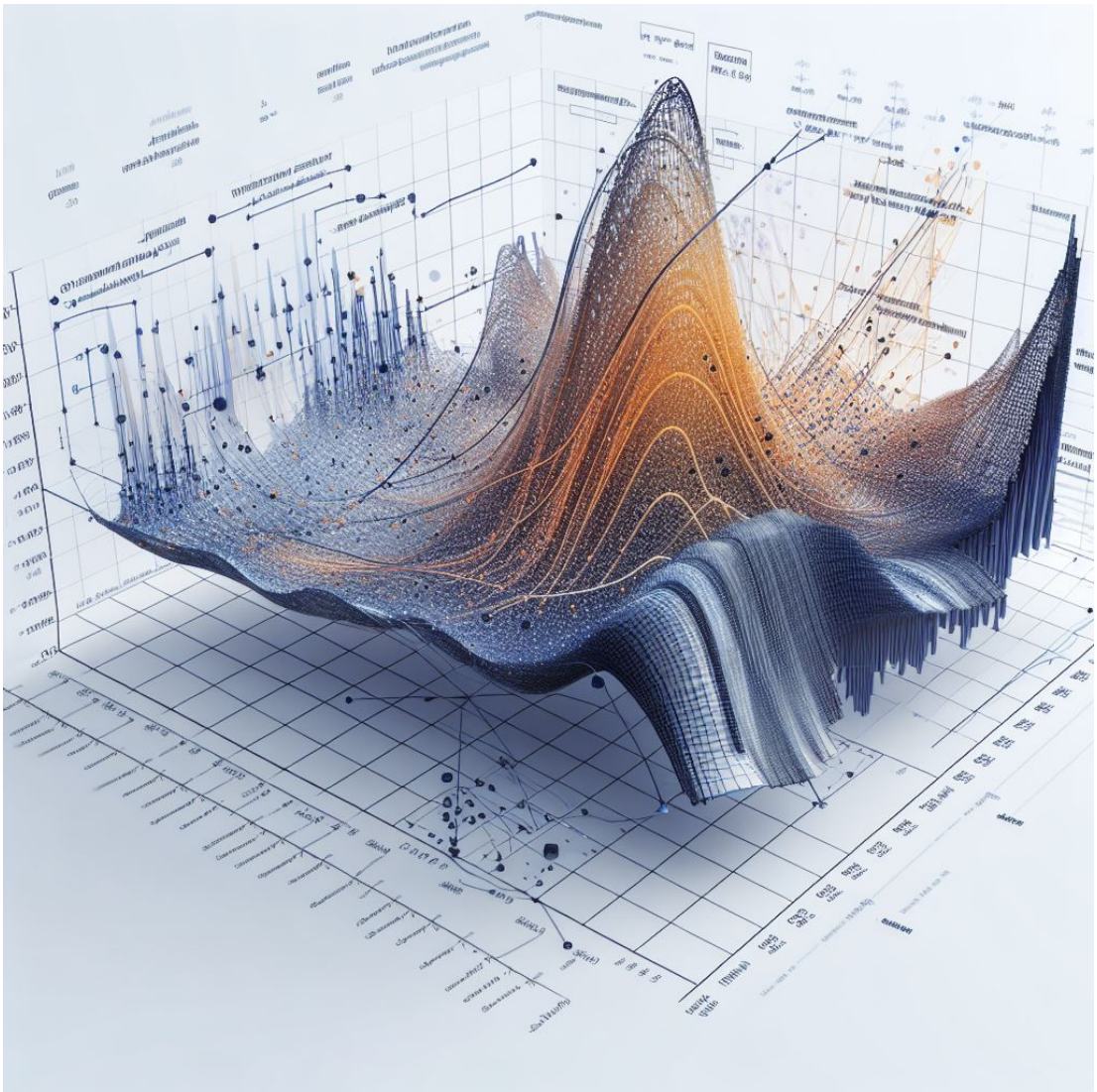
ISTT1003 Rapport H2023

Gruppe 8

Av: 10021, 10121, 10108

Dato: 19.11.2023



Bildet generert av DALL-E

Table of Contents

Sammendrag

Denne oppgaven har undersøkt om man betaler mer for LEGO-produkter som er ikke-originale konsepter, enn de som er originale LEGO-produkter. I et forsøk på å svare på denne problemstillingen ble det tatt i bruk regresjonsanalyse gjennom å bruke multippel lineær regresjon for å finne gode forklaringsvariabler, samt enkel og multippel lineær regresjon for å undersøke sammenhengen mellom pris og forklaringsvariablene. Oppgaven kommer til slutt fram til en konklusjon basert på resultatene av analysen og kvalitetsevalueringen.

Innledning

Gruppen fikk utdelt en oppgave som gikk ut på å analysere data om LEGO, ved hjelp av teori fra faget statistikk (ISTT1003). Gruppen ville diskutere prisendringer basert på om forskjellige LEGO-sett var av originale ideer eller ikke. Problemstillingen så da slik ut: "Er LEGO-sett av ikke-originale konsepter dyrere enn LEGO-sett av originale konsepter?".

Teori

I analysearbeidet har ulike begreper fra statistisk modellering en viktig rolle. Responsvariabler utgjør det som observeres under endringer av forklaringsvariablene. For eksempel kan prisen på et produkt være en responsvariabel der tilbud og etterspørsel kan være forklaringsvariabler. Multippel lineær regresjon brukes til å forutsi responsvariabler basert på flere forklaringsvariabler, i motsetning til enkel lineær regresjon, som bare har en enkelt forklaringsvariabel. Nøyaktigheten til regresjonen kan vurderes ved hjelp av den justerte R^2-verdien, som forteller hvor mye av dataene som er forklart av modellen. (Langaas, 2020). Kvantiler og Q-Q plotter visualiserer datasettets fordeling og residualene i forhold til en normalfordeling, som også kan vurdere usikkerheten til en regresjon. (Sørmoen, 2023) En annen måte å vurdere usikkerheten på er å bruke konfidensintervall for å med stor sannsynlighet vite at en gjennomsnittlig verdi ligger innenfor et visst intervall. En analyse tar utgangspunkt i hypoteser, der en nullhypotese antar ingen forskjell eller sammenheng, i motsetning til en alternativ hypotese som representerer det man søker å bevise eller avdekke gjennom et forsøk. (Langaas, 2020). Disse begrepene danner grunnlaget gruppen har for å finne og analysere resultater.

Databeskrivelse

Før datasettet kunne bli tatt i bruk og analysen kunne starte måtte alle observasjoner som var samlet renses og forberedes. Observasjoner som inneholdt manglende informasjon, ble utelatt fra det endelige datasettet. I det rensede datasettet ble variabelen LegoGroup opprettet og brukt til å skille mellom to typer LEGO-konsepter; LEGO-original og ikke LEGO-original. Denne variabelen ble basert på det overordnede temaet til hvert LEGO-sett. "LEGO original" refererer til konsepter som

ble oppfunnet og utviklet internt av LEGO selv, som for eksempel "LEGO City" eller "LEGO Friends". På den andre siden omfatter "ikke LEGO-original" sett som er basert på andres oppfinnelser eller ideer, for eksempel "Batman" eller "DC". “LEGO Original" fikk verdien 1, mens "Eksterne LEGO-konsepter" fikk verdien 0.

Hypotese

For å undersøke problemstillingen definerte gruppen en nullhypotese som sa at det ikke er noen forskjell i pris mellom originale og ikke-originale LEGO-sett. Det ble også definert en alternativ hypotese som sa at det ville vært dyrere med ikke-originale konsepter sammenlignet med originale konsepter.

Grunnlag for modeller

En måte å sjekke om LegoGroup har en betydning på prisen, er å bruke pris som responsvariabel og LegoGroup som forklaringsvariabel. LegoGroup som forklaringsvariabel alene er forventet å predikere prisen dårlig, siden denne er binær. For å finne andre forklaringsvariabler som passer til modellen, ble det systematisk testet flere kombinasjoner av kontinuerlige forklaringsvariabler for å oppnå best dekning (R^2) og høyt signifikansnivå.

Valget av forklaringsvariabler til kombinasjonstestingen ble gjort ved å kjøre en OLS regresjon med alle mulige forklaringsvariabler og observere hvem som var signifikante for pris, altså hadde en p-verdi < 0.05. Gruppen kom da fram til at forklaringsvariablene skulle være sider og antall brikker, i tillegg til den kategoriske forklaringsvariablen LegoGroup som gruppen opprettet. Disse tre forklaringsvariablene ble satt sammen i en formel med pris som responsvariabel, som ble brukt i resten av modellene.

$$Pris_i = \beta_0 + \beta_{LegoGroup} * x_{LegoGroup} + \beta_{Brikker} * x_{Brikker} + \beta_{Sider} * x_{Sider}$$

Resultater

I modellen (se figur 1) basert på formelen gruppen kom fram til, var det tydelig at alle variablene var signifikante til pris med en p-verdi på godt under 0.05. Modellen viser at koeffisientverdien til LegoGroup er -3.84. Ettersom LegoGroup er en binær variabel, viser en koeffisientverdi på -3.84 at originale LEGO-sett i gjennomsnitt har en responsverdi på 3.84 prisenheter lavere enn ikke LEGO-originale sett.

For å se nærmere på dette resultatet, ble modellen visualisert med et delvis regresjonsplott, hvor det var fokus på pris og LegoGroup. Grafen nedenfor (se figur 2) viser tydelig at det er to grupper hvor regresjonslinjen minker når x-verdien øker. Dette betyr at jo høyere LegoGroup verdien er, jo lavere er prisen på LEGO-settene. Originale konsepter har høyest LegoGroup-verdi i figuren, ergo er LEGO-originale konsepter billigere enn ikke LEGO-originale konsepter ifølge denne modellen. I et delvis regresjonsplott er effektene av antall brikker og antall sider fjernet.

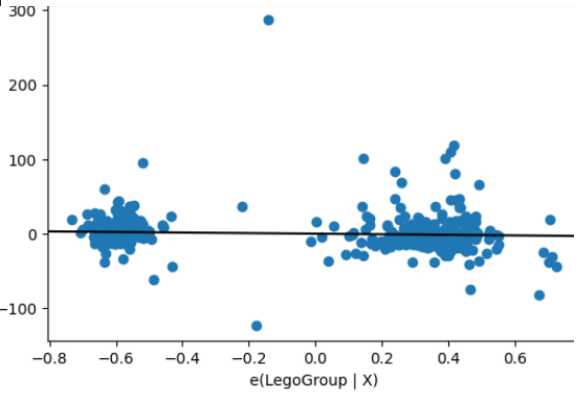
OLS Regression Results

Dep. Variable:	Price	R-squared:	0.848
Model:	OLS	Adj. R-squared:	0.848
Method:	Least Squares	F-statistic:	1324.
Date:	Sat, 18 Nov 2023	Prob (F-statistic):	3.40e-290
Time:	16:05:41	Log-Likelihood:	-3180.3
No. Observations:	714	AIC:	6369.
Df Residuals:	710	BIC:	6387.
Df Model:	3		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	8.5377	1.511	5.651	0.000	5.572	11.504
LegoGroup	-3.8479	1.624	-2.370	0.018	-7.036	-0.660
Pages	0.0801	0.016	5.100	0.000	0.049	0.111
Pieces	0.0771	0.002	31.580	0.000	0.072	0.082

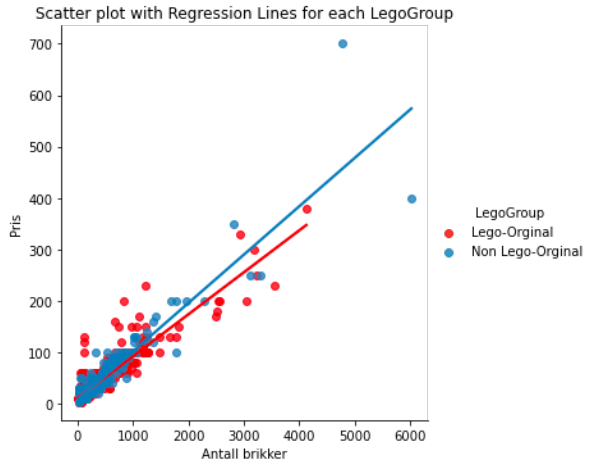
Omnibus:	753.759	Durbin-Watson:	1.773
Prob(Omnibus):	0.000	Jarque-Bera (JB):	98943.187
Skew:	4.558	Prob(JB):	0.00
Kurtosis:	59.945	Cond. No.	1.78e+03

Figur 1

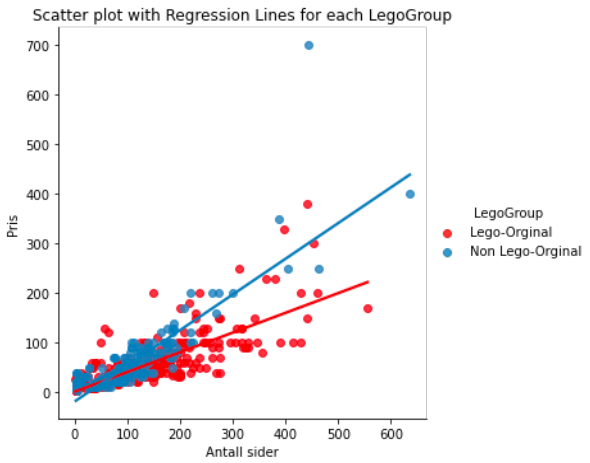


Figur 2

Det ble i tillegg tatt i bruk en felles modell med egne skjæringspunkt og stigningstall for hver gruppe, som ble brukt for å visualisere sammenhengen mellom pris og antall brikker (se figur 3), og pris og antall sider (se figur 4). I begge grafene kan man se at LEGO av kategorien "ikke LEGO original" (blå linje) har et større stigningstall enn kategorien “LEGO original” (rød linje).



Figur 3

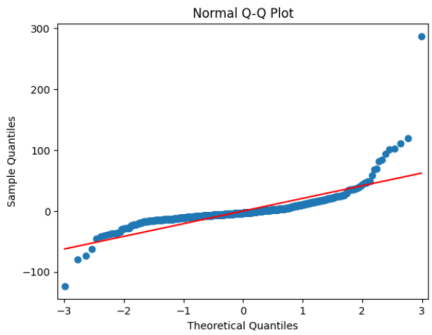


Figur 4

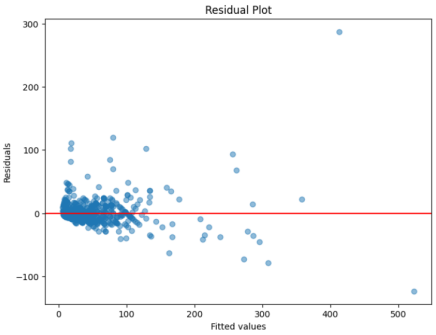
Analysene av koeffisientene, delvis regresjonsplott og felles modellen styrker alle den alternative hypotesen, og en konklusjon kan bli lagd basert på observasjonene om modellene er pålitelige.

Evaluering av modellene

For å sikre kvaliteten av modellen, ble modellen visualisert ved hjelp av grafiske verktøy. Den første visualiseringen som ble gjort var å benytte et QQ-plott. I QQ-plottet plottes teoretiske normalfordelte kvantiler mot de faktiske residualene i modellen. Residualene er forskjellen i faktiske observasjoner og modellresultatene for hver observasjon i datasettet. I QQ-plottet (se figur 5) kan man se at de fleste punktene ligger langs linjen.

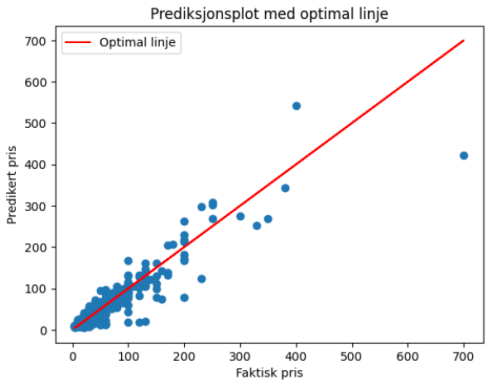


Figur 5



Figur 6

Ettersom de fleste punktene ligger langs linjen, tyder det på at forutsetningen om en normalfordeling for residualene i modellen er oppfylt. Dette styrker modellens kvalitet og troverdighet. Selv om de fleste punktene ligger langs linjen, er det likevel et stort avvik i de høyere kvantilene, og et lite avvik i de lavere kvantilene, som kalles tunge haler (Sørmoen, 2023). Dette kan tyde på at modellen ikke predikerer pris særlig godt for de dyreste og billigste LEGO-settene. Når gruppen så nærmere på disse forskjellene mellom modell og observert pris, ble det oppdaget at antall brikker og sider varierer veldig med prisen i høyere og lavere prisklasser. For eksempel viste det seg at i det dyreste LEGO-settet «Imperial Star Destroyer» hadde 4784 antall brikker og 444 antall sider, mens i det nest dyreste LEGO-settet «Hogwarts Castle» var antall brikker 6020 og antall sider 636. Basert på denne observasjonen konkluderes det med at brikker og sider ikke er særlig egnet til å predikere de dyreste og billigste LEGO-settene. Det påvirker likevel ikke konklusjonen om at LegoGroup er en viktig forklaringsvariabel for prisen på et LEGO-sett. I tillegg viser residualplottet (se figur 6), en tilnærmet jevn fordeling av residualene rundt null, noe som indikerer at modellen ikke systematisk undervurderer eller overvurderer prisene. Dette styrker modellens troverdighet.



Figur 7

Grafen til høyre (se figur 7) viser et plott av predikerte priser mot observerte priser. X-aksen står for den faktiske prisen på LEGO-settet og y-aksen står for den predikerte prisen. Grafen gir et visuelt inntrykk på modellens styrke, hvor linjen viser optimal predikering. Grafen viser at flere punkter ligger langt utenfor den optimale linjen, noe som kan indikere at modellen har vanskeligheter med å forutsi disse spesifikke prisene med god nok presisjon. Modellen vil kunne gi dårlige prediksjoner i de høyere prisklassene, slik som QQ-plottet også viste. Dette kommer av få observasjoner i de høyere prisklassene som påvirker nøyaktigheten til modellen. For de lavere prisene vil modellen predikere prisen bedre. Det er likevel tydelig at observasjonene følger linjens stigning, som vil si at prediksjonsevnen til modellen er pålitelig.

Det ble vurdert å fjerne observasjoner som hadde en negativ påvirkning på påliteligheten til modellen. Det å fjerne disse observasjonene ville føre til en bedre modell som kunne predikere typiske prissatte LEGO-sett bedre. Likevel ble det valgt å beholde dataen for å unngå cherry-picking. Dersom disse variablene hadde blitt fjernet ville modellen sannsynligvis blitt noe bedre for majoriteten av dataen, men det ville også tvunget fram en modell som egnet seg mer til vår problemstilling. Evalueringen av datapunktene som var med, var derfor viktig for modellens utfall.

Til tross for at de nevnte unøyaktighetene, spesielt i høyere prisklasser, kan modellen brukes for å slå fast en konklusjon og besvare problemstillingen. Det kan begrunnes med at modellen har en høy justert R^2 og tilfredsstillende resultater av de visuelle verktøyene for majoriteten av observasjonene.

Konklusjon

Basert på resultatene og evaluering av data og modeller, har gruppen grunnlag for å forkaste nullhypotesen til fordel for den alternative hypotesen. Problemstillingen kan dermed besvares med at LEGO-sett av ikke-originale konsepter er dyrere enn originale konsepter. h

Kilder

Langaas, M. (2020). *IST[A/G/T]1003: Statistisk læring og data science* [Kompendium]. IMF/NTNU. https://www.math.ntnu.no/emner/IST100x/ISTx1003/Regresjon.html

Sørmoen, I. (2023). *Statistisk læring og data science Forelesning 2: Introduksjon og multippel lineær regresjon*. [Lysarkpresentasjon]. Institutt for matematiske fag, NTNU. Presentasjon

Vedlegg

oppg1

November 19, 2023

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from scipy import stats
import statsmodels.formula.api as smf
import statsmodels.api as sm

from sklearn.metrics import confusion_matrix
```

[42]:
```
df = pd.read_csv("data/lego.population.csv", sep = ",", encoding = "latin1")
df.head()
df
```

[42]:

	Item_Number	Set_Name	Theme	Pieces	\
0	41916	Extra Dots - Series 2	DOTS	109.0	
1	41908	Extra Dots - Series 1	DOTS	109.0	
2	11006	Creative Blue Bricks	Classic	52.0	
3	11007	Creative Green Bricks	Classic	60.0	
4	41901	Funky Animals Bracelet	DOTS	33.0	
…	…	…	…	…	
1299	45678	SPIKE Prime Set	LEGO® Education	528.0	
1300	71367	Mario's House & Yoshi	LEGO® Super Mario	205.0	
1301	71368	Toad's Treasure Hunt	LEGO® Super Mario	464.0	
1302	71369	Bowser's Castle Boss Battle	LEGO® Super Mario	1010.0	
1303	71371	Propeller Mario Power-Up Pack	LEGO® Super Mario	13.0	

	Price	Amazon_Price	Year	Ages	Pages	Minifigures	Packaging	\
0	$3.99	$3.44	2020	Ages_6+	NaN	NaN	Foil pack	
1	$3.99	$3.99	2020	Ages_6+	NaN	NaN	Foil pack	
2	$4.99	$4.93	2020	Ages_4+	37.0	NaN	Box	
3	$4.99	$4.93	2020	Ages_4+	37.0	NaN	Box	
4	$4.99	$4.99	2020	Ages_6+	NaN	NaN	Foil pack	
…	…	…	…	…	…	…	…	
1299	$329.95	NaN	2020	Ages_10+	NaN	2.0	NaN	
1300	$29.99	NaN	2020	Ages_6+	NaN	2.0	Box	

```
1301   $69.99           NaN  2020   Ages_8+    NaN          4.0        Box
1302   $99.99           NaN  2020   Ages_8+    NaN          NaN        Box
1303    $9.99           NaN  2020   Ages_6+    NaN          NaN        Box

     Weight  Unique_Pieces Availability   Size
0       NaN            6.0       Retail  Small
1       NaN            6.0       Retail  Small
2       NaN           28.0       Retail  Small
3       NaN           36.0       Retail  Small
4       NaN           10.0       Retail  Small
…       …              …           …      …
1299    NaN          108.0          NaN  Small
1300    NaN          114.0       Retail  Small
1301    NaN          195.0       Retail  Small
1302    NaN          346.0       Retail  Small
1303    NaN           11.0       Retail  Small

[1304 rows x 15 columns]
```

[43]:
```
# fjerner forklaringsvariabler vi ikke trenger
df2 = df[['Set_Name', 'Theme', 'Pieces', 'Price', 'Pages', 'Minifigures',␣
 ↪'Unique_Pieces']]

# fjerner observasjoner med manglende datapunkter (I virkeligheten er det␣
 ↪risikabelt å fjerne manglende data punkter uten videre. Dette kan påvirke␣
 ↪resultatene på en måte som er vanskelig å avdekke.)
# Det finnes metoder som tar hensyn til tomme dataceller men dette prosjektet␣
 ↪har ikke dette som en del av oppgaven, slik at vi velger å ta utgangspunkt i␣
 ↪det originale datasettet.
df2 = df2.dropna()

# gjør themes om til string og fjern alle tegn vi ikke vil ha med
df2['Theme'] = df2['Theme'].astype(str)
df2['Theme'] = df2['Theme'].str.replace(r'[^a-zA-Z0-9\s-]', '', regex = True)

# fjerner dollartegn og trademark-tegn fra datasettet
df2['Price'] = df2['Price'].str.replace('\$', '', regex = True)

# og gjør så prisen om til float
df2['Price'] = df2['Price'].astype(float)

df2

df2['LegoGroup'] = np.where(df2['Theme'].isin(['Harry Potter', 'Star Wars',␣
 ↪'Batman', 'DC', 'Disney', 'Marvel', 'Minecraft', 'LEGO Frozen 2', 'Trolls␣
 ↪World Tour', 'Minions', 'Powerpuff Girls', 'Jurassic World', 'Overwatch',␣
 ↪'Spider-Man', 'Stranger Things']), 0,
```

```
                    np.where(df2['Theme'].isin(['City', 'Friends',␣
 ↪'Unikitty', 'NINJAGO', 'DUPLO', 'THE LEGO MOVIE 2', 'Speed Champions',␣
 ↪'Hidden Side', 'Classic', 'Juniors', 'Creator 3-in-1', 'Ideas', 'Creator␣
 ↪Expert', 'Monkie Kid', 'Minifigures', 'Powered UP']), 1, 2))
df2.groupby(['LegoGroup']).size().reset_index(name = 'Count')
```

```
<>:13: SyntaxWarning: invalid escape sequence '\$'
<>:13: SyntaxWarning: invalid escape sequence '\$'
C:\Users\vikto\AppData\Local\Temp\ipykernel_30304\3269387580.py:13:
SyntaxWarning: invalid escape sequence '\$'
  df2['Price'] = df2['Price'].str.replace('\$', '', regex = True)
```

[43]:

	LegoGroup	Count
0	0	282
1	1	432

1 Pris som responsvariabel

[44]:
```
formel = 'Price ~ LegoGroup + Pages +  Pieces + Minifigures + Unique_Pieces'

modell = smf.ols(formel, data = df2)
resultat = modell.fit()
```

[45]:
```
print(resultat.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Price   R-squared:                       0.849
Model:                            OLS   Adj. R-squared:                  0.848
Method:                 Least Squares   F-statistic:                     796.2
Date:                Sun, 19 Nov 2023   Prob (F-statistic):          8.75e-288
Time:                        21:05:38   Log-Likelihood:                -3178.6
No. Observations:                 714   AIC:                             6369.
Df Residuals:                     708   BIC:                             6397.
Df Model:                           5
Covariance Type:            nonrobust
==============================================================================
=
                 coef    std err          t      P>|t|      [0.025
0.975]
------------------------------------------------------------------------------
-
Intercept      8.2712      1.732      4.776      0.000       4.871
11.672
LegoGroup     -3.6380      1.629     -2.234      0.026      -6.835
-0.441
```

```
Pages             0.0953      0.021      4.562      0.000       0.054
0.136
Pieces            0.0784      0.003     27.353      0.000       0.073
0.084
Minifigures       0.6685      0.434      1.541      0.124      -0.183
1.520
Unique_Pieces    -0.0285      0.020     -1.444      0.149      -0.067
0.010
==============================================================================
Omnibus:                      737.256   Durbin-Watson:                   1.753
Prob(Omnibus):                  0.000   Jarque-Bera (JB):           100662.839
Skew:                           4.363   Prob(JB):                         0.00
Kurtosis:                      60.511   Cond. No.                     1.94e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 1.94e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

2 Visualiseringer av dataen

[46]:
```
formel = 'Price ~  Pages + Pieces + LegoGroup'

modell = smf.ols(formel, data = df2)
resultat = modell.fit()

print(resultat.summary())
```

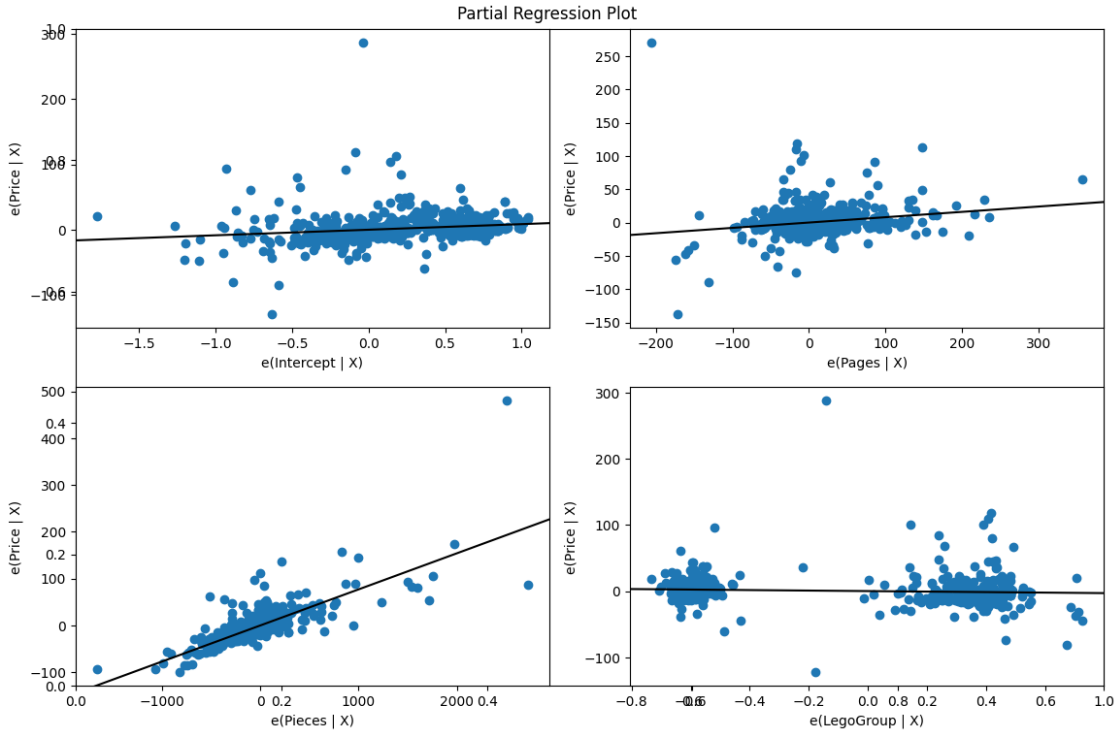
```
                            OLS Regression Results
==============================================================================
Dep. Variable:                  Price   R-squared:                       0.848
Model:                            OLS   Adj. R-squared:                  0.848
Method:                 Least Squares   F-statistic:                     1324.
Date:                Sun, 19 Nov 2023   Prob (F-statistic):          3.40e-290
Time:                        21:05:38   Log-Likelihood:                -3180.3
No. Observations:                 714   AIC:                             6369.
Df Residuals:                     710   BIC:                             6387.
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
Intercept      8.5377      1.511      5.651      0.000       5.572      11.504
Pages          0.0801      0.016      5.100      0.000       0.049       0.111
Pieces         0.0771      0.002     31.580      0.000       0.072       0.082
```

```
LegoGroup      -3.8479      1.624     -2.370      0.018     -7.036     -0.660
==============================================================================
Omnibus:                      753.759   Durbin-Watson:                   1.773
Prob(Omnibus):                  0.000   Jarque-Bera (JB):            98943.187
Skew:                           4.558   Prob(JB):                         0.00
Kurtosis:                      59.945   Cond. No.                     1.78e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly
specified.
[2] The condition number is large, 1.78e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

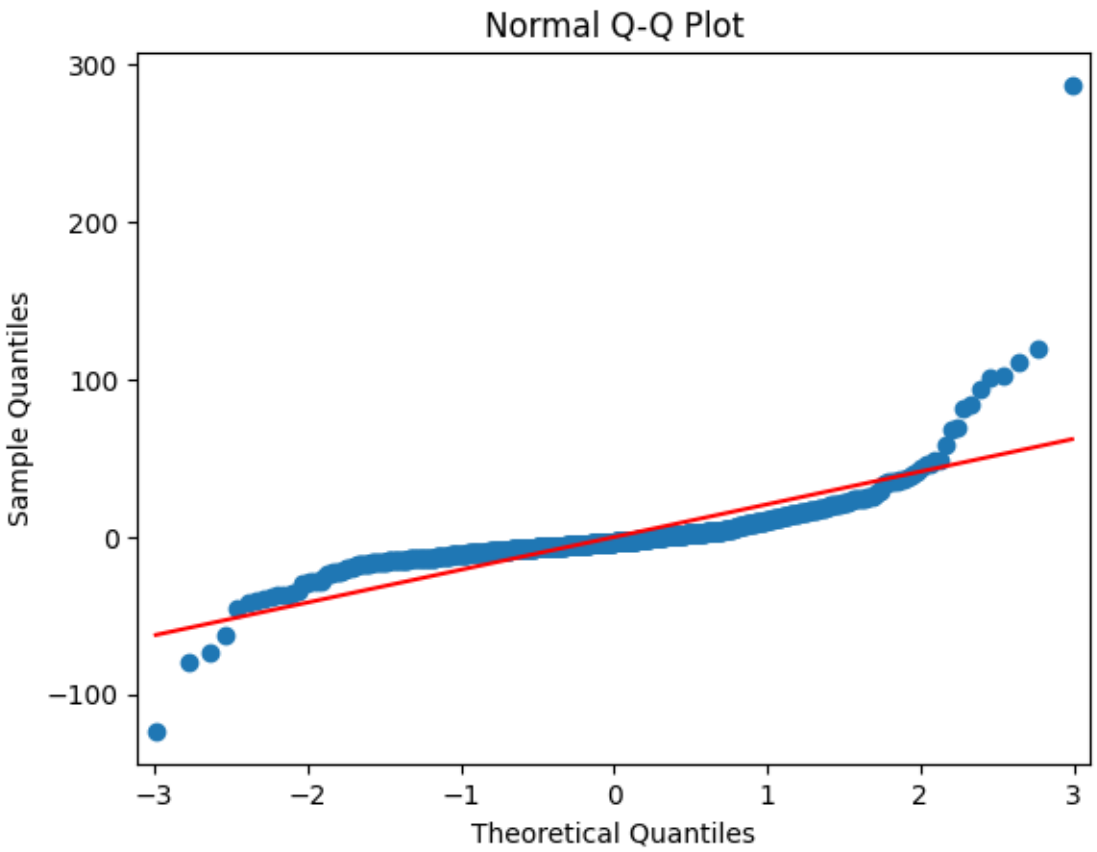
[47]:

```python
from statsmodels.graphics.regressionplots import plot_partregress_grid
fig, ax = plt.subplots(figsize=(12, 8))
plot_partregress_grid(resultat, fig=fig)
plt.suptitle('Partial Regression Plot')
plt.show()
```

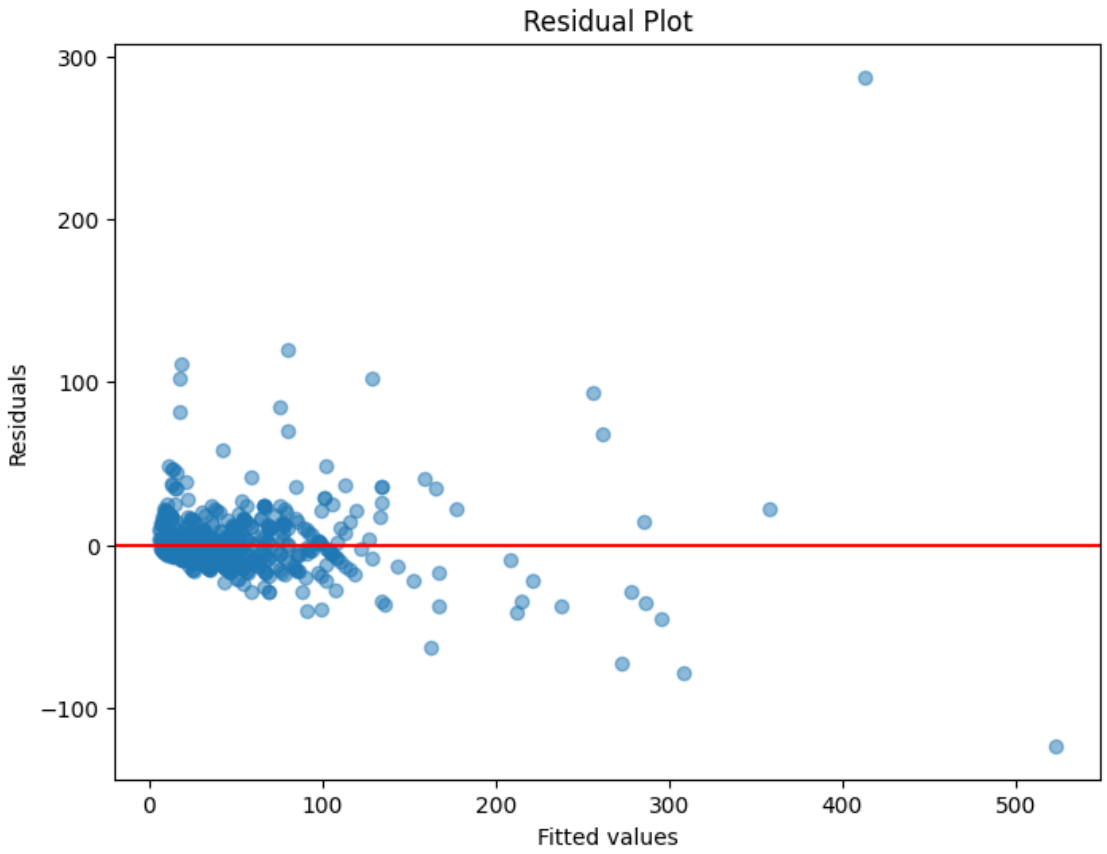
```
c:\Users\vikto\anaconda3\envs\venv\Lib\site-
packages\statsmodels\graphics\regressionplots.py:430: FutureWarning:
Series.__getitem__ treating keys as positions is deprecated. In a future
version, integer keys will always be treated as labels (consistent with
DataFrame behavior). To access a value by position, use `ser.iloc[pos]`
  fig = abline_plot(0, fitted_line.params[0], color='k', ax=ax)
c:\Users\vikto\anaconda3\envs\venv\Lib\site-
packages\statsmodels\graphics\regressionplots.py:430: FutureWarning:
Series.__getitem__ treating keys as positions is deprecated. In a future
version, integer keys will always be treated as labels (consistent with
DataFrame behavior). To access a value by position, use `ser.iloc[pos]`
  fig = abline_plot(0, fitted_line.params[0], color='k', ax=ax)
c:\Users\vikto\anaconda3\envs\venv\Lib\site-
packages\statsmodels\graphics\regressionplots.py:430: FutureWarning:
Series.__getitem__ treating keys as positions is deprecated. In a future
version, integer keys will always be treated as labels (consistent with
DataFrame behavior). To access a value by position, use `ser.iloc[pos]`
  fig = abline_plot(0, fitted_line.params[0], color='k', ax=ax)
c:\Users\vikto\anaconda3\envs\venv\Lib\site-
packages\statsmodels\graphics\regressionplots.py:430: FutureWarning:
Series.__getitem__ treating keys as positions is deprecated. In a future
version, integer keys will always be treated as labels (consistent with
DataFrame behavior). To access a value by position, use `ser.iloc[pos]`
  fig = abline_plot(0, fitted_line.params[0], color='k', ax=ax)
```



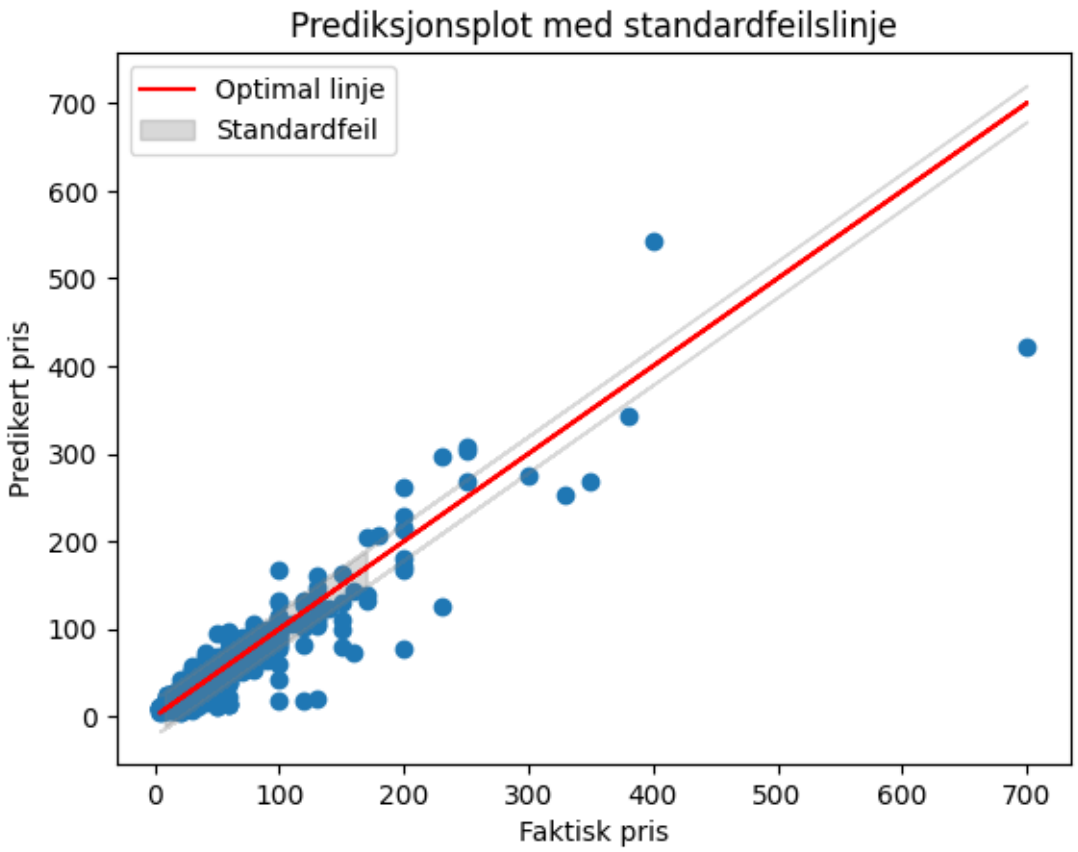
[48]:
```
# Normal Q-Q plot
sm.qqplot(resultat.resid, line='s')  # 's' gir en referanselinje for standard␣
↪normal distribusjon
plt.title('Normal Q-Q Plot')
plt.show()
```



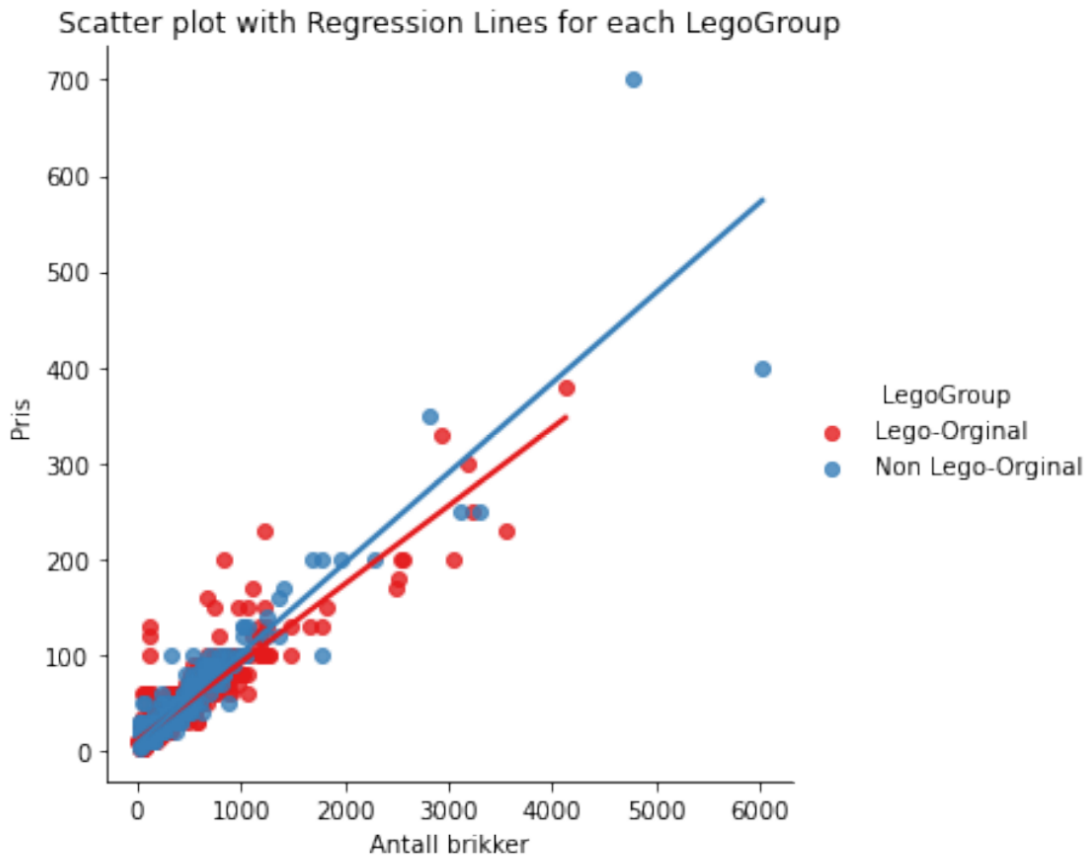
[49]:
```
# Hent residualene og tilpassede verdiene
residuals = resultat.resid
fitted = resultat.fittedvalues

# Residual plot
plt.figure(figsize=(8, 6))
plt.scatter(fitted, residuals, alpha=0.5)
plt.axhline(y=0, color='r', linestyle='-')
plt.title('Residual Plot')
plt.xlabel('Fitted values')
plt.ylabel('Residuals')
plt.show()
```



[50]:
```
# Beregn standardfeilen for prediksjonen
residuals = df2['Price'] - predicted_values
standard_error = residuals.std()

# Lag øvre og nedre grenser basert på standardfeilen
upper_bound = df2['Price'] + standard_error
lower_bound = df2['Price'] - standard_error

# Plot scatterplott og optimal linje
plt.scatter(df2['Price'], predicted_values)
plt.plot(df2['Price'], df2['Price'], color='red', label='Optimal linje')  #␣
 ↪Optimal linje - y = x
plt.fill_between(df2['Price'], upper_bound, lower_bound, color='grey', alpha=0.
 ↪3, label='Standardfeil')
plt.xlabel('Faktisk pris')
plt.ylabel('Predikert pris')
plt.title('Prediksjonsplot med standardfeilslinje')
plt.legend()
plt.show()
```


Prediksjonsplot med standardfeilslinje
Optimal linje
Standardfeil
Predikert pris
Faktisk pris
0
100
200
300
400
500
600
700

```
# Display the regression equation on the plot
for lg_group in df2['LegoGroup'].unique():
    subset = df2[df2['LegoGroup'] == lg_group]
    slope, intercept = resultatt.params['Pieces'], resultatt.params['Intercept']
    #plt.text(subset['NumPieces'].mean(), slope * subset['Pieces'].mean() +␣
 ↪intercept, f'{lg_group} fit', ha='center', va='center')
    plt.title('Scatter plot with Regression Lines for each LegoGroup')
    plt.xlabel('Antall brikker')
    plt.ylabel('Pris')
    plt.show()
```



```
# Create a scatter plot with regression lines for each LegoGroup
plt.figure(figsize=(10, 6))
sns.lmplot(x='Pages', y='Price', hue='LegoGroup', data=df2, palette='Set1',␣
 ↪aspect=1, ci=None, markers='o')  # 'o' for circles

# Display the regression equation on the plot
for lg_group in df2['LegoGroup'].unique():
    subset = df2[df2['LegoGroup'] == lg_group]
    slope, intercept = resultatt.params['Pieces'], resultatt.params['Intercept']
    #plt.text(subset['NumPieces'].mean(), slope * subset['Pieces'].mean() +␣
 ↪intercept, f'{lg_group} fit', ha='center', va='center')
    plt.title('Scatter plot with Regression Lines for each LegoGroup')
    plt.xlabel('Antall sider')
    plt.ylabel('Pris')
    plt.show()
```