Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Национальный исследовательский Томский политехнический Университет»



Инженерная школа ядерных технологий

Отделение математики и математической физики

01.03.02 Прикладная математика и информатика

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Лабораторная работа № 3

**Вариант - 6**

по дисциплине:

**Математическая статистика**

|  |  |  |  |
| --- | --- | --- | --- |
| **Выполнил:** |  | | |
| студент группы | 0В21 | Дзебан А.А. |  |
|  |  |  |  |
| **Проверил:** | Шинкеев М.Л. | | |
|  |  |  |  |
|  |  |  |

Томск – 2024

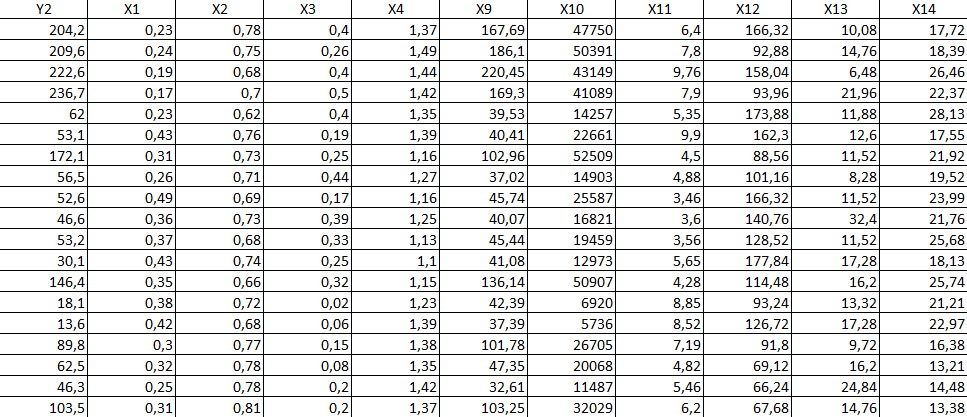
**Цели работы:**

Оценка параметров уравнения множественной линейной регрессии

**Ход работы:**

**Требуется,** для выбранных в соответствии с вариантом задания показателей (смотри таблицу 1), построить модель множественной линейной регрессии, связывающую результирующий показатель и определяющие факторы. Для устранения эффекта мультиколлинеарности, использовать метод пошагового включения факторов в уравнение регрессии (первым добавляем фактор, наиболее коррелирующий с откликом, затем последовательно добавляем факторы, наиболее коррелирующие с остатком). Указать уровни значимости коэффициентов модели и модели регрессии в целом, а также полученные оценки для коэффициента детерминации и остаточной дисперсии. Исследовать остатки модели на нормальность и однородность.

Выберем данные согласно варианту:



Действовать будем следующим образом:

1. Будем включать наиболее коррелирующий с остатком (откликом на шаге 1) фактором.
2. Временно добавляем фактор в модель.
3. Просчитываем её с добавленным коэффициентом
4. Если коэффициент не является значимым – отказываемся от его добавления
5. Если является значимым – добавляем его в финальную модель, обновляем остатки, пересчитываем модель со всеми коэффициентами.
6. Выполнять 1-5, пока либо модель перестанет быть значимой, либо коэффициентов не останется.

Выходные данные и важные метрики для моделей:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Добавл Фактор** | **Коэффициент** | **P-значение** | **R^2** | **Скорр R^2** | **F-стат** | **P-значение (F-стат)** | **Статистика ДУ** |
| X10 | 0,005170025 | 1,62444E-16 | 0,739525928 | 0,734418593 | 144,7968394 | 1,62444E-16 | 2,016737942 |
| X9 | 0,608048239 | 8,0761E-10 | 0,878420291 | 0,873557103 | 180,6264183 | 1,3229E-23 | 2,557630234 |
| X1 | -222,0450777 | 0,001080035 | 0,902455113 | 0,896482977 | 151,1109455 | 9,28405E-25 | 2,892406961 |

Итоговая модель имеет следующий вид:

Уровни значимости для всех коэффициентов итоговой модели:

|  |  |
| --- | --- |
| Const | 0.069 |
| X10 | 0.00 |
| X9 | 0.00 |
| X1 | 0.001 |

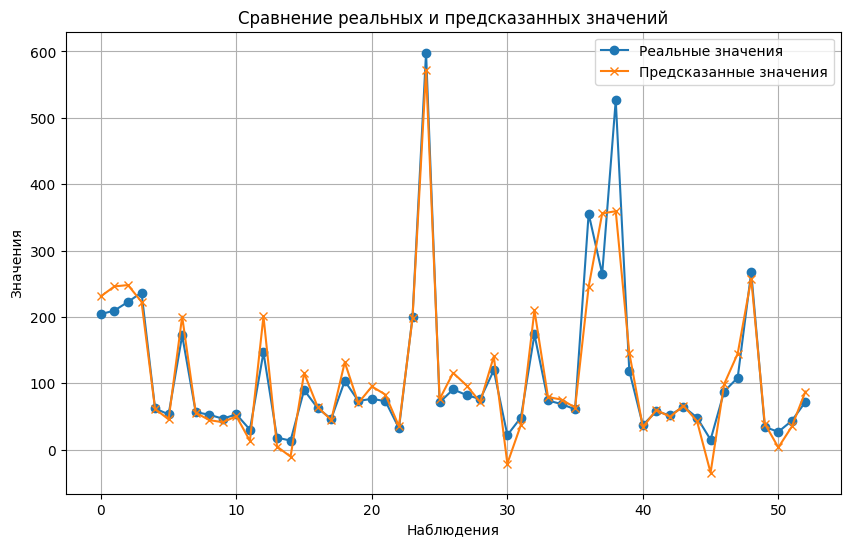
Уровень статистик:

F-statistic: 151.1

Prob (F-statistic): 9.28e-25

R-squared: 0.902

График предсказанных и реальных значений



Для проверки остатков на однородность воспользуемся тестом Шапиро-Уилка:

Тест Шапиро-Уилка: статистика = 0.7925, p-value = 0.0000

Остатки не распределены нормально (p-value ≤ 0.05).

Для проверки остатков на однородность воспользуемся тестом Бреуша-Пагана:

Тест Бреуша-Пагана: статистика = 15.9015, p-value = 0.0012

Остатки гетероскедастичны (p-value ≤ 0.05).

**Вывод:** хотя модель обладает высоким значением R^2 ~ 0.902, неоднородность и гетероскедастичность остатков может сигнализировать о ненадежности будущих прогнозов, для улучшения модели имеет смысл добавить статистически важные переменные.

Приложение:

# %%

import numpy as np

import pandas as pd

import sklearn

import matplotlib.pyplot as plt

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, r2\_score

import statsmodels.api as sm

from scipy.fft import fft, fftfreq

from statsmodels.stats.stattools import durbin\_watson

# %%

data=pd.read\_csv('data\_lab3.csv', delimiter=',')

# %%

data.head(15)

# %%

y = data["Y2"]

X = data.drop(columns=["Y2"])

# %%

X.head()

# %%

y.head()

# %%

included = []

residuals = y

model =None

model\_results = []

# %%

# %%

# Пошаговое включение факторов

while True:

    # Вычисление корреляций остатков с факторами

    correlations = X.apply(lambda col: np.corrcoef(col, residuals)[0, 1])

    # Исключение уже включенных факторов

    correlations = correlations.drop(included, errors='ignore')

    # Если не осталось факторов, завершаем процесс

    if correlations.empty:

        print("Нет факторов для добавления. Процесс завершен.")

        break

    # Выбор фактора с максимальной корреляцией

    best\_factor = correlations.idxmax()

    # Временное добавление фактора в модель

    temp\_included = included + [best\_factor]

    X\_model = sm.add\_constant(X[temp\_included])

    temp\_model = sm.OLS(y, X\_model).fit()

    # Проверка значимости коэффициента добавленного фактора

    p\_value = temp\_model.pvalues[best\_factor]

    if p\_value <= 0.05:  # Если коэффициент значим

        included.append(best\_factor)  # Добавляем фактор в модель

        model = temp\_model  # Обновляем модель

        residuals = y - model.predict(X\_model)  # Обновляем остатки

        # Статистика Дарбина-Уотсона

        dw\_statistic = durbin\_watson(residuals)

        # Сохранение результатов модели

        model\_results.append({

            "Фактор": best\_factor,

            "Коэффициент": model.params[best\_factor],

            "P-значение": p\_value,

            "R^2": model.rsquared,

            "Скорректированный R^2": model.rsquared\_adj,

            "F-статистика": model.fvalue,

            "P-значение (F-статистика)": model.f\_pvalue,

            "Статистика Дарбина-Уотсона": dw\_statistic,

        })

        print(f"Добавлен фактор: {best\_factor}")

        print(model.summary())

        print(f"Статистика Дарбина-Уотсона: {dw\_statistic:.4f}")

    else:  # Если коэффициент незначим

        print(f"Фактор {best\_factor} незначим (p-value = {p\_value:.4f}). Пропускаем.")

        X = X.drop(columns=[best\_factor])  # Удаляем фактор из дальнейшего рассмотрения

# Создание DataFrame с результатами

results\_df = pd.DataFrame(model\_results)

# Экспорт в Excel

results\_df.to\_excel("Результаты\_моделей.xlsx", index=False)

print("Результаты экспортированы в файл 'Результаты\_моделей.xlsx'.")

# %%

# Создание DataFrame с результатами

results\_df = pd.DataFrame(model\_results)

# Экспорт в Excel

results\_df.to\_excel("Результаты\_моделей.xlsx", index=False)

print("Результаты экспортированы в файл 'Результаты\_моделей.xlsx'.")

# %%

X\_final = sm.add\_constant(X[included])

y\_pred = model.predict(X\_final)

# %%

y.head(15)

# %%

X\_final = sm.add\_constant(X[included])  # Добавляем константу

y\_pred = model.predict(X\_final)  # Предсказанные значения

# %%

plt.figure(figsize=(10, 6))

plt.plot(y.to\_numpy(), label="Реальные значения", marker="o")

plt.plot(y\_pred, label="Предсказанные значения", marker="x")

plt.xlabel("Наблюдения")

plt.ylabel("Значения")

plt.title("Сравнение реальных и предсказанных значений")

plt.legend()

plt.grid()

plt.show()

# %%

# Остаточная дисперсия из модели

residual\_variance\_model = model.mse\_resid

print(f"Остаточная дисперсия для модели: {residual\_variance\_model:.4f}")

# %%

# Тест Шапиро-Уилка

from scipy.stats import shapiro

shapiro\_test = shapiro(model.resid)

print(f"Тест Шапиро-Уилка: статистика = {shapiro\_test[0]:.4f}, p-value = {shapiro\_test[1]:.4f}")

# Интерпретация

if shapiro\_test[1] > 0.05:

    print("Остатки распределены нормально (p-value > 0.05).")

else:

    print("Остатки не распределены нормально (p-value ≤ 0.05).")

# %%

from statsmodels.stats.diagnostic import het\_breuschpagan

# Тест Бреуша-Пагана

bp\_test = het\_breuschpagan(residuals, X\_final)

print(f"Тест Бреуша-Пагана: статистика = {bp\_test[0]:.4f}, p-value = {bp\_test[1]:.4f}")

# Интерпретация

if bp\_test[1] > 0.05:

    print("Остатки гомоскедастичны (p-value > 0.05).")

else:

    print("Остатки гетероскедастичны (p-value ≤ 0.05).")