Описание проекта

Допустим, вы работаете в добывающей компании «ГлавРосГосНефть». Нужно решить, где бурить новую скважину.

Вам предоставлены пробы нефти в трёх регионах: в каждом 10 000 месторождений, где измерили качество нефти и объём её запасов. Постройте модель машинного обучения, которая поможет определить регион, где добыча принесёт наибольшую прибыль. Проанализируйте возможную прибыль и риски техникой *Bootstrap.*

Шаги для выбора локации:

- В избранном регионе ищут месторождения, для каждого определяют значения признаков;
- Строят модель и оценивают объём запасов;
- Выбирают месторождения с самым высокими оценками значений. Количество месторождений зависит от бюджета компании и стоимости разработки одной скважины;
- Прибыль равна суммарной прибыли отобранных месторождений.

Условия задачи:

Для обучения модели подходит только линейная регрессия (остальные — недостаточно предсказуемые).

При разведке региона исследуют 500 точек, из которых с помощью машинного обучения выбирают 200 лучших для разработки.

Бюджет на разработку скважин в регионе — 10 млрд рублей.

При нынешних ценах один баррель сырья приносит 450 рублей дохода. Доход с каждой единицы продукта составляет 450 тыс. рублей, поскольку объём указан в тысячах баррелей.

После оценки рисков нужно оставить лишь те регионы, в которых вероятность убытков меньше 2.5%. Среди них выбирают регион с наибольшей средней прибылью.

Данные синтетические: детали контрактов и характеристики месторождений не разглашаются.

Признаки в датасетах:

- id — уникальный идентификатор скважины;
- f0, f1, f2 — три признака точек (неважно, что они означают, но сами признаки значимы);
- product — объём запасов в скважине (тыс. баррелей).

План

1. Подготовка данных

2. Обучение и проверка моделей

- Разделение данных на обучающую и валидационную выборки в соотношении 75:25.
- Обучиние моделей
- Предсказания на валидационной выборке
- Получение среднего запаса предсказанного сырья и RMSE модели.

3. Подготовка к расчёту прибыли

- Рассчитёт достаточного объёма сырья для безубыточной разработки новой скважины.
- Сравнение полученного объёма сырья со средним запасом в каждом регионе.

4. Расчёт прибыли

- Выбор скважины с максимальными значениями предсказаний.
- Расчёт прибыли для полученного объёма сырья.

5. Риски и прибыль по регионам

- Применение технику Bootstrap с 1000 выборок, для нахождения распределение прибыли.
- Нахождение средней прибыли, 95%-го доверительного интервала и рисков убытков.

B [1]:
```
import pandas as pd
import numpy as np

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error as mse,\
                            mean_absolute_error as mae,\
                            r2_score as r2
from sklearn.preprocessing import StandardScaler

import plotly.express as px

import os
import urllib

pd.options.display.float_format = "{:,.3f}".format
state = np.random.RandomState(1)
```

Подготовка данных

Данные геологоразведки трёх регионов находятся в файлах

B [2]:
```
data_names_urls = [
    ['datasets/geo_data_0.csv','https://code.s3.yandex.net/datasets/geo_data_0.cs
    ['datasets/geo_data_1.csv','https://code.s3.yandex.net/datasets/geo_data_1.cs
    ['datasets/geo_data_2.csv','https://code.s3.yandex.net/datasets/geo_data_2.cs
]

def get_data(data):
    if not os.path.exists(data[0][0].split('/')[0]):
        os.makedirs(data[0][0].split('/')[0])
    for name, url in data:
        if not os.path.exists(name):
            _ = urllib.request.urlretrieve(url, name)

get_data(data_names_urls)
```

B [3]:
```
data0 = pd.read_csv(data_names_urls[0][0])
data1 = pd.read_csv(data_names_urls[1][0])
data2 = pd.read_csv(data_names_urls[2][0])
```

B [4]:
```
for df in (data0, data1, data2):
    display(df.describe().T)
```

	count	mean	std	min	25%	50%	75%	max
f0	100,000.000	0.500	0.872	-1.409	-0.073	0.502	1.074	2.362
f1	100,000.000	0.250	0.504	-0.848	-0.201	0.250	0.701	1.344
f2	100,000.000	2.503	3.248	-12.088	0.288	2.516	4.715	16.004
product	100,000.000	92.500	44.289	0.000	56.498	91.850	128.564	185.364

	count	mean	std	min	25%	50%	75%	max
f0	100,000.000	1.141	8.966	-31.610	-6.299	1.153	8.621	29.422
f1	100,000.000	-4.797	5.120	-26.359	-8.268	-4.813	-1.333	18.734
f2	100,000.000	2.495	1.704	-0.018	1.000	2.011	4.000	5.020
product	100,000.000	68.825	45.944	0.000	26.953	57.086	107.813	137.945

	count	mean	std	min	25%	50%	75%	max
f0	100,000.000	0.002	1.732	-8.760	-1.162	0.009	1.159	7.238
f1	100,000.000	-0.002	1.730	-7.084	-1.175	-0.009	1.164	7.845
f2	100,000.000	2.495	3.473	-11.970	0.130	2.484	4.859	16.739
product	100,000.000	95.000	44.750	0.000	59.450	94.926	130.595	190.030

B [5]:
```
features_columns = ['f0', 'f1', 'f2']
if not any(map(lambda x: (x[features_columns].corr() > 5).sum().sum() > 3,\
    (data0, data1, data2))):
    print("Thers's no correaltion in input datasets")
```

```
Thers's no correaltion in input datasets
```

B [6]:
```
data = pd.concat([data0, data1, data2], keys=[0, 1, 2])
data = data.reset_index().drop(['level_1', 'id'], axis=1)
display(data.head(3))
data.columns = ['region', 'f0', 'f1', 'f2', 'product']
```

	level_0	f0	f1	f2	product
0	0	0.706	-0.498	1.221	105.280
1	0	1.335	-0.340	4.365	73.038
2	0	1.023	0.152	1.420	85.266

Даже смешав признаки для всех трёх регионов видим сильные различия между ними

B [7]:
```
scaler = StandardScaler()
scaler.fit(data[features_columns])
data.loc[:, features_columns] = scaler.transform(data[features_columns])
```

B [8]:
```
data.f0 = data.f0.astype(np.float32())
data.f1 = data.f1.astype(np.float32())
data.f2 = data.f2.astype(np.float32())
data.region = data.region.astype(np.uint8())
```

B [9]:
```
data0 = data[data.region == 0].drop(['region'], axis=1)
data1 = data[data.region == 1].drop(['region'], axis=1)
data2 = data[data.region == 2].drop(['region'], axis=1)
```

B [10]:
```
data0.head(3)
```

Out[10]:

	f0	f1	f2	product
0	0.030	0.261	-0.438	105.280
1	0.148	0.302	0.640	73.038
2	0.089	0.428	-0.369	85.266

Вывод:

Масштабировали данные для коррекного обучения регрессии. Изменили типы данных (для оптимального использования памяти) Удалили ненужные для модели признаки (`id`). Проверили признаки на линейную зависимость (корреляцию). Пропусков в данных нет. Оставили 3 отдельных датасета для дальнейшего построения моделей (требуется по условию)

2. Обучение и проверка моделей

B [11]:
```
features = [df[features_columns] for df in (data0, data1, data2)]
targets = [df['product'] for df in (data0, data1, data2)]
```

B [12]:
```
features_trn, features_vld, target_trn, target_vld = [], [], [], []
for i in range(3):
    out = train_test_split(features[i], targets[i], test_size=.25, random_state=s
    features_trn.append(out[0])
    features_vld.append(out[1])
    target_trn.append(out[2])
    target_vld.append(out[3])
```

B [13]:
```
models = [LinearRegression(), LinearRegression(), LinearRegression()]
predicted_trn, predicted_vld, rmse_trn, rmse_vld, mae_vld, r2_vld = [], [], [], [
for i in range(3):
    models[i].fit(features_trn[i], target_trn[i])
    predicted_trn.append(pd.Series(models[i].predict(features_trn[i])))
    predicted_vld.append(pd.Series(models[i].predict(features_vld[i])))
    rmse_trn.append(mse(target_trn[i], predicted_trn[i]) ** .5)
    rmse_vld.append(mse(target_vld[i], predicted_vld[i]) ** .5)
    mae_vld.append(mae(target_vld[i], predicted_vld[i]))
    r2_vld.append(r2(target_vld[i], predicted_vld[i]))
```

B [14]:
```
result = {'mean_predicted': [], 'RMSE_train' :[], 'RMSE' :[], 'MAE' :[], 'R2' :[]
for i in range(3):
    result['mean_predicted'].append(predicted_vld[i].mean())
    result['RMSE_train'].append(rmse_trn[i])
    result['RMSE'].append(rmse_vld[i])
    result['MAE'].append(mae_vld[i])
    result['R2'].append(r2_vld[i])
result_df = pd.DataFrame(result, index=[f'Reg {i + 1}' for i in range(3)])
result_df
```

Out[14]:

	mean_predicted	RMSE_train	RMSE	MAE	R2
Reg 1	92.493	37.675	37.743	31.080	0.277
Reg 2	69.036	0.890	0.893	0.719	1.000
Reg 3	95.018	40.140	39.799	32.581	0.204

Вывод:

Обучили 3 модели и предсказали для 3-ёх регионов значения целевого признака. На таблице выше убедились, что переобучения нет. Лучше всех обучилась модель 2-го региона.

Подготовка к расчёту прибыли

B [15]:
```
budget = 1e7 # тыс.руб
price_per_bar = 450 # тыс.руб
amount_wells = 500
amount_best_wells = 200
bars_per_well = int(np.ceil(budget / (price_per_bar * amount_best_wells)))
```

В [16]:
```
print('Для безубыточной разработки новой скважины в среднем '
      f'необходимо {bars_per_well} тыс. баррелей')
```

```
Для безубыточной разработки новой скважины в среднем необходимо 112 тыс. баррел
ей
```

В [17]:
```
data.groupby('region').agg({'product': [np.mean]}).T
```

Out[17]:

	region	0	1	2
product	**mean**	92.500	68.825	95.000

Вывод:

Определили, что для безубыточной разработки новой скважины в среднем необходимо 112 тыс. баррелей . В свою очередь средние значения не всех регионов не превышают 95 тыс.

4. Расчёт прибыли

В [18]:
```
def get_revenue_from_region(actual, predicted=None):
    if predicted is not None:
        for_result = predicted.sort_values(ascending=False)
        return sum(actual[for_result.index][:amount_best_wells]) * price_per_bar
    return sum(actual.sort_values(ascending=False)[:amount_best_wells]) * price_p

def get_B_from_k(ipt):
    return round(ipt / 1e6, 3)
```

В [19]:
```
income_data = []
for i in range(3):
    actual_product = data[data.region == i]['product'].reset_index(drop=True)
    revenue_by_pred = get_B_from_k(get_revenue_from_region(actual_product, predic
    revenue_by_real = get_B_from_k(get_revenue_from_region(actual_product))
    income_data.append((int(i + 1), revenue_by_pred, revenue_by_real))
pd.DataFrame(income_data, columns=['Region', 'Total income (predict)', 'Total inc
```

Out[19]:

	Region	Total income (predict)	Total income (real)
0	1	8.670	16.635
1	2	6.373	12.415
2	3	8.352	17.060

Вывод:

Нашли доходы для регионов по предсказаням модели (ориентируясь на предсказанятия, на суммарный объём сырья брали по реальным показателям) и фактические (если находить скважины с максимальным объёмом) - чтобы увидеть ошибки моделей не по метрикам, а на

конкретном примере

5. [Риски и прибыль по регионам]

- Примените технику Bootstrap с 1000 выборок, чтобы найти распределение прибыли.
- Найдите среднюю прибыль, 95%-й доверительный интервал и риск убытков. Убыток — это отрицательная прибыль.

B [22]:
```
%%time
result = {'mean': [], 'min': [], 'max': [], 'risk %': []}
for reg in range(3):
    bootstrap_result = []
    sub_actual_ = target_vld[reg].reset_index(drop=True)
    predicted = predicted_vld[reg]
    for i in range(1000):
        sub_actual = sub_actual_.sample(random_state=state, replace=True, n=500)
        sub_pred = pd.Series(predicted[sub_actual.index])
        bootstrap_result.append(get_revenue_from_region(sub_actual, sub_pred) - b
    bootstrap_result = pd.Series(bootstrap_result)
    result['mean'].append(int(bootstrap_result.mean()))
    result['min'].append(int(bootstrap_result.quantile(.025)))
    result['max'].append(int(bootstrap_result.quantile(.975)))
    result['risk %'].append(round((bootstrap_result < 0).sum() / 1000 * 100, 3))

get_res = pd.DataFrame(result, index=[f'Reg {i + 1}' for i in range(3)])
get_res
```

Итоговый вывод

Найдя точку безубыточности (`для безубыточной разработки новой скважины в среднем необходимо 112 тыс. баррелей`) и средние показатели по регионам стало ясно, что в среднем регионы не удовлетворяют желания не потерять денег на разработку месторождений в них.

По результатам обучения моделей (`LinearRegression`), а также взяв во внимание тот факт, что для проверки моделей (решения поставленной задачи нахождения как минимум безубыточных месторождений и вероятности этой самой безубыточности) имеются валидационные данные, но неизвестно к какому распределению эта часть данных ГС относится (ГС распределена номрмально, но низвестно о тестовой/валидационной выборке) - поэтому использовать t-критерий нельзя, поэтому воспользовались техникой `Bootstrap` , сгенерировав для каждого региона 1000 подвыборок (результаты представлены на таблице ниже). На основании получившихся данных можно сделать преположение об целесообраности ведения разработки месторождений в конкретном регионе (границы 95-го доверительного интервала и процент показатель безубыточности)

B [21]:
```
get_res
```

Out[21]:

	mean	min	max	risk %
Reg 1	461391	-64744	1011105	5.000
Reg 2	511588	58840	941519	0.900
Reg 3	427235	-80230	951400	6.000

Из которой видно, что следует разрабатывать месторождения во 2-ом регионе: меньше всего рисков. С вероятностью 2.5% в данном регионе компания не столкнётся с убытками (используя обученную модель), но и сможет заработать на месторождениях (как минимум 93.095 руб)