1. Feladat

December 4, 2024

0 Előkészületek

0.1 Szükséges könyvtárak importálása

```
import pandas as pd
import numpy as np
from scipy import stats
```

0.2 Adatok beolvasása

```
# Kategóriák definiálása
kategoriak = ['szuperhos', 'anti-hos', 'mellekszereplo', 'fogonosz',␣
 ↪'allatsegito']

# Adatok beolvasása string-ként
with open('data/bead1.csv', 'r') as file:
    lines = file.readlines()

# Az első sor elhagyása (mivel az a kategóriákat tartalmazza)
# Az értékek átalakítása soronként listává
data = [list(map(float, line.strip().strip('"').split(','))) for line in lines[1:
 ↪]]

# DataFrame létrehozása
df = pd.DataFrame(data, columns=kategoriak)


# Adatok átalakítása long formátumba
df_long = df.melt(var_name='Kategória', value_name='Gumimaci pontszám')
# Alapvető statisztikai jellemzők
print("Alapvető statisztikai jellemzők kategóriánként:")
print(df_long.groupby('Kategória')['Gumimaci pontszám'].describe())
```

```
Alapvető statisztikai jellemzők kategóriánként:
                count    mean       std   min     25%    50%     75%    max
Kategória
allatsegito      50.0  5.4096  3.130664  1.37  2.8850  4.335  9.6700  10.00
anti-hos         50.0  2.0552  1.655914  0.14  0.8300  1.545  3.0850   6.71
```

```
fogonosz          50.0   5.4766   2.125727   1.30   3.7325   5.690   7.1775    9.21
mellekszereplo    50.0   2.9046   1.635708   0.17   1.5675   2.735   4.0050    7.24
szuperhos         50.0   4.4258   2.879298   0.52   1.8700   3.770   6.0975    9.90
```

1 Hipotézisvizsgálat

1.1 Próba meghatározása

Az adatok eloszlásáról nem tudunk semmit, csak hogy számok és a nagyságuk sorrendje számít, így ordinális változóknak tekintjük a gumimaci pontszámokat. A szereplők egymástól függetlenek és 5 mintánk van, így Kruskal-Wallis próbát hajtunk végre.

1.2 Kruskal-Wallis próba

1.2.1 Hipotézisek megfogalmazása

Hipotézispár:

H_0: A kategóriák pontszámainak eloszlása azonos
H_1: Van két olyan kategória, amelyek pontszámainak eloszlása különbözik
Szignifikanciaszint: $\varepsilon = 0.05$

1.2.2 Próbastatisztika számítása

```
h_stat, p_value = stats.kruskal(*[group['Gumimaci pontszám'].values
                                  for name, group in df_long.
 ↪groupby('Kategória')])
print("Kruskal-Wallis teszt eredménye:")
print(f"H-statisztika = {h_stat:.4f}")
```

```
Kruskal-Wallis teszt eredménye:
H-statisztika = 68.1814
```

1.2.3 Döntés a kritikus érték alapján

Paraméterek:

Kategóriák száma (k) = 5
Szabadságfok (df) = k-1 = 4
Szignifikanciaszint (ε) = 0.05
H-statisztika = 68.1814
χ^2(0.05,4) kritikus érték (táblázat alapján) = 9.49

Döntési szabály:

Ha H > $\chi^2(\varepsilon,df)$ → elvetjük H0-t
Ha H ≤ $\chi^2(\varepsilon,df)$ → nem vetjük el H0-t

Összehasonlítás:

$68.1814 > 9.49$
A H-statisztika értéke nagyobb, mint a kritikus érték

1.2.4 Következtetés:

A H-statisztika meghaladja a kritikus értéket, ezért $\varepsilon = 0.05$ szignifikanciaszinten elvetjük a nullhipotézist.
Azaz statisztikailag kimutatható, hogy van különbség a kategóriák gumimaci pontszámai között.

2 Post-hoc tesztek

Mivel szignifikáns eltérést találtunk, ezért páronként meg kell vizsgálnunk a kategóriákat. A változóink ordinálisak, páronként végezzük a teszteket (tehát minden teszt esetén 2 mintát vetünk össze), a mintáink nem összefüggők.
Páronként 2 független mintás ordinális próbát, azaz Mann-Whitney próbát hajtunk végre.

2.1 Mann-Whitney Z teszt páronként

```
kategoriak = df_long['Kategória'].unique()
alpha = 0.05 # szignifikanciaszint

# Kritikus érték (kétoldali próba) normális eloszlás táblázatból
z_critical = 1.96  # z0.975 = 1.96

print(f"\nPáronkénti Mann-Whitney Z teszt eredményei:")
print(f"Kritikus érték (z{1-alpha/2:.3f}): {z_critical}")
print("-" * 50)

results = []
for i in range(len(kategoriak)):
    for j in range(i+1, len(kategoriak)):
        x = df_long[df_long['Kategória'] == kategoriak[i]]['Gumimaci pontszám'].
 ↪values
        y = df_long[df_long['Kategória'] == kategoriak[j]]['Gumimaci pontszám'].
 ↪values

        # Mann-Whitney teszt
        stat, p_value = stats.mannwhitneyu(x, y, alternative='two-sided')

        # Z-érték kiszámítása a p-értékből
        z_stat = stats.norm.ppf(1 - p_value/2)

        results.append({
            'Kategória 1': kategoriak[i],
            'Kategória 2': kategoriak[j],
```

```
            'Z-érték': z_stat,
            '|Z|': abs(z_stat),
            'Szignifikáns': abs(z_stat) > z_critical
        })
        print(f"{kategoriak[i]} vs {kategoriak[j]}: |Z| = {abs(z_stat):.4f} {'*'␣
↪if abs(z_stat) > z_critical else ''}")

# Eredmények DataFrame-be rendezése és megjelenítése
results_df = pd.DataFrame(results)
print("\nÖsszes páronkénti összehasonlítás eredménye:")
print(results_df)

# Szignifikáns különbségek kiírása
print("\nSzignifikáns különbségek:")
sig_pairs = results_df[results_df['Szignifikáns']].apply(
    lambda x: f"{x['Kategória 1']} vs {x['Kategória 2']} (|Z| = {x['|Z|']:.
↪4f})", axis=1
)
for pair in sig_pairs:
    print(f"{pair}")

# Nem szignifikáns különbségek kiírása
print("\nNem szignifikáns különbségek:")
nonsig_pairs = results_df[~results_df['Szignifikáns']].apply(
    lambda x: f"{x['Kategória 1']} vs {x['Kategória 2']} (|Z| = {x['|Z|']:.
↪4f})", axis=1
)
for pair in nonsig_pairs:
    print(f"{pair}")
```

```
Páronkénti Mann-Whitney Z teszt eredményei:
Kritikus érték (z0.975): 1.96
--------------------------------------------------
szuperhos vs anti-hos: |Z| = 4.4880 *
szuperhos vs mellekszereplo: |Z| = 2.4611 *
szuperhos vs fogonosz: |Z| = 2.2543 *
szuperhos vs allatsegito: |Z| = 1.8492
anti-hos vs mellekszereplo: |Z| = 2.7404 *
anti-hos vs fogonosz: |Z| = 6.7423 *
anti-hos vs allatsegito: |Z| = 5.8649 *
mellekszereplo vs fogonosz: |Z| = 5.5497 *
mellekszereplo vs allatsegito: |Z| = 4.1192 *
fogonosz vs allatsegito: |Z| = 0.7314

Összes páronkénti összehasonlítás eredménye:
      Kategória 1     Kategória 2   Z-érték       |Z|  Szignifikáns
```

```
0       szuperhos         anti-hos  4.487958  4.487958          True
1       szuperhos  mellekszereplo  2.461145  2.461145          True
2       szuperhos         fogonosz  2.254313  2.254313          True
3       szuperhos     allatsegito  1.849159  1.849159         False
4        anti-hos  mellekszereplo  2.740367  2.740367          True
5        anti-hos         fogonosz  6.742276  6.742276          True
6        anti-hos     allatsegito  5.864852  5.864852          True
7  mellekszereplo         fogonosz  5.549675  5.549675          True
8  mellekszereplo     allatsegito  4.119246  4.119246          True
9        fogonosz     allatsegito  0.731384  0.731384         False

Szignifikáns különbségek:
szuperhos vs anti-hos (|Z| = 4.4880)
szuperhos vs mellekszereplo (|Z| = 2.4611)
szuperhos vs fogonosz (|Z| = 2.2543)
anti-hos vs mellekszereplo (|Z| = 2.7404)
anti-hos vs fogonosz (|Z| = 6.7423)
anti-hos vs allatsegito (|Z| = 5.8649)
mellekszereplo vs fogonosz (|Z| = 5.5497)
mellekszereplo vs allatsegito (|Z| = 4.1192)

Nem szignifikáns különbségek:
szuperhos vs allatsegito (|Z| = 1.8492)
fogonosz vs allatsegito (|Z| = 0.7314)
```

Látható tehát, hogy a legtöbb kategória között szignifikáns különbség van az eloszlásuk tekintetében. Egyedül a szuperhos-allatsegito és fogonosz-allatsegito párosok eloszlásában nincs szignifikáns különbség.

2. Feladat

December 4, 2024

0 Előkészületek

0.1 Szükséges könyvtárak importálása

```
%reset -f

import pandas as pd
from sklearn.linear_model import LinearRegression
from sklearn.preprocessing import StandardScaler
import statsmodels.api as sm
from scipy import stats
from statsmodels.stats.outliers_influence import variance_inflation_factor
import numpy as np
import matplotlib.pyplot as plt
```

0.2 Adatok beolvasása

```
# Oszlopok definiálása
cols = ['Y', 'X_1', 'X_2']

# Adatok beolvasása string-ként
with open('data/bead2.csv', 'r') as file:
    lines = file.readlines()

# Az első sor elhagyása (mivel az az oszlopokat tartalmazza)
# Az értékek átalakítása soronként listává
data = [list(map(float, line.strip().strip('"').split(','))) for line in lines[1:
↪]]

# DataFrame létrehozása
df = pd.DataFrame(data, columns=cols)

# Adatok szétválasztása
X = df[['X_1', 'X_2']]  # magyarázó változók
y = df['Y']             # eredményváltozó

# Alapvető statisztikák
print("\nAlapvető statisztikák:")
```

```
print(df.describe())
```

```
Alapvető statisztikák:
               Y        X_1        X_2
count  50.000000  50.000000  50.000000
mean    6.130800   4.994800   5.082600
std     4.188834   2.909244   2.786417
min     0.000000   0.520000   0.340000
25%     1.335000   2.557500   2.612500
50%     7.915000   4.945000   5.130000
75%    10.000000   7.552500   7.927500
max    10.000000   9.900000   9.400000
```

1 Becslések

1.1 Az együtthatók pontbecslése

1.1.1 Regressziós együtthatók pontbecslése

```
# Modell illesztése
model = LinearRegression()
model.fit(X, y)

# Együtthatók és tengelymetszet
print("\nRegressziós együtthatók:")
print(f"b_0 (tengelymetszet) = {model.intercept_:.4f}")
print(f"b_1 (küzdőképesség) = {model.coef_[0]:.4f}")
print(f"b_2 (gumimaci pontszám) = {model.coef_[1]:.4f}")
```

```
Regressziós együtthatók:
b_0 (tengelymetszet) = 4.1082
b_1 (küzdőképesség) = 1.0282
b_2 (gumimaci pontszám) = -0.6124
```

1.1.2 Standardizált regressziós együtthatók pontbecslése

```
# Standardizálás
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
y_scaled = scaler.fit_transform(y.values.reshape(-1, 1)).ravel()
# A StandardScaler() 2D adatot vár, ezért y-t átalakítjuk azzá, majd a␣
↪ravel()-lel visszaalakítjuk 1D-vé, mert a regresszióhoz úgy kell

# Standardizált modell illesztése
model_scaled = LinearRegression()
model_scaled.fit(X_scaled, y_scaled)
```

```python
# Standardizált együtthatók
print("\nStandardizált regressziós együtthatók:")
print(f"b_1* (küzdőképesség) = {model_scaled.coef_[0]:.4f}")
print(f"b_2* (gumimaci pontszám) = {model_scaled.coef_[1]:.4f}")
```

```
Standardizált regressziós együtthatók:
b_1* (küzdőképesség) = 0.7141
b_2* (gumimaci pontszám) = -0.4074
```

1.1.3 Lineáris modell:

OLS Lineáris regresszió

1.1.4 Eredmények értelmezése

Az együtthatók közvetlenül összehasonlíthatók, mert azonos skálán vannak.
Látható, hogy az X_1 változó hatása erősebb az Y-ra, mint X_2-é, mert egységnyi változás X_1 változóban 0.7141 egységnyi hatással van Y-ra, míg egységnyi változás X_2 változóban csak 0.4074 hatással van Y-ra.

1.2 Előrejelzés készítése

```python
# Új megfigyelés
X_new = pd.DataFrame({
    'X_1': [85],
    'X_2': [8.5]
})

# Előrejelzés
prediction = model.predict(X_new)

print("\nElőrejelzés eredménye:")
print(f"Input értékek:")
print(f"- Küzdőképesség (X_1) = {X_new['X_1'].values[0]}")
print(f"- Gumimaci pontszám (X_2) = {X_new['X_2'].values[0]}")
print(f"\nBecsült erő (Y) = {prediction[0]:.4f}")
```

```
Előrejelzés eredménye:
Input értékek:
- Küzdőképesség (X_1) = 85
- Gumimaci pontszám (X_2) = 8.5

Becsült erő (Y) = 86.2962
```

1.3 Konfidenciaintervallum az együtthatókra

1.3.1 Kód és eredmény

```python
X_sm = sm.add_constant(X)
model_sm = sm.OLS(y, X_sm).fit()

# 95%-os konfidencia intervallumok az együtthatókra
conf_int = model_sm.conf_int(alpha=0.05)
print(model_sm.summary())
print("\n")
print(conf_int)
print("\nEgyütthatók 95%-os konfidencia intervallumai:")
print("-" * 50)
print("b_0 (tengelymetszet):")
print(f"[{conf_int.iloc[0,0]:.4f}, {conf_int.iloc[0,1]:.4f}]")
print("\nb_1 (küzdőképesség):")
print(f"[{conf_int.iloc[1,0]:.4f}, {conf_int.iloc[1,1]:.4f}]")
print("\nb_2 (gumimaci pontszám):")
print(f"[{conf_int.iloc[2,0]:.4f}, {conf_int.iloc[2,1]:.4f}]")
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      Y   R-squared:                       0.708
Model:                            OLS   Adj. R-squared:                  0.695
Method:                 Least Squares   F-statistic:                     56.88
Date:                Wed, 04 Dec 2024   Prob (F-statistic):           2.81e-13
Time:                        13:20:03   Log-Likelihood:                -111.32
No. Observations:                  50   AIC:                             228.6
Df Residuals:                      47   BIC:                             234.4
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          4.1082      0.912      4.506      0.000       2.274       5.942
X_1            1.0282      0.114      9.041      0.000       0.799       1.257
X_2           -0.6124      0.119     -5.158      0.000      -0.851      -0.374
==============================================================================
Omnibus:                        2.782   Durbin-Watson:                   1.569
Prob(Omnibus):                  0.249   Jarque-Bera (JB):                1.544
Skew:                          -0.087   Prob(JB):                        0.462
Kurtosis:                       2.157   Cond. No.                         21.6
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
```

```
              0         1
const  2.273973  5.942342
X_1    0.799377  1.256950
X_2   -0.851320 -0.373576

Együtthatók 95%-os konfidencia intervallumai:
--------------------------------------------------
b_0 (tengelymetszet):
[2.2740, 5.9423]

b_1 (küzdőképesség):
[0.7994, 1.2570]

b_2 (gumimaci pontszám):
[-0.8513, -0.3736]
```

1.3.2 Eredmények értelmezése

A konfidencia intervallumok jelentése: 95%-os valószínűséggel a valódi együttható értéke a megadott intervallumon belül van. Az intervallum szélessége a becslés pontosságát jelzi (minél szélesebb, annál bizonytalanabb a becslés).
Ha az intervallum nem tartalmazza a 0-t, akkor az adott változó hatása szignifikáns ($\alpha = 0.05$ mellett).

Következtetések: A változók szignifikánsak.

1.4 Előrejelzési intervallum

```python
# Konstans hozzáadása
X_new_sm = sm.add_constant(X_new, has_constant='add')

# Előrejelzési intervallum számítása
prediction = model_sm.get_prediction(X_new_sm)
pred_summary = prediction.summary_frame(alpha=0.05)

print("\nElőrejelzés és intervallumok:")
print("-" * 50)
print(f"Pontbecslés: {pred_summary['mean'].values[0]:.4f}")
print(f"95%-os előrejelzési intervallum:")
print(f"[{pred_summary['obs_ci_lower'].values[0]:.4f}, 
 ↪{pred_summary['obs_ci_upper'].values[0]:.4f}]")
```

```
Előrejelzés és intervallumok:
--------------------------------------------------
Pontbecslés: 86.2962
```

```
95%-os előrejelzési intervallum:
[67.3380, 105.2545]
```

2 Illeszkedésdiagnosztika

2.1 Determinációs együttható (R^2) és korrigált R^2

2.1.1 Kód és eredmények

```
r2 = model_sm.rsquared
adj_r2 = model_sm.rsquared_adj

print("\nDeterminációs együtthatók:")
print("-" * 50)
print(f"R² = {r2:.4f}")
print(f"Korrigált R² = {adj_r2:.4f}")
print(f"Különbség = {(r2-adj_r2):.4f}")
```

```
Determinációs együtthatók:
--------------------------------------------------
R² = 0.7077
Korrigált R² = 0.6952
Különbség = 0.0124
```

2.1.2 Értelmezés

R^2 (Determinációs együttható):

A determinációs együttható értéke 0.7077, ami a modell által magyarázott variancia arányát mutatja.
Az R^2 a teljes varianciához viszonyítva fejezi ki a modell által megmagyarázott hányadot.
Értéke 0 és 1 közé esik, ahol 0 esetén a modell semmit nem magyaráz, 1 esetén tökéletes az illeszkedés.
Az R^2 = 1 - (SSE/SST) képlettel számolható, ahol SSE a hiba szórásnégyzetösszeg, SST a teljes szórásnégyzetösszeg.

Korrigált R^2:

A korrigált R^2 értéke 0.6952, ami figyelembe veszi a magyarázó változók számát is.
A korrigált R^2 = 1 - (1-R^2)*(n-1)/(n-k-1) képlettel számolható, ahol n a mintaelemszám (jelen esetben 50), k a magyarázó változók száma (jelen esetben 2).
Ez a mutató bünteti a felesleges magyarázó változók bevonását.
Értéke mindig kisebb vagy egyenlő, mint az R^2.

A két mutató jelentősége:

Az R^2 érték sosem csökken új változó bevonásakor, akkor sem, ha az valójában nem javít a modellen.
A korrigált R^2 ezzel szemben csökkenhet, ha nem hasznos változót vonunk be a modellbe.

Modellek összehasonlítására ezért a korrigált R^2 alkalmasabb.
Ha nagy a különbség a két érték között, az felesleges változók jelenlétére utalhat.

Értékelés:

A kapott $R^2 = 0.7077$ azt jelenti, hogy modellünk a variancia 70.77%-át magyarázza meg. A korrigált $R^2 = 0.6952$ érték a modell tényleges magyarázó erejét mutatja.

3 Modelldiagnosztika

3.1 Modelldiagnosztikai tesztek

3.1.1 Kód és eredmények

```
# F-próba statisztikái
f_stat = model_sm.fvalue
f_pvalue = model_sm.f_pvalue
df_reg = 2  # magyarázó változók száma
df_res = len(df) - df_reg - 1
f_crit = stats.f.ppf(0.95, df_reg, df_res)

print(f"F-statisztika: {f_stat:.4f}")
print(f"p-érték: {f_pvalue}")
print(f"Kritikus érték (F0.95({df_reg},{df_res})): {f_crit:.4f}")
```

```
F-statisztika: 56.8848
p-érték: 2.808718819001525e-13
Kritikus érték (F0.95(2,47)): 3.1951
```

3.1.2 Értelmezés

Hipotézisek:

H_0: A modell nem magyarázza az eredményváltozó varianciáját ($X_1 = X_2 = 0$)
H_1: A modell szignifikánsan magyarázza az eredményváltozó varianciáját ($X_1 \neq 0$ és/vagy $X_2 \neq 0$)
Szignifikanciaszint: $\alpha = 0.05$

F-próba eredménye:

F-statisztika értéke: 56.8848
p-érték: 2.808718819001525e-13
Kritikus érték (F0.95(2,47)): 3.1951

Döntés:

Az F-próba p-értéke (2.808718819001525e-13) kisebb, mint $\alpha = 0.05$, ezért elvetjük a nullhipotézist 95%-os konfidenciaszinten.

Következtetés:

A kapott eredmények alapján a modellünk szignifikáns $\alpha = 0.05$ szignifikanciaszint mellett.
Ez azt jelenti, hogy a küzdőképesség és gumimaci pontszám együttesen magyarázzák szignifikánsan a mesehős erejét.
A modell alkalmas előrejelzésre és további elemzésre.
Az eredmény összhangban van a korábban számolt R^2 értékkel.

A teszt jelentősége:

Az F-próba a modell egészének magyarázó erejét vizsgálja.
Azt teszteli, hogy a magyarázó változók együttesen szignifikáns hatással vannak-e az eredményváltozóra.
Az F-próba a determinációs együttható nullától való eltérését vizsgálja.
A teszt a regressziós modell gyakorlati használhatóságáról ad információt.

3.2 Változók szignifikanciájának tesztelése

3.2.1 Kód és eredmények

```
# Kritikus érték meghatározása (kétoldali próba)
df_res = len(df) - df_reg - 1
t_crit = stats.t.ppf(0.975, df_res)  # 0.975 a kétoldali próba miatt

print("\nKritikus érték:")
print(f"t_krit = ±{t_crit:.4f} (szabadságfok = {df_res})")
print("\nEgyütthatók tesztjei:")
print(model_sm.summary().tables[1])
```

```
Kritikus érték:
t_krit = ±2.0117 (szabadságfok = 47)

Együtthatók tesztjei:
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          4.1082      0.912      4.506      0.000       2.274       5.942
X_1            1.0282      0.114      9.041      0.000       0.799       1.257
X_2           -0.6124      0.119     -5.158      0.000      -0.851      -0.374
==============================================================================
```

3.2.2 Értelmezés

Hipotézispárok:

Tengelymetszet (b_0):
H_0: $b_0 = 0$
H_1: $b_0 \neq 0$

Küzdőképesség (b_1):
H_0: $b_1 = 0$
H_1: $b_1 \neq 0$

Gumimaci pontszám (b_2):
H_0: $b_2 = 0$
H_1: $b_2 \neq 0$

Eredmények:

Tengelymetszet (b_0):
|t-érték| = 4.506 > 2.0117 (t_krit)
Döntés: 5%-os szignifikanciaszinten elvetjük H_0-t

Küzdőképesség (b_1):
|t-érték| = 9.041 > 2.0117 (t_krit)
Döntés: 5%-os szignifikanciaszinten elvetjük H_0-t

Gumimaci pontszám (b_2):
|t-érték| = 5.158 > 2.0117 (t_krit)
Döntés: 5%-os szignifikanciaszinten elvetjük H_0-t

Következtetések:

A t-próba kritikus értéke ±2.0117 (47 szabadságfok mellett, 5%-os szignifikanciaszinten).
A tengelymetszet |t| = 4.506 értéke meghaladja a kritikus értéket, ami azt jelenti, hogy amikor mindkét magyarázó változó 0, akkor a várható Y érték (4.1082) szignifikánsan különbözik nullától.
A küzdőképesség |t| = 9.041 értéke jelentősen meghaladja a kritikus értéket, tehát erős szignifikáns hatást mutat.
A gumimaci pontszám |t| = 5.158 értéke szintén meghaladja a kritikus értéket, így ez a hatás is szignifikáns.
Mindhárom változó esetében elvetjük a nullhipotézist, ami azt jelenti, hogy mindegyik hatása szignifikáns.

3.3 Multikollinearitás vizsgálata

3.3.1 Kód és eredmények

```
vif_data = pd.DataFrame()
vif_data["Változó"] = X.columns
vif_data["VIF"] = [variance_inflation_factor(X.values, i) for i in range(X.
↪shape[1])]

print("\nVIF értékek:")
print(vif_data)
```

```
VIF értékek:
  Változó       VIF
```

```
0      X_1  2.273206
1      X_2  2.273206
```

3.3.2 Értelmezés

Döntési szabály:

VIF > 5: erős multikollinearitás
VIF > 10: súlyos multikollinearitás
VIF ≈ 1: nincs multikollinearitás

VIF érték:

A VIF érték: 2.273206
A VIF érték azt mutatja, hogy egy változó mennyire magyarázható a többi magyarázó változóval.
VIF = $1/(1-R^2)$, ahol R^2 az adott változónak a többi magyarázó változóval vett determinációs együtthatója.
A kapott VIF értékek alapján nincs jelentős multikollinearitás a modellben.

Miért probléma a multikollinearitás?

A multikollinearitás növeli az együtthatók standard hibáját. Bizonytalanabbá teszi a paraméterek becslését. Nehézzé teszi az egyes változók egyedi hatásának elkülönítését. Instabillá teheti a modellt: kis változás az adatokban nagy változást okozhat az együtthatókban.

3.4 Hibatagok vizsgálata

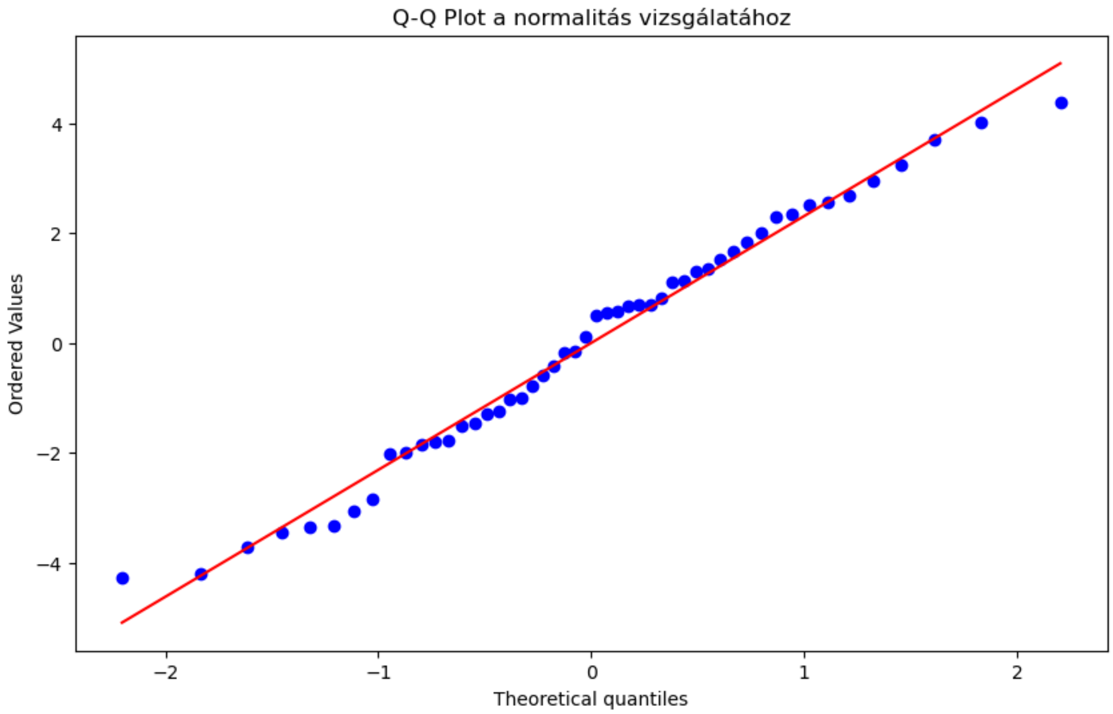
3.4.1 Kód és eredmények

```python
# Reziduálisok kiszámítása
residuals = model_sm.resid

# 1. Várható érték vizsgálata
resid_mean = np.mean(residuals)
resid_std = np.std(residuals, ddof=len(X_sm.columns))
t_stat = resid_mean / (resid_std/np.sqrt(len(residuals)))
p_value_mean = 2 * stats.t.cdf(-abs(t_stat), len(residuals)-1)

# 2. Normalitás vizsgálata (Shapiro-Wilk teszt)
shapiro_stat, shapiro_p = stats.shapiro(residuals)

# 3. Függetlenség vizsgálata (Durbin-Watson teszt)
dw_stat = sm.stats.stattools.durbin_watson(residuals)

# 4. Homoszkedaszticitás vizsgálata (Breusch-Pagan teszt)
bp_test = sm.stats.diagnostic.het_breuschpagan(residuals, X_sm)

# 5. Variancia becslése
```

```python
variance = np.var(residuals, ddof=len(X_sm.columns))

print("\nHibatagok vizsgálata:")
print("-" * 50)

print("\nVárható érték vizsgálata:")
print(f"Átlag (várható érték becslése): {resid_mean}")
print(f"t-statisztika: {t_stat}")
print(f"p-érték: {p_value_mean}")

print("\nNormalitás vizsgálata (Shapiro-Wilk):")
print(f"Teszt statisztika: {shapiro_stat:.4f}")
print(f"p-érték: {shapiro_p:.4f}")

print("\nFüggetlenség vizsgálata (Durbin-Watson):")
print(f"DW statisztika: {dw_stat:.4f}")

print("\nHomoszkedaszticitás vizsgálata (Breusch-Pagan):")
print(f"Teszt statisztika: {bp_test[0]:.4f}")
print(f"p-érték: {bp_test[1]:.4f}")

print("\nVariancia becslése:")
print(f"Becsült variancia: {variance:.4f}")

plt.figure(figsize=(10, 6))
stats.probplot(residuals, dist="norm", plot=plt)
plt.title('Q-Q Plot a normalitás vizsgálatához')
plt.show()
```

```
Hibatagok vizsgálata:
--------------------------------------------------

Várható érték vizsgálata:
Átlag (várható érték becslése): -4.263256414560601e-16
t-statisztika: -1.3035784262394252e-15
p-érték: 0.9999999999999989

Normalitás vizsgálata (Shapiro-Wilk):
Teszt statisztika: 0.9779
p-érték: 0.4678

Függetlenség vizsgálata (Durbin-Watson):
DW statisztika: 1.5689

Homoszkedaszticitás vizsgálata (Breusch-Pagan):
Teszt statisztika: 1.3786
p-érték: 0.5019
```

```
Variancia becslése:
Becsült variancia: 5.3478
```



3.4.2 Értelmezés

Várható érték vizsgálata:

H_0: $\mathrm{E}(\varepsilon) = 0$
H_1: $\mathrm{E}(\varepsilon) \neq 0$
t-statisztika értéke: -1.3036e-15
p-érték: 1.0000
Döntés: $1.0000 > 0.05$, tehát nem vetjük el H_0-t
A lineáris regresszióban, ha a modell tartalmaz konstans tagot (interceptet), akkor a reziduálisok összege nulla lesz, és így az átlaguk is nulla, ezért ez nem túlzottan meglepő.

Normalitás vizsgálata (Shapiro-Wilk teszt):

H_0: A hibatagok normális eloszlásúak
H_1: A hibatagok nem normális eloszlásúak
Teszt statisztika: 0.9779
p-érték: 0.4678
Döntés: $0.4678 > 0.05$, tehát nem vetjük el H_0-t

Függetlenség vizsgálata (Durbin-Watson teszt):

H_0: A hibatagok függetlenek
H_1: A hibatagok autokorreláltak
DW statisztika: 1.5689
Kritikus értékek 5%-os szinten: dL = 1.46, dU = 1.63 (DW táblázatból: https://www3.nd.edu/~wevans1/econ30331/durbin_watson_tables.pdf)
Döntés: 1.5689 beleesik az [1.46, 1.63] intervallumba, így nem tudunk egyértelmű döntést hozni

Homoszkedaszticitás vizsgálata (Breusch-Pagan teszt):

H_0: A hibatagok homoszkedasztikusak
H_1: A hibatagok heteroszkedasztikusak
Teszt statisztika: 1.3786
p-érték: 0.5019
Döntés: $0.5019 > 0.05$, tehát nem vetjük el H_0-t

Variancia becslése:

A hibatagok becsült varianciája: 5.3478
A variancia a reziduálisok szóródását méri a regressziós egyenes körül.

Összefoglaló értékelés:

A várható érték feltétel teljesül.
A normalitás feltétele teljesül.
A függetlenség feltételéről nem tudunk egyértelmű döntést hozni.
A homoszkedaszticitás feltétele teljesül (a szórás állandó).

3. Feladat

December 4, 2024

0 Előkészületek

0.1 Szükséges könyvtárak importálása

```
%reset -f
import pandas as pd
import numpy as np
import statsmodels.api as sm
import matplotlib.pyplot as plt
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from scipy import stats
from statsmodels.tsa.holtwinters import SimpleExpSmoothing
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.tsa.stattools import adfuller
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.stats.diagnostic import acorr_ljungbox
from statsmodels.graphics.gofplots import qqplot
```

0.2 Adatok beolvasása

```
# Oszlopok definiálása
cols = ['Idő', 'Érték']

# Adatok beolvasása string-ként
with open('data/bead3.csv', 'r', encoding='latin-1') as file:
   lines = file.readlines()

# Az első sor elhagyása és értékek átalakítása
data = [list(map(float, line.strip().strip('"').split(','))) for line in lines[1:
 ↪]]

# DataFrame létrehozása
df = pd.DataFrame(data, columns=cols)
```

1 Determinisztikus modell illesztése

1.1 Kód és eredmények

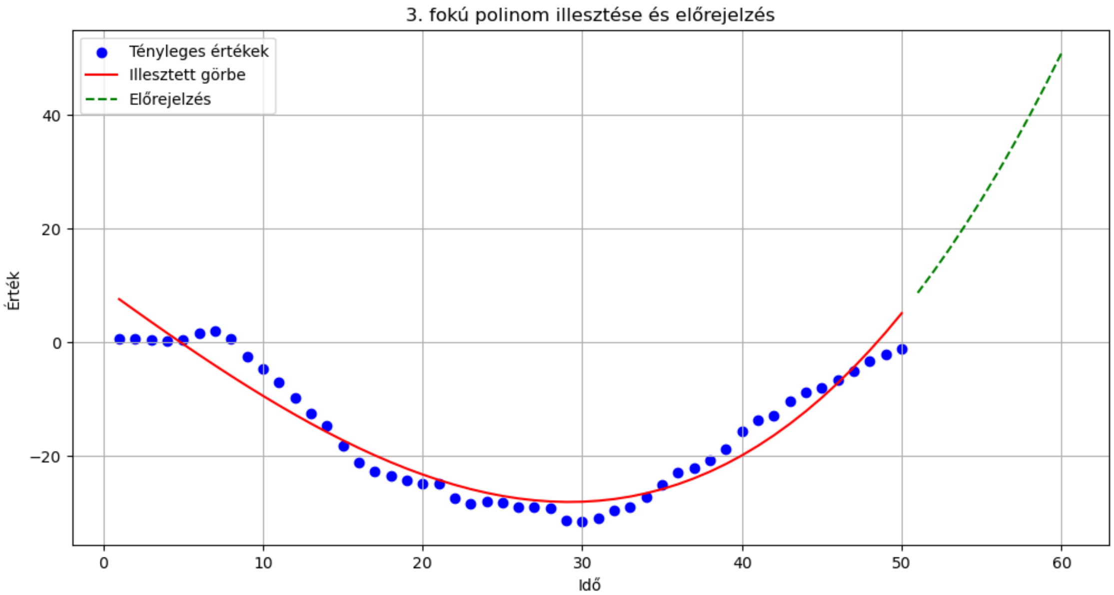
```
# Modellek összehasonlítása
max_degree = 10
selected_degree = 3  # a kiválasztott fokszám
results = []

# Különböző fokszámú modellek összehasonlítása
for degree in range(0, max_degree):
   X = np.vander(df['Idő'], degree + 1)
   model = sm.OLS(df['Érték'], X).fit()
   r2 = r2_score(df['Érték'], model.fittedvalues)

   results.append({
       'Fokszám': degree,
       'R²': r2,
       'AIC': model.aic,
       'BIC': model.bic
   })

# Eredmények kiíratása
results_df = pd.DataFrame(results)
print("\nModellek összehasonlítása:")
print(results_df)

# Kiválasztott fokszámú polinom illesztése
X = np.vander(df['Idő'], selected_degree + 1)
model = sm.OLS(df['Érték'], X).fit()

# Eredmények kiíratása
print(f"\n{selected_degree}. fokú polinom illesztése:")
print(model.summary().tables[0])
print(model.summary().tables[1])

# Reziduálisok vizsgálata
residuals = model.resid

# 1. Várható érték vizsgálata
resid_mean = np.mean(residuals)
resid_std = np.std(residuals, ddof=selected_degree+1)
t_stat = resid_mean / (resid_std/np.sqrt(len(residuals)))
p_value_mean = 2 * stats.t.cdf(-abs(t_stat), len(residuals)-1)

# 2. Normalitás vizsgálata (Shapiro-Wilk teszt)
shapiro_stat, shapiro_p = stats.shapiro(residuals)
```

```python
# 3. Függetlenség vizsgálata (Ljung-Box teszt)
lb_stat = sm.stats.diagnostic.acorr_ljungbox(residuals)

# 4. Homoszkedaszticitás vizsgálata (Breusch-Pagan teszt)
bp_test = sm.stats.diagnostic.het_breuschpagan(residuals, model.model.exog)

print("\nHibatagok vizsgálata - eredmények:")
print("-" * 50)
print(f"1. Várható érték vizsgálata:")
print(f"Átlag (várható érték becslése): {resid_mean:.6f}")
print(f"t-statisztika: {t_stat}")
print(f"p-érték: {p_value_mean}")

print(f"\n2. Normalitás vizsgálata (Shapiro-Wilk):")
print(f"Teszt statisztika: {shapiro_stat:.6f}")
print(f"p-érték: {shapiro_p:.6f}")

print(f"\n3. Függetlenség vizsgálata (Ljung-Box):")
print("Lag  Teszt statisztika   p-érték")
print("-" * 35)
for lag, row in lb_stat.iterrows():
    print(f"{lag:2d}   {row['lb_stat']:14.6f}   {row['lb_pvalue']:.6e}")

print(f"\n4. Homoszkedaszticitás vizsgálata (Breusch-Pagan):")
print(f"Teszt statisztika: {bp_test[0]:.6f}")
print(f"p-érték: {bp_test[1]:.6f}")

# Előrejelzés a következő 10 időpontra
future_points = np.arange(len(df) + 1, len(df) + 11)
X_future = np.vander(future_points, selected_degree + 1)
predictions = model.predict(X_future)

print("\nElőrejelzések:")
for i, pred in enumerate(predictions):
   print(f"{future_points[i]}. időpont: {pred:.2f}")

# Ábrázolás
plt.figure(figsize=(12, 6))
plt.scatter(df['Idő'], df['Érték'], color='blue', label='Tényleges értékek')
plt.plot(df['Idő'], model.fittedvalues, color='red', label='Illesztett görbe')
plt.plot(future_points, predictions, color='green', linestyle='--',␣
 ↪label='Előrejelzés')
plt.xlabel('Idő')
plt.ylabel('Érték')
plt.title(f'{selected_degree}. fokú polinom illesztése és előrejelzés')
plt.legend()
```

```
plt.grid(True)
plt.show()
```

```
Modellek összehasonlítása:
   Fokszám        R²         AIC         BIC
0        0  0.000000  387.062930  388.974953
1        1  0.062344  385.844343  389.668389
2        2  0.902997  274.412442  280.148511
3        3  0.921054  266.113215  273.761307
4        4  0.986175  180.999703  190.559818
5        5  0.987419  178.285939  189.758077
6        6  0.988357  176.408735  189.792896
7        7  0.988364  178.381243  193.677427
8        8  0.994452  143.342890  160.551097
9        9  0.992497  156.440981  171.737165

3. fokú polinom illesztése:
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Érték   R-squared:                       0.921
Model:                            OLS   Adj. R-squared:                  0.916
Method:                 Least Squares   F-statistic:                     178.9
Date:                Wed, 04 Dec 2024   Prob (F-statistic):           2.30e-25
Time:                        20:18:41   Log-Likelihood:                -129.06
No. Observations:                  50   AIC:                             266.1
Df Residuals:                      46   BIC:                             273.8
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
x1             0.0006      0.000      3.244      0.002       0.000       0.001
x2             0.0064      0.016      0.412      0.682      -0.025       0.038
x3            -2.0320      0.342     -5.935      0.000      -2.721      -1.343
const          9.5845      2.036      4.707      0.000       5.486      13.683
==============================================================================

Hibatagok vizsgálata - eredmények:
--------------------------------------------------
1. Várható érték vizsgálata:
Átlag (várható érték becslése): 0.000000
t-statisztika: 3.9543447227832004e-13
p-érték: 0.999999999999686

2. Normalitás vizsgálata (Shapiro-Wilk):
Teszt statisztika: 0.971069
```

```
p-érték: 0.255692

3. Függetlenség vizsgálata (Ljung-Box):
Lag  Teszt statisztika   p-érték
-----------------------------------
 1       36.706257   1.373379e-09
 2       57.588516   3.124732e-13
 3       66.733136   2.135808e-14
 4       68.558895   4.572428e-14
 5       68.644769   1.961309e-13
 6       70.969881   2.585887e-13
 7       75.831668   9.719502e-14
 8       83.001311   1.214121e-14
 9       92.734283   4.596171e-16
10      102.056297   2.110987e-17

4. Homoszkedaszticitás vizsgálata (Breusch-Pagan):
Teszt statisztika: 16.254783
p-érték: 0.001005

Előrejelzések:
51. időpont: 8.67
52. időpont: 12.46
53. időpont: 16.46
54. időpont: 20.69
55. időpont: 25.14
56. időpont: 29.81
57. időpont: 34.72
58. időpont: 39.86
59. időpont: 45.24
60. időpont: 50.86
```



1.2 Értelmezés $\varepsilon = 0.05$ szignifikanciaszint mellett

1.2.1 Modellválasztás

Az AIC és BIC értékek alapján a 8. fokú polinom adná a legjobb illeszkedést, azonban a 3. fokú polinom mellett döntöttem az overfitting elkerülése végett.

1.2.2 Várható érték vizsgálata

H_0: $E(\varepsilon) = 0$
H_1: $E(\varepsilon) \neq 0$
t-statisztika értéke: 0.0000
p-érték: 0.9999
Döntés: $0.9999 > 0.05$, tehát nem vetjük el H_0-t

1.2.3 Normalitás vizsgálata (Shapiro-Wilk teszt)

H_0: A hibatagok normális eloszlásúak
H_1: A hibatagok nem normális eloszlásúak
Teszt statisztika: 0.9711
p-érték: 0.2557
Döntés: $0.2557 > 0.05$, tehát nem vetjük el H_0-t

1.2.4 Függetlenség vizsgálata (Ljung-Box teszt)

H_0: A hibatagok függetlenek
H_1: A hibatagok autokorreláltak
A teszt minden vizsgált késleltetésre (1-10 lag) erősen szignifikáns autokorrelációt mutat
Döntés: Minden késleltetésre $p < 0.05$, tehát elvetjük H_0-t

1.2.5 Homoszkedaszticitás vizsgálata (Breusch-Pagan teszt)

H_0: A hibatagok homoszkedasztikusak
H_1: A hibatagok heteroszkedasztikusak
Teszt statisztika: 16.2548
p-érték: 0.0010
Döntés: $0.0010 < 0.05$, tehát elvetjük H_0-t

1.2.6 Összefoglaló értékelés

A hibatagok diagnosztikai vizsgálata alapján:
- A várható érték feltétel teljesül (az átlag gyakorlatilag 0).
- A normalitás feltétele teljesül (a hibatagok normális eloszlásúak).
- A függetlenség feltétele nem teljesül, erős pozitív autokorreláció van jelen.
- A homoszkedaszticitás feltétele nem teljesül, a hibatagok heteroszkedasztikusak.

2 Exponenciális simítás alkalmazása

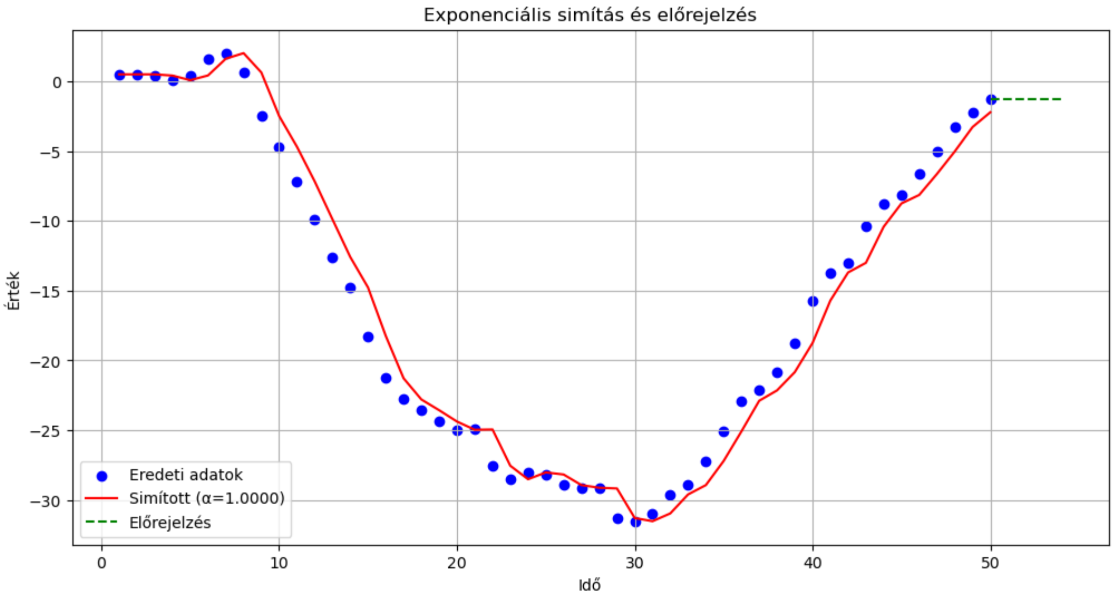
2.1 Kód és eredmények

```
# Exponenciális simítás (optimális alpha meghatározása)
model = SimpleExpSmoothing(df['Érték']).fit()
alpha = model.model.params['smoothing_level']

# Illesztett értékek és előrejelzések
fitted_values = model.fittedvalues
forecast = model.forecast(5)

# Illeszkedés vizsgálata
mae = mean_absolute_error(df['Érték'], fitted_values)
mse = mean_squared_error(df['Érték'], fitted_values)
rmse = np.sqrt(mse)

# Reziduálisok vizsgálata
residuals = model.resid

# 1. Várható érték vizsgálata
resid_mean = np.mean(residuals)
resid_std = np.std(residuals, ddof=1)
t_stat = resid_mean / (resid_std / np.sqrt(len(residuals)))
p_value_mean = 2 * stats.t.cdf(-abs(t_stat), len(residuals) - 1)

# 2. Normalitás vizsgálata (Shapiro-Wilk teszt)
shapiro_stat, shapiro_p = stats.shapiro(residuals)

# 3. Függetlenség vizsgálata (Ljung-Box teszt)
lb_stat = sm.stats.diagnostic.acorr_ljungbox(residuals)
```

```python
# 4. Homoszkedaszticitás vizsgálata (Breusch-Pagan teszt)
exog = sm.add_constant(fitted_values)
bp_test = sm.stats.diagnostic.het_breuschpagan(residuals, exog)

# Eredmények kiírása
print("\nIlleszkedési mutatók:")
print(f"MAE = {mae:.4f}")
print(f"MSE = {mse:.4f}")
print(f"RMSE = {rmse:.4f}")
print(f"Smoothing level (alpha) = {alpha:.4f}")

print("\nHibatagok vizsgálata - eredmények:")
print("-" * 50)
print("1. Várható érték vizsgálata:")
print(f"Átlag: {resid_mean:.6f}")
print(f"t-statisztika: {t_stat:.6f}")
print(f"p-érték: {p_value_mean:.6f}")

print("\n2. Normalitás vizsgálata (Shapiro-Wilk):")
print(f"Teszt statisztika: {shapiro_stat:.6f}")
print(f"p-érték: {shapiro_p:.6f}")

print(f"\n3. Függetlenség vizsgálata (Ljung-Box):")
print("Lag  Teszt statisztika   p-érték")
print("-" * 35)
for lag, row in lb_stat.iterrows():
    print(f"{lag:2d}   {row['lb_stat']:14.6f}   {row['lb_pvalue']:.6e}")

print("\n4. Homoszkedaszticitás vizsgálata (Breusch-Pagan teszt):")
print(f"Teszt statisztika: {bp_test[0]:.6f}")
print(f"p-érték: {bp_test[1]:.6f}")

print("\nElőrejelzések:")
for i, pred in enumerate(forecast, 1):
    print(f"{len(df) + i}. időpont: {pred:.2f}")

# Ábrázolás
plt.figure(figsize=(12, 6))
plt.scatter(df['Idő'], df['Érték'], color='blue', label='Eredeti adatok')
plt.plot(df['Idő'], fitted_values, 'r-', label=f'Simított (α={alpha:.4f})')

future_points = np.arange(len(df), len(df) + len(forecast))
plt.plot(future_points, forecast, 'g--', label='Előrejelzés')

plt.title('Exponenciális simítás és előrejelzés')
plt.xlabel('Idő')
plt.ylabel('Érték')
```

```
plt.legend()
plt.grid(True)
plt.show()
```

```
Illeszkedési mutatók:
MAE = 1.3446
MSE = 2.7510
RMSE = 1.6586
Smoothing level (alpha) = 1.0000

Hibatagok vizsgálata - eredmények:
--------------------------------------------------
1. Várható érték vizsgálata:
Átlag: -0.035400
t-statisztika: -0.149436
p-érték: 0.881823

2. Normalitás vizsgálata (Shapiro-Wilk):
Teszt statisztika: 0.961779
p-érték: 0.105540

3. Függetlenség vizsgálata (Ljung-Box):
Lag  Teszt statisztika   p-érték
-----------------------------------
 1        34.645889   3.954729e-09
 2        59.005126   1.538862e-13
 3        82.658607   8.253867e-18
 4       104.418817   1.126498e-21
 5       118.862058   5.466764e-24
 6       131.665637   5.730822e-26
 7       144.778551   5.062839e-28
 8       151.797860   8.266950e-29
 9       155.033991   7.976957e-29
10       156.828963   1.461608e-28

4. Homoszkedaszticitás vizsgálata (Breusch-Pagan teszt):
Teszt statisztika: 0.487821
p-érték: 0.484901

Előrejelzések:
51. időpont: -1.25
52. időpont: -1.25
53. időpont: -1.25
54. időpont: -1.25
55. időpont: -1.25
```



2.2 Exponenciális simítás eredményei

2.2.1 Modell specifikációk

A modellben a SimpleExpSmoothing függvény által meghatározott $\alpha = 1.0000$ simítási paramétert használtuk.

2.2.2 Illeszkedési mutatók

MAE (Mean Absolute Error): 1.3446
Az átlagos abszolút hiba azt mutatja, hogy az előrejelzéseink átlagosan 1.3446 egységgel térnek el a tényleges értékektől.

MSE (Mean Squared Error): 2.7510
Az átlagos négyzetes hiba az előrejelzési hibák négyzetének átlaga, jelen esetben 2.7510. Ez a mutató érzékeny a nagyobb eltérésekre, mivel a hibákat négyzetre emeli.

RMSE (Root Mean Squared Error): 1.6586
A négyzetes átlaggyök hiba az MSE négyzetgyöke, ami az előrejelzési hibák átlagos nagyságát adja meg az eredeti mértékegységben.

Simítási paraméter (alpha): 1.0000
Az alpha értéke 1.0, ami azt jelenti, hogy a modell teljes mértékben az utolsó megfigyelésre támaszkodik az előrejelzés során. Ebben az esetben a modell nem simítja az adatokat, hanem minden előrejelzés az utolsó ismert érték lesz.

2.3 Hibatagok tulajdonságainak vizsgálata $\varepsilon = 0.05$ szignifikanciaszint mellett

2.3.1 Várható érték vizsgálata

H_0: E(ε) = 0
H_1: E(ε) ≠ 0
t-statisztika értéke: -0.1494
p-érték: 0.8818
Döntés: 0.8818 > 0.05, tehát nem vetjük el H_0-t

2.3.2 Normalitás vizsgálata (Shapiro-Wilk teszt)

H_0: A hibatagok normális eloszlásúak
H_1: A hibatagok nem normális eloszlásúak
Teszt statisztika: 0.9618
p-érték: 0.1055
Döntés: 0.1055 > 0.05, tehát nem vetjük el H_0-t

2.3.3 Függetlenség vizsgálata (Ljung-Box teszt)

H_0: A hibatagok függetlenek
H_1: A hibatagok autokorreláltak
A teszt minden vizsgált késleltetésre (1-10 lag) erősen szignifikáns autokorrelációt mutat
Döntés: Minden késleltetésre p < 0.05, tehát elvetjük H_0-t

2.3.4 Homoszkedaszticitás vizsgálata (Breusch-Pagan teszt)

H_0: A hibatagok homoszkedasztikusak
H_1: A hibatagok heteroszkedasztikusak
Teszt statisztika: 0.4878
p-érték: 0.4849
Döntés: 0.4849 > 0.05, tehát nem vetjük el H_0-t

2.4 Összefoglaló értékelés

A hibatagok diagnosztikai vizsgálata alapján:
- A várható érték feltétel teljesül (az átlag gyakorlatilag 0)
- A normalitás feltétele teljesül (a hibatagok normális eloszlásúak)
- A függetlenség feltétele nem teljesül, erős pozitív autokorreláció van jelen
- A homoszkedaszticitás feltétele teljesül (a szórás állandó)

3 Box-Jenkins modell

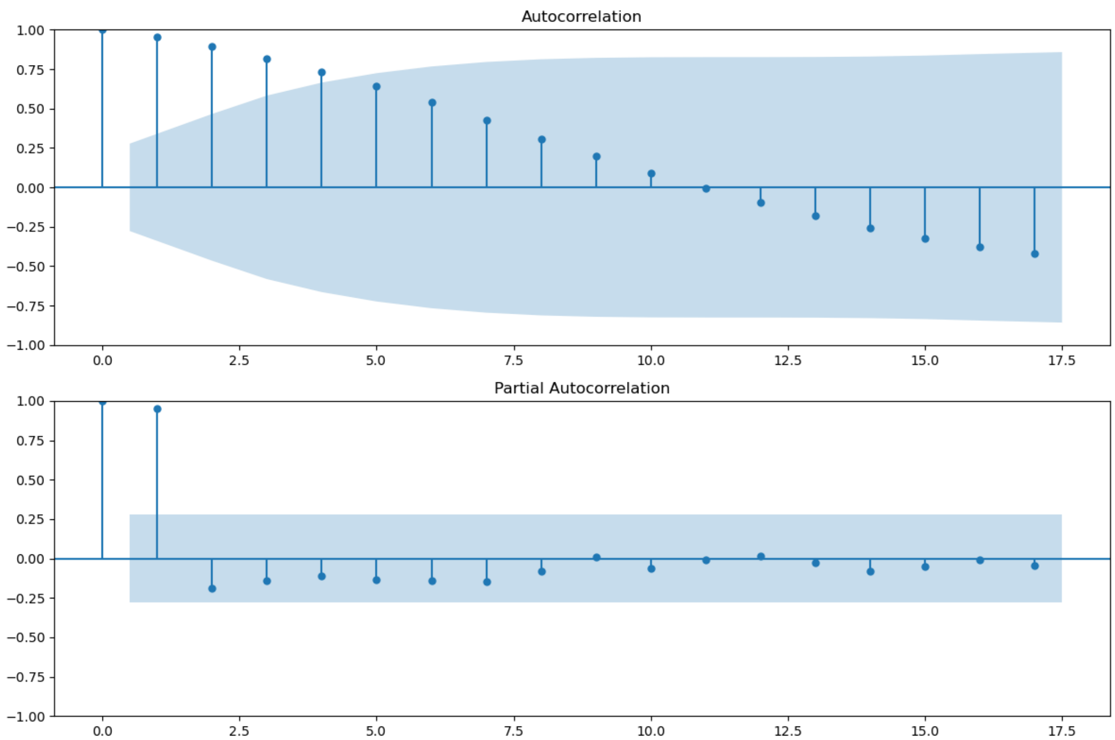
3.1 Kód és eredmények

```
# Idősor stacionaritásának vizsgálata (ADF teszt)
adf_result = adfuller(df['Érték'])
print('\nADF Teszt eredménye:')
print(f'ADF Statisztika: {adf_result[0]:.4f}')
print(f'p-érték: {adf_result[1]:.4f}')
```

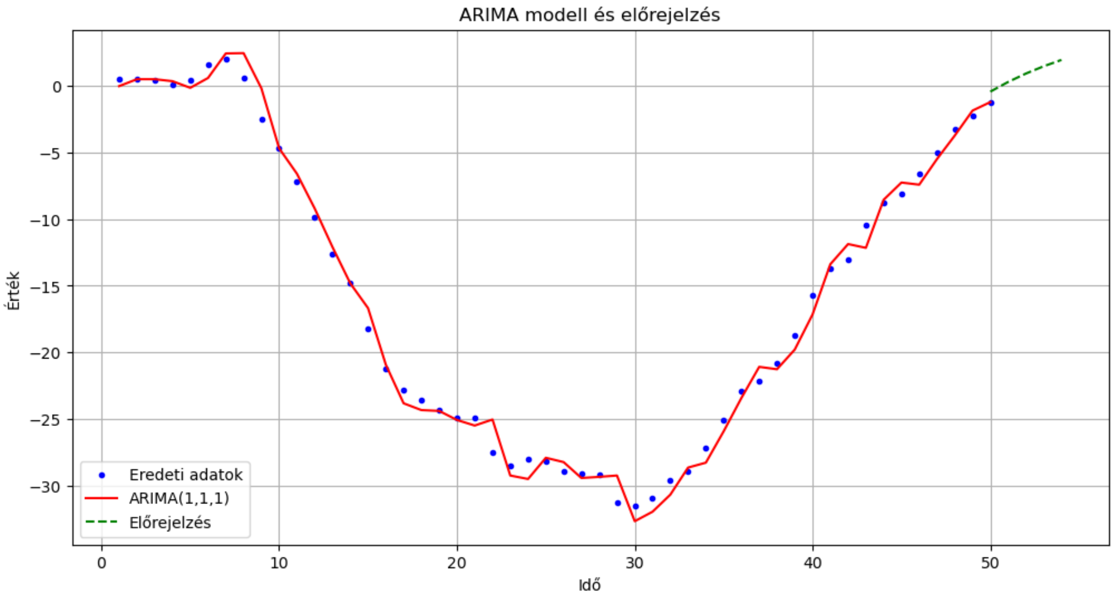
```python
# ACF és PACF ábrák a paraméterek meghatározásához
fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(12, 8))
plot_acf(df['Érték'], ax=ax1)
plot_pacf(df['Érték'], ax=ax2)
plt.tight_layout()
plt.show()

# ARIMA modell illesztése
p, d, q = 1, 1, 1
model = ARIMA(df['Érték'], order=(p, d, q))
results = model.fit()

# Illesztett értékek és előrejelzések
fitted_values = results.fittedvalues
forecast = results.forecast(steps=5)

# Reziduálisok vizsgálata
residuals = results.resid

# 1. Várható érték vizsgálata
resid_mean = np.mean(residuals)
resid_std = np.std(residuals, ddof=1)
t_stat = resid_mean / (resid_std/np.sqrt(len(residuals)))
p_value_mean = 2 * stats.t.cdf(-abs(t_stat), len(residuals)-1)

# 2. Normalitás vizsgálata
shapiro_stat, shapiro_p = stats.shapiro(residuals)

# 3. Függetlenség vizsgálata (Ljung-Box teszt)
lb_stat = sm.stats.diagnostic.acorr_ljungbox(residuals)

# 4. Homoszkedaszticitás vizsgálata
exog = sm.add_constant(fitted_values)
bp_test = sm.stats.diagnostic.het_breuschpagan(residuals, exog)

print('\nModell eredmények:')
print(results.summary().tables[0])
print(results.summary().tables[1])

print('\nHibatagok vizsgálata:')
print(f'Várható érték teszt p-érték: {p_value_mean:.4f}')
print(f'Shapiro-Wilk teszt p-érték: {shapiro_p:.4f}')
print(f"Ljung-Box teszt:")
print("Lag  Teszt statisztika   p-érték")
print("-" * 35)
for lag, row in lb_stat.iterrows():
```

```
    print(f"{lag:2d}   {row['lb_stat']:14.6f}   {row['lb_pvalue']:.6e}")
print(f'\nBreusch-Pagan teszt p-érték: {bp_test[1]:.4f}')

print('\nElőrejelzések:')
for i, pred in enumerate(forecast, 1):
    print(f'{len(df) + i}. időpont: {pred:.2f}')

# Ábrázolás
plt.figure(figsize=(12, 6))
plt.plot(df['Idő'], df['Érték'], 'b.', label='Eredeti adatok')
plt.plot(df['Idő'], fitted_values, 'r-', label=f'ARIMA({p},{d},{q})')
future_points = np.arange(len(df), len(df) + 5)
plt.plot(future_points, forecast, 'g--', label='Előrejelzés')
plt.title('ARIMA modell és előrejelzés')
plt.xlabel('Idő')
plt.ylabel('Érték')
plt.legend()
plt.grid(True)
plt.show()
```

```
ADF Teszt eredménye:
ADF Statisztika: -2.5644
p-érték: 0.1006
```



```
Modell eredmények:
                               SARIMAX Results
==============================================================================
Dep. Variable:                  Érték   No. Observations:                   50
Model:                 ARIMA(1, 1, 1)   Log Likelihood                 -68.798
Date:                Wed, 04 Dec 2024   AIC                            143.597
Time:                        20:18:42   BIC                            149.272
Sample:                             0   HQIC                           145.750
                                 - 50
Covariance Type:                  opg
==============================================================================
==============================================================================
                 coef    std err          z      P>|z|      [0.025      0.975]
------------------------------------------------------------------------------
ar.L1          0.8688      0.090      9.658      0.000       0.692       1.045
ma.L1         -0.2181      0.159     -1.372      0.170      -0.530       0.094
sigma2         0.9504      0.193      4.913      0.000       0.571       1.329
==============================================================================

Hibatagok vizsgálata:
Várható érték teszt p-érték: 0.8523
Shapiro-Wilk teszt p-érték: 0.0627
Ljung-Box teszt:
Lag  Teszt statisztika   p-érték
-----------------------------------
 1          0.212220   6.450331e-01
 2          3.389275   1.836658e-01
 3          3.694513   2.963968e-01
 4          7.022499   1.347040e-01
 5          7.781213   1.687128e-01
 6          7.867512   2.479714e-01
 7         13.683233   5.711086e-02
 8         13.708309   8.969225e-02
 9         15.190585   8.583220e-02
10         16.478098   8.673992e-02

Breusch-Pagan teszt p-érték: 0.5577

Előrejelzések:
51. időpont: -0.41
52. időpont: 0.31
53. időpont: 0.94
54. időpont: 1.49
55. időpont: 1.97
```



3.2 Értelmezés $\varepsilon = 0.05$ szignifikanciaszint mellett

3.2.1 ADF teszt eredménye

H_0: Az idősor nem stacionárius
H_1: Az idősor stacionárius
ADF Statisztika: -2.5644
p-érték: 0.1006
Döntés: $0.1006 > 0.05$, tehát nem vetjük el H_0-t, az idősor nem stacionárius, ezért szükséges differenciálnunk

3.2.2 Modell paraméterek

ARIMA(1,1,1) modellt illesztettünk, ahol:
- p = 1 (autoregresszív tag), mert a PACF ábrán az első késleltetés volt szignifikáns
- d = 1 (differenciálás rendje), mert az idősor nem stacionárius
- q = 1 (mozgóátlag tag), mert az ACF ábra az első késleltetésnél mutat szignifikáns értéket

3.2.3 Paraméterek szignifikanciája

- AR(1) tag: p-érték $= 0.000 < 0.05$, szignifikáns

- MA(1) tag: p-érték $= 0.170 > 0.05$, nem szignifikáns

3.2.4 Modell illeszkedési mutatók

AIC: 143.597
BIC: 149.272
Log Likelihood: -68.798

3.3 Hibatagok tulajdonságainak vizsgálata

3.3.1 Várható érték vizsgálata

p-érték: $0.8523 > 0.05$, tehát nem vetjük el H_0-t

3.3.2 Normalitás vizsgálata (Shapiro-Wilk teszt)

p-érték: $0.0627 > 0.05$, tehát nem vetjük el H_0-t

3.3.3 Függetlenség vizsgálata (Ljung-Box teszt)

Minden késleltetésre p > 0.05, tehát nem vetjük el H_0-t

3.3.4 Homoszkedaszticitás vizsgálata (Breusch-Pagan teszt)

p-érték: $0.5577 > 0.05$, tehát nem vetjük el H_0-t

3.4 Összefoglaló értékelés

- A modell diagnosztikája megfelelő:
  - A várható érték feltétel teljesül
  - A hibatagok normális eloszlásúak
  - A függetlenség feltétele teljesül
  - A homoszkedaszticitás feltétele teljesül
- Az AR(1) tag szignifikáns, míg az MA(1) tag nem
- Az előrejelzések növekvő trendet mutatnak