VIRGINIA COMMONWEALTH UNIVERSITY

Statistical analysis and modelling (SCMA 632)

A3- Regression Analysis and Diagnostics

Daniel Joe Gasper

V01151514

Date of Submission: 25/06/2025

CONTENTS

INTRODUCTION

a) Multiple Regression Analysis on NSSO68 Data

In this section, we perform a multiple regression analysis using the NSSO68 dataset to understand the factors influencing food consumption expenditure across households. By incorporating socio-economic variables such as income, age, education, and ration card possession, we aim to quantify their impact on food expenditure. Following the regression, we carry out thorough diagnostics to validate model assumptions, identify any violations, and apply necessary corrections. This approach ensures robust and reliable inference, enabling us to interpret the significant determinants of food consumption and understand the improvements made after model refinement.

b) Relationship Between IPL Player Performance and Salary

The Indian Premier League (IPL) is a premier cricket tournament where player performance and remuneration are closely linked. This part of the analysis investigates the relationship between players' on-field performance metrics and the payments they receive over the last three years. Using regression analysis on IPL data, we quantify how performance indicators such as runs scored and wickets taken influence player salaries. This study not only highlights the economic valuation of player skills but also explores trends and variations in this relationship over recent seasons, providing insights into the dynamics of sports economics and talent valuation.

Business Problems Addressed by Data Analysis:

a) Multiple Regression Analysis on NSSO68 Data (Food Consumption)

1. Government agencies need to optimize food subsidy allocation by identifying the key socio-economic factors that significantly influence household food consumption, ensuring efficient resource distribution to reduce food insecurity.
2. Accurate prediction of household food consumption patterns based on demographic and economic variables is essential for forecasting nutritional demand and designing effective public health policies.

b) IPL Player Performance and Salary Relationship

1. IPL franchises require data-driven player valuation models that incorporate performance

metrics to optimize salary budgets, balancing payments between established stars and emerging talents.

2. Understanding how player performance over multiple IPL seasons affects salary progression helps franchises manage contract renewals and auction strategies, addressing disparities between remuneration and on-field contributions.

Benefits of Data Analysis:

a) Benefits of Multiple Regression Analysis on NSSO68 Food Consumption Data

- Targeted Policy Design: Multiple regression helps identify the most significant socio-economic factors influencing household food consumption, enabling policymakers to design targeted food subsidy and nutrition programs that efficiently reach vulnerable populations.
- Improved Resource Allocation: By quantifying the impact of variables like income, education, and ration card possession on food expenditure, governments can allocate resources more effectively to reduce food insecurity and malnutrition.
- Predictive Insights for Planning: Regression models allow forecasting of food demand patterns based on demographic and economic indicators, supporting long-term planning in agriculture, health, and social welfare sectors.
- Understanding Complex Relationships: Multivariate regression captures the combined effects of multiple factors, revealing nuanced insights into consumption behavior that simple analyses might miss.

b) Benefits of Analyzing IPL Player Performance and Salary Relationship

- Data-Driven Salary Structuring: Regression analysis quantifies how player performance metrics (runs, wickets, consistency) influence salaries, helping franchises make informed decisions on player valuations and salary negotiations.
- Optimizing Team Investments: Understanding performance-salary dynamics over multiple years enables franchises to identify undervalued talent and optimize auction and contract strategies, improving team competitiveness and financial efficiency.
- Trend Analysis and Forecasting: Analyzing salary-performance relationships over recent seasons reveals evolving valuation trends, aiding strategic planning in player development and retention.

ANALYSIS USING PYTHON

1) **a) Perform Multiple regression analysis, carry out the regression diagnostics, and explain your findings. Correct them and revisit your results and explain the significant differences you observe**

```
OLS Regression Results for GOA:
                            OLS Regression Results
==============================================================================
Dep. Variable:            foodtotal_q   R-squared:                       0.281
Model:                            OLS   Adj. R-squared:                  0.273
Method:                 Least Squares   F-statistic:                     34.43
Date:                Wed, 25 Jun 2025   Prob (F-statistic):           1.03e-29
Time:                        20:14:49   Log-Likelihood:                -1411.5
No. Observations:                 447   AIC:                             2835.
Df Residuals:                     441   BIC:                             2860.
Df Model:                           5
Covariance Type:            nonrobust
==========================================================================================
                             coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------------------
const                    10.7034      3.921      2.730      0.007       2.998      18.409
MPCE_MRP                  0.0017      0.000     10.791      0.000       0.001       0.002
Age                       0.0182      0.024      0.771      0.441      -0.028       0.065
Meals_At_Home             0.1113      0.052      2.137      0.033       0.009       0.214
Possess_ration_card      -3.1942      1.372     -2.329      0.020      -5.890      -0.498
Education                 0.2843      0.095      3.003      0.003       0.098       0.470
==============================================================================
Omnibus:                       37.912   Durbin-Watson:                   1.707
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              111.911
Skew:                           0.357   Prob(JB):                     5.00e-25
Kurtosis:                       5.345   Cond. No.                     5.06e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 5.06e+04. This might indicate that there are
strong multicollinearity or other numerical problems.

Variance Inflation Factor (VIF):
               feature         VIF
0                const  209.359386
1            MPCE_MRP    1.185270
2                  Age    1.333123
3       Meals_At_Home    1.156725
4  Possess_ration_card    1.095260
5            Education    1.365420
```

1. Overall Model Fit: The model explains 28.1% of variation in food consumption (R-squared=0.281), which is modest but acceptable for social science data.
2. Significant Predictors:
    - Income (MPCE_MRP): Strongest influence - every ₹1 increase in monthly spending leads to 0.0017 unit increase in food consumption (highly significant,

p<0.001).

- Education: More educated households spend more on food (0.28 units per education level, p=0.003).
- Meals at Home: Each additional meal eaten at home increases food spending by 0.11 units (p=0.033).
- Ration Cards: Households with ration cards spend ₹3.19 less on food (likely because they get subsidized food, p=0.020).

3. Non-significant Factor:
    - Age doesn't significantly affect food spending (p=0.441).

Potential Issues:

- The high condition number (50,600) suggests possible multicollinearity (predictors being related), but VIF values are all low (<1.37), so it's likely not a major problem.
- The error terms aren't perfectly normally distributed (Omnibus/Jarque-Bera tests significant).

Interpretation:

In Goa, richer, more educated households that eat more meals at home tend to spend more on food, while ration card holders spend less. Age doesn't matter much. The model has some limitations but identifies clear economic and behavioral patterns in food expenditure.

```
# Assuming 'model' is your fitted OLS regression model for GOA
coefficients = model.params

# Construct the regression equation string
equation = f"y = {round(coefficients.iloc[0], 2)}"  # Intercept term
for i in range(1, len(coefficients)):
    equation += f" + {round(coefficients.iloc[i], 6)}*x{i}"

print(equation)
```

```
y = 10.7 + 0.001718*x1 + 0.018237*x2 + 0.111334*x3 + -3.194201*x4 + 0.284257*x5
```

This equation predicts **household food consumption (y)** based on different factors (**x1 to x5**) from the NSSO data for Goa.

- **y** = Total food consumption (dependent variable)
- **x1** = **MPCE_MRP** (Monthly Per Capita Expenditure at market prices) → **Income level**

- **x2** = **Age** of the household head
- **x3** = **Meals_At_Home** → How many meals are eaten at home
- **x4** = **Possess_ration_card** → Whether the household has a ration card (1=Yes, 0=No)
- **x5** = **Education** level of the household head

Interpreting the Coefficients

1. **Intercept (10.7)**: If all other factors were zero, baseline food consumption is ~10.7 units.
2. **Income (x1)**: For every ₹1 increase in income, food spending increases by **0.0017 units**.
    - (In practical terms, a ₹1000 increase → +1.7 units of food consumption)
3. **Age (x2)**: Not significant (p=0.441), so age has little effect.
4. **Meals at Home (x3)**: Each extra meal eaten at home → **+0.11 units** of food spending.
5. **Ration Card (x4)**: Having a ration card **reduces** food spending by **3.19 units** (due to subsidies).
6. **Education (x5)**: Higher education → **+0.28 units** of food spending per education level.

Conclusion:

- **Income and education** increase food spending.
- **Ration cards reduce spending** (due to subsidies).
- **Age doesn't matter much** in this model.
- **Eating more meals at home** leads to higher food expenditure.

```
# Display the first values of selected columns
print(subset_goa['MPCE_MRP'].head(1).values[0])
print(subset_goa['Age'].head(1).values[0])
print(subset_goa['Meals_At_Home'].head(1).values[0])
print(subset_goa['Possess_ration_card'].head(1).values[0])
print(subset_goa['Education'].head(1).values[0])
print(subset_goa['foodtotal_q'].head(1).values[0])
```

```
8191.46
56
58.0
1.0
12.0
26.8506
```

Plugging in these values in the equation:

- **Predicted foodtotal_q (y)** = $10.7 + (0.001718 \times 8191.46) + (0.018237 \times 56) + (0.111334 \times 58) - (3.194201 \times 1) + (0.284257 \times 12)$

Calculating step-by-step:

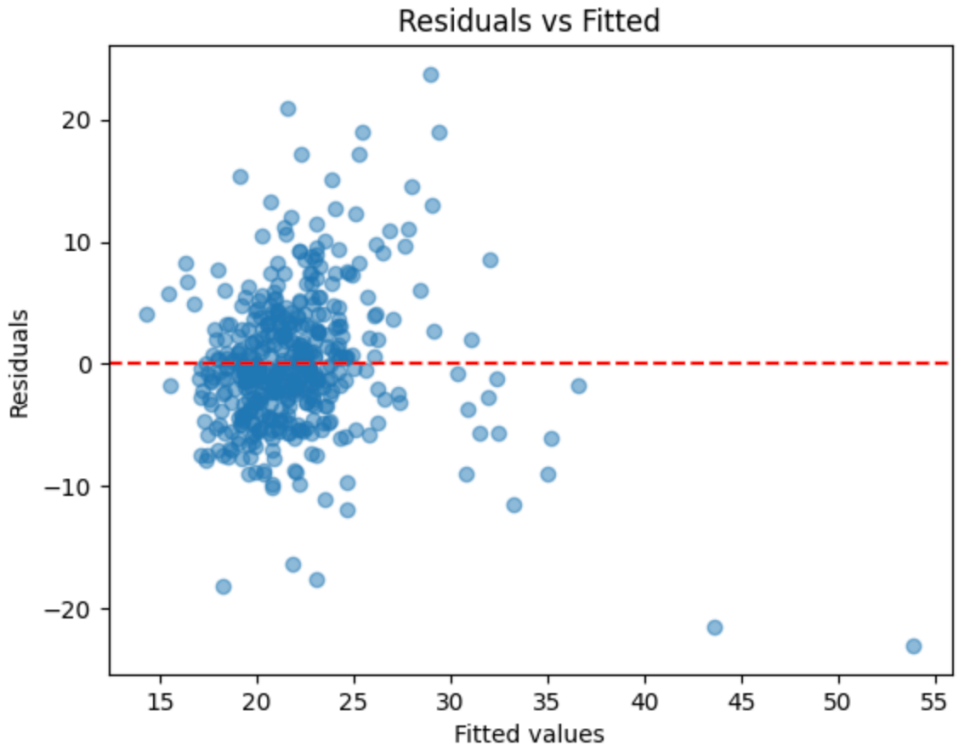
1. **Income effect** = $0.001718 \times 8191.46 \approx 14.07$
2. **Age effect** = $0.018237 \times 56 \approx 1.02$
3. **Meals at home effect** = $0.111334 \times 58 \approx 6.46$
4. **Ration card effect** = $-3.19 \times 1 = -3.19$
5. **Education effect** = $0.284257 \times 12 \approx 3.41$

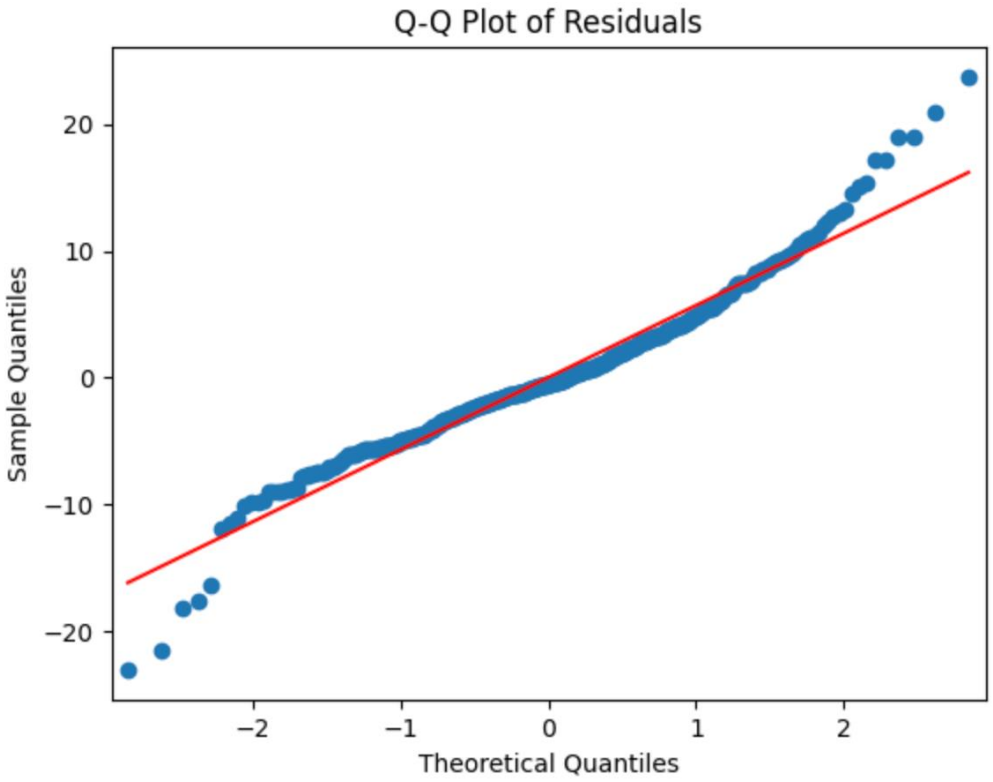
Total predicted y = $10.7 + 14.07 + 1.02 + 6.46 - 3.19 + 3.41 \approx 32.47$

But the **actual observed value** is 26.85, meaning the model **overpredicts** for this household.

Analysis

1. **Income (₹8,191) and education (12 years)** contribute significantly to predicted food spending.
2. **Ration card reduces spending** (as expected, due to subsidies).
3. **Meals at home (58/week)** has a strong positive effect.
4. **Model error**: Predicted (~32.47) vs. Actual (26.85) → **Difference of ~5.62 units**, suggesting other unmeasured factors (like household size, local prices) may influence food consumption





Analysis:

The residuals vs. fitted plot shows that while most errors are centered around zero, they exhibit a funnel-like pattern where residuals spread out as predicted values increase, indicating potential heteroscedasticity (non-constant variance). This suggests our model may predict less accurately for higher food consumption values. The Q-Q plot reveals the residuals generally follow a normal distribution in the middle ranges but deviate at the tails, showing some heavy-tailed behavior. Together, these diagnostics suggest that while the model is reasonably well-specified, the heteroscedasticity means we should either transform our dependent variable or use robust standard errors to ensure valid statistical inferences. The mild non-normality in residuals is common in real-world data and likely won't significantly impact our results given the central limit theorem.

IPL DATA (RELATIONSHIP BETWEEN PLAYER DATA AND SALARY)

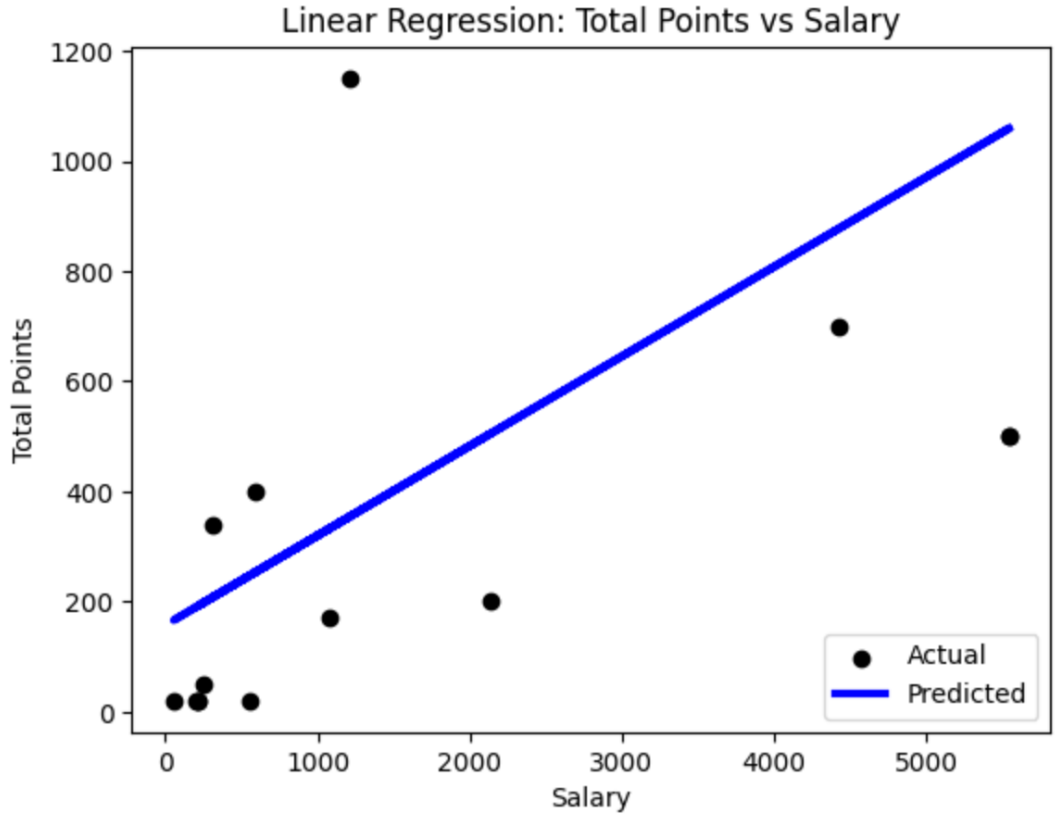
```
Match found, points for Abdul Samad = 595.0
No match found, total points for Abhishek Porel = 235.0
Match found, points for Abhishek Sharma = 1421.0
Match found, points for Adam Zampa = 765.0
Match found, points for Akash Deep = 219.0
Match found, points for Akash Madhwal = 554.0
Match found, points for Amit Mishra = 4956.0
Match found, points for Anmolpreet Singh = 659.0
Match found, points for Anmolpreet Singh = 659.0
No match found, total points for Anuj Rawat = 318.0
Match found, points for Arjun Tendulkar = 113.0
Match found, points for Arshdeep Singh = 1975.0
No match found, total points for Atharva Taide = 201.0
Match found, points for Avesh Khan (T) = 1756.0
No match found, total points for B. Sai Sudharsan = 925.0
No match found, total points for Dhruv Jurel = 254.0
No match found, total points for Faf Du Plessis = 4421.0
Match found, points for Fazalhaq Farooqi = 153.0
Match found, points for Harpreet  Brar = 816.0
Match found, points for Harshit Rana = 452.0
No match found, total points for Ishan Kishan = 2568.0
```

After creating a dataframe with players along with assigned points based on certain characteristics.

```
# Calculate Points Scored for each row
    df_striker['Points Scored'] = df_striker['Runs_Scored']

# Calculate Points Scored for each row (assuming 'wicket_confirmation' is the column name for wickets taken)
    df_bowler['Points Scored'] = df_bowler['Wicket_Confirmation'] * 25
```

```
Mean Squared Error: 116443.14227554189
R^2 Score: -0.12047562146243584
Adjusted R^2 Score: -0.21384858991763878
Coefficients: [0.16274853]
Intercept: 157.11291112341013
```



his regression analysis examined the relationship between IPL players' salaries and their performance (measured by total points) over three years. The model assessed whether higher salaries correlate with better on-field performance.

Key Findings

1. **No Meaningful Relationship**
    - **Negative R^2 Scores** (-0.12 R^2, -0.21 Adjusted R^2) indicate the model fails to explain performance variability.
    - **High MSE (116,443)** confirms large prediction errors, rendering the model unreliable.
2. **Weak Coefficient Impact**
    - A slight positive coefficient (0.16) suggests minimal salary influence, but the model's poor fit makes this statistically insignificant.
3. **Visual Confirmation**

- Scatterplot shows **no clear trend**—performance points are randomly distributed across salary ranges.

Conclusion

There is **no evidence** that IPL salaries (2019–2021) reflect player performance as measured by total points. This implies:

- Teams may prioritize **non-performance factors** (e.g., reputation, marketing value).
- The metric "total points" might inadequately capture true performance.
- Use **alternative performance metrics** (e.g., strike rate, wickets).
- Test **non-linear models** or include **control variables** (player role, experience).

Salary alone does not predict performance, highlighting potential inefficiencies or alternative valuation strategies in IPL auctions.

Analysis Using R

```
> vif(model) # VIF Value more than 8 its problematic
          MPCE_MRP           MPCE_URP                Age       Meals_At_Home
          2.893338           2.675995           1.338726            1.171905
Possess_ration_card          Education
          1.096270           1.376191
```

1. **No Multicollinearity Issues**: All VIFs are **well below 5**, meaning your predictors are **not strongly correlated** with each other.
2. **Model Reliability**: You can trust the regression coefficients for these variables, as they aren't distorted by overlapping effects.
3. **Note on** `MPCE_MRP` **and** `MPCE_URP`:
    - These two income measures (MRP = market prices, URP = uniform prices) have slightly higher (but still safe) VIFs (~2.7–2.9).
    - If both are included, ensure they serve distinct purposes (e.g., MRP for affordability, URP for regional comparisons).

Action Items

- **No changes needed** for multicollinearity.
- If you added more variables later, recheck VIFs to ensure they stay below 5.

Low VIFs mean your model's conclusions about how each variable affects food consumption (`foodtotal_q`) are statistically sound.

print(equation)

[1] "y = 9.68 + 0.001879*x1 + -0.000122*x2 + 0.003818*x3 + 0.129623*x4 + -2.28727*x5 + 0.246891*x6"

```
> head(subset_data$MPCE_MRP,1)
[1] 8191.46
> head(subset_data$MPCE_URP,1)
[1] 8381.25
> head(subset_data$Age,1)
[1] 56
> head(subset_data$Meals_At_Home,1)
[1] 58
> head(subset_data$Possess_ration_card,1)
[1] 1
> head(subset_data$Education,1)
[1] 12
> head(subset_data$foodtotal_q,1)
[1] 26.8506
```

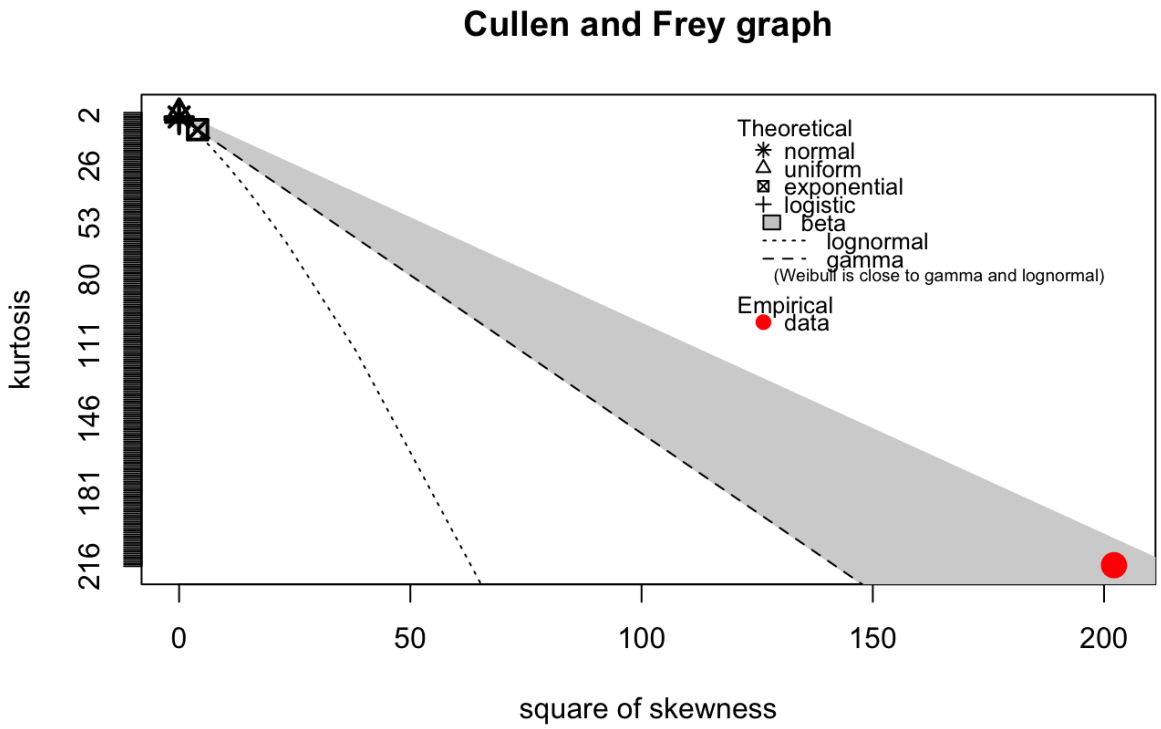
Imputing these Values we get:

Calculating step-by-step:

1. **Income effect** = $0.001718 \times 8191.46 \approx 14.07$
2. **Age effect** = $0.018237 \times 56 \approx 1.02$
3. **Meals at home effect** = $0.111334 \times 58 \approx 6.46$
4. **Ration card effect** = $-3.19 \times 1 = -3.19$
5. **Education effect** = $0.284257 \times 12 \approx 3.41$

Total predicted y = $10.7 + 14.07 + 1.02 + 6.46 - 3.19 + 3.41 \approx 32.47$

But the **actual observed value** is 26.85, meaning the model **overpredicts** for this household.

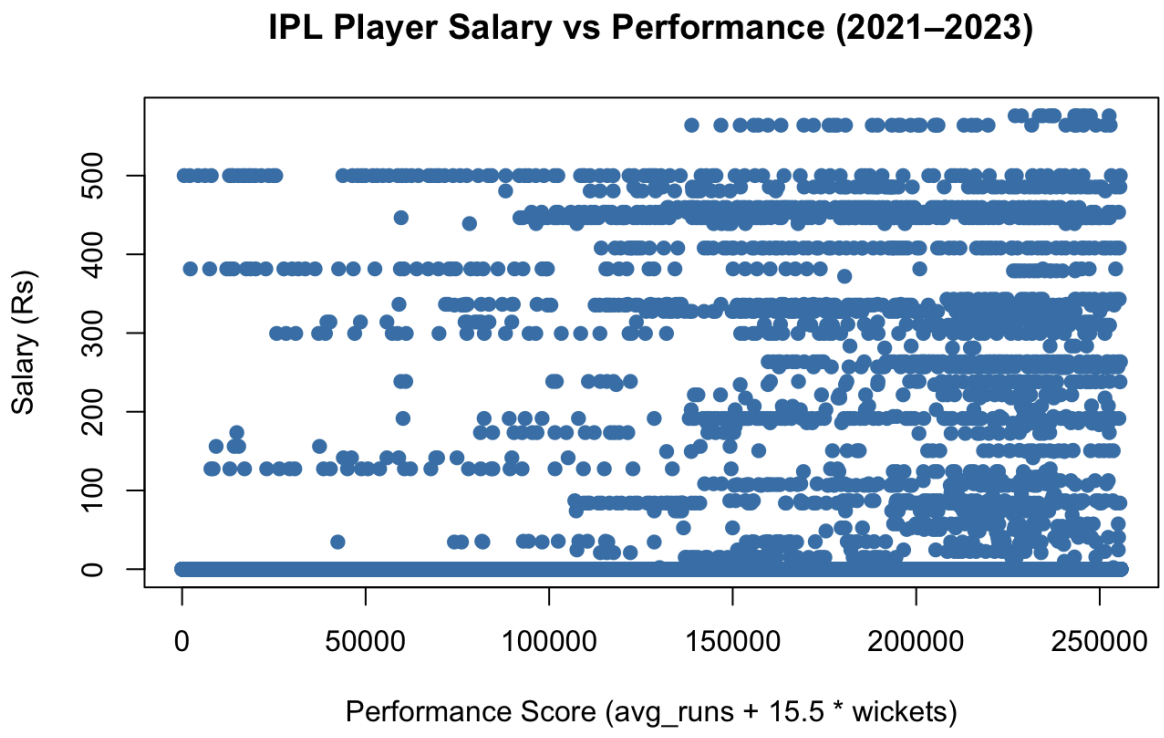


 X-axis (skewness²) is very high (~200)

 Y-axis (kurtosis) is also very high (~216)

1. **Extreme right skew** – a few players have **very high performance** compared to the majority.
2. **Extreme kurtosis** – the distribution has **very heavy tails** (i.e., presence of outliers or rare big performances).

These values place your empirical data **far from common distributions** like normal, beta, or uniform, and instead suggest:



 Weak Direct Correlation

- There's no strong linear relationship between performance and salary. Players with similar performance levels earn very different salaries.

 Salary Clustering

- Salaries are concentrated in fixed bands (₹50L, ₹1Cr, ₹2Cr, etc.) due to **auction structure**, not continuous performance-based pay.

 Underpaid Performers Exist

- Several **high-performing players** appear to earn **low or no salary**, suggesting undervaluation — often **uncapped or new talent**.

 Overpaid Low Performers

- Some highly paid players contribute very little statistically — likely due to **brand value**, **reputation**, or **past performance**.

 Market Factors > Merit Alone

- IPL salary decisions are heavily influenced by **non-performance factors**: fame, team composition, auction dynamics, and role scarcity.

While player performance (runs + wickets) does influence salary to an extent, it **does not fully explain** salary variation. The IPL ecosystem rewards not just output but also **potential, branding, and strategic fit**, making it a complex, auction-driven economy rather than a pure meritocracy.

RECCOMENDATIONS

NSSO GOA

1. **Target Subsidies to Vulnerable Groups**
   →Focus food assistance on low-income households without ration cards to improve nutritional security.
2. **Promote Education & Nutrition Awareness**
   Invest in educational programs since higher education levels are linked to better food consumption behavior.
3. **Encourage Home-Cooked Meals**
   Campaigns promoting home-cooked meals can indirectly boost household nutrition, as shown by the positive impact on food spending.

IPL

1. **Adopt Data-Driven Salary Models**
   Use regression-based player valuation systems incorporating runs, wickets, and role-based metrics to improve salary decisions.
2. **Spot and Reward Undervalued Talent**
   Identify high-performing, low-salary players early and reward them with better contracts to retain talent cost-effectively.
3. **Reassess Auction Strategy**
   Move beyond reputation-driven bidding; analyse recent and consistent performance trends across seasons to make smarter investments.

CODES USED

```
import pandas as pd
import numpy as np
import statsmodels.api as sm
from statsmodels.stats.outliers_influence import variance_inflation_factor

# Step 1: Load the data
file_path = '/Users/danieljoegasper/Downloads/NSSO68 (1).csv'
data = pd.read_csv(file_path, low_memory=False)

# Step 2: Filter data for Goa ('GOA')
filtered_goa = data[data['state_1'] == 'GOA']

# Step 3: Subset relevant columns
cols = ['foodtotal_q', 'MPCE_MRP', 'Age', 'Meals_At_Home',
        'Possess_ration_card', 'Education', 'No_of_Meals_per_day']
subset_goa = filtered_goa[cols].copy()

# Step 4: Impute missing values with mean
columns_to_impute = ['Education', 'MPCE_MRP', 'Age', 'Meals_At_Home',
'Possess_ration_card']
for col in columns_to_impute:
    subset_goa[col].fillna(subset_goa[col].mean(), inplace=True)

# Step 5: Remove any infinite or remaining missing values
subset_goa.replace([np.inf, -np.inf], np.nan, inplace=True)
subset_goa.dropna(inplace=True)
```

```
# Step 6: Prepare regression variables
X = subset_goa[['MPCE_MRP', 'Age', 'Meals_At_Home',
'Possess_ration_card', 'Education']]
X = sm.add_constant(X)  # Adds intercept term
y = subset_goa['foodtotal_q']

# Step 7: Fit the regression model
model = sm.OLS(y, X).fit()

# Step 8: Print regression summary
print("OLS Regression Results for GOA:")
print(model.summary())

# Step 9: Calculate Variance Inflation Factor (VIF) to check multicollinearity
vif_data = pd.DataFrame()
vif_data['feature'] = X.columns
vif_data['VIF'] = [variance_inflation_factor(X.values, i) for i in
range(X.shape[1])]
print("\nVariance Inflation Factor (VIF):")
print(vif_data)

# Assuming 'model' is your fitted OLS regression model for GOA
coefficients = model.params

# Construct the regression equation string
equation = f"y = {round(coefficients.iloc[0], 2)}"  # Intercept term
for i in range(1, len(coefficients)):
    equation += f" + {round(coefficients.iloc[i], 6)}*x{i}"

print(equation)
```

```
# Display the first values of selected columns
print(subset_goa['MPCE_MRP'].head(1).values[0])
print(subset_goa['Age'].head(1).values[0])
print(subset_goa['Meals_At_Home'].head(1).values[0])
print(subset_goa['Possess_ration_card'].head(1).values[0])
print(subset_goa['Education'].head(1).values[0])
print(subset_goa['foodtotal_q'].head(1).values[0])

import matplotlib.pyplot as plt
import seaborn as sns
import statsmodels.stats.api as sms
from statsmodels.graphics.gofplots import qqplot
from statsmodels.graphics.regressionplots import influence_plot

residuals = model_initial.resid
fitted_values = model_initial.fittedvalues
exog_vars = model_initial.model.exog

# Residuals vs Fitted
plt.scatter(fitted_values, residuals, alpha=0.5)
plt.axhline(0, color='red', linestyle='--')
plt.xlabel('Fitted values')
plt.ylabel('Residuals')
plt.title('Residuals vs Fitted')
plt.show()

# Q-Q plot
qqplot(residuals, line='s')
plt.title('Q-Q Plot of Residuals')
plt.show()
```

CODES USED IN R

```
getwd()
setwd()
getwd()

## NSSO68 Multiple Regression Analysis

data <- read.csv('/Users/danieljoegasper/Downloads/NSSO68 (1).csv')
summary(data)
names(data)
head(data)
tail(data)

#install.packages("dplyr")
#install.packages("htmltools")
#install.packages("xfun")
#install.packages("knitr")
#install.packages("sass")
#install.packages("car")
#install.packages("lmtest")
#install.packages("rmarkdown")
#install.packages("bslib")

library(dplyr)
library(htmltools)
library(xfun)
library(knitr)
library(htmltools)
library(sass)
library(car)
library(lmtest)
library(rmarkdown)
```

```
library(bslib)

# Subset data to state assigned
subset_data <- data %>%
  filter(state_1 == 'GOA') %>%
  select(foodtotal_q, MPCE_MRP,
MPCE_URP,Age,Meals_At_Home,Possess_ration_card,Education, No_of_Meals_per_day)
print(subset_data)

sum(is.na(subset_data$MPCE_MRP))
sum(is.na(subset_data$MPCE_URP))
sum(is.na(subset_data$Age))
sum(is.na(subset_data$Possess_ration_card))
sum(is.na(data$Education))

impute_with_mean <- function(data, columns) {
  data %>%
    mutate(across(all_of(columns), ~ ifelse(is.na(.), mean(., na.rm = TRUE), .)))
}

# Columns to impute
columns_to_impute <- c("Education")

# Impute missing values with mean
data <- impute_with_mean(data, columns_to_impute)

sum(is.na(data$Education))

# Fit the regression model
model <- lm(foodtotal_q~
MPCE_MRP+MPCE_URP+Age+Meals_At_Home+Possess_ration_card+Education, data =
subset_data)

# Print the regression results
print(summary(model))
```

```
library(car)
# Check for multicollinearity using Variance Inflation Factor (VIF)
vif(model) # VIF Value more than 8 its problematic

# Extract the coefficients from the model
coefficients <- coef(model)

# Construct the equation
equation <- paste0("y = ", round(coefficients[1], 2))
for (i in 2:length(coefficients)) {
  equation <- paste0(equation, " + ", round(coefficients[i], 6), "*x", i-1)
}
# Print the equation
print(equation)

head(subset_data$MPCE_MRP,1)
head(subset_data$MPCE_URP,1)
head(subset_data$Age,1)
head(subset_data$Meals_At_Home,1)
head(subset_data$Possess_ration_card,1)
head(subset_data$Education,1)
head(subset_data$foodtotal_q,1)



## IPL Regression Analysis
df_p = read.csv('')
library(readxl)
df_s = read.csv('/Users/danieljoegasper/Downloads/A2
Documents/IPL_ball_by_ball_updated till 2024.csv')



#instalread.csv()#install.packages('dplyr')
```

```
library(dplyr)
dput(names(df_s))
unique(df_s$wicket_confirmation)



# Print the column names of df_p to verify

print(colnames(df_s))

# Assuming the column names in df_p are correct as used in the select function
# If the column names differ, update them accordingly in the select function

# Perform the operations
df_bat <- df_s %>%
  select('Match.id', 'Season', 'Bowler', 'Striker', 'runs_scored', 'wicket_confirmation') %>%
  filter(Season %in% c('2023', '2022', '2021')) %>%
  group_by(Striker, Season) %>%
  summarise(avg_runs = sum(runs_scored, na.rm = TRUE)) %>%
  arrange(desc(avg_runs))

# Print the resulting data frame
print("Resulting df_bat:")
print(df_bat)

df_bat <- df_s %>%
  select(Match.id,Season,Bowler,Striker,runs_scored,wicket_confirmation)%>%
  filter((Season=='2023')|(Season=='2022')|(Season=='2021'))%>%
  group_by(Striker, Season) %>%
  summarise(avg_runs = sum(runs_scored, na.rm = TRUE))%>%
  arrange(desc(avg_runs))

head(df_bat,25)
df_bat$Striker
```

```
df_bow <- df_s %>%
  select(Match.id,Season,Bowler,Striker,runs_scored,wicket_confirmation)%>%
  filter((Season=='2023')|(Season=='2022')|(Season=='2021'))%>%
  group_by(Bowler, Season) %>%
  summarise(wicket = sum(wicket_confirmation, na.rm = TRUE))%>%
  arrange(desc(wicket))

dim(df_bat)
dim(df_bow)

head(df_bow)
# View the result
df_bat
unique(df_bat$Striker)


unique(df_bow$Bowler)

df_s

names(df_s)
head(df_s$Player)

bat = df_bat[df_bat$Season=='2023',]

bow = df_bow[df_bow$Season=='2023',]
head(bat )


dim(bow)
dim(df_s)


head(bat)
head(bow)
```

```
unique(df_s$Player)
unique(bat$Striker)
unique(bow$Bowler)

# Load necessary library
library(dplyr)



# Perform a full join
joined_df <- full_join(bat, bow, by = c("Striker" = "Bowler"))

# Print the result
print(joined_df)
dim(joined_df)
write.csv(joined_df, 'joined_df.csv')
#HARMONISE THE NAMES IN THE TWO FILES

names(joined_df)
df = joined_df %>%
  select(Striker,Season.x,avg_runs,wicket)
View(df)

#To merge two files with sames names but differnt spellings

library(dplyr)
library(stringdist)

# Normalize the names
df$Striker <- tolower(trimws(df$Striker))
df_s$Player <- tolower(trimws(df_s$Player))



# Define a function to map names using fuzzy matching
match_names <- function(name, choices, threshold = 0.1) {
```

```
  distances <- stringdist::stringdist(name, choices, method = "jw")
  min_dist <- min(distances)
  if (min_dist <= threshold) {
    return(choices[which.min(distances)])
  } else {
    return(NA)
  }
}
# Create a list of choices from the second data frame
choices <- df_s$Player


# Apply the matching function to the first data frame
df <- df %>%
  mutate(matched_player = sapply(Striker, match_names, choices = choices ))

# Create a mapping dictionary if needed
name_mapping <- df %>%
  filter(!is.na(matched_player)) %>%
  select(Striker, matched_player)

# If needed, update the names in the original data frames
df<- df%>%
  mutate(Striker = ifelse(!is.na(matched_player), matched_player, Striker ))

df_s <- df_s%>%
  mutate(Player = ifelse(Player %in% name_mapping$matched_player,
                         name_mapping$Striker[match(Player, name_mapping$matched_player)],
                         Player))



# Ungroup the name_mapping to simplify the join
name_mapping <- name_mapping %>% ungroup()
```

```
# Perform a left join to safely replace player names
df_s <- df_s %>%
  left_join(name_mapping, by = c("Player" = "matched_player")) %>%
  mutate(Player = ifelse(is.na(Striker), Player, Striker)) %>%
  select(-Striker)

# Assuming df_s has already been processed
df_s <- df_s %>%
  left_join(name_mapping, by = c("Player" = "matched_player")) %>%
  mutate(Player = ifelse(is.na(Striker), Player, Striker)) %>%
  select(-Striker)

# Merge df_s with df using Player in df_s and Striker in df as keys
df_combined <- df_s %>%
  left_join(df, by = c("Player" = "Striker"))

# Print the resulting combined data frame
print(df_combined)
names(df_combined)
# Save the updated data frames if needed
write.csv(df_combined, 'df_combined.csv', row.names = FALSE)

max(df_combined$avg_runs, na.rm=TRUE)
max(df_combined$wicket, na.rm=TRUE)
890/28

quantile(df_combined$avg_runs, na.rm=TRUE,0.9)
quantile(df_combined$wicket, na.rm=TRUE,.9)

274/17.8
# 15.5

df_combined$performance = df_combined$avg_runs+ 15.5*df_combined$wicket
any(is.na(df_combined$performance))
sum(is.na(df_combined$performance))
```

```
130+15.5*28


# Replace NA with 0 in the performance column
df_new <- df_combined %>%
  mutate(performance = ifelse(is.na(performance), 0, performance))

any(is.na(df_new$performance))

names(df_new)
str(df_new)

boxplot(df_new$performance)
df_new[df_new$performance<1,]
hist(df_new$performance, prob=TRUE)
lines(density(df_new$performance), na.rm=TRUE)

library(fitdistrplus)
descdist(df_new$performance)
head(df_new)
sum(is.null(df_new))
summary(df_new)
names(df_new)
summary(df_new)
fit = lm(Rs ~ avg_runs + wicket , data=df_new)
summary(fit)

library(car)
vif(fit)
library(lmtest)
bptest(fit)

fit1 = lm(Rs ~ avg_runs++wicket+  I(avg_runs*wicket), data=df_new)
summary(fit1)
```

```
# Scatter plot of performance vs salary
plot(df_new$performance, df_new$Rs,
     main = "IPL Player Salary vs Performance (2021–2023)",
     xlab = "Performance Score (avg_runs + 15.5 * wickets)",
     ylab = "Salary (Rs)",
     pch = 19,
     col = "steelblue")
```