Научный доклад на тему “Ошибки в разметках”

Зад. 8

Макарчук Алексей МФТИ ФПМИ 4курс Б05-903в

Введение (актуальность)

Проблема:

В задачах МО качество разметки играет основную роль в получении точных моделей, решающих поставленную задачу.

Проблемы в разметке обычно следующие:

1. Когда данные берутся из открытых источников вместе с готовой разметкой, невозможно гарантировать безошибочность меток.

2. Когда данные отдаются на разметку разметчикам на сторонних платформах, нужно ожидать погрешности из-за человеческого фактора.

Постановка задачи

Изучить влияние некачественных меток в разметке на качество классических моделей машинного обучения. Проанализировать результаты. Предложить варианты решения.

Для этого необходимо решить следующие задачи:

1. Исследовать модели классического МО, решающие задачу бинарной классификации на устойчивость к шуму. Построить графики для различных метрик.
2. Исследовать модели классического МО, решающие задачу бинарной классификации на устойчивость к размеру данных train-а. Построить графики для различных метрик.
3. Проанализировать результаты.
4. Сделать выводы.

Обзор существующих решений

Существует ряд классических способов избавиться от шума в разметке:

- При разметке человеком, использовать протокол разметки. Иметь второго эксперта. Контролировать коэффициент согласия
- Поиск аномалий и проверка на наличие null-values в датасете
- Использование робастных или ансамблевых методов обучения
- Контроль распределения таргетов

Используемый датасет

Датасет "MAGIC Gamma Telescope":

- 19 020 записей
- 10 числовых признаков
- target - событие сигнала гамма-кванта или фонового шума

Предварительный анализ датасета:

- проверка на наличие null-values
- построение гистограмм для всех столбцов-признаков, анализ распределений
- смысловой анализ столбцов-признаков

https://archive.ics.uci.edu/ml/machine-learning-databases/magic/

Пример данных датасета

```
"""
fLength: major axis of ellipse
fWidth: minor axis of ellipse
fSize: 10-log of sum of content of all pixels
fConc: ratio of sum of two highest pixels over fSize
fConc1: ratio of highest pixel over fSize
fAsym: distance from highest pixel to center, projected onto major axis
fM3Long: 3rd root of third moment along major axis
fM3Trans: 3rd root of third moment along minor axis
fAlpha: angle of major axis with vector to origin
fDist: distance from origin to center of ellipse
class: g (signal), h (background)
"""

dataset = pd.read_csv("magic04.data", header=None, names=column_names, delimiter=',')

print(dataset.head())
```

```
    fLength    fWidth   fSize   fConc  fConc1     fAsym  fM3Long  fM3Trans  \
0   28.7967   16.0021  2.6449  0.3918  0.1982   27.7004  22.0110   -8.2027
1   31.6036   11.7235  2.5185  0.5303  0.3773   26.2722  23.8238   -9.9574
2  162.0520  136.0310  4.0612  0.0374  0.0187  116.7410 -64.8580  -45.2160
3   23.8172    9.5728  2.3385  0.6147  0.3922   27.2107  -6.4633   -7.1513
4   75.1362   30.9205  3.1611  0.3168  0.1832   -5.5277  28.5525   21.8393

    fAlpha     fDist class
0  40.0920   81.8828     g
1   6.3609  205.2610     g
2  76.9600  256.7880     g
3  10.4490  116.7370     g
4   4.6480  356.4620     g
```

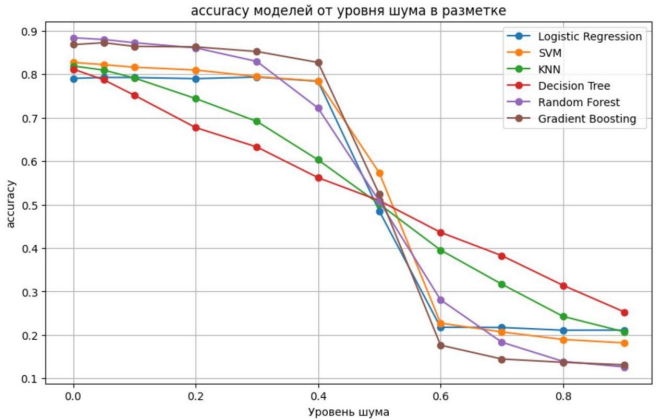
Рассматриваемые модели и метрики

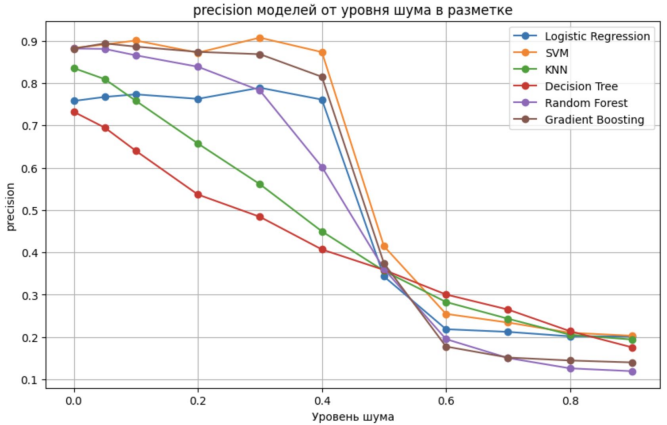
```
models = {
    "Logistic Regression": LogisticRegression(max_iter=1000, random_state=42),
    "SVM": SVC(random_state=42),
    "KNN": KNeighborsClassifier(),
    "Decision Tree": DecisionTreeClassifier(random_state=42),
    "Random Forest": RandomForestClassifier(random_state=42),
    "Gradient Boosting": GradientBoostingClassifier(n_estimators=100, max_depth=4, random_state=42)
}
```

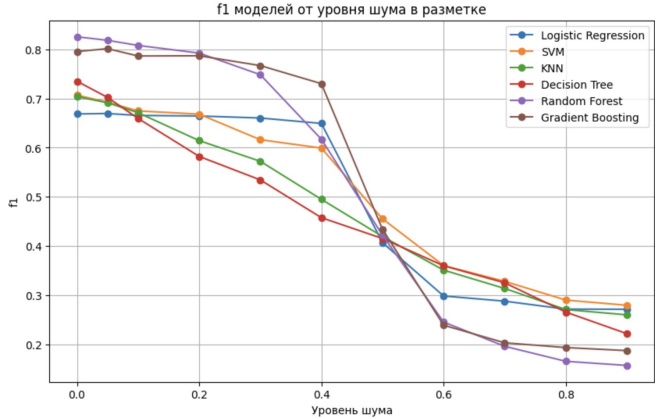
Метрики:

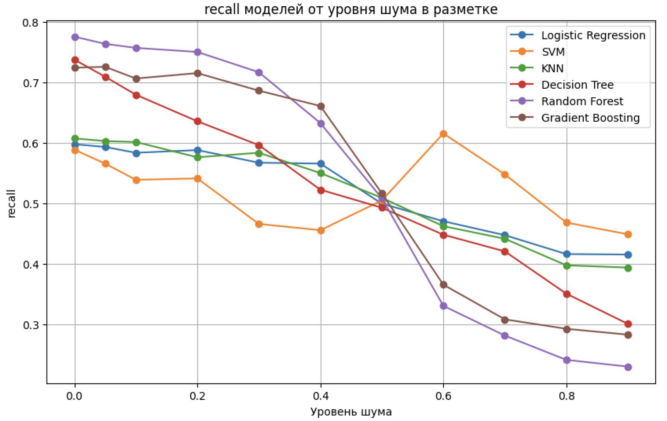
- accuracy
- precision
- recall
- f1-score

Графики зависимости метрика(шум)

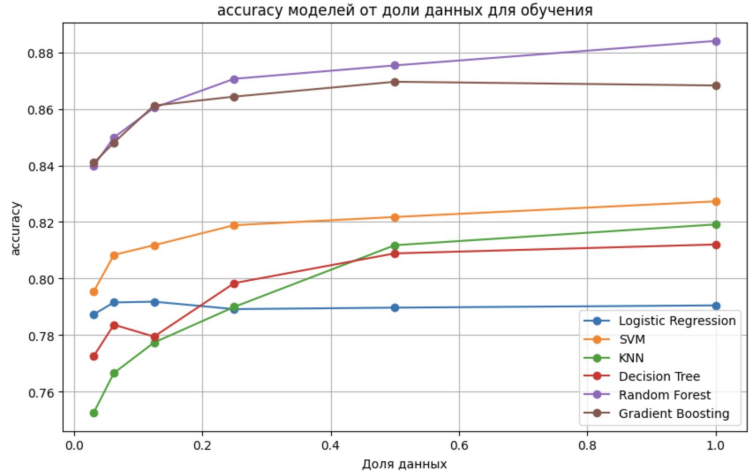


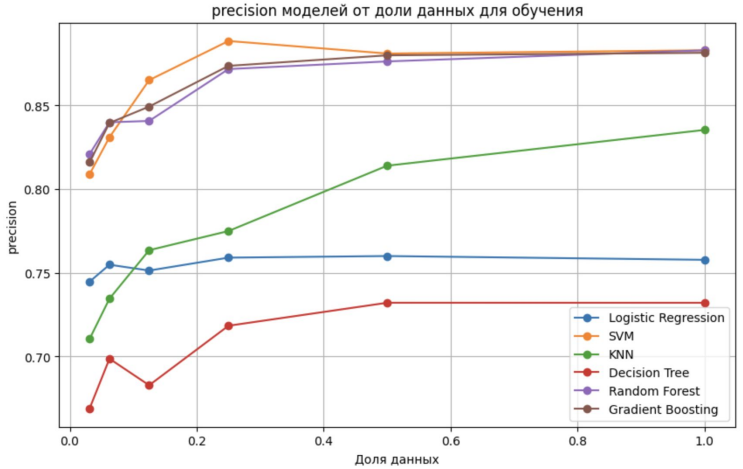


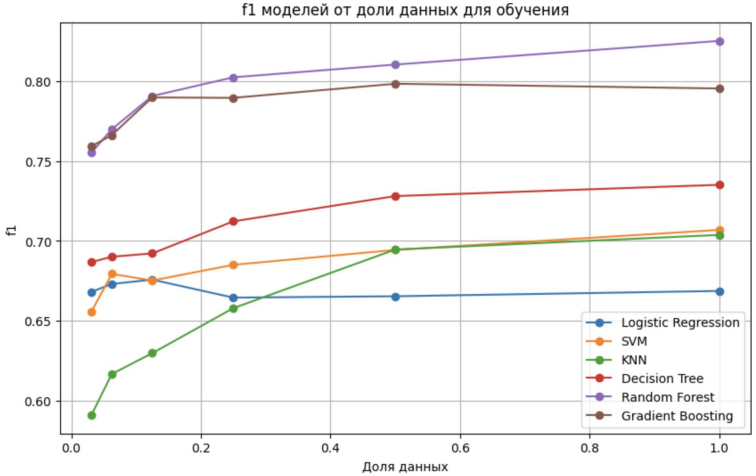


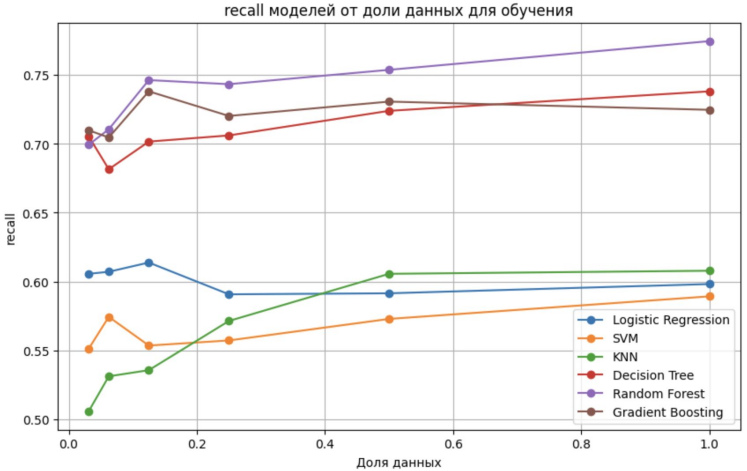


Графики зависимости метрика(доля данных)









Анализ Результатов

Более чувствительные к ошибкам в разметке методы(шум до 30%):

- Decision Tree
- KNN

Менее чувствительные к ошибкам в разметке методы(шум до 30%):

- Gradient Boosting
- Random Forest
- Log Reg
- SVM

Более точные модели на меньшем количестве данных:

- Random Forest
- Gradient Boosting

Менее точные модели на меньшем количестве данных:

- KNN
- Log reg

Спасибо за внимание

Весь код, который использовался при решении задачи, можно найти по ссылке ниже

https://github.com/alexmak123/neichev_mipt_ml/blob/main2/Ippi_test/