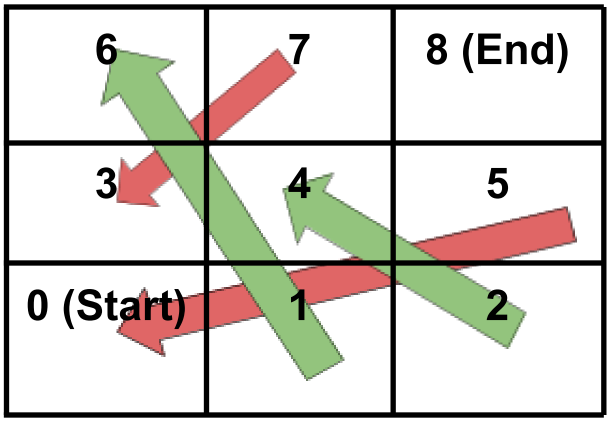
Coventry University Assessment

Name: Abhi Ashokbhai Thumar
Course: Msc. Data Science (EEST036)
Module: 7042SCN - Data Analytics - 2526NOVJAN

Task 1 (Individual Task) – Markov Chains



Part - I

```
import numpy as np

P = np.zeros((9,9))

def final_destination(j_dash):
    if j_dash == 1:
        return 6
    if j_dash == 2:
        return 4
    if j_dash == 5:
        return 0
    if j_dash == 7:
        return 3
    return j_dash

for i in range(8):
    # Heads (Moves 1 step ahead)
    j1_dash = i + 1
    if j1_dash <=8:
        j1 = final_destination(j1_dash)
        P[i, j1] += 0.5

    # Tails (Moves 2 step ahead)
    j2_dash = i + 2
    if j2_dash <= 8:
        j2 = final_destination(j2_dash)
        P[i,j2] += 0.5
    else:
        P[i, j2] += 0.5 #for 7 --> 9 is also game complete


P[8,8] = 1
P = P.T
print("transition matrix\n",P)
```

Output:

```
transition matrix
 [[0.  0.  0.  0.5 0.5 0.  0.  0.  0. ]
 [0.  0.  0.  0.  0.  0.  0.  0.  0. ]
 [0.  0.  0.  0.  0.  0.  0.  0.  0. ]
 [0.  0.5 0.5 0.  0.  0.5 0.5 0.  0. ]
 [0.5 0.5 0.5 0.5 0.  0.  0.  0.  0. ]
 [0.  0.  0.  0.  0.  0.  0.  0.  0. ]
 [0.5 0.  0.  0.  0.5 0.5 0.  0.  0. ]
 [0.  0.  0.  0.  0.  0.  0.  0.  0. ]
 [0.  0.  0.  0.  0.  0.  0.5 1.  1. ]]
```

Part - II

Here I Took k = 8 and removed the 8th row and 8th column from the matrix P

```
N = P[:8, :8]

I = np.identity(8)
M = np.linalg.inv(I - N)


print("\n Matrix M = \n",M)

u_col_sum = np.sum(M, axis=0)

print("\nExpected number of coin flips required:")
for i in range(8):
    print(f"{i} ------> {u_col_sum[i]:.4f}")
```

Output:

```
Matrix M =
 [[2.33333333 1.83333333 1.83333333 2.         1.66666667 1.5
  1.         0.        ]
 [0.         1.         0.         0.         0.         0.
  0.         0.        ]
 [0.         0.         1.         0.         0.         0.
  0.         0.        ]
 [1.         1.5        1.5        2.         1.         1.5
  1.         0.        ]
 [1.66666667 2.16666667 2.16666667 2.         2.33333333 1.5
  1.         0.        ]
 [0.         0.         0.         0.         0.         1.
  0.         0.        ]
 [2.         2.         2.         2.         2.         2.
  2.         0.        ]
 [0.         0.         0.         0.         0.         0.
  0.         1.        ]]
```

```
Expected number of coin flips required:
0 ------> 7.0000
1 ------> 8.5000
2 ------> 8.5000
3 ------> 8.0000
4 ------> 7.0000
5 ------> 7.5000
6 ------> 5.0000
7 ------> 1.0000
```

Part - III

We have to move from middle Square(4) to the end(8) without getting onto the snake. So, we start on square 4 and we have to get tails in order to avoid the snake on square 5 and reach at square 6. From square 6 we have to flip tails again to reach the end while also avoiding the snake at square 7.

The probability of hitting 2 tails in row is $(0.5)^2$ = 0.25
Conclusion: The probability of completing the game (reaching square 8) before slipping back from the middle square (4) is **0.25** or **1/4.**

Task 2 (Individual Task) — Linear Models

Part - I

In typical machine learning practice, AIC is most commonly utilized when testing the model's performance on a test set is difficult.

- The acronym stands for Akaike Information Criterion.
- The formula for AIC is
  AIC=2K-2ln(L)
- **K** is the number of independent variables or parameters in the model.
- **L** is the maximum likelihood estimate of the model, which indicates how likely the model is to have produced the observed data.
- A lower AIC value indicates a better model.

Reference:

Zajic, A. (2022, November 29). *What is akaike information criterion (AIC)?* Built In. https://builtin.com/data-science/what-is-aic

Part - II

```
library(tidyverse)
library(dplyr)
library(olsrr)
library(ggfortify)

# (2)
df <- read.csv("imd2025_individual.csv")
#Model 1
model1 <- lm(Overall ~ Employment + Living, data = df)
summary(model1)
AIC(model1)
# The best two-predictor linear mode is Income and Employment as per olsrr
```

```
#Model 2
df1 <- df%>% select(-Rank,-LAD24CD,-LAD24NM)
full_model <- lm(Overall ~ ., data = df1)
best <- ols_step_best_subset(full_model)
best

model2 <- lm(Overall ~ Income+Employment+Education+Living, data = df)
summary(model2)
AIC(model2)

# The best overall that uses at most four quantitative predictors
(excluding Rank) is
# Income Employment Education Living


df_london <- df1%>% filter(Region == "London") %>% select(-Region)
london_model <- lm(Overall ~ ., data = df_london)
best2 <- ols_step_best_subset(london_model, include = "Crime")
best2
# Best four quantitative predictors in region London including "Crime" is
# Income Education Crime Living


df_non_london <- df1%>% filter(Region != "London") %>% select(-Region)
non_london_model <- lm(Overall ~ ., data = df_non_london)
best3 <- ols_step_best_subset(non_london_model, include = "Crime")
best3

# Best four quantitative predictors excluding region London including
"Crime" is
# Income Employment Crime Living


model3_london <- lm(Overall ~ Income+Education+Crime+Living, data = df)
summary(model3_london)
AIC(model3_london)

model3_non_london <- lm(Overall ~ Income+Employment+Crime+Living, data =
df)
summary(model3_non_london)
AIC(model3_non_london)
```

Model #1:

- AIC of Model 1 = **595.8703**
- Employment and Living are not the best two-predictor linear model to predict Overall.
- The best two-predictor linear model to predict Overall is Income and Employment.

Model #2:

- The best linear model to predict Overall that uses at most four quantitative predictors is Income, Employment, Education, Living
- AIC of Model 2 = **404.8591**

Model #3:

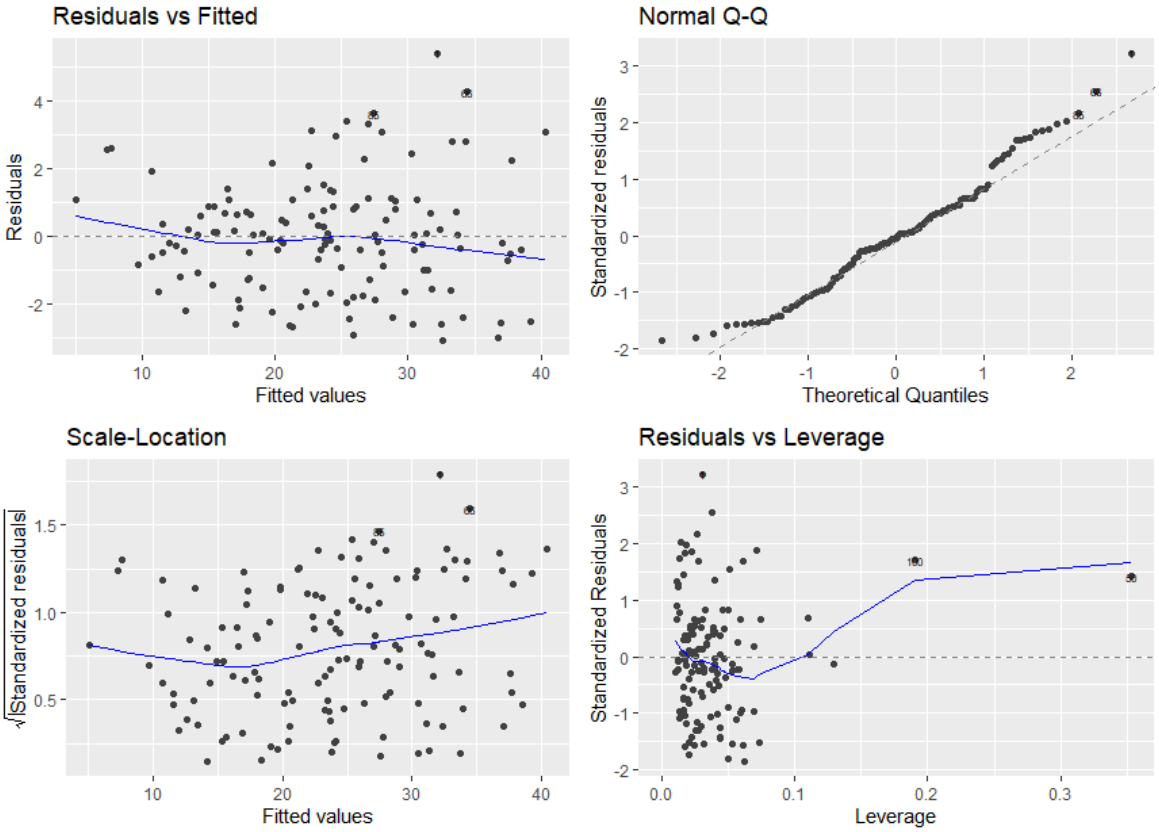
- **London_Model:**
    - The best linear model to predict Overall in London only, that uses at most four quantitative predictors (excluding Rank) but must include Crime is Income, Education, Crime, Living
    - AIC of Model3_London = **524.2588**
- **Non_London_Model:**
    - The best linear model to predict Overall excluding London, that uses at most four quantitative predictors (excluding Rank) but must include Crime is Income, Employment, Crime, Living
    - AIC of Model3_Non_London = **495.3492**

Part - III

```
autoplot(model3_london, label = TRUE, label.size = 2)

autoplot(model3_non_london, label = TRUE, label.size = 2)
```

London Model



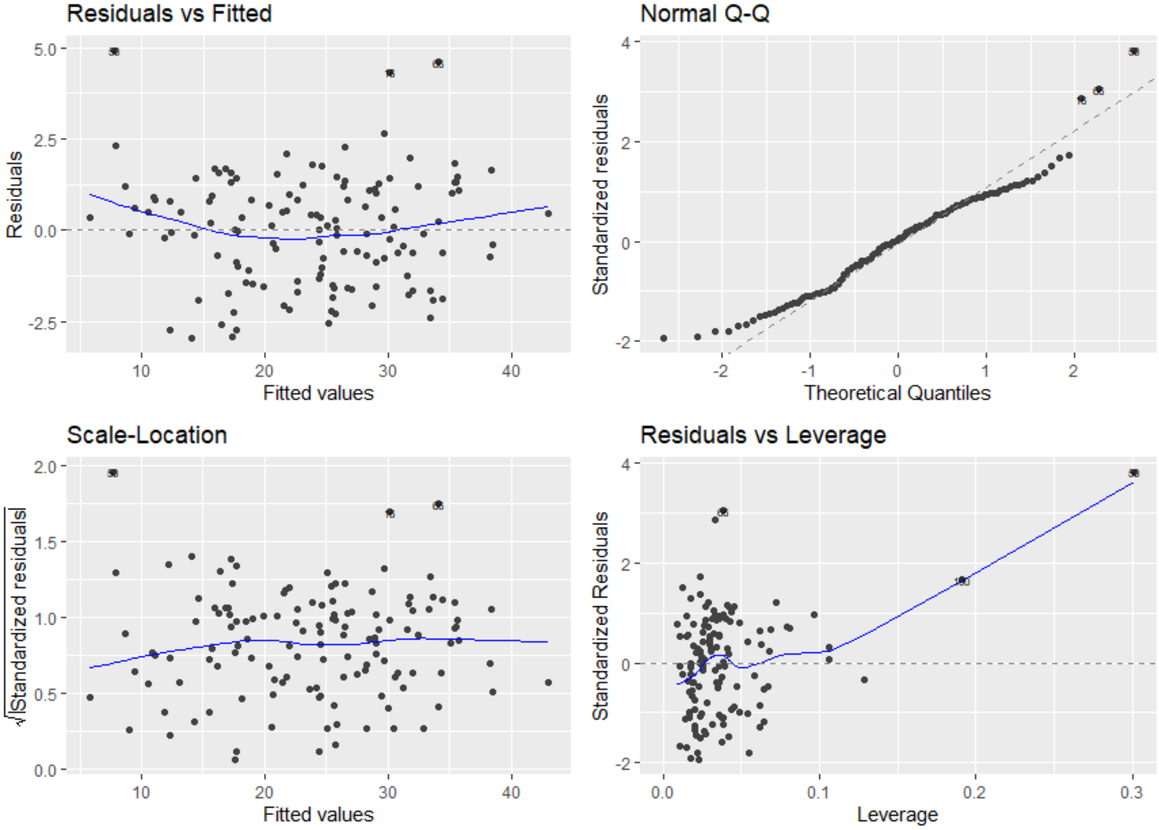
Residuals vs Fitted

Normal Q-Q

Scale-Location

Residuals vs Leverage

Non London Model



Residuals vs Fitted

Normal Q-Q

Scale-Location

Residuals vs Leverage