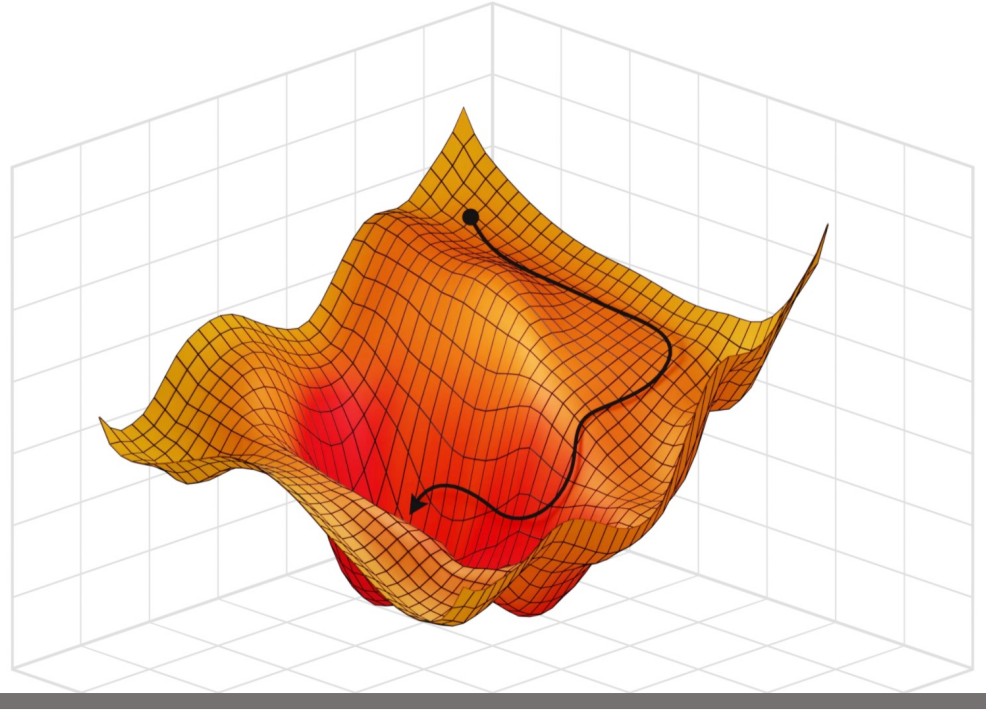
ЕВРАЗ

Команда: Dsteam

Буслов Дмитрий

Фиронов Сергей

Задача 1: Продуйте металл через Data Science

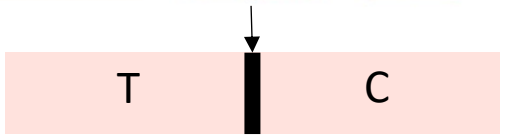


EVRAZ AI CHALLENGE

Постановка задачи

Задача - разработать алгоритм предсказания содержание углерода и температуру чугуна.

Данные - produv lom plavki sip chronom chugun target gas



Дополнительная задача - придумать как обобщить алгоритм прогноза на ситуацию реального производства. В реальном времени мы не знаем, когда наступит конец продувки, а должны определить это сами на основании прогнозируемых параметров чугуна.

- Что нужно добавить?
- Как перейти от модельной задачи к задаче из реального мира?
- Как развить алгоритм так, чтобы это было максимально полезно бизнесу?

Справочная минутка по чтению данных для DS

Пропустить, если мало времени

Когда лень писать код:

```
files_train = glob.glob(DATA_DIR+'*train.csv')
files_test = glob.glob(DATA_DIR+'*test.csv')
```

```
def gen_read(files,train_type):
    for i in files:
        print(f'df_{i[8:].split("_")[0]}_{train_type}=fread(\'{i}\').to_pandas()')
gen_read(files_train,'train')
print('-'*50)
gen_read(files_test,'test')
```

```
df_sip_train=fread('../data/sip_train.csv').to_pandas()
df_lom_train=fread('../data/lom_train.csv').to_pandas()
df_chronom_train=fread('../data/chronom_train.csv').to_pandas()
df_produv_train=fread('../data/produv_train.csv').to_pandas()
df_target_train=fread('../data/target_train.csv').to_pandas()
df_chugun_train=fread('../data/chugun_train.csv').to_pandas()
df_gas_train=fread('../data/gas_train.csv').to_pandas()
df_plavki_train=fread('../data/plavki_train.csv').to_pandas()
--------------------------------------------------
df_gas_test=fread('../data/gas_test.csv').to_pandas()
df_produv_test=fread('../data/produv_test.csv').to_pandas()
df_lom_test=fread('../data/lom_test.csv').to_pandas()
df_chugun_test=fread('../data/chugun_test.csv').to_pandas()
df_chronom_test=fread('../data/chronom_test.csv').to_pandas()
df_plavki_test=fread('../data/plavki_test.csv').to_pandas()
df_sip_test=fread('../data/sip_test.csv').to_pandas()
```

Когда pandas заставляет ждать…

```
%%time
df_sip=pd.read_csv('../data/sip_train.csv')
df_lom=pd.read_csv('../data/lom_train.csv')
df_chronom=pd.read_csv('../data/chronom_train.csv')
df_produv=pd.read_csv('../data/produv_train.csv')
df_target=pd.read_csv('../data/target_train.csv')
df_chugun=pd.read_csv('../data/chugun_train.csv')
df_gas=pd.read_csv('../data/gas_train.csv')
df_plavki=pd.read_csv('../data/plavki_train.csv')
```

```
CPU times: user 15.1 s, sys: 1.4 s, total: 16.5 s
Wall time: 16.5 s
```

```
%%time
df_sip=fread('../data/sip_train.csv').to_pandas()
df_lom=fread('../data/lom_train.csv').to_pandas()
df_chronom=fread('../data/chronom_train.csv').to_pandas()
df_produv=fread('../data/produv_train.csv').to_pandas()
df_target=fread('../data/target_train.csv').to_pandas()
df_chugun=fread('../data/chugun_train.csv').to_pandas()
df_gas=fread('../data/gas_train.csv').to_pandas()
df_plavki=fread('../data/plavki_train.csv').to_pandas()
```

```
CPU times: user 7.08 s, sys: 834 ms, total: 7.91 s
Wall time: 2.41 s
```

Нюансы в данных

Adversarial валидация - дает 0.96 ROC_AUC

Плавка 511135 - 75 разных стойкостей фурмы и головки фурмы

	Марка только в тесте
0	СТ3ПС.7
1	Ст3сп/ЭТ
2	СТ3ГСП.4
3	Ст4сп/ЭТ
4	Ст3сп/Т
5	3SP.z24/ЭТ
6	С121ТМ/ЭТ
7	5SP.z24/ЭТ
8	Ст3пс
9	08YU.z02/ЭТ
10	Ст1пс.z05/ЭТ

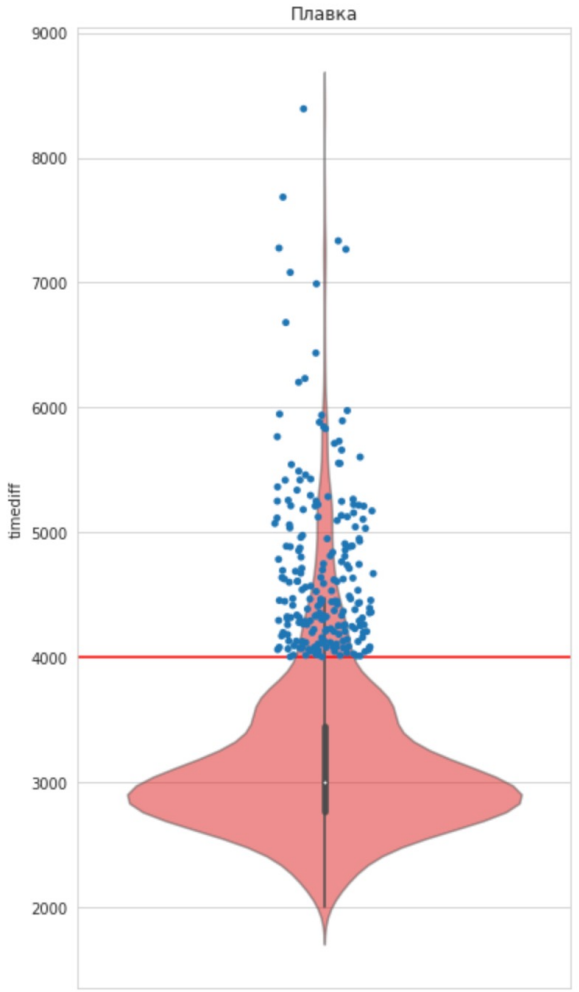
Отличия в сыпучих

Train dataset

	index	NMSYP
0	изв_ЦОИ	0.603312
1	Уголь ТО	0.245282
2	ФЛЮМАГ	0.092768
3	Флюс ФОМИ	0.051418
4	изв отсев	0.007156
5	Долом(сыр)	0.000063

Test dataset

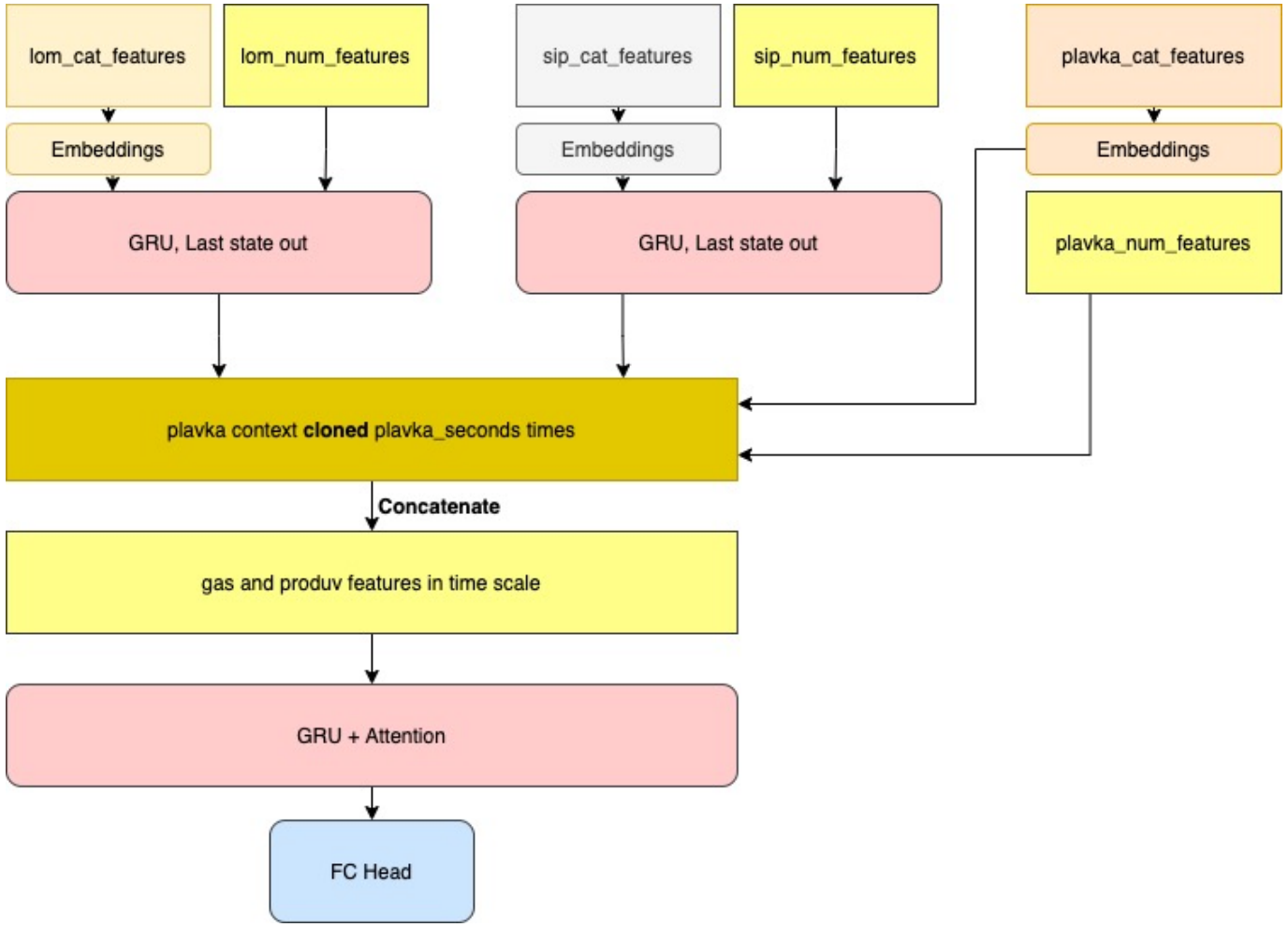
	index	NMSYP
0	изв_ЦОИ	0.737717
1	Уголь ТО	0.100268
2	ФЛЮМАГ	0.080482
3	Флюс ФОМИ	0.032594
4	изв_отсев	0.019499
5	агл_офл_с	0.014720
6	долом_сыр	0.014051
7	кокс_25_40	0.000669



Здесь должен был быть этот мем, но у нас *“серьезная”* презентация…

Поэтому, он у нас находится в суперпозиции - он вроде как есть, но его как бы и нет…

Модель



Ключевые моменты:

На вход модели подаются данные до конца плавки (не заглядываем в будущее и обрезаем)

Низкочастотные категории отбрасываем, заменяй их на ‘none’ (для ембединга)

Аналогичный подход для категорий теста

Мы индексируем события по времени с начала плавки и модель может предиктить значения в любой момент времени, они меняются плавно

Интерфейс

Что важно бизнесу?

- ~~Когда какая будет температура~~
- ~~Когда какой будет углерод~~
- **Какие управляющие параметры менять и когда**
- **Как изменение параметров повлияет на результат**
- **Какая экономия**

	NPLV	plavka_VR_NACH	plavka_VR_KON	plavka_NMZ	plavka_NAPR_ZAD	plavka_STFUT	plavka_TIPE_FUR	plavka_ST_FURM	plavka_TIPE_GOL	plavka_ST_GOL	TST	C
2	510010	2021-01-01T05:12:29	2021-01-01T06:00:53	Ст3пс/Э	Изл	973	цилиндрическая	13	5 сопловая	13	1662	0.0910
12	510020	2021-01-01T16:31:53	2021-01-01T17:19:16	Ст3пс/Э	Изл	983	цилиндрическая	23	5 сопловая	23	1606	0.1070
23	510033	2021-01-02T07:46:17	2021-01-02T08:38:22	Ст3пс/Э	Изл	996	цилиндрическая	36	5 сопловая	36	1649	0.0680
40	510060	2021-01-03T15:04:52	2021-01-03T15:57:40	Ст3пс/Э	Изл	1023	цилиндрическая	4	5 сопловая	4	1663	0.0670
41	510061	2021-01-03T16:19:54	2021-01-03T17:07:49	Ст3пс/Э	Изл	1024	цилиндрическая	5	5 сопловая	5	1697	0.0650

Предложения

Быстрый результат - это только прогноз для информации оператора и поиск аналогичных плавок.

Оценка влияния на результат статичного изменения отдельных управляющих параметров

Оценка влияния на результат совместного изменения управляющих параметров

Использования моделей для формирования среды симуляции (gym)

Использование Decision Transformer (RL) https://arxiv.org/abs/2106.01345

А еще может можно собрать больше данных с других заводов?

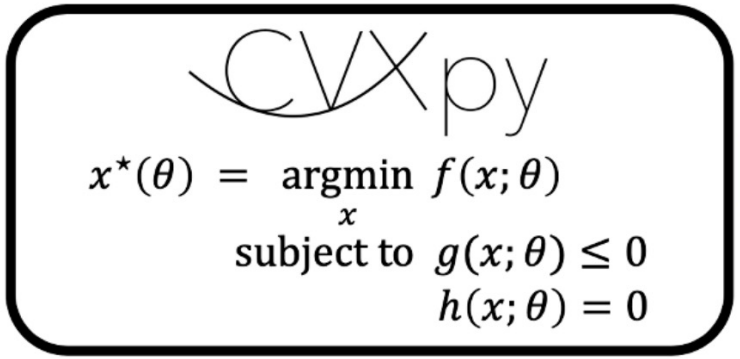
Доменные знания -> модель

Ограничения на отдельные признаки и их интеракции имеет смысл закладывать сразу в модель.

Для нейронок - дифференциируемые слои **cvxpylayers**

Для бустинга - монотонные ограничения

Мы не успели, но очень хотели…

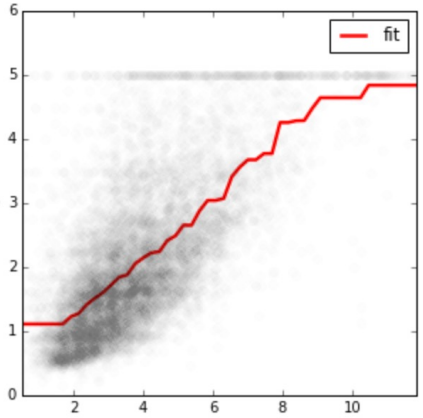


PyTorch

TensorFlow

dmlc XGBoost

Monotonic Constraints



Металл

В любом исходе - мы сможем говорить, что мы **продули**!