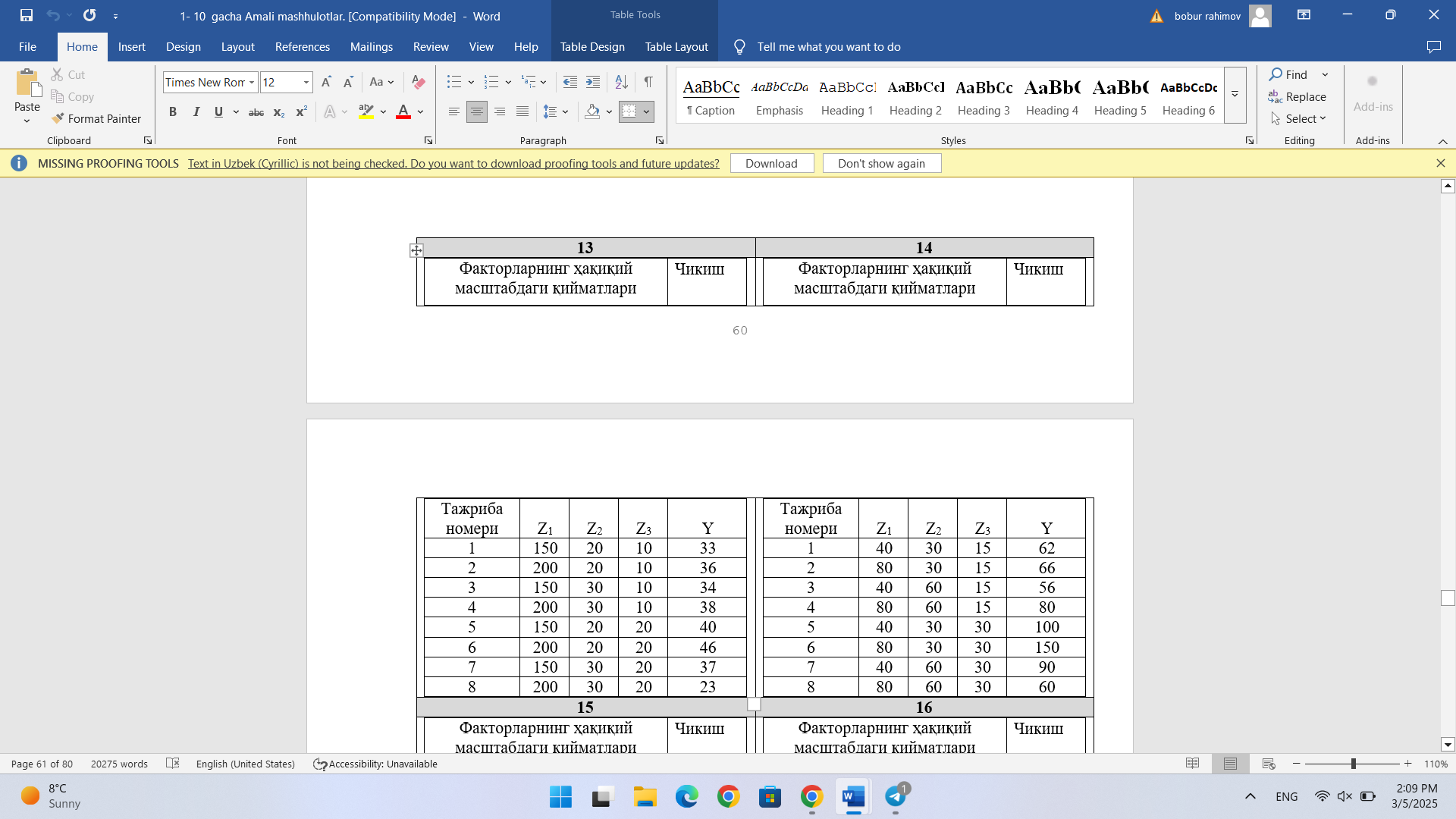
**6. Тажриба иши**

**ЖАРАЕННИНГ НОЧИЗИҚ МОДЕЛИНИ ТАЖРИБАНИ РЕЖАЛАШТИРИШ (2К - РЕЖА) УСУЛИДАН ФОЙДАЛАНИБ ҚУРИШ ВА МОДЕЛНИ ТАҲЛИЛ ҚИЛИШ**

**Ишдан мақсад.** Кўп факторли, параметрли объектларнинг математик моделларини тажриба-статистик усуллари билан қуришда ўтказиладиган тўлиқ тажрибалар сони ҳаддан ташқари кўпайиб кетади. Бундай ҳолларда объектнинг математик моделини тажрибани режалаштириш усулларидан фойдаланиб қуриш мумкин. Шу боисдан тажрибани режалаштириш усулларини ўрганиш мақсадга мувофиқ ҳисобланади.

Тажрибани режалаштириш усуллари бир вақтнинг ўзида тажриба ўтказишлар сонини минимумга келтиришга ва изланаётган функциянинг оптимал қийматини аниқлашга ёрдам беради. Тажрибани режалаштириш усулларидан бири икки поғонали режалаштиришдир. Икки поғонали режа (режа 2к) бундай режалаштиришда ҳар бир кирувчи параметр учун икки поғона мавжуд. Агар тажриба икки поғонада факторларнинг иккита қийматида ҳамма факторлар бўйича (k) тажрибада барча комбинациялари кўриб чиқилса бундай тажрибани тўлик факторли тажриба

4) Chiqish signalining kvadripolning kirish signaliga bog'liqligi modelini qurish va uni baholash talab qilinadi. Namuna ma'lumotlariga ko'ra to'rt kutupli kirish va chiqish signali haqida ma'lumot jadvalda keltirilgan.



import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.linear\_model import LinearRegression

from sklearn.preprocessing import PolynomialFeatures

from sklearn.pipeline import make\_pipeline

data = {

"Z1": np.array([30, 60, 30, 60, 30, 50, 30, 60]),

"Z2": np.array([20, 20, 30, 30, 20, 20, 30, 30]),

"Z3": np.array([20, 20, 20, 20, 30, 30, 30, 30]),

"Y": np.array([33, 36, 34, 38, 40, 46, 37, 23])

}

df = pd.DataFrame(data)

X = df["Z1"].values.reshape(-1, 1)

y = df["Y"].values

X\_sorted = np.linspace(min(X), max(X), 100).reshape(-1, 1)

plt.figure(figsize=(12, 4))

plt.subplot(1, 3, 1)

sns.scatterplot(x=df["Z1"], y=df["Y"], color="blue", label="Haqiqiy ma'lumotlar")

linear\_model = LinearRegression()

linear\_model.fit(X, y)

y\_pred\_linear = linear\_model.predict(X\_sorted)

plt.plot(X\_sorted, y\_pred\_linear, color="red", linewidth=2, linestyle="--", label="Chiziqli regressiya")

plt.xlabel("Z1")

plt.ylabel("Y")

plt.title("1. Chiziqli Regressiya")

plt.legend()

plt.grid(True)

plt.subplot(1, 3, 2)

sns.scatterplot(x=df["Z1"], y=df["Y"], color="blue", label="Haqiqiy ma'lumotlar")

poly\_2 = make\_pipeline(PolynomialFeatures(degree=2), LinearRegression())

poly\_2.fit(X, y)

y\_pred\_poly\_2 = poly\_2.predict(X\_sorted)

plt.plot(X\_sorted, y\_pred\_poly\_2, color="green", linewidth=2, linestyle="-", label="2-darajali (parabola)")

plt.xlabel("Z1")

plt.ylabel("Y")

plt.title("2. Parabola Regressiya")

plt.legend()

plt.grid(True)

plt.subplot(1, 3, 3)

sns.scatterplot(x=df["Z1"], y=df["Y"], color="blue", label="Haqiqiy ma'lumotlar")

colors = ["purple", "orange", "brown"]

degrees = [2, 4, 8]

for i, deg in enumerate(degrees):

poly\_model = make\_pipeline(PolynomialFeatures(degree=deg), LinearRegression())

poly\_model.fit(X, y)

y\_pred\_poly = poly\_model.predict(X\_sorted)

plt.plot(X\_sorted, y\_pred\_poly, color=colors[i], linewidth=2, linestyle="-", label=f"{deg}-darajali regressiya")

plt.xlabel("Z1")

plt.ylabel("Y")

plt.title("3. 2, 4, 8-darajali regressiya")

plt.legend()

plt.grid(True)

plt.tight\_layout()

plt.show()

