Курсовая работа

Создание ML-решения для предсказания вероятности подключения услуги абонентом

14.02.2022

Модель

Разработанная модель включает в себя пайплайн, состоящий из трех этапов:

1. предобработка признаков
2. отбор наиболее важных признаков
3. непосредственное обучение на тренировочных данных

В качестве основного ML-решения был выбран классификатор **LGBMClassifier**.

```
lgbm_fs_pipe = make_pipeline(
    f_prep_pipe,
    SelectFromModel(LogisticRegression(penalty='l2', random_state=RANDOM_STATE, solver='sag'), threshold=1e-2),
    lgb.LGBMClassifier(is_unbalance=True, max_depth=10, learning_rate=0.1, random_state=RANDOM_STATE)
)

lgbm_fs_pipe.fit(X_train, y_train)
```

Предобработка и отбор признаков

Предобработка признаков включает в себя простой набор действий:

- стандартизацию числовых признаков
- OneHot-кодирование категориальных признаков

В данном случае не включен этап заполнения пропусков, так как предоставленные тренировочный и тестовый наборы данных пропусков не имеют. При необходимости возможно скорректировать пайплайн.

```
# OneHotEncoder(handle_unknown='ignore') - игнорируем значение,
# которого не было при обучении
f_prep_pipe = make_pipeline(
    ColumnSelector(columns=f_ok),
    FeatureUnion(transformer_list=[
        ('numeric_features', make_pipeline(
            ColumnSelector(f_numeric),
            StandardScaler()
        )),
        ('categorical_features', make_pipeline(
            ColumnSelector(f_categorical),
            OneHotEncoder(handle_unknown='ignore')
        ))
    ])
)
```

```
# Воспользуемся классом SelectFromModel для отбора значимых признаков
fs_pipe = make_pipeline(
    f_prep_pipe,
    SelectFromModel(LogisticRegression(penalty='l2', random_state=RANDOM_STATE, solver='sag'), threshold=1e-2),
)
```

```
# f_prep_pipe.fit(X_train)
# f_prep_pipe.transform(X_test).shape
# # (89664, 258)
```

```
# # Логистическая регрессия из SelectFromModel обнулила 246 признаков при пороге 1e-2.
# fs_pipe.fit(X_train, y_train)
# fs_pipe.transform(X_test).shape
# # (89664, 12)
```

```
# # Логистическая регрессия из SelectFromModel обнулила 73 признака при пороге 1e-3.
# fs_pipe.fit(X_train, y_train)
# fs_pipe.transform(X_test).shape
# # (89664, 185)
```

Отбор признаков происходит с помощью класса SelectFromModel, в котором селективную функцию выполняет модель LogisticRegression.

При пороге в **0.01** удалось сократить значительное количество признаков (**95%**), при этом оставить качество работы модели на приемлемом уровне (**f1_macro score: 0.61**).

Классификатор LGBMClassifier

Причины выбора классификатора:

- быстрота вычислений
- учет дисбаланса классов без дополнительной обработки данных
- лучший показатель по метрике **f1_score('macro')**

```
param_grid = {
    'lgbmclassifier__max_depth': [3, 5, 10],
    'lgbmclassifier__is_unbalance': [True, False],
    'lgbmclassifier__learning_rate': [0.1, 1.0]
}

lgbm_fs_gsc = run_grid_search(lgbm_fs_pipe, X_train, y_train, param_grid, kfold_cv)
```

```
Best f1_macro score: 0.74

Best parameters set found on development set:

{'lgbmclassifier__is_unbalance': True, 'lgbmclassifier__learning_rate': 0.1, 'lgbmclassifier__max_depth': 10}
```

Благодаря перебору некоторых параметров, удалось достигнуть качества работы модели на тренировочных данных — **0.74**, на отложенных — **0.73**.

```
y_pred_3 = model_report(X_train, y_train, X_test, lgbm_fs_pipe)
```

```
print(classification_report(y_test, y_pred_3 > 0.68))
```

	precision	recall	f1-score	support
0.0	1.00	0.86	0.93	83129
1.0	0.36	1.00	0.53	6535
accuracy			0.87	89664
macro avg	0.68	0.93	0.73	89664
weighted avg	0.95	0.87	0.90	89664

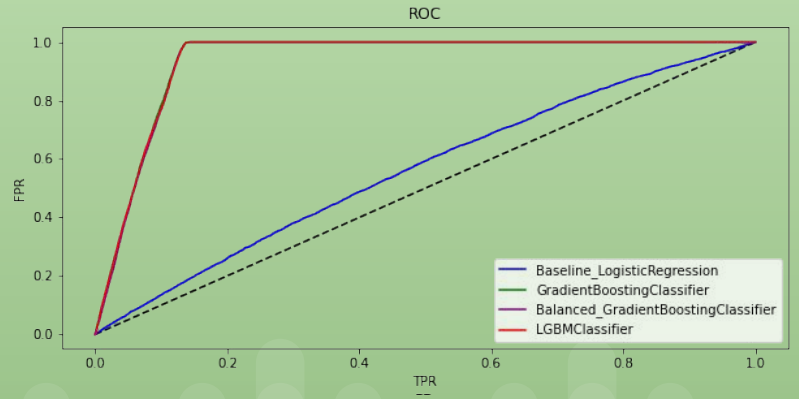
Эксперименты с моделями

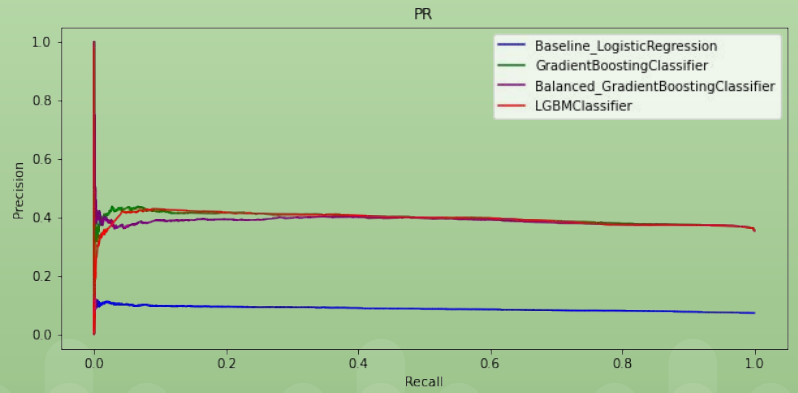
Комментарий	Model_name	F1_macro
Базовая модель	Baseline_LogisticRegression	**0.47**
Базовая модель + обработка признаков	Prep_features_LogisticRegression	**0.61**
Базовая модель + обработка и отбор признаков	Prep_f_SelectFromModel_LogisticRegression	**0.61**
Базовая модель + уменьшение размерности	Tsvd_LogisticRegression	**0.47**
Градиентный бустинг + обработка и отбор признаков	Prep_f_SelectFromModel_GradientBoostingClassifier	**0.72**
Градиентный бустинг + обработка и отбор признаков + балансировка классов	Balanced_Prep_f_SelectFromModel_GradientBoostingClassifier	**0.89**
Случайный лес + обработка и отбор признаков	Prep_f_SelectFromModel_RandomForestClassifier	**0.49**
LGBM + обработка и отбор признаков	Prep_f_SelectFromModel_LGBMClassifier	**0.74**

Сравнение моделей

Модели с наибольшей метрикой показали примерно одинаковое качество работы, согласно ROC и PR кривым.

Однако на отложенной выборке лучше отработали модели градиентного бустинга со сбалансированными классами и LGBM. Метрика **f1_score('macro')** установилась на уровне **0.73**.





Выбор порога для 1-го класса

В качестве обоснования порога для отнесения предсказания к 1-му классу предлагается грубая оценка прибыли при соотношении правильно и ошибочно предсказанных значений.

Таким образом, перебрав несколько вариантов порога, мы можем найти наиболее подходящий вариант соотношения предсказаний.

В нашем случае, наилучший порог — **0.68**.

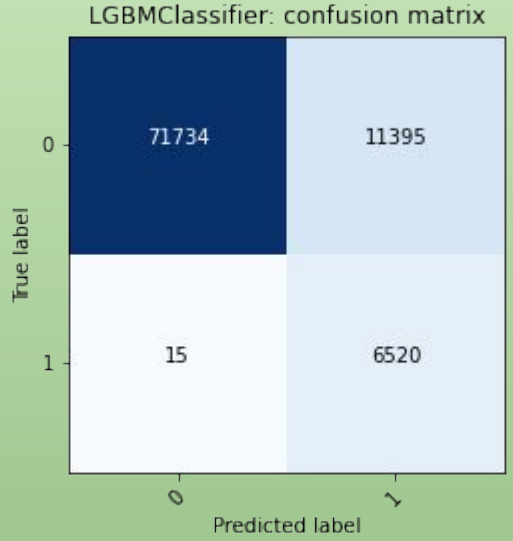
```
# Посчитаем выгоду на условных единицах
price_caller = 2        # затраты на обзвон 1-го абонента
price_service = 10      # стоимость подключаемой услуги

cost_call = (fp + tp)*price_caller
profit = tp*price_service
lost = fn*price_service
```

```
# Найдем оптимальный порог
choice_th(2, 10, 100)
```

```
Затраты на обзвон: 35830 руб.
Выручка: 65200 руб.
Упущенная выгода: 150 руб.
Прибыль: 29220 руб.

Максимальная прибыль: 29220
Порог: 0.6818181818181818
```



Спасибо за внимание!