Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования
Московский государственный технический университет имени Н.Э. Баумана

Лабораторная работа №4
«Анализ результатов проб нефти»
по курсу «Моделирование»

Выполнил:
студент группы ИУ9-82Б
Егоров Алексей

Москва, 2023

1. Цель работы

Целью данной работы является построение регрессионной модели для данных о пробах нефти с предварительной очисткой результатов наблюдения с использованием статистических методов для оценки прибыльности разработки месторождений.

2. Постановка задачи

Предоставлены пробы нефти в трёх регионах: в каждом 100 000 месторождений, где измерили качество нефти и объём её запасов. Необходимо построить модель, которая поможет определить регион, где добыча принесёт наибольшую прибыль. Шаги для выбора локации:

1. в избранном регионе ищут месторождения, для каждого определяют значения признаков;
2. строят модель и оценивают объём запасов;
3. выбирают месторождения с самым высокими оценками значений, количество месторождений зависит от бюджета компании и стоимости разработки одной скважины;
4. прибыль равна суммарной прибыли отобранных месторождений.

Предоставлены три набора данных, соответствующие трем разным исследуемым локациям, в них 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). Необходимо провести предвварительную обработку данных. Выявить выбросы (если есть), рассчитать квартили, интерквартильный размах, выборочную дисперсию для всех столбцов каждого набора данных. Определить корреляцию целевого признака (product) с зависимыми признаками для каждого набора данных.

3. Теоретические сведения

Представленные наборы данных представляют собой функциональную зависимость. Функциональная зависимость – это закон, ставящий в соответствие каждому действительному числу x из множества X действительное число y из множества Y.

Стохастическая (случайная) зависимость между величинами X и Y - это зависимость, при которой строго определенному значению величины X может соответствовать множество значений величины Y. Зависимость носит вероятностный характер, то есть случайная величина Y принимает разные значения с некоторой вероятностью. Функциональная зависимость является предельным случаем стохастической - при наиболее тесной связи.

При оценивании стохастической зависимости различают корреляцию (существует ли взаимосвязь между переменными) и регрессию (какая зависимость).

В математической статистике регрессионным анализом называют совокупность приемов для установления связей между независимой переменной Y и одной или несколькими переменными X_1,X_2,...,X_m. Регрессия - условное математическое ожидание случайной переменной Y при условии, что другая условная переменная X приняла значение x. Моделью линейной регрессии является модель, в которой теоретическое среднее значение наблюдаемой величины y является линейной комбинацией независимых переменных:

$$\mu = \beta_0 + \beta_1 x_1 + ... + \beta_k x_k$$

Множители β_0,β_1,...,β_k представляют собой параметры модели, значения которых должны быть установлены. Они называются коэффициентами регрессии, а β_0 называется свободным или постоянным членом. Модель, более чем с одной переменной x называется моделью множественной регрессии.

При статистическом анализе данных используются следующие понятия: квантиль, квартиль и интерквартильный размах.

Для эмпирического распределения α-квантиль (x_α) задается следующим образом:

1. составляется вариационный ряд значений $V_0 \leq V_1 \leq ... \leq V_{N-1}$ (выборка имеет объём N), а также $V_N = V_{N-1}$;
2. вычисляется величина $\lfloor \alpha(N-1) \rfloor$
3. сравниваются K и αN
   — если $K+1 < \alpha N$, то $x_\alpha = V_{K+1}$
   — если $K+1 = \alpha N$, то $x_\alpha = \frac{V_K + V_{K+1}}{2}$
   — если $K+1 > \alpha N$, то $x_\alpha = V_K$

— 0.25-квантиль называется первым (или нижним) квартилем;

— 0.5-квантиль называется медианой или вторым квартилем;

— 0.75-квантиль называется третьим (или верхним квартилем).

Интерквартильным размахом называется разность между третьим и первым квартилями. Интерквартильный размах является характеристикой распределения величины и является аналогом дисперсии.

Выброс в статистике - результат измерения, выделяющийся из общей выборки. Простейшие способы определения выбросов основаны на интерквартильном размахе - например всё, что не попадает в следующий диапазон считается выбросом:

$$[(x_{0.25} - 1.5(x_{0.75} - x_{0.25})),(x_{0.75} + 1.5(x_{0.75} - x_{0.25}))]$$

Выборочное среднее:

$$\overline{X} = \frac{1}{n}\sum_{i=1}^{N} X_i$$

Выборочная дисперсия:

$$S_n^2 = \frac{1}{n}\sum_{i=1}^{N}(X_i - \overline{X})^2$$

Коэффициент корреляции Пирсона:

$$r_{XY} = \frac{\sum_{i=1}^{N}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{N}(X_i - \overline{X})^2 \sum_{i=1}^{N}(Y_i - \overline{Y})^2}}$$

4. Реализация

```python
import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
import sklearn.pipeline as pipe
first = pd.read_csv('document.txt', sep=r'\s+')
first = first.drop('ind', axis=1)
# first = first[first['product']>0]
second = pd.read_csv('1.txt', sep=r'\s+')
second = second.drop('ind', axis=1)
third = pd.read_csv('2.txt', sep=r'\s+')
third = third.drop('ind', axis=1)
def quant(data: pd.DataFrame, field: str, alpha: float):
    d = data.copy()
    d = d.sort_values(field)
    N = len(d)
    K = int(alpha*(N-1))
    d = pd.concat([d, d.tail(1)])
    # print(float(d.iloc[[K+1]]['product']))
    if K + 1 < alpha*N:
      return float(d.iloc[[K+1]][field])
    elif K+1 == alpha*N:
      return (float(d.iloc[[K]][field])+float(d.iloc[[K+1]][field]))/2
    else:
       return float(d.iloc[[K]][field])

def correl(a: list, b: list) -> float:
    mean_a = sum(a)/len(a)
    mean_b = sum(b)/len(b)
    desp_a = sum([(i - mean_a) ** 2 for i in a])
    desp_b = sum([(i - mean_b) ** 2 for i in b])
    return sum([(a[i]-mean_a)*(b[i]-mean_b) for i in range(len(a))])/(desp_a*desp_b)**0.5
def get_outliers(data):
  x25 = quant(data, 'product', 0.25)
  x75 = quant(data, 'product', 0.75)
  a = x25-1.5*(x75-x25)
```

```
  b = x75+1.5*(x75-x25)
  return data[ (data['product'] < a) | (data['product'] > b)]
def stat(data: pd.DataFrame):
    print('квантиль 0.25: ', quant(data, 'product', 0.25))
    print('квантиль 0.5: ', quant(data, 'product', 0.5))
    print('квантиль 0.75: ', quant(data, 'product', 0.75))
    print('интерквартильный размах: ', quant(data, 'product', 0.75) - quant(data, 'product', 0.25))
    product = data['product'].to_list()
    f0 = data['f0'].to_list()
    f1 = data['f1'].to_list()
    f2 = data['f2'].to_list()
    mean_p = sum(product)/len(product)
    desp = sum([(i - mean_p) ** 2 for i in product])
    print('выборочная дисперсия: ', 1/len(product)*desp)
    print('cov product f0: ', correl(product, f0))
    print('cov product f1: ', correl(product, f1))
    print('cov product f2: ', correl(product, f2))
    print('cov f0 f2: ', correl(f0, f2))
    print('cov f0 f1: ', correl(f0, f1))
    print('cov f1 f2: ', correl(f1, f2))
#Небольшая процедура для предварительного анализа данных
def define_dataset(df):
      print(df.shape)
      print(df.info())
      print(df.head(40))
      print(df.describe())
      return df['id'].value_counts().head(20) #определение индексов-дубликатов Функция get_dummies
#Формируем наборы признаков и вектор целевого признака для всех трех локаций, одинакого исключая из списка признаков
#идентификатор (индекс) месторождения - он никак не может влиять на объем добытой нефти

features_1 = first.drop(['id','product'], axis=1)
features_ohe_1 = pd.get_dummies(features_1, drop_first=True)
target_1 = first['product']

features_2 = second.drop(['id','product'], axis=1)
```

```
features_ohe_2 = pd.get_dummies(features_2, drop_first=True)
target_2 = second['product']

features_3 = third.drop(['id','product'], axis=1)
features_ohe_3 = pd.get_dummies(features_3, drop_first=True)
target_3 = third['product']

#Разбиваем данные на обучающую и валидационную выборки в соотношении
↪ 75:25.
features_train_1, features_valid_1, target_train_1, target_valid_1 =
↪ train_test_split(features_ohe_1, target_1, test_size=0.25,
↪ random_state=12345)
features_train_2, features_valid_2, target_train_2, target_valid_2 =
↪ train_test_split(features_ohe_2, target_2, test_size=0.25,
↪ random_state=12345)
features_train_3, features_valid_3, target_train_3, target_valid_3 =
↪ train_test_split(features_ohe_3, target_3, test_size=0.25,
↪ random_state=12345)
model_1 = LinearRegression() #Применяем модель линейной регрессии
model_2 = LinearRegression()
model_3 = LinearRegression()

#Признаки кодируем во избежание доминирования одного из них
numeric = ['f0','f1','f2']
def scale(features_train, features_valid = None, numeric=['f0','f1','f2']):
      scaler = StandardScaler()
      scaler.fit(features_train_1[numeric])
      features_train[numeric] = scaler.transform(features_train[numeric])
      if features_valid is not None:
            features_valid[numeric] =
            ↪ scaler.transform(features_valid[numeric])
      return

#Обучаем модель и проводим предсказания на первой валидационной выборке.
def study(model: LinearRegression, features_train, features_valid,
↪ target_train, target_valid, number_location):
      model.fit(features_train,target_train) # обучите модель на первой
      ↪ тренировочной выборке
```

```
        predictions_valid = model.predict(features_valid) # получите
        ↪  предсказания модели на первой валидационной выборке
        #Выводим на печать средний запас предсказанного сырья и RMSE модели
        ↪  для первой локации.
        mse = mean_squared_error(target_valid, predictions_valid)

        # < извлекаем корень из MSE >
        result =  mse ** 0.5
        print("Средний запас предсказанного на валидационной выборке",
        ↪  number_location, "сырья:", predictions_valid.mean(), '(тыс.
        ↪  баррелей)')
        print("RMSE модели линейной регрессии на валидационной выборке",
        ↪  number_location, ":", result)
        return predictions_valid
model_1 = pipe.Pipeline([
    ('scaler', StandardScaler()),
    ('model', LinearRegression())
])

# study(model, features_train_1, features_valid_1, target_train_1,
↪  target_valid_1, 1)
study(model_1, features_train_1, features_valid_1, target_train_1,
↪  target_valid_1, 1)
model_2 = pipe.Pipeline([
    ('scaler', StandardScaler()),
    ('model', LinearRegression())
])
study(model_2, features_train_2, features_valid_2, target_train_2,
↪  target_valid_2, 2)
model_3 = pipe.Pipeline([
    ('scaler', StandardScaler()),
    ('model', LinearRegression())
])
study(model_3, features_train_3, features_valid_3, target_train_3,
↪  target_valid_3, 3)
r1 = pd.read_csv('place1.csv', sep=',')
r2 = pd.read_csv('place2.csv', sep=',')
r3 = pd.read_csv('place3.csv', sep=',')
COSTS = 500_000 #бюджет на разработку
```

```
INCOME = 450 #доход с одного бареля нефти
COUNT_REGION = 30 #количество исследуемых точек в одном регионе
BOREHOLES = 16 #количество выбранных скважин для разработки  месторождения
loss_threshold = COSTS/(BOREHOLES*INCOME) #Минимальная средняя продуктивность скважины для достижения порога окупаемости
region_threshold = round(BOREHOLES*loss_threshold,1) #Минимальная продуктивность 200 скважин региона для достижения порога окупаемости
print('Минимальная средняя продуктивность скважины для достижения порога окупаемости:', round(loss_threshold,1), '(тыс. баррелей)')
def calc_profit(data: pd.DataFrame, model: LinearRegression):
  d = data.copy()
  # d.sample()
  d_product = model.predict(d[numeric])
  # print(d_product)
  d['product'] = d_product
  d.sort_values(by='product', inplace=True)
  top_d = d.tail(BOREHOLES)
  # print(top_d)
  return top_d['product'].sum() * INCOME - COSTS

print("Прибыль в первой локации:", calc_profit(r1, model_1).round(),'тыс рублей')
print("Прибыль в первой локации:", calc_profit(r2, model_2).round(),'тыс рублей')
print("Прибыль в первой локации:", calc_profit(r3, model_3).round(),'тыс рублей')
import numpy as np
state = np.random.RandomState(12345) #обеспечим случайность формируемых выборок
def bootstrapped(data: pd.DataFrame, model: LinearRegression):
  values = []
  d = data.copy()
  for _ in range(1000):
    profit = calc_profit(d.sample(COUNT_REGION, replace=False, random_state=state), model)
    values.append(profit.round())

  values = pd.Series(values)
  mean = values.mean() #расчет средней прибыли
```

```
    print('Средняя прибыль, тыс руб.: {:,.2f}'.format(mean))

    lower = values.quantile(.025) #строим доверительный интервал
    upper = values.quantile(.975)

    print('95% доверительный интервал:', '{:,.2f}'.format(lower), ':', '{:,.2f}'.format(upper))
bootstrapped(r1, model_1)
bootstrapped(r2, model_2)
bootstrapped(r3, model_3)
```

5. Результат

С помощью статистического анализа была выявлена высокая (99%) корреляция признака f_2 и целевого признака $product$ для первого региона. Таким образом для данного региона при построении линейной регрессии можно исключить признаки f_0, f_1.

Статистический анализ показал наличие одного выброса данных во втором наборе. Однако удаление данной записи кажется не целесообразной из-за крайне небольшого размера выборки.

В исходной задаче требовалось выбрать 200 точек из 500 с бюджетом 10 млрд. руб. на 200 точек (50 млн. руб. на точку). В наборе данных содержалось 40 записей, поэтому для соблюдения пропорции, выбиралось 16 точек. Общий бюджет составил 800 млн. руб. (всё те же 50 млн. руб. на точку). Доход с одного барреля был оставлен таким же, как в условии —– 450 рублей.

Также из-за небольшого объёма набора данных, выбор разбиения на обучающую и валидационную выборку может оказывать существенное влияние на результат. С целью уменьшения данного фактора, использовалась технология bootstrap. Из 40 месторождений выбирались 30 и проводилось предсказание для уменьшенной выборки. В качестве конечного результата, было взято среднее от результатов на 1000 итерациях.

В таблице 1 приведены результат работы модели.

Таблица 1 — Результат анализа

	Регион 1	Регион 2	Регион 3
Средний запас нефти, тыс баррелей	101	79	99
RMSE на валидационной выборке	1.14	39.4	45.5
Средняя прибыль, тыс руб. (bootstrap)	213163	-36786	105355
95% доверительный интервал (bootstrap)	(78394; 326257)	(-88196; 7432)	(67463; 135708)

6. Вывод

В ходе данной лабораторной работы была построена регрессионная модель для данных о пробах нефти. Был произведен статистический анализ выбросов, корреляции признаков, в результате которого для первого региона была обнаружена корреляция между признаками f_2 и $product$. С помощью линейной регрессии была произведена оценка прибыли для каждого региона. В результате оценки 1 регион был выбран, как наиболее перспективный, на основе показателей средней прибыли и доверительного интервала. Также стоит отметить, что и для 3 региона 95% доверительный интервал показывает, что данный регион является безубыточным, чего нельзя сказать о 2 регионе, у которого нижняя граница интервала находится в отрицательной зоне.