GAN-Augmented Startup Success Predictor

Хузин Эльдар Русланович

Постановка задачи

- **Сухой остаток:** бинарная классификация стартапов на успешные (получившие следующий раунд финансирования, IPO или acquisition) и неуспешные (закрытые или не получившие финансирования) -- в точности, предсказание данной характеристики.
- **Решаемая проблема:** дисбаланс классов в данных (только 3-7% стартапов успешны) -- на таких данных сложно обучать модели.
- **Основа решения:** обучение и применение Wasserstein GAN для генерации синтетических данных -- мотивированно этой статьёй.
- **Ценность:** система может использоваться венчурными фондами для первичного скрининга стартапов, снижая время на due diligence и улучшая качество инвестиционных решений.

Формат входных и выходных данных

- **Input:** вектор признаков стартапа
  - Post запрос из словарей формата `{ features: { feature_i: float }, categorical: { category_i: int } }` - т.е. отдельно real features и categorical features
- **Output:** response в формате словаря `{"success": bool, "prob": float}`
- **Протокол:** http, будет использоваться сервер на fastapi, обрабатывающий соответствующий post запрос.
- ***Замечание:*** *названия могут чуть меняться.*

Метрики

- **Основные метрики:**
  - AUROC (Area Under ROC Curve): Ожидаемое значение > 0.80. Основная метрика, инвариантная к threshold и учитывающая дисбаланс классов.
  - AUPRC (Area Under Precision-Recall Curve): Ожидаемое значение > 0.45. Более информативна при сильном дисбалансе классов (baseline = доля положительного класса ≈ 5%).
  - F1-score: Ожидаемое значение > 0.55. Гармоническое среднее precision и recall.
- Дополнительные метрики:

- Precision@k: Точность в топ-k предсказаниях (k=100, 500). Имитирует реальный сценарий VC — выбор лучших стартапов из пула.
- Recall миноритарного класса: Ожидаемое значение > 0.70. Важно не пропустить потенциально успешные стартапы.

Ожидаю согласно статье: модели на данных Crunchbase достигают accuracy 82-85% и AUROC > 0.80 при правильной работе с class imbalance.

Валидация и тест

- Temporal Split:
  - Train: стартапы до 2015 года,
  - Validation: 2015-2017,
  - Test: 2018+.
- Stratified K-Fold: 5 фолдов с сохранением пропорции классов для оценки стабильности модели.
- Для воспроизводимости:
  - Фиксированный random_seed=30
  - Версионирование через DVC
  - Логирование экспериментов в MLFlow
  - UV для версионирования библиотек (uv.lock)
- ***Замечание:*** *GAN обучается* ***только*** *на train set. Синтетические данные генерируются для train set и не используются в validation/test для честной оценки -- в точности, как в статье.*

Датасеты

- **Kaggle:** Startup Investments
  - Размер: ~50+ MB (несколько связанных CSV файлов)
  - Количество записей: ~54,000 компаний
  - Есть дисбаланс: большинство компаний имеют статус operating, что создаёт соотношение примерно 90-95% негативных примеров vs 5-10% позитивных — идеальный сценарий для применения GAN-аугментации.
- **Проблемы:**
  - Missing values
  - Feature engineering
  - Просто немного устаревший датасет.
  - Влияние периода времени на успешность стартапа.

Моделирование

Бейзлайн

- Обученный MLP по данным категориальным признакам -- т.е. когда нет исправления имбаланса классов
    - **Замечание:** вероятно попробую поиграться с моделями Яндекса типа TabM
- Дополнительно совсем baseline: Gradient Boosting (XGBoost/CatBoost) + SMOTE как традиционное для классификации на табличных данных.

Основная модель

- Двухэтапный pipeline:
    - Этап 1: WGAN-GP для генерации синтетических данных
        - Wasserstein GAN with Gradient Penalty
        - Generator: Linear(latent_dim=100 → 256) → BatchNorm → LeakyReLU → Linear(256 → 128) → BatchNorm → LeakyReLU → Linear(128 → num_features) → Tanh
        - Critic (Discriminator): Linear(num_features → 128) → LeakyReLU → Dropout(0.3) → Linear(128 → 64) → LeakyReLU → Linear(64 → 1)
        - Loss: Wasserstein loss
    - Этап 2: Classifier
        - Архитектура: MLP [num_features → 256 → 128 → 64 → 1] с BatchNorm, LeakyReLU, Dropout…
        - Loss: Binary Cross-Entropy
        - Optimizer: Adam с регуляризацией

Внедрение

RESTapi сервис на FastAPI, завёрнутый в docker.