

ИТОГОВАЯ АТТЕСТАЦИЯ

Тема: Разработка и тестирование приложения для анализа и прогнозирования метеорологических данных, влияющих на промышленные объекты, с использованием машинного обучения

Автор работы:
Сорокин Максим Евгеньевич

Руководитель:
Корнеева Елена Игоревна

Иннополис 2023



Одним из наиболее важных достижений добычи нефти и газа является бурение скважин. На сегодняшний день нефтегазовый комплекс играет важнейшую роль в развитии экономики Российской Федерации. Такие комплексы должны работать в непрерывном цикле. Анализируя одну из нефтяных скважин, можно выявить проблему в раннем оповещении поломок и нестабильной работы. Рутинной задачей является сбор, обработка и визуализация работы системы по добыче нефти. Внедрение технологий искусственного интеллекта могут ускорить процесс раннего выявления работы нефтяных скважин, качество работы и анализ в зависимости от метеорологических данных.

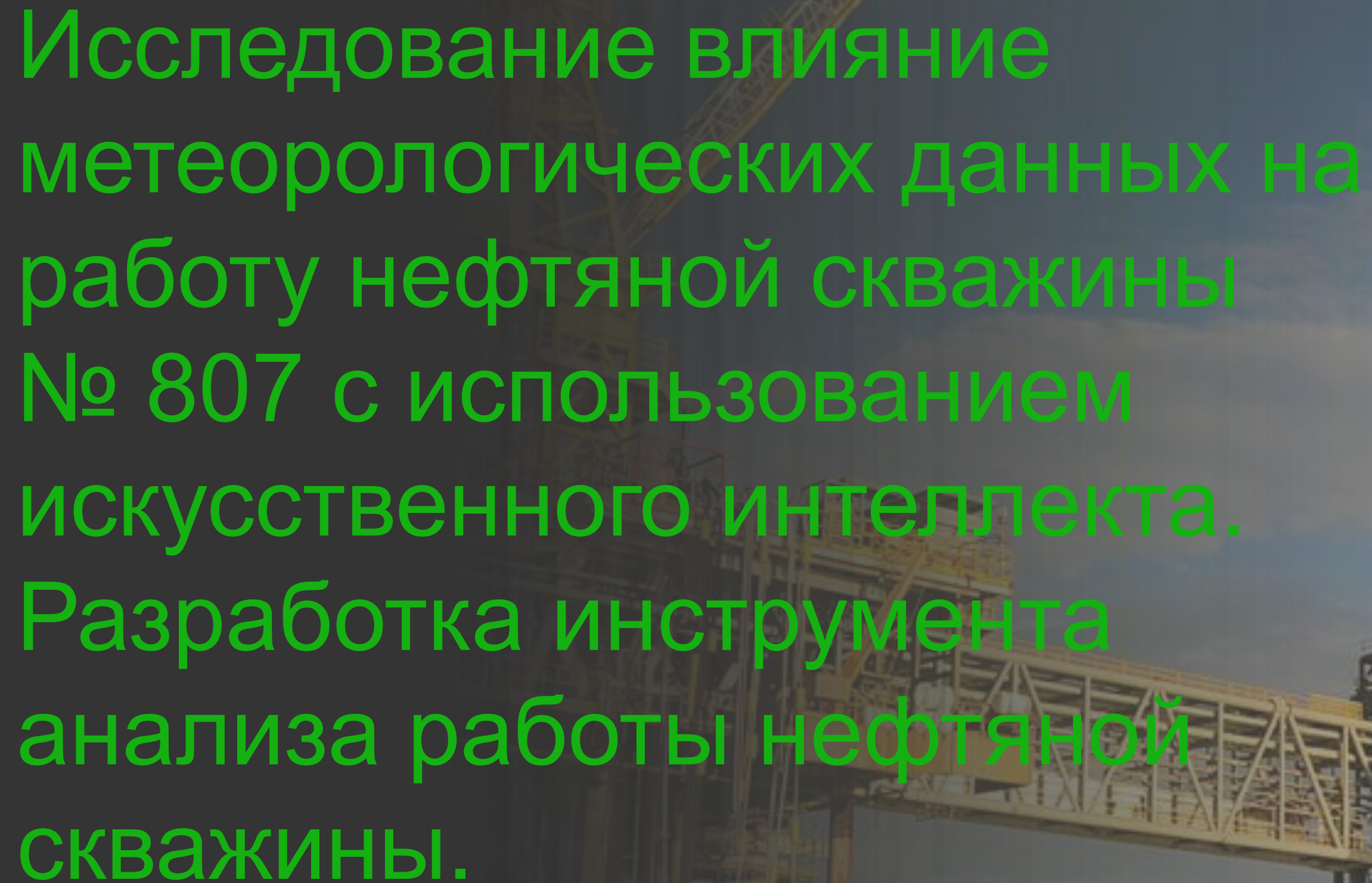
Объект исследования – нефтяная скважина № 807.

Предмет исследования – влияние метеорологических данных на работу нефтяной скважины № 807 с использованием искусственного интеллекта.

Цель работы – создать инструмент анализа работы нефтяной скважины в зависимости от погодных условий и явлений для раннего обнаружения неисправностей.

Решаемые задачи:

1. Сформулировать гипотезу по данным;
2. Собрать данные из разрозненных источников;
3. Провести исследовательский анализ данных и визуализировать основные распределения;
4. Запустить базовые модели машинного обучения;
5. Оценить качество результата по релевантным для задачи метрикам;
6. Получить отчеты по результатам;
7. Сделать выводы и дать дальнейшие рекомендации.



Исследование влияние
метеорологических данных на
работу нефтяной скважины
№ 807 с использованием
искусственного интеллекта.
Разработка инструмента
анализа работы нефтяной
скважины.

Сбор данных из разрозненных источников



ИСТОЧНИК:

Веб-сайт с архивными данными о погоде в село Самбург: <https://pogoda1.ru/samburg/arkhiv/>. Набор содержит данные с 07.06.2017 года по настоящее время. Ближайшее к нефтяной скважине поселение (находится в 24 км) с известными метеорологическими данными. Село расположено в Ямало-Ненецком автономном округе, Пуровский район.

Разработан и применен парсер HTML страниц в CSV файл для представления данных в удобочитаемый формат. Собрано 2350 строк данных.

Основные метеорологические параметры:

[illegible]

Сбор данных из разрозненных источников

Источник:

Датасет нефтяной скважины № 807: <https://www.kaggle.com/datasets/ruslanzalevskikh/oil-well>. Набор содержит 2939 строки данных о работе нефтяной скважины. Площадь 21,05155 га. Глубина скважины 4100 м. Тюменская область, ЯНАО, Пуровский район. Объект находится в 114 км по азимуту 341,21° от аэропорта г. Тарко-Сале, и в 27,1 км по азимуту 135,49° от аэропорта г. Новый Уренгой. (широта: 66.80035786146756, долгота: 78.38975066524623).

Разработан и применен парсер Excel документа в CSV файл для представления данных в удобочитаемый формат.

Основные параметры эксплуатации нефтяной скважины:

	id	год	месяц	день	объем нефти (м3/сутки)	объем жидкости (м3/сутки)	объем газа (м3/сутки)	объем воды (м3/сутки)	обводненность (%)	рабочее время	динамический уровень (м)	пластовое давление (атм)
0	0	2013	1	1	49	70	13055	21	29	24	1819	214
1	1	2013	1	2	49	70	13055	21	29	24	1836	214
2	2	2013	1	3	49	70	13055	21	29	24	1788	214
3	3	2013	1	4	49	70	13055	21	29	24	1789	214
4	4	2013	1	5	44	70	11768	26	36	24	1825	214
...
2934	2934	2021	1	14	6	20	1593	15	70	16	2101	100
2935	2935	2021	1	15	6	20	1593	15	70	16	2113	100
2936	2936	2021	1	16	6	20	1583	14	70	16	2125	100
2937	2937	2021	1	17	6	20	1573	14	70	16	2125	100
2938	2938	2021	1	18	5	18	1418	13	70	15	2125	100

2939 rows × 12 columns

Исследовательский анализ данных

Произведена подготовка данных в плане заполнения пустых значений – наиболее встречающимися значениями. Перевод категориальных данных в числовые. Изменены типы данных в необходимые числовые. Слияние данных по известным датам из двух датасетов. Произведено удаление ненужных столбцов (типа идентификаторов и ссылочных данных).

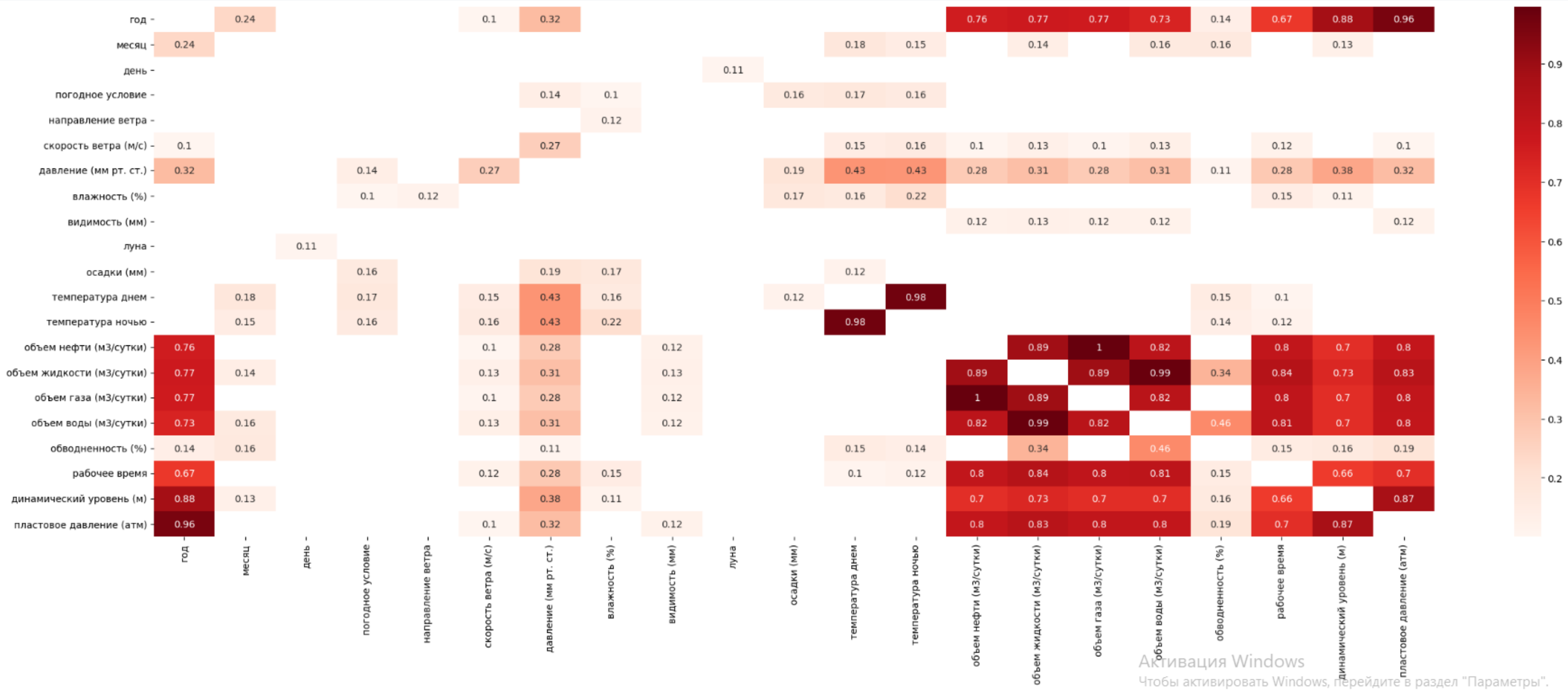
Итоговый объединенный датасет:

	год	месяц	день	погодное условие	направление ветра	скорость ветра (м/с)	давление (мм рт. ст.)	влажность (%)	видимость (мм)	луна	...	температура днем	температура ночью	объем нефти (м3/ сутки)	объем жидкости (м3/ сутки)	объем газа (м3/ сутки)	объем воды (м3/ сутки)	обводненность (%)	рабочее время	д
0	2017	6	7	4	5	4.0	764	56	10.0	3	...	9	13	16	52	4210	36	69	24	
1	2017	6	8	6	5	3.0	762	63	10.0	3	...	9	13	16	52	4210	36	69	24	
2	2017	6	9	2	3	3.0	756	51	10.0	2	...	12	15	16	52	4210	36	69	24	
3	2017	6	10	4	5	8.0	752	60	10.0	4	...	10	10	16	52	4210	36	69	24	
4	2017	6	11	4	2	6.0	756	50	10.0	4	...	4	2	16	52	4210	36	69	24	
...
1310	2021	1	14	5	2	8.0	763	95	380.0	3	...	-13	-13	6	20	1593	15	70	16	
1311	2021	1	15	4	7	3.0	770	87	10.0	3	...	-20	-21	6	20	1593	15	70	16	
1312	2021	1	16	4	6	5.0	759	93	7.4	3	...	-17	-16	6	20	1583	14	70	16	
1313	2021	1	17	5	1	8.0	747	93	132.0	3	...	-19	-19	6	20	1573	14	70	16	
1314	2021	1	18	4	7	4.0	744	89	4.2	3	...	-18	-24	5	18	1418	13	70	15	

1315 rows × 21 columns

Исследовательский анализ данных

Произведен расчет коэффициентов корреляции для всех столбцов (не связаны ли между собой какие-либо атрибуты).
Отсечена корреляция менее 10%:



Исследовательский анализ данных

Собраны пары с высокой корреляцией:

	0
-----	-----
('год', 'год')	1
('объем газа (м3/сутки)', 'объем нефти (м3/сутки)')	0.996308
('объем воды (м3/сутки)', 'объем жидкости (м3/сутки)')	0.990395
('температура днем', 'температура ночью')	0.977463
('пластовое давление (атм)', 'год')	0.96275
('объем жидкости (м3/сутки)', 'объем газа (м3/сутки)')	0.893276
('объем жидкости (м3/сутки)', 'объем нефти (м3/сутки)')	0.889227
('год', 'динамический уровень (м)')	0.882661
('динамический уровень (м)', 'пластовое давление (атм)')	0.873967
('объем жидкости (м3/сутки)', 'рабочее время')	0.838319
('объем жидкости (м3/сутки)', 'пластовое давление (атм)')	0.826001
('объем воды (м3/сутки)', 'объем газа (м3/сутки)')	0.824716
('объем воды (м3/сутки)', 'объем нефти (м3/сутки)')	0.820455
('объем воды (м3/сутки)', 'рабочее время')	0.814131
('объем газа (м3/сутки)', 'рабочее время')	0.804058
('рабочее время', 'объем нефти (м3/сутки)')	0.802173
('объем газа (м3/сутки)', 'пластовое давление (атм)')	0.801352
('объем воды (м3/сутки)', 'пластовое давление (атм)')	0.79939
('объем нефти (м3/сутки)', 'пластовое давление (атм)')	0.796269
('объем жидкости (м3/сутки)', 'год')	0.76557
('объем газа (м3/сутки)', 'год')	0.765069
('год', 'объем нефти (м3/сутки)')	0.759238
('год', 'объем воды (м3/сутки)')	0.733988
...	
('давление (мм рт. ст.)', 'видимость (мм)')	0.00101883
('день', 'скорость ветра (м/с)')	0.000920772
('скорость ветра (м/с)', 'видимость (мм)')	0.000428875
('луна', 'объем жидкости (м3/сутки)')	0.000198837



Запуск базовых моделей машинного обучения

Произведено разделение данных на X (экзогенные переменные, т.е. регрессоры или независимые) и y (эндогенные переменные или зависимые) и нормализция данных:

```
X = data.drop([
    'объем нефти (м3/сутки)',
    'объем жидкости (м3/сутки)',
    'объем газа (м3/сутки)',
    'объем воды (м3/сутки)',
    'обводненность (%)',
    'рабочее время',
    'динамический уровень (м)',
    'пластовое давление (атм)'
], axis=1)

y = data[
    'объем нефти (м3/сутки)',
    'объем жидкости (м3/сутки)',
    'объем газа (м3/сутки)',
    'объем воды (м3/сутки)',
    'обводненность (%)',
    'рабочее время',
    'динамический уровень (м)',
    'пластовое давление (атм)'
]
```

```
# Нормализация (MinMaxScaler)
scalar = MinMaxScaler()
features = scalar.fit_transform(X, y)
X_normalised = pd.DataFrame(features, columns=X.columns)
X_normalised
```

	погодное условие	направление ветра	скорость ветра (м/с)	давление (мм рт. ст.)	влажность (%)	видимость (мм)	луна	осадки (мм)	температура днем	температура ночью
0	0.6	0.571429	0.069767	0.454545	0.56	0.009738	0.666667	0.0	0.820896	0.776316
1	1.0	0.571429	0.046512	0.428571	0.63	0.009738	0.666667	0.0	0.820896	0.776316
2	0.2	0.285714	0.046512	0.350649	0.51	0.009738	0.333333	0.0	0.865672	0.802632
3	0.6	0.571429	0.162791	0.298701	0.60	0.009738	1.000000	0.0	0.835821	0.736842
4	0.6	0.142857	0.116279	0.350649	0.50	0.009738	1.000000	0.0	0.746269	0.631579
...
1310	0.8	0.142857	0.162791	0.441558	0.95	0.414597	0.666667	0.0	0.492537	0.434211
1311	0.6	0.857143	0.046512	0.532468	0.87	0.009738	0.666667	0.0	0.388060	0.328947
1312	0.6	0.714286	0.093023	0.389610	0.93	0.006894	0.666667	0.0	0.432836	0.394737
1313	0.8	0.000000	0.162791	0.233766	0.93	0.143232	0.666667	0.0	0.402985	0.355263
1314	0.6	0.857143	0.069767	0.194805	0.89	0.003392	0.666667	0.0	0.417910	0.289474

1315 rows × 10 columns

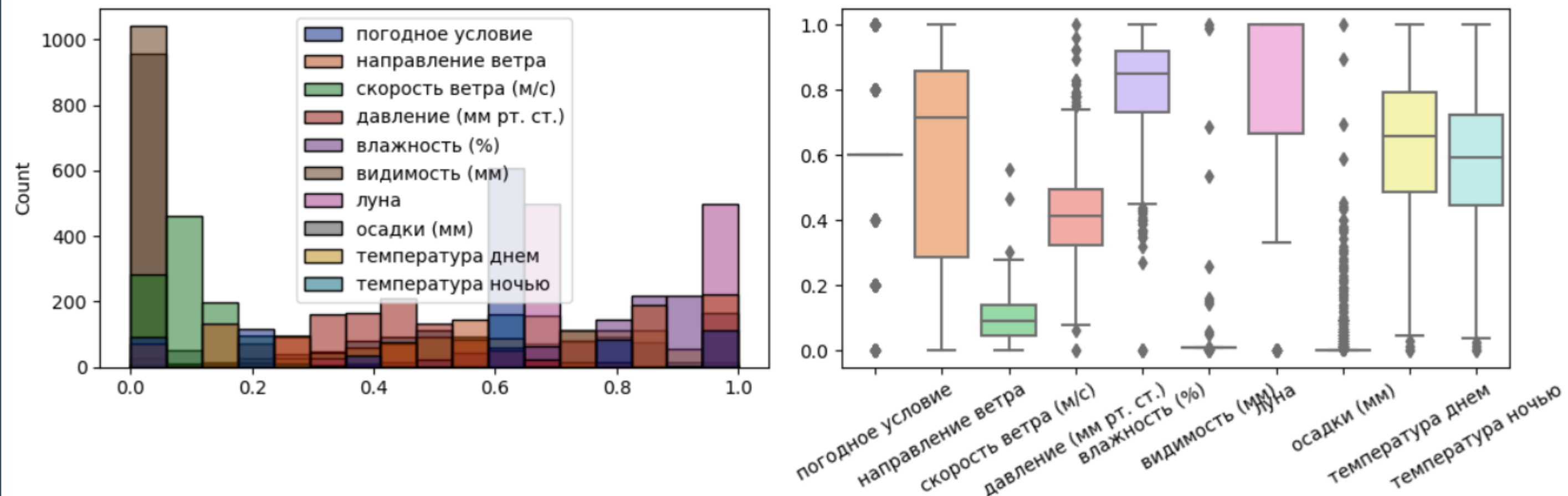
Запуск базовых моделей машинного обучения

Данные также были разделены на тренировочную (80%), тестовую (10%) и валидационную (10%) выборку:

```
# Разделение на тренировочную (80%), тестовую (10%) и валидационную (10%)
X_train, X_test, y_train, y_test = train_test_split(X_normalised, y, test_size=0.2, random_state=1024, shuffle=True)
X_test, X_val, y_test, y_val = train_test_split(X_test, y_test, test_size=0.5, random_state=1024, shuffle=True)
```

Построен график тренировочных данных:

```
# Строю графики тренировочных данных
fig, axes = plt.subplots(nrows=1, ncols=2, figsize=(12, 4))
sns.histplot(data=X_train, ax=axes[0], palette='dark')
boxplot = sns.boxplot(data=X_train, ax=axes[1], palette='pastel')
boxplot.tick_params(axis='x', rotation=30)
plt.tight_layout()
```



Запуск и оценка качества моделей машинного обучения

Произведен запуск обучения данных на алгоритмах: метод наименьших квадратов, случайный лес и метод ближайших соседей. Получены результаты коэффициента детерминации по R в квадрате.

```
Модель LinearRegression:
Правильность на обучающем наборе: 0.128
Правильность на тестовом наборе: 0.079

Модель LinearRegression:
Правильность на обучающем наборе: 0.142
Правильность на тестовом наборе: 0.122

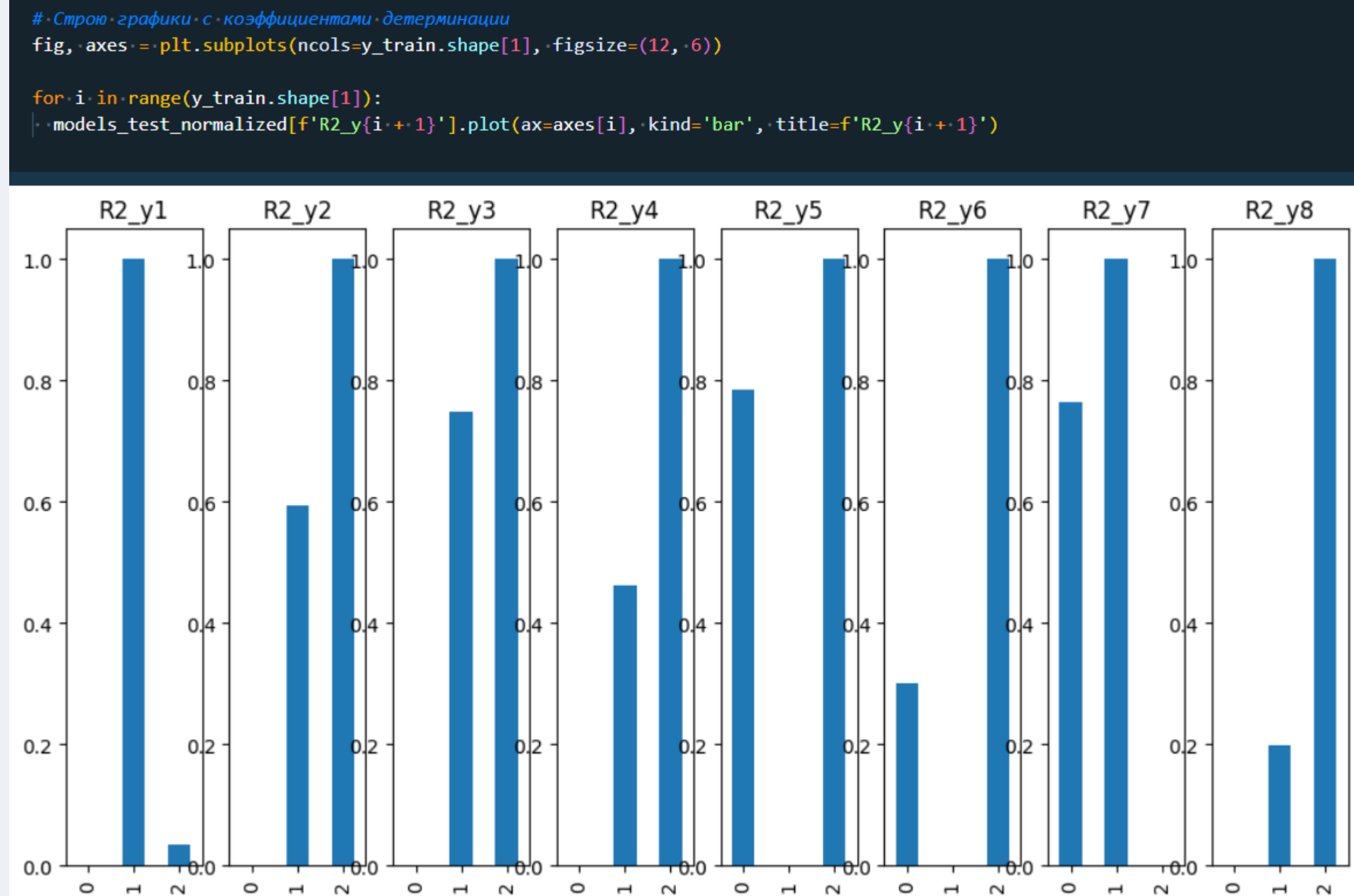
Модель LinearRegression:
Правильность на обучающем наборе: 0.129
Правильность на тестовом наборе: 0.075

Модель LinearRegression:
Правильность на обучающем наборе: 0.139
Правильность на тестовом наборе: 0.122

Модель LinearRegression:
Правильность на обучающем наборе: 0.044
Правильность на тестовом наборе: -0.011

Модель LinearRegression:
Правильность на обучающем наборе: 0.182
Правильность на тестовом наборе: 0.165
...

Модель KNeighborsRegressor:
Правильность на обучающем наборе: 0.368
Правильность на тестовом наборе: 0.082
```



Запуск и оценка качества моделей машинного обучения

Произведен анализ и обучение лучших признаков.

```
# Проведу анализ лучших признаков

# Для каждого столбца результирующего набора
for i in range(y_train.shape[1]):
    # Выбор 2 лучших признаков из 8 по Хи-квадрат
    selector = SelectKBest(chi2, k=2)
    X_train_best = selector.fit_transform(X_train, y_train.iloc[:, i])
    # print(X_train_best.shape)
    X.columns[selector.get_support(indices=True)]
    vector_names = list(X.columns[selector.get_support(indices=True)])
    print(f'Для {y.columns[i]} лучшие признаки: {vector_names}')
    # X_train_best_df = pd.DataFrame(X_train_best, columns=selector.get_support(indices=True))
    # print(X_train_best_df)
```

```
Для ['объем нефти (м3/сутки)'] лучшие признаки: ['видимость (мм)', 'осадки (мм)']
Для ['объем жидкости (м3/сутки)'] лучшие признаки: ['видимость (мм)', 'осадки (мм)']
Для ['объем газа (м3/сутки)'] лучшие признаки: ['направление ветра', 'осадки (мм)']
Для ['объем воды (м3/сутки)'] лучшие признаки: ['видимость (мм)', 'температура ночью']
Для ['обводненность (%)'] лучшие признаки: ['температура днем', 'температура ночью']
Для ['рабочее время'] лучшие признаки: ['давление (мм рт. ст.)', 'видимость (мм)']
Для ['динамический уровень (м)'] лучшие признаки: ['осадки (мм)', 'температура днем']
Для ['пластовое давление (атм)'] лучшие признаки: ['температура днем', 'температура ночью']
```

```
# обучение лучшей модели с лучшими признаками
model_best = models[2]
model_best.fit(X_train, y_train)

# качество модели
score = model.score(X_test, y_test)
print("Accuracy: ", score*100) # A, P, R, E
```

Accuracy: -0.3609410049226727

Запуск и оценка качества моделей машинного обучения

Произведено обучение влияние температуры на отдельные параметры нефтяной скважины.

```
# Сделаю обучение (KNeighborsRegressor) влияние температуры на обводненность
KNeighbors = KNeighborsRegressor(n_neighbors=6)
KNeighbors.fit(X_train[['температура-днем', 'температура-ночью']], y_train[['обводненность-(%)']])
print(f"Тренировочный набор: {KNeighbors.score(X_train[['температура-днем', 'температура-ночью']], y_train[['обводненность-(%)']])*100}")
print(f"Тестовый набор: {KNeighbors.score(X_test[['температура-днем', 'температура-ночью']], y_test[['обводненность-(%)']])*100}")
```

Тренировочный набор: 16.148008250982006

Тестовый набор: -19.805909411103695

```
# Сделаю обучение (KNeighborsRegressor) влияние температуры на рабочее время
KNeighbors = KNeighborsRegressor(n_neighbors=6)
KNeighbors.fit(X_train[['температура-днем', 'температура-ночью']], y_train[['рабочее-время']])
print(f"Тренировочный набор: {KNeighbors.score(X_train[['температура-днем', 'температура-ночью']], y_train[['рабочее-время']])*100}")
print(f"Тестовый набор: {KNeighbors.score(X_test[['температура-днем', 'температура-ночью']], y_test[['рабочее-время']])*100}")
```

Тренировочный набор - 16.626827484781614

Тестовый набор - -9.418121337384001

```
# Сделаю обучение (KNeighborsRegressor) влияние температуры на динамический уровень
KNeighbors = KNeighborsRegressor(n_neighbors=6)
KNeighbors.fit(X_train[['температура-днем', 'температура-ночью']], y_train[['динамический-уровень-(м)']])
print(f"Тренировочный набор: {KNeighbors.score(X_train[['температура-днем', 'температура-ночью']], y_train[['динамический-уровень-(м)']])*100}")
print(f"Тестовый набор: {KNeighbors.score(X_test[['температура-днем', 'температура-ночью']], y_test[['динамический-уровень-(м)']])*100}")
```

Тренировочный набор - 16.71026765166179

Тестовый набор - -13.303811550795386

```
# Сделаю обучение (KNeighborsRegressor) влияние температуры на пластовое давление
KNeighbors = KNeighborsRegressor(n_neighbors=6)
KNeighbors.fit(X_train[['температура-днем', 'температура-ночью']], y_train[['пластовое-давление-(атм)']])
print(f"Тренировочный набор: {KNeighbors.score(X_train[['температура-днем', 'температура-ночью']], y_train[['пластовое-давление-(атм)']])*100}")
print(f"Тестовый набор: {KNeighbors.score(X_test[['температура-днем', 'температура-ночью']], y_test[['пластовое-давление-(атм)']])*100}")
```

Тренировочный набор - 17.569929148467523

Тестовый набор - -19.35995902961767

Запуск и оценка качества моделей машинного обучения

Как видно, показатели получены плохие. Исходя из логического мышления предметной области. Искусственное поднятие пластового давления зависит от закачки воды. Естественное поднятие пластового давления зависит от водоносного слоя, который подпирает нефть, что приводит к увеличению давления. Сам же водоносный слой зависит от осадков, чем больше осадков, тем выше водоносный слой.

Произведена группировка данных по месяцам и годам (среднее), а также обучение исходя из зависимой переменной:
 y = пластовое давление (атм).

```
# Случайный лес
RandomForest = RandomForestRegressor(n_estimators=100)
RandomForest.fit(X_train[X_values], y_train[y_values])
print(f"Тренировочный набор - {RandomForest.score(X_train[X_values], y_train[y_values])*100}")
print(f"Тестовый набор - {RandomForest.score(X_test[X_values], y_test[y_values])*100}")
```

Тренировочный набор - 91.30609429516709
Тестовый набор - 53.437838027831084

```
# Bagging
Bagging = BaggingRegressor()
Bagging.fit(X_train[X_values], y_train[y_values])
print(f"Тренировочный набор - {Bagging.score(X_train[X_values], y_train[y_values])*100}")
print(f"Тестовый набор - {Bagging.score(X_test[X_values], y_test[y_values])*100}")
```

Тренировочный набор - 85.7371119844786
Тестовый набор - 45.253817042952946

```
# AdaBoost
AdaBoost = AdaBoostRegressor(n_estimators=100)
AdaBoost.fit(X_train[X_values], y_train[y_values])
print(f"Тренировочный набор - {AdaBoost.score(X_train[X_values], y_train[y_values])*100}")
print(f"Тестовый набор - {AdaBoost.score(X_test[X_values], y_test[y_values])*100}")
```

Тренировочный набор - 96.43614361731873
Тестовый набор - 58.07004937014552

```
# CatBoost
CatBoost = CatBoostRegressor(n_estimators=100)
CatBoost.fit(X_train[X_values], y_train[y_values])
print(f"Тренировочный набор - {CatBoost.score(X_train[X_values], y_train[y_values])*100}")
print(f"Тестовый набор - {CatBoost.score(X_test[X_values], y_test[y_values])*100}")
```

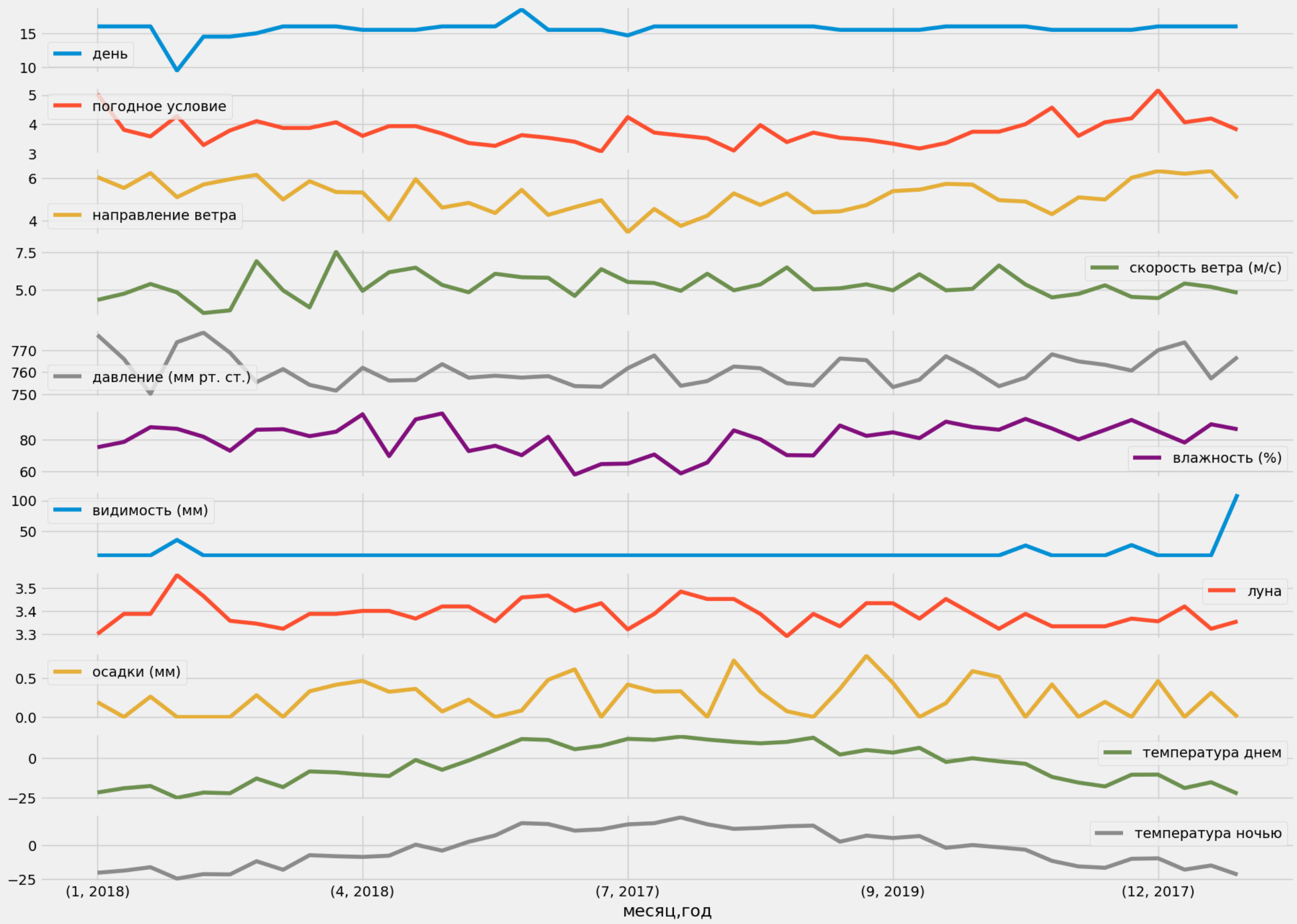
Тренировочный набор - 99.97442511995243
Тестовый набор - 38.59452456007237

```
# XGBoost
XGBoost = XGBRegressor(n_estimators=100)
XGBoost.fit(X_train[X_values], y_train[y_values])
print(f"Тренировочный набор - {XGBoost.score(X_train[X_values], y_train[y_values])*100}")
print(f"Тестовый набор - {XGBoost.score(X_test[X_values], y_test[y_values])*100}")
```

Тренировочный набор - 99.99999986367433
Тестовый набор - 31.968238098896464

Feature engineering

На данный момент, лучшие результаты показывают алгоритмы: RandomForest и AdaBoost. Частотность данных:



Feature engineering

Извлечение признаков

```
# Минимальный параметр
settings_min = settings.MinimalFCParameters()
settings_min
```

Python

```
{'sum_values': None, 'median': None, 'mean': None, 'length': None, 'standard_deviation': None, 'variance': None, 'root_mean_square': None, 'maximum': None, 'a
```

⌵ ⏪ ⏩ 🗑

```
extracted_features = extract_features(X_drop, column_id="id", impute_function=impute, default_fc_parameters=settings_min)
```

Python

Feature Extraction: 100% 13150/13150 [00:05<00:00, 2628.42it/s]

```
extracted_features
```

Python

	погодное условие_sum_values	погодное условие_median	погодное условие_mean	погодное условие_length	погодное условие_standard_deviation	погодное условие_variance	погодное условие_root_mean_square	погодное условие_p
0	4.0	4.0	4.0	1.0	0.0	0.0	4.0	
1	6.0	6.0	6.0	1.0	0.0	0.0	6.0	
2	2.0	2.0	2.0	1.0	0.0	0.0	2.0	
3	4.0	4.0	4.0	1.0	0.0	0.0	4.0	
4	4.0	4.0	4.0	1.0	0.0	0.0	4.0	
...	Активация Windows	...	
1310	5.0	5.0	5.0	1.0	0.0	Чтобы активировать Windows, перейдите в раздел "Параметры".	5.0	

Feature engineering

Обучение модели и получение оценки качества (без группировки по месяцам и годам и с группировкой).

```
# БЕЗ ГРУППИРОВКИ
# разделим данные на тестовую (20%) и тренировочную выборки (80%)
X_train_ext, X_test_ext, y_train_ext, y_test_ext = train_test_split(extracted_features, y_original_1, test_size=0.2, random_state=1024, shuffle=True)
X_test_ext, X_val_ext, y_test_ext, y_val_ext = train_test_split(extracted_features, y_original_1, test_size=0.5, random_state=32, shuffle=True)

# X_values = [
# ... 'погодное условие',
# ... 'направление ветра',
# ... 'скорость ветра (м/с)',
# ... 'давление (мм.рт.ст.)',
# ... 'влажность (%)',
# ... 'видимость (м)',
# ... 'луна',
# ... 'осадки (мм)',
# ... 'температура днем',
# ... 'температура ночью'
# ]
y_values = [
    'пластовое давление (атм)'
]

# Случайный лес
forest = RandomForestRegressor(n_estimators=100)
forest.fit(X_train_ext, y_train_ext[y_values])
print(f"Тренировочный набор - {forest.score(X_train_ext, y_train_ext[y_values])*100}")
print(f"Тестовый набор - {forest.score(X_test_ext, y_test_ext[y_values])*100}")
```

Тренировочный набор - 91.25749337651345
Тестовый набор - 79.65666408586509

```
# С ГРУППИРОВКОЙ
X_drop = X
X_drop['id'] = X_drop.index
X_drop = X_drop.drop(['день'], axis=1)

extracted_features = extract_features(X_drop, column_id="id", impute_function=impute, default_fc_parameters=settings_min)

# разделим данные на тестовую (20%) и тренировочную выборки (80%)
X_train_ext, X_test_ext, y_train_ext, y_test_ext = train_test_split(extracted_features, y, test_size=0.2, random_state=1024, shuffle=True)
X_test_ext, X_val_ext, y_test_ext, y_val_ext = train_test_split(extracted_features, y, test_size=0.5, random_state=32, shuffle=True)

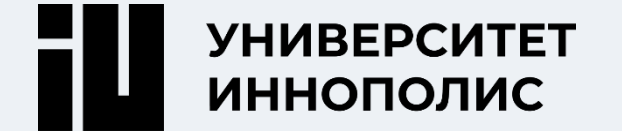
# X_values = [
# ... 'погодное условие',
# ... 'направление ветра',
# ... 'скорость ветра (м/с)',
# ... 'давление (мм.рт.ст.)',
# ... 'влажность (%)',
# ... 'видимость (м)',
# ... 'луна',
# ... 'осадки (мм)',
# ... 'температура днем',
# ... 'температура ночью'
# ]
y_values = [
    'пластовое давление (атм)'
]

X_train_original_2 = X_train_ext
y_train_original_2 = y_train_ext
X_test_original_2 = X_test_ext
y_test_original_2 = y_test_ext

# Случайный лес
forest = RandomForestRegressor(n_estimators=100)
forest.fit(X_train_ext, y_train_ext[y_values])
print(f"Тренировочный набор - {forest.score(X_train_ext, y_train_ext[y_values])*100}")
print(f"Тестовый набор - {forest.score(X_test_ext, y_test_ext[y_values])*100}")
```

Тренировочный набор - 91.01880118964779
Тестовый набор - 86.77089427483158

Отчет по результатам



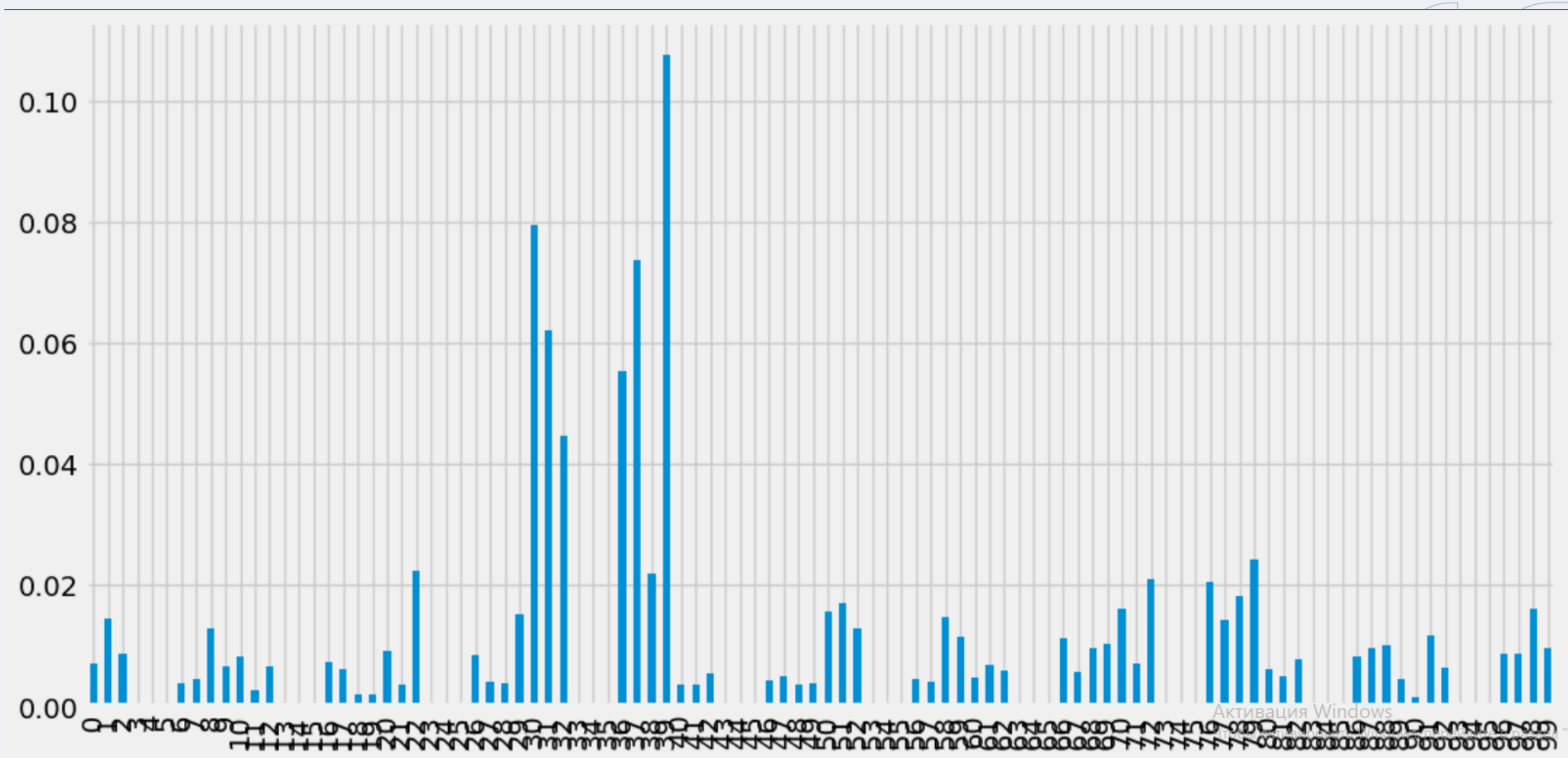
Получил показатели благодаря изучению предметной области, группировки данных по году и месяцу, а также применению Feature engineering.

Оценка качества:

Тренировочный набор - 91.01880118964779

Тестовый набор - 86.77089427483158

Вес каждого фактора в итоговой модели:



Отчет по результатам

Сохранение и загрузка итоговой модели и предсказание на тестовых данных:

```
# Сохраню модели
joblib.dump(RandomForest, 'model.pkl') # без Feature engineering
joblib.dump(forest, 'model_fe.pkl') # с Feature engineering

['model_fe.pkl']

# Загружу модели
RandomForest = joblib.load('model.pkl') # без Feature engineering
forest = joblib.load('model_fe.pkl') # с Feature engineering
```



```
# Использую загруженные модели для предсказания
# без Feature engineering
# Тренировочный набор -- 91.07214646869724
# Тестовый набор -- 53.455281999856
X_test_pred = X_test_original_1.drop(['день'], axis=1)
y_pred = RandomForest.predict(X_test_pred)
y_pred
```

```
array([137.85267773, 111.84754542, 124.09439488, 140.23862796])
```

```
# Использую загруженные модели для предсказания
# с Feature engineering
# Тренировочный набор -- 91.25531053440886
# Тестовый набор -- 88.85372762794663
y_pred = forest.predict(X_test_original_2)
y_pred
```

```
array([104.79806452, 110.90151489, 142.45441022, 109.62976661,
       121.87562366, 138.19558011, 127.88614009, 142.98012366,
       114.82838765, 128.50568049, 131.09269124, 135.64498065,
       120.30869596, 137.33242396, 114.58752725, 108.79166052,
       134.35352358, 125.64131797, 108.85301075, 143.73223328,
       129.42535522, 111.19276559])
```


Отчет по результатам



Предсказание на собственных данных:

```
#-Пример ·ввода ·собственных ·данных
#-без ·Feature ·engineering
y_pred = RandomForest.predict([[
    -1, #- погодное ·условие
    1, #- направление ·ветра
    -1, #- скорость ·ветра ·(м/с)
    -1, #- давление ·(мм ·рт. ·ст.)
    -1, #- влажность ·(%)
    -1, #- видимость ·(мм)
    -1, #- луна
    -1, #- осадки ·(мм)
    1, #- температура ·днем
    -1, #- температура ·ночью
]])
y_pred
```

```
c:\Python311\Lib\site-packages\sklearn\base.py:465: UserWarning:
```

X does not have valid feature names, but RandomForestRegressor was fitted with feature names

```
array([120.63871613])
```

[illegible]

```
c:\Python311\Lib\site-packages\sklearn\base.py:465: UserWarning:
```

X does not have valid feature names, but RandomForestRegressor was fitted with feature names

```
array([119.99001075])
```

Выводы и рекомендации

Разработка модели прогнозирования пластового давления влияет на увеличение давления нефти, что может привести к поломке оборудования нефтяной скважины № 807. Такая модель представляет собой комплексный подход, требующий анализа множества переменных.

Так как село Самбург находится в 24 км от скважины №807, результаты могут быть с отклонениями. Для более точного анализа и построения модели, необходимо снимать погодные показания непосредственно рядом с нефтяной скважиной.

В дальнейшем планируется разработка веб-интерфейса с загрузкой готовой модели. Такое приложение позволяет подавать данные непосредственной через веб-формы и получать результаты в базу данных типа Postgres.

Следует отметить, что полученные выводы и рекомендации могут служить отправной точкой для последующих исследований всех нефтяных скважин, а также в области прогнозирования устойчивой работы нефтяной отрасли.

Список литературы

Параметры эксплуатации нефтяной скважины № 807 (2013-2021):
<https://www.kaggle.com/datasets/ruslanzalevskikh/oil-well>

Еремин Н. А., Селенгинский Д. А. О возможностях применения методов искусственного интеллекта в решении нефтегазовых задач

Дмитриевский А. Н., Столяров В. Е., Еремин Н. А. - Роль информации в применении технологий искусственного интеллекта при строительстве скважин для нефтегазовых месторождений

Дмитриевский А. Н., Столяров В. Е., Еремин Н. А. - Актуальные вопросы и индикаторы цифровой трансформации на заключительной стадии нефтегазодобычи промыслов

Дмитриевский А.Н. [и др.] - Анализ рисков при использовании технологий искусственного интеллекта в нефтегазодобывающем комплексе



ИНСТИТУТ
ДОПОЛНИТЕЛЬНОГО
ОБРАЗОВАНИЯ
УНИВЕРСИТЕТА ИННОПОЛИС

Спасибо за внимание

Сорокин Максим Евгеньевич
sotge@hotmail.com