) - Предсказание значений - Оценка качества модели (mae).

Формула получилась объёмная, но результативная. Погрешность небольшая, а значит работает она хорошо.

- Работа выполнена: *Гузюкиным Матвеем*.
- *Mean Absolute Error: 0.056*
- *Методы выполнения: ML, python, pandas, linear algebra*
- *Подробнее:* g1zman/imnotadatascientist