Хакатон от комании YADRO по ИИ и МО

Команда "Нищета и собаки"

Описание проблемы

Задача

Методами машинного обучения предсказать погодные условия по истории измерений

Данные

- 43 часа
- Координатная сетка размером 30×30
- 7 параметров: высота, температура, влажность, давление, скорость и направление ветра, облачность

Целевая метрика

MAPE (Mean Average Percentage Error).

Способы решения

ML

Использование методов **классического машинного обучения** для предсказания погодных показателей на основе данных об атмосферных явлениях. Преимуществом является наглядность и строгое математическое описание, а также скорость обучения. Стоит отметить, что показатели точности моделей ML достаточно высоки.

NN

Применение **нейронных сетей** для построения более сложных прогностических моделей. Имеет потенциал к более точным прогнозам и адаптации к данным, имеющим слабые корреляции. Однако требует много ресурсов и времени на тонкую настройку и обучение.

Концепция решения

1. Найти наиболее оптимальную модель для каждого из параметров, поддающихся эффективному предсказанию (температура, влажность, давление)
2. Для остальных параметров подобрать эффективный и обоснованный подход для минимизации целевой метрики.

Более точную информацию вы сможете найти в ноутбуке

Архитектура решения

1

Обработка данных

Мы детально проанализировали данные, чтобы понять, какие из них не являются для нас ключевыми, как они распределены и какие модели стоит выбрать для реализации поставленной задачи.

2

Тестирование моделей

Проверили, какая из выбранных моделей окажется наиболее точной для предсказания каждого таргета.

3

Построение модели

По итогам тестов для каждого таргета сформировали модель, которая включает в себя результаты работы с данными и их анализа.

Обработка данных

Структура данных

Данные имеют структуру, представленную на картинке:

	day	time	y	x	z	temperature	pressure	humidity	wind_speed	wind_dir	cloud_cover
0	1	0	1	1	129.0	19.900000	1006.000000	74.0	20.1	111.0	2.0
1	1	0	1	2	116.0	20.000000	1006.000000	74.0	20.1	111.0	2.0
2	1	0	1	3	113.0	20.000000	1006.000000	74.0	20.1	111.0	2.0
3	1	0	1	4	115.0	19.400000	1006.299988	73.0	18.4	107.0	3.0
4	1	0	1	5	122.0	19.299999	1006.299988	73.0	18.4	107.0	3.0
...	...	...	...	...	...	...	...	...	...	...	...
38695	2	18	30	26	134.0	21.600000	1000.299988	85.0	9.3	28.0	14.0
38696	2	18	30	27	110.0	21.700001	1000.400024	83.0	5.4	37.0	26.0
38697	2	18	30	28	109.0	21.700001	1000.400024	83.0	5.4	37.0	26.0
38698	2	18	30	29	124.0	21.600000	1000.400024	83.0	5.4	37.0	26.0
38699	2	18	30	30	140.0	21.500000	1000.400024	83.0	5.4	37.0	26.0

38700 rows × 11 columns

Немного дополнительной информации о датафрейме (распределения значений, пропущенные значения):

	day	time	y	x	z	temperature	pressure	humidity	wind_speed	wind_dir	cloud_cover
count	38700.000000	38700.000000	38700.000000	38700.000000	38700.000000	38700.000000	38700.000000	38700.000000	38700.000000	38700.000000	38700.000000
mean	1.441860	10.395349	15.500000	15.500000	123.945557	25.320974	1003.612028	62.899948	12.809393	144.326202	21.193953
std	0.496615	6.445343	8.655553	8.655553	36.028423	3.994402	1.601819	14.660645	4.048162	41.743641	19.025041
min	1.000000	0.000000	1.000000	1.000000	63.000000	16.799999	1000.200012	26.000000	0.000000	2.000000	0.000000
25%	1.000000	5.000000	8.000000	8.000000	95.000000	21.600000	1002.299988	53.000000	10.200000	125.000000	4.000000
50%	1.000000	10.000000	15.500000	15.500000	118.500000	25.500000	1003.799988	63.000000	13.000000	149.000000	19.000000
75%	2.000000	16.000000	23.000000	23.000000	147.000000	28.400000	1004.599976	74.000000	15.300000	166.000000	33.000000
max	2.000000	23.000000	30.000000	30.000000	235.000000	34.099998	1007.700012	94.000000	29.799999	360.000000	100.000000

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 38700 entries, 0 to 38699
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   day          38700 non-null  int64
 1   time         38700 non-null  int64
 2   y            38700 non-null  int64
 3   x            38700 non-null  int64
 4   z            38700 non-null  float32
 5   temperature  38700 non-null  float64
 6   pressure     38700 non-null  float64
 7   humidity     38700 non-null  float64
 8   wind_speed   38700 non-null  float64
 9   wind_dir     38700 non-null  float64
 10  cloud_cover  38700 non-null  float64
dtypes: float32(1), float64(6), int64(4)
memory usage: 3.1 MB
```

Отчетливо видно, что данные являются достаточно "чистыми" и заполнять пропуски не требуется. Однако все еще остаются трудности с интерпретацией направления ветра и облачности, которые не поддаются простому описанию. Забегая вперед, для их описания нужно либо собрать большой массив данных, чтобы построить data-driven модель необходимой точности, либо создавать математическую модель движения воздушных масс.

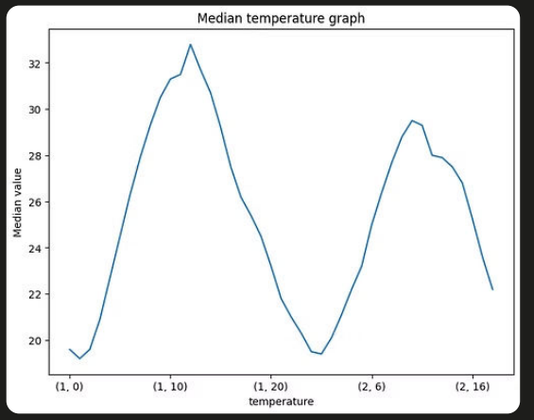
Оба подхода требуют слишком много времени и ресурсов для того, чтобы использоваться в рамках хакатона.

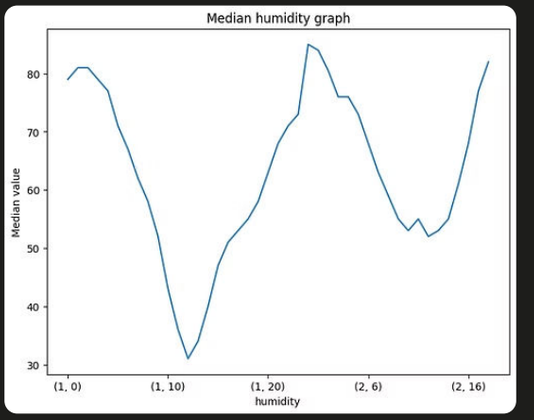
Распределение данных

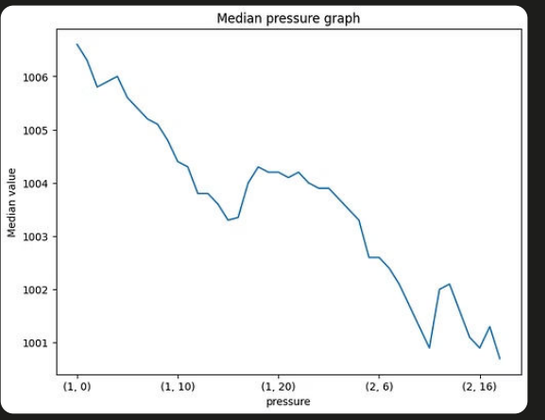
Далее представлены два типа визуализации (см. ноутбук и папку gif):

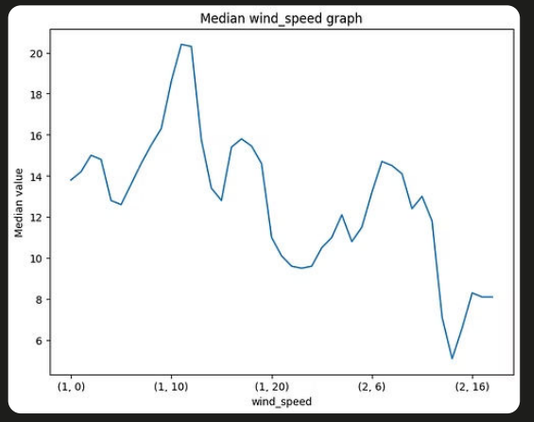
Графики распределения

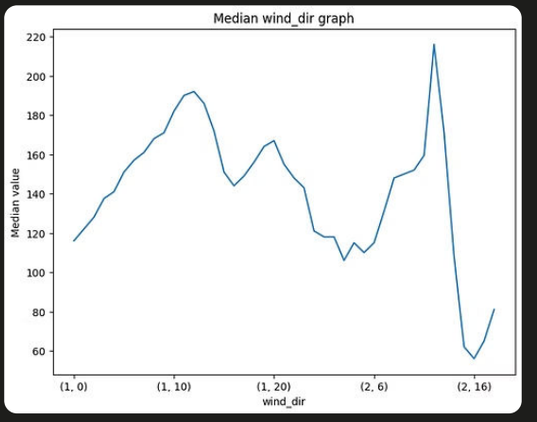
Графики медианных значений (по сетке) признаков по часам (слева направо, первый ряд: температура, влажность, давление; второй ряд: скорость ветра, направление ветра, облачность)

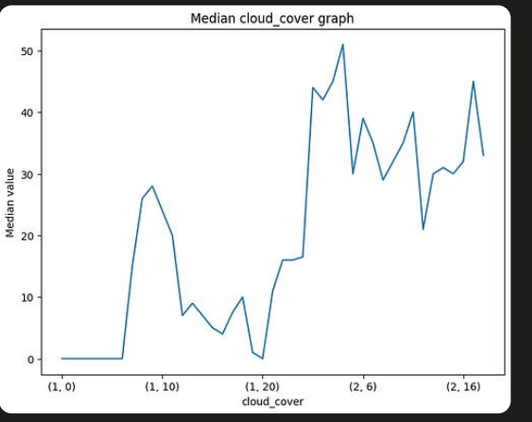






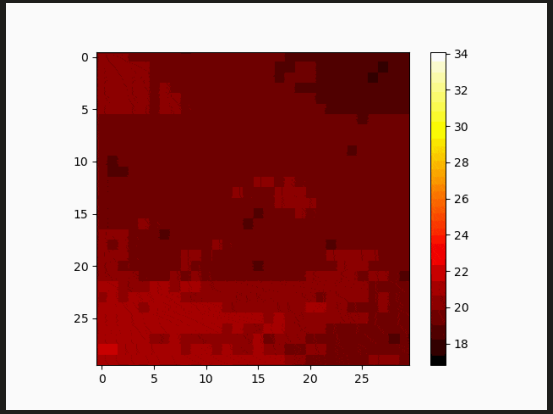




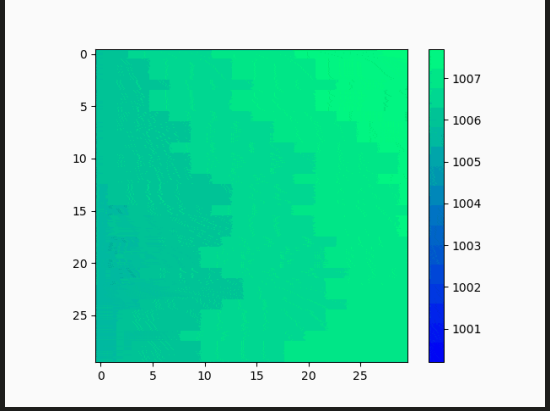


Анимированная визуализация

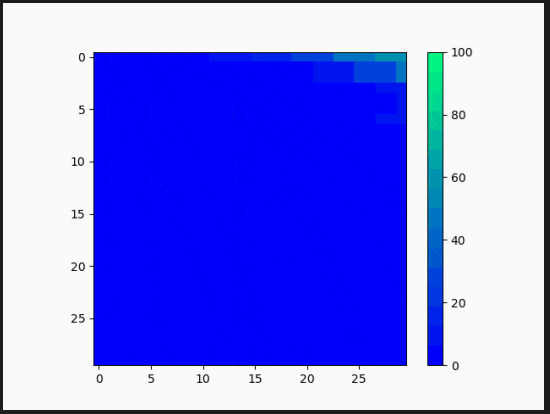
Температура



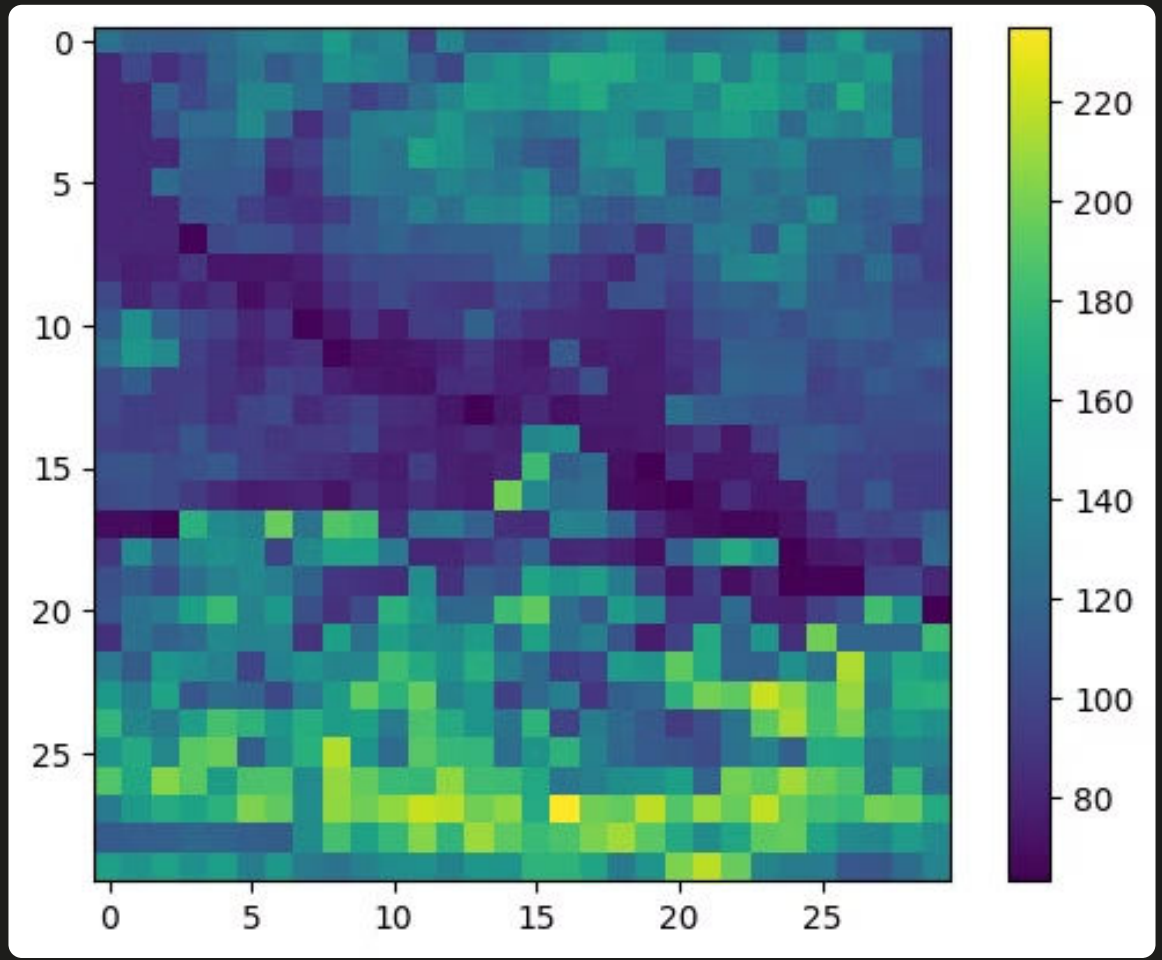
Давление



Облачность



Карта высот



Результаты анализа

Если мы посмотрим полученные gif-ки и графики, то по ним видно, что температура и влажность меняются примерно соответственно суточным ритмам. На основе этого предполагаем, что их можно будет обработать как time-series данные. Предварительно для них в качестве базовых моделей возьмем градиентный бустинг и случайный лес, в качестве продвинутой - RNN.

Давление меняется регрессионно - будем использовать линейную регрессию, либо так же "деревянные" решения.

Данные по облачности и ветру выглядят сильно распредленными и довольно случайными. В качестве базового решения будем предсказывать их по treshold-у, который выберем исходя из метрики качества. Также попробуем посчитать ковариацию этих данных с остальными и, возможно, попробуем предсказывать их на основе предсказаний для первых трех столбцов.

Обоснование подхода к решению

См. подробнее в ноутбуке

В нашем случае мы считаем итоговую метрику как MAPE по двумерной матрице ответов:

При этом очевидно, что мы можем представить ее как среднюю MAPE по столбцам.

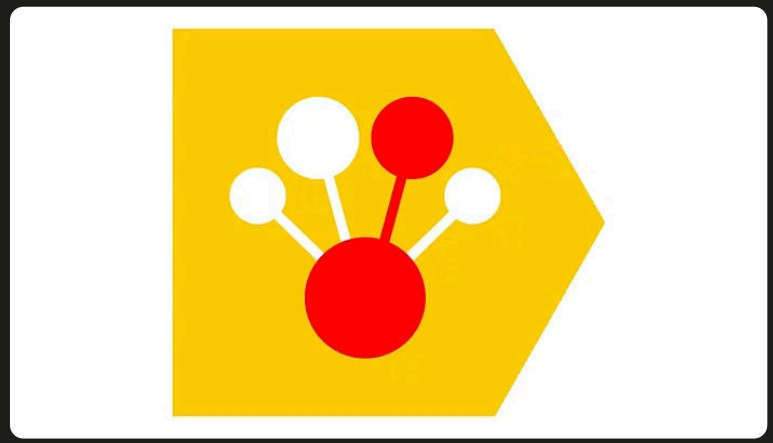
Таким образом, для оптимизации итоговой метрики нам нужно оптимизировать ее по каждому столбцу. Идея очевидная, но для нас это означает, что если мы будем (а мы действительно будем) предсказывать какие-то отдельные значения сильно лучше других, дальнейшее их уточнение даст нам куда меньший итоговый выигрыш, чем уточнение "проблемных" столбцов.

Теперь выберем treshold-значения. Нам нужно такое константное решение $\hat{a}$, которое позволит минимизировать MAPE по столбцу, то есть наше оптимальное значение - это медиана по столбцу. Поэтому treshold-значения будем выбирать как медиану нужного столбца по всем клеткам за последний час.

Выбор моделей

Мы остановились на двух наиболее популярных моделях бустинга: CatBoost и XGBoost.

В ноутбуке также есть сравнение со случайным лесом и линейной регрессией. В силу их неэффективности по сравнению с бустингом, мы не стали вставлять их в презентацию, как основные модели.



В дополнение к выбранным моделям мы решили обучить RNN (рекуррентную нейронную сеть), для того, чтобы сравнить ее эффективность в данной задаче с классическим машинным обучением.

Тестирование

Тестирование было проведено на всех таргетах, однако для наглядности мы оставили только тест на столбце "Температура", как наиболее предсказуемом и наглядном. Результаты прочих тестов показали схожие результаты.

Точность моделей МО

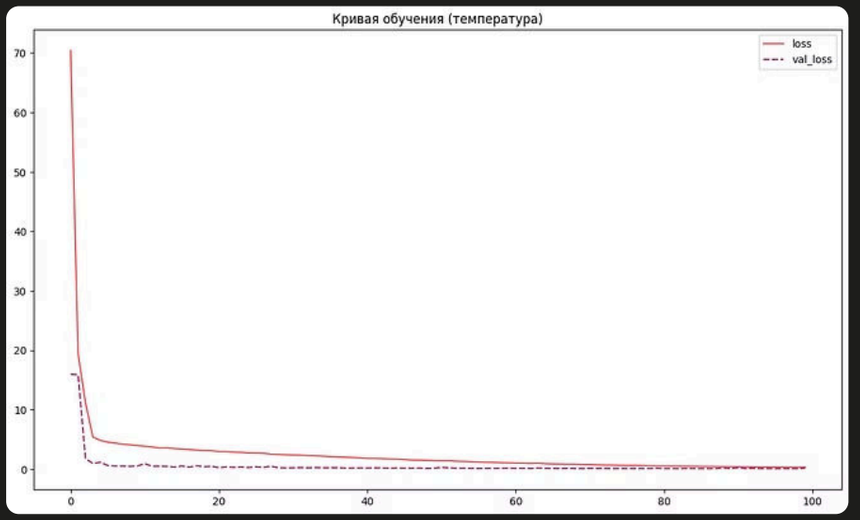
```
Модель Линейная регрессия
MAPE on train:      0.1306873930607974
MAPE on validation: 0.1319994045960414

Модель Случайный лес
MAPE on train:      0.001973779215924481
MAPE on validation: 0.005408247765133007

Модель XG-Boost
MAPE on train:      0.004976274713740796
MAPE on validation: 0.005689957901703988

Модель CatBoost
MAPE on train:      0.008373532357272164
MAPE on validation: 0.008576696334532879
```

Кривая обучения RNN



Точность RNN

```
908/908 [==============================] - 4s 5ms/step
MAPE on train:      0.010537361580977808
303/303 [==============================] - 2s 5ms/step
MAPE on validation: 0.010712377233502058
```

Результаты

Температура

Линейная регрессия явно уступает.	Улучшать случайный лес нет смысла. С метрикой абсолютной ошибки он обучается значительно дольше, особого улучшения результата при этом не дает.
XGBoost лучший при обучении, но на валидации выигрывает CatBoost. Их мы и оставляем в качестве baseline-модели.	RNN на наших данных не выигрывает, результаты сравнимы с XGBoost и CatBoost. Однако на отложенной выборке (при загрузке решения в бота) выигрывает на 2%, поэтому **лучшей считаем ее.**

Давление и влажность

Вообще, с учетом того, что целевая метрика - MAPE, давление можно предсказывать константой и все равно иметь очень высокую точность по этому показателю, просто потому что порядок самой величины - 1000, а ее разброса - 7. Соответственно, максимальная ошибка будет 0.7%.

Однако мы планируем использовать данные первых трех столбцов в качестве входных для предсказания последних трех. Поэтому нам необходимо предсказывать их с максимально возможной точностью.

Итоговая модель

Давление предсказываем с помощью **XGBoost**

```
MAPE on train:      5.0754213509818 89e-05
MAPE on validation: 5.6842261999894 96e-05
```

Для температуры и влажности влажности используем **RNN**

```
908/908 [==============================] - 4s 5ms/step
MAPE on train:      0.0105373615809778 08
303/303 [==============================] - 2s 5ms/step
MAPE on validation: 0.0107123772335020 58
```

```
908/908 [==============================] - 4s 5ms/step
MAPE on train:      0.0187039173378555 9
303/303 [==============================] - 2s 5ms/step
MAPE on validation: 0.0190238411317520 22
```

Скорость ветра

Скорость ветра в целом +/- постоянна и хорошо аппроксимируется медианой с последнего часа. В качестве продвинутого решения - бустинг, который берет в качестве аргументов дополнительно температуру, давление и влажность. Соответственно ему на вход подаются предсказания с этих трех столбцов.

Данный метод показывает большую эффективность, чем просто медиана.

Направление ветра

Лучшие показатели выходят, если использовать просто медиану из предсказаний линейной регрессии на синтетически предсказанных [температуре, влажности, давлении] в качестве константы, дополнительно занизив ее на стандартное отклонение (потому что здесь у нас данные могут быть близки к нулю, и из-за MAPE нам лучше предсказание занизить, чем завысить).

Работает лучше минимума за последний час.

Облачность

По графику медианной облачности видно, что в целом она имеет суточный режим, при этом имея явный возрастающий тренд. Поэтому ее мы построим следующим образом. Возьмем распределение за последние 5 часов предыдущего дня (то есть ровно сутки назад) и поднимем на минимальное значение с последнего часа.

Результат

Оценка MAPE для submission_finaaaaaaaaal.csv:

0.33290354544848527

Ошибка отдельно по значениям:

temperature: 0.047593807002872 81

pressure: 0.0005631976567073602

humidity: 0.0604622778 19842446

wind_speed: 0.4325689381528198

wind_dir: 0.8893911332206957

cloud_cover: 0.5668419188379733

Преимущества решения

1

Скорость

Решение не основывается на сложных нейронных сетях и не предусматривает долгого обучения моделей перед использованием. В общей сложности требуется до 30 минут на полной обучение для получения наибольшей точности. При упрощении модели время обучения может быть снижено до нескольких секунд (при использовании XGBoost и медиан для "проблемных" столбцов)

2

Наглядность

Для предсказания большинства признаков используются стандартные методы машинного обучения, а также базовые математические методы, что позволяет легко описать процесс обучения и интерпретировать результаты.

3

Точность

Приведенные в решении обобщения позволили добиться максимальной точности при сохранении универсальности решения. Дальнейшее увеличение точности достижимо при увеличении датасета по времени и использования более сложных моделей (CNN).

Команда

Игорь Дубинкин

Нищий

ИФИБ Б21-311

Артемий Терещенко

Собака

ИФИБ Б21-302