МИНОБРНАУКИ РОССИИ
САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ
ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА (ЛЕНИНА)

ОТЧЕТ

по задаче № 1 (Модуль 2)

по дисциплине «Интеллектуальный анализ больших данных»

ТЕМА: «ПРЕДСКАЗАНИЕ ВЫЖИВАЕМОСТИ ДЕТЕЙ(13 ДАТАСЕТ)»

Студент гр. 3314 _____ Кокорев С.С

Санкт-Петербург
2025

ЦЕЛЬ РАБОТЫ

Создание модели для предсказания риска смерти ребёнка на основе социально-демографических и медицинских данных.

ОПИСАНИЕ ДАННЫХ

Этот набор данных содержит информацию из Ежегодного обследования здоровья: Анкеты для женщин. Этот набор данных включает информацию о результатах беременностей (живорождение/мертворождение/аборт), истории родов, типе медицинского наблюдения при родах, деталях материнского здравоохранения (дородовый/родовый/послеродовый периоды), вакцинации детей, практике грудного вскармливания и использования добавок, случаях детских заболеваний, регистрации рождений и т.д. Также рассматриваются использование, источники и практика методов планирования семьи, детали будущего использования контрацептивов и неудовлетворенные потребности, осведомленность о ИППП/ЗППП, ВИЧ/СПИДе.
Всего в наборе данных 197 переменных/столбцов и 8 299 069 строк.

Данные были собраны в разных областях Индии, в которых условия жизни сильно отличаются:

- Уттаракханд (05)
- Раджастхан (08)
- Уттар-Прадеш (09)
- Бихар (10)
- Джаркханд (20)
- Одиша (21)
- Чхаттисгарх (22)
- Мадхья-Прадеш (23)
- Ассам (18)

Это официальных официальный комментарий к датасету. Но важнейший параметр (результат беременности) (живорождение/мертворождение/аборт) представляет собой подобную колонку:

outcome_pregnancy	Has the outcome of any pregnancy(s) resulted in live birth/still birth/abortion	Yes-1 No-2

Она несет информацию только о факте завершения беременности и неважно как. Т.е. из неё не получить информацию о том, родился ли ребёнок живим или мёртвым. Поэтому как тему этого проекта я выбрал «Предсказание выживаемости детей» уже рождённых.

ПРЕДОБРАБОТКА ДАННЫХ

Датасет содержит множество полезных и не очень параметров. И чтобы самостоятельно не выбирать какие будут наиболее информативными, я оставлю достаточно большое их количество. Я планирую использовать XGBoostClassifier, который может сам выбрать нужные ему параметры.

Для начала я объединил все файлы в один, т.к. вычислительные мощности Yandex DataSphere позволяли обрабатывать такой объём данных.

Очевидно бесполезные параметры я удалил вручную. Руководствуясь файлом *AHS_Woman_Data_Dictionary.xlsx*, который шёл в комплекте с датасетом, я отобрал относительно информативные колонки. Также я смотрел на то, какая доля данных в столбце была null. И если процент пропусков был слишком высок, колонка отбрасывалась.

Параметры, которые остались:

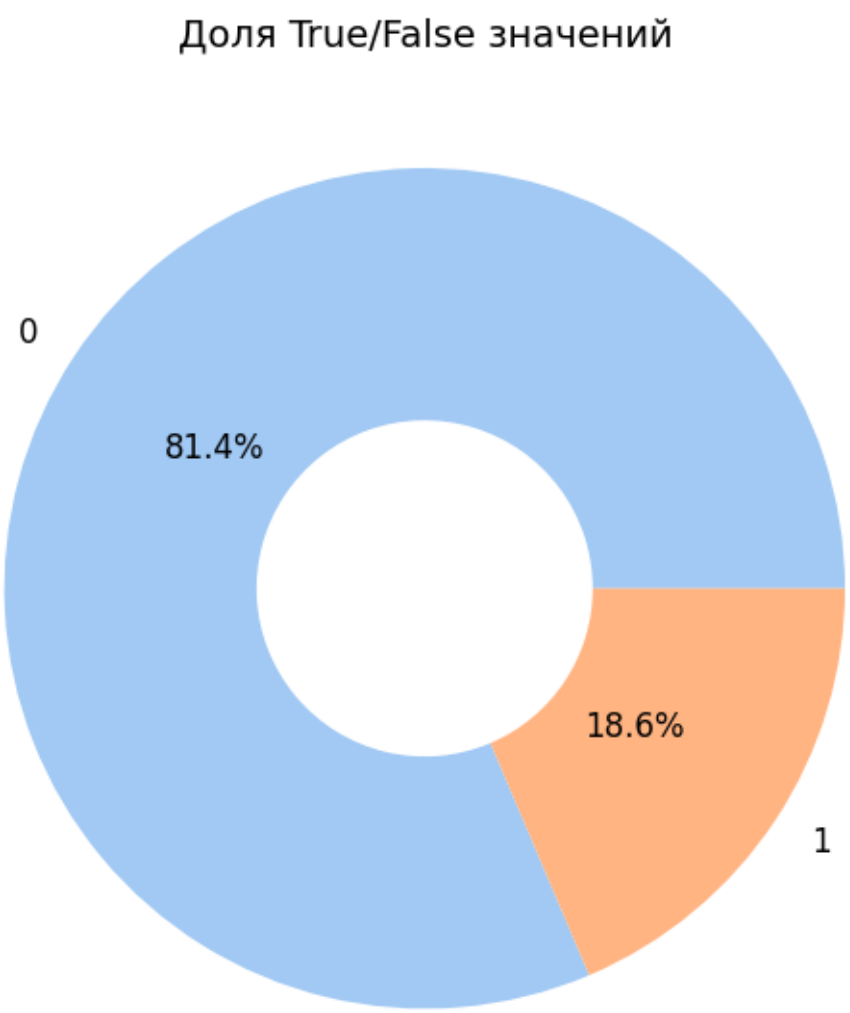
```
imp_cols = ['state', 'district', 'rural', 'stratum_code', 'age', 'marital_status',
   'born_alive_female', 'born_alive_male', 'born_alive_total',
   'mother_age_when_baby_was_born', 'outcome_pregnancy', 'is_tubectomy',
   'is_vasectomy', 'is_copper_t', 'is_pills_daily', 'is_piils_weekly',
   'is_emergency_contraceptive', 'is_condom', 'is_moder_methods',
   'is_contraceptive', 'is_periodic_abstinence', 'is_withdrawal',
   'is_amenorrahoea', 'is_other_traditional_method',
   'is_currently_pregnant', 'is_currently_menstruating',
   'when_you_bcome_mother_last_time', 'is_any_fp_methos_used',
   'fp_method_used', 'want_more_childern', 'anm_in_last_3_months',
   'during_pregnancy', 'during_lactation', 'aware_abt_rti',
   'aware_abt_hiv', 'aware_of_haf', 'aware_of_the_danger_signs',
   'religion', 'social_group_code', 'year_of_marriage', 'year_of_intr',
   'highest_qualification', 'occupation_status', 'disability_status',
   'injury_treatment_type', 'illness_type', 'treatment_source',
   'symptoms_pertaining_illness', 'diagnosed_for', 'chew', 'smoke',
   'alcohol', 'drinking_water_source', 'is_water_filter', 'toilet_used',
   'household_have_electricity', 'cooking_fuel', 'is_car',
   'ever_conceived', 'age_at_first_conception',
   'is_husband_living_with_you', 'counselled_for_menstrual_hyg',
   'aware_abt_danger_signs_new_born', 'iscoveredbyhealthscheme', 'wt',
   'as_binned', 'surviving_total', 'surviving_female', 'surviving_male', 'delivered_any_baby']
```

Целевую переменную я создал таким образом:

```
df['child_death_risk'] = np.where(df['born_alive_total'] > df['surviving_total'], 1, 0)
```

Если у женщины есть погибшие дети, то значение переменной 1, иначе 0.

Распределение данных неравномерно:



ОБРАБОТКА NULL-ЗНАЧЕНИЙ

Для обработки пропусков был написан такой код.

```
for col in imp_cols:
    null_percent = df[col].isna().mean()
    if null_percent < 0.5:
        if df[col].nunique() < 20:
            df[col] = df[col].fillna(df[col].mode()[0])
        else:
            df[col] = df[col].fillna(df[col].median())

    else:
        df[col] = df[col].fillna(-99)
```

Он определяет, насколько много null значений в каждой колонке. И если пропусков меньше половины, выполняется заполнение пропусков модой для категориальных параметров, и медианой для числовых. Если же пропусков больше 50%. Значение помечается как -99, чтобы модель поняла, что это пропуск.

РАБОТА С ВЫБРОСАМИ

Чтобы увидеть, в каких колонках находятся неестественные данных, я написал следующую функцию.

```
def data_analise(df):
    for col in df.columns:
        print(col)
        print(df[col].value_counts(), '\n')
```

Она выводит уникальные значения каждого параметра. Благодаря этому можно легко заметить ошибки или странные значения и удалить их.

```
def outliers_1(df):
    df = df[df['aware_of_the_danger_signs'] != 3]
    df = df[df['as_binned'] != 200806003340102.0]
    df = df[(df['want_more_childern'] != 8.0) & (df['want_more_childern'] != 0.0)]
    df = df[(df['stratum_code'] != 3) & (df['stratum_code'] != 5) & (df['stratum_code'] != 6)]
    df = df[(14 < df['age']) & (df['age'] <= 51)]
    df = df[(11 <= df['mother_age_when_baby_was_born']) & (df['mother_age_when_baby_was_born'] <= 49)]
    df = df[(11 <= df['age_at_first_conception']) & (df['age_at_first_conception'] <= 49)]
    df = df[df['is_currently_menstruating'] != 0]
    df = df[df['treatment_source'] != 12]
    df = df[df['smoke'] != 7]
    df = df[df['alcohol'] != 7]

    return df
```

После всех вышеперечисленных обработок осталось

6 912 332 строк из 8 299 069 (около 83.3%)
50 из 197 столбцов

СОЗДАНИЕ НОВЫХ ПАРАМЕТРОВ

Используя существующие колонки, можно создать новые, возможно более полезные для модели.

Например, я объединил 13 колонок о различных видах контрацепции в 2 (‘modern_contraception_score’, ‘traditional_contraception_score’)

или создал новую переменную «возраст выхода замуж», вместо «года свадьбы»

```
df['age_when_married'] = df['age'] - (df['year_of_intr'] - df['year_of_marriage'])
```

и т.д.

Такие переменные будут лучше интерпретироваться некоторыми моделями.

ВЫБОР МОДЕЛИ

Для данной задачи был выбран XGBoostClassifier по следующим причинам:

- **Распределенные вычисления**:
  Поддержка многопоточности (n_jobs=32) вычисления ведутся сразу на 32 ядрах процессора.
- **Оптимизация памяти**:
  Алгоритм tree_method="hist" (гистограммный подход) группирует данные в бины, снижая потребление памяти.
- **Работы с категориальными фичами**:
  Использование enable_categorical=True для работы с категориальными фичами без one-hot encoding (экономия памяти).
- **Градиентный бустинг**:
  Последовательное построение деревьев, где каждое новое дерево корректирует ошибки предыдущего. Это требует меньше ресурсов, чем Random Forest.
- **Регуляция**:
  Параметры gamma, reg_alpha, reg_lambda контролируют сложность модели.
- **Автоматический баланс классов**:
  Параметр scale_pos_weight корректирует вес положительного класса.

ПОДБОР ГИПЕРПАРАМЕТРОВ ДЛЯ МОДЕЛИ

Для начала я рассчитываю смещение классов, как отношение 0 к 1:

```
scale_pos_weight = (len(y_tr) - y_tr.sum()) / y_tr.sum()
```

Далее определяется функция **def objective(trial)**, которая обучает модель на основе ограниченного объёма данных и рассчитывает и возвращает её Average Precision Score.

Average Precision Score (AP) — метрика качества для бинарной классификации, особенно полезная при работе с несбалансированными данными.

Затем с помощью Optuna подбираются такие параметры, чтобы AP было максимальным. В итоге вышли такие значения:

```
xgb = XGBClassifier(
    n_estimators=1000,
    max_depth = 25,
    learning_rate = 0.05,
    subsample = 0.928,
    colsample_bytree =0.6,
    gamma = 1.01,
    reg_alpha = 0.65,
    reg_lambda = 1.43,
    early_stopping_rounds=20,
    random_state=42,
    eval_metric='aucpr', #eval_metric='logloss',
    objective="binary:logistic",
    enable_categorical=True,
    tree_method = "hist",
    n_jobs = 32,
    scale_pos_weight=scale_pos_weight,
    importance_type='gain'
)
xgb.fit(X_tr, y_tr, eval_set=[(X_val, y_val)])
```

ОЦЕНКА КАЧЕСТВА МОДЕЛИ

Для оценки качества была написана функция **analise(model)**, которая выводит данные в таком формате:

```
Accuracy: 0.8073
Death predicted percent: 20.7703
Правильность предсказания смертности: 0.4823
Процент найденных случаев смертности: 0.5406
Final ROC-AUC: 0.7043
              precision    recall  f1-score   support

           0       0.89      0.87      0.88    883026
           1       0.48      0.54      0.51    200827

    accuracy                           0.81   1083853
   macro avg       0.69      0.70      0.69   1083853
weighted avg       0.82      0.81      0.81   1083853
```

В этом исследовании разумно ориентироваться на recall для класса 1(смертность). Т.е. процент найденный случаев смертности, чтобы выявлять потенциальных матерей, дети которых рискуют погибнуть.

Обращать внимание на accuracy не стоит, потому что данные очень не сбалансированы.

На первый взгляд может показаться, что результат не очень хороший: модель выявляется только 54% процента смертей и правильность предсказания смертности 48%. Однако это связано со спецификой задачи: смертность ребёнка во многом зависит от генов, что не учтено и не могло быть учтено в датасете.

Несмотря на невысокие показатели модель может быть полезна для выявления матерей, чей уровень жизни настолько низок, что вероятность смерти их детей существенно завышена.

Если повысить scale_pos_weight, например, до 8, то можно получить такой результат:

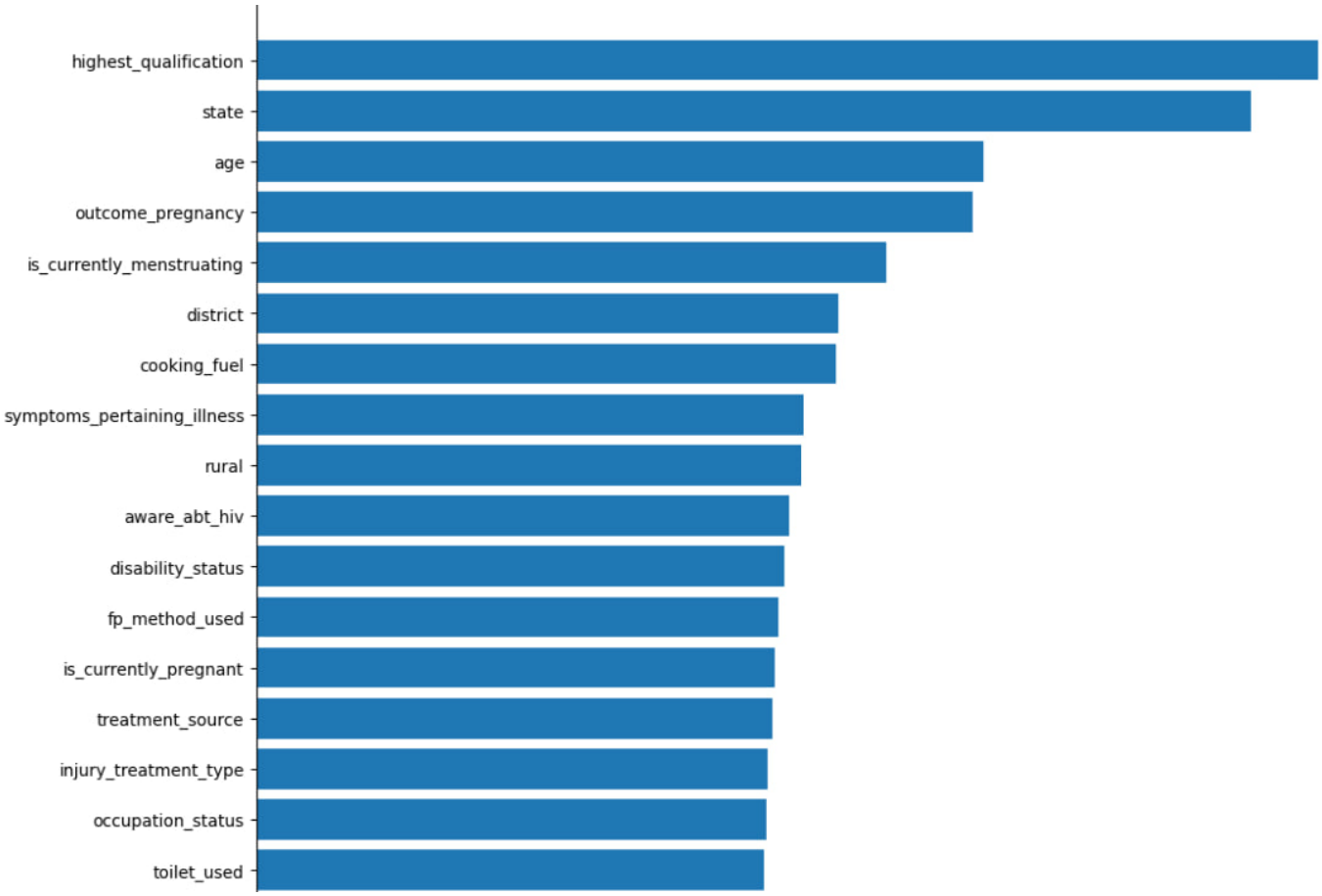
```
Accuracy: 0.7745
Death predicted percent: 26.6509
Правильность предсказания смертности: 0.4242
Процент найденных случаев смертности: 0.6108
Final ROC-AUC: 0.7113
              precision    recall  f1-score   support

           0       0.90      0.81      0.85    883235
           1       0.42      0.61      0.50    200618

    accuracy                           0.77   1083853
   macro avg       0.66      0.71      0.68   1083853
weighted avg       0.81      0.77      0.79   1083853
```

Видно, что recall для класса 1 повысился, а значить больше случаев смертности выявлено, однако precision(“правильность”) для класса 1 упала, что не очень хорошо, так как много женщин было классифицировано неверно. Но всё же, это тоже может быть полезно, если надо выявить группу риска.

Также эта модель способна предсказать вероятность смертности ребёнка для отдельно взятой женщины, если чуть поменять код.

НАИБОЛЕЕ ПОЛЕЗНЫЕ ПРИЗНАКИ ДЛЯ МОДЕЛИ

В ходе обучения, модель выбрала следующие признаки наиболее полезными для прогнозирования результата:



Как можно заметить самым важным, с отрывом, признаком являются:

Наличие высшего образования – что может характеризовать женщину как хорошо обеспеченную.

Возраст – ребёнок молодой женщины родится более здоровым, что логично

Область – в разных областях Индии разные условия жизни, что сказывается на здоровье матерей, следовательно и детей.

РЕСУРСЫ ЗАТРАЧЕННЫЕ НА ОБУЧЕНИЕ

Для анализа этого датасета были затрачены следующие временные ресурсы:

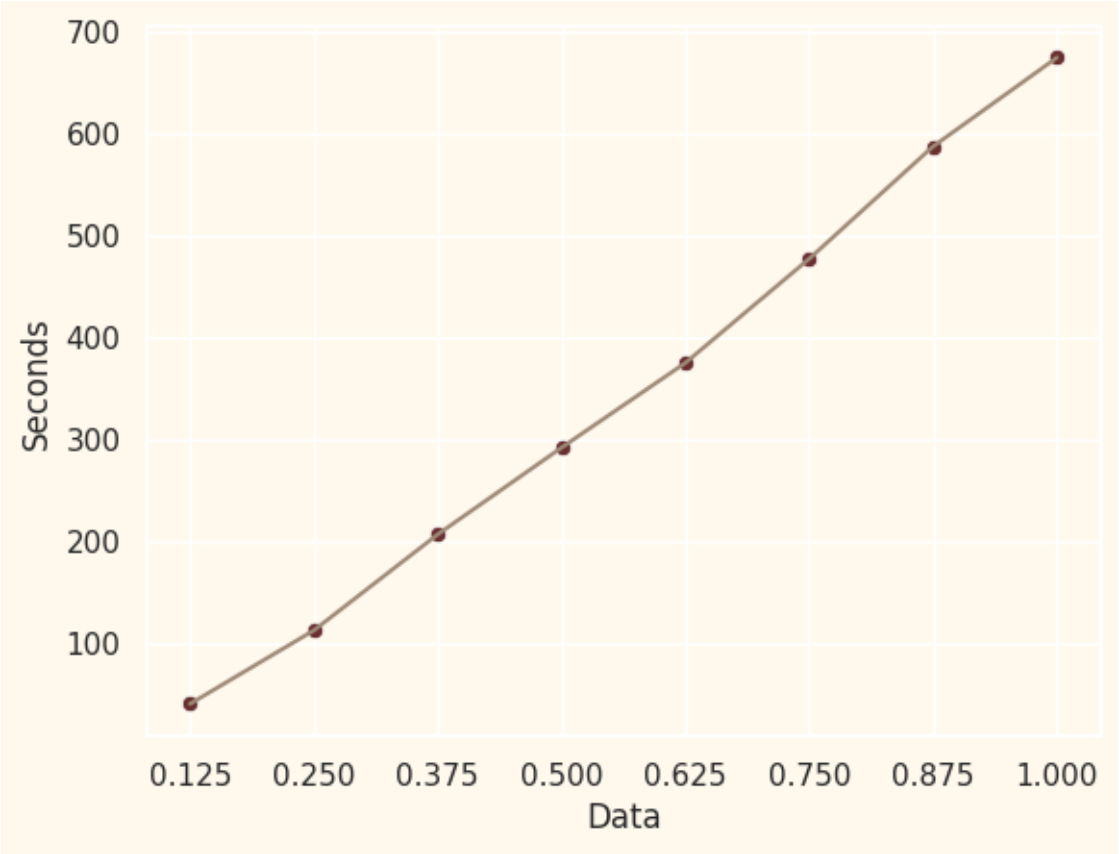
Считывание файлов	5m 51.06s
Работа с null-значениями и отбрасывание лишних колонок	3m 18.80s
Работа с выбросами	15.49s + 9m 15.48s
Feature engineering	40.07s
Деление выборки	4.85s
Подбор гиперпараметров	2h 44m 11s
Обучение модели	11m 14s
Оценка модели	11.89s

Потребление оперативной памяти достигало 128 ГБ.

Использовались следующие ВМ: *CPU 32 ядра; RAM 256 гб*

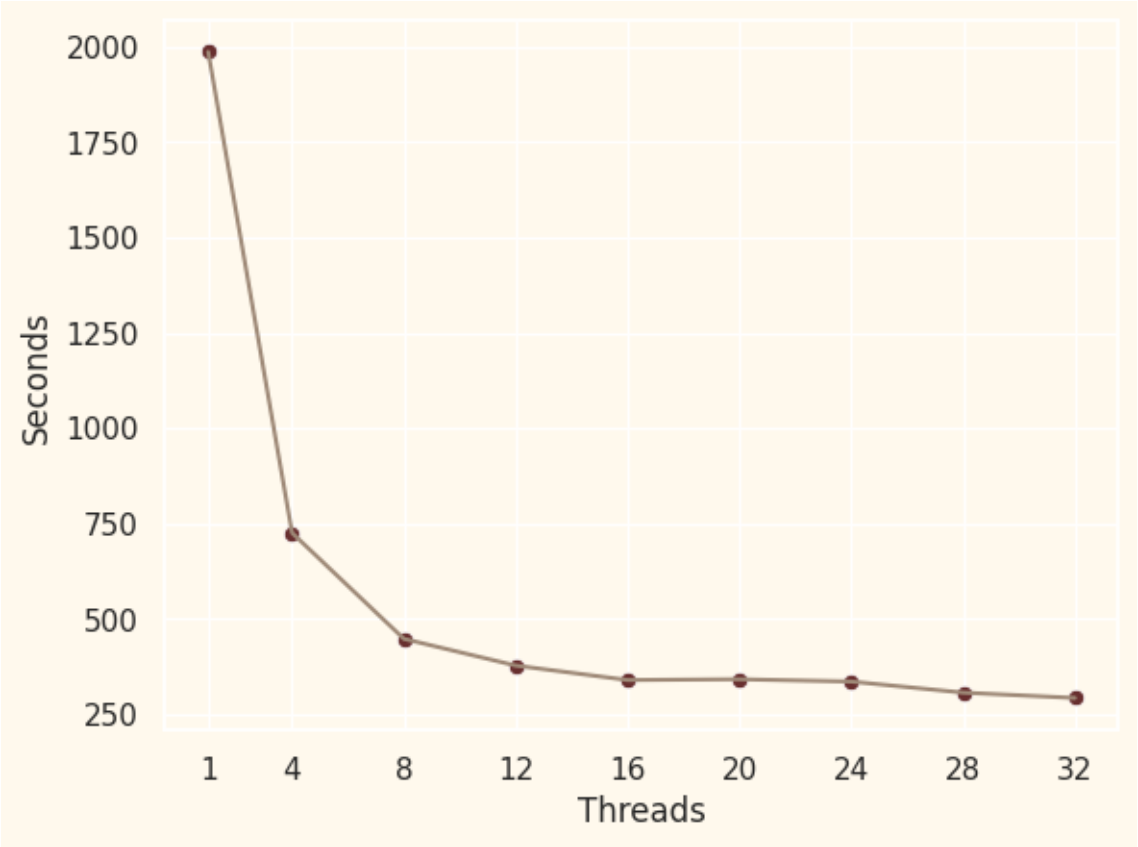
ТЕСТЫ ВРЕМЕНИ ОБУЧЕНИЯ МОДЕЛИ ПРИ РАЗНОМ ОБЪЁМЕ ДАННЫХ

На графике представлено время обучения модели (без подбора гиперпараметров) при вычислении на 32 ядрах. По оси X представлена часть от полной тренировочной выборки (5 419 268 объектов).



ТЕСТЫ ВРЕМЕНИ ОБУЧЕНИЯ МОДЕЛИ ПРИ РАЗНОМ КОЛИЧЕСТВЕ ВЫЧИСЛИТЕЛЕЙ

На графике представлено время обучения модели при разном количестве ядер процессора, задействованных при обучении модели при объёме данных 0.5 от тренировочной выборки. Количество задействованных ядер регулировалось параметром n_jobs в XGBoost.



КОД

Полный код можно найти по ссылке:

https://github.com/mole88/ChildSurvivalPredictor