Terningdropp

Inge Johan Johansson

2024-02-29

R Markdown

Denne markdown filen viser min python kode og en autogennerert kode i R. Siden det er en dag gjenn til fristen så gidder eg ikkje å finne ut om det er en bedre måte å bruke python til å importere i overleaf. VScode gjir brukeren muligheten til å skrive i LaTeX, men man må fremdeles importere filene og koden in i LaTeX filen.

```
import pandas as pa
import matplotlib.pyplot as plt
import numpy as np
import math

"Nvidia contact: 1-408-486-2000 support time: 24/7"

# Importing csv files
p1 = pa.read_csv("oblig_1b/Terning_20.csv")

# Create a new DataFrame with columns "1" and "0"
hoyde_subset = pa.DataFrame({
    "0": np.ones(5),
    "1": p1["Hoyde"].iloc[0:5]
})
# Remove name of columns
hoyde_subset.columns = [None]*len(hoyde_subset.columns)


# Save the DataFrame as csv
hoyde_subset.to_csv("oblig_1b/hoyde_subset.csv", index=False)

hoyde_subset_transpose = hoyde_subset.transpose()  # Define "hoyde_subset_transpose" by transposing "hoy

hoyde_subset_transpose.columns = [None]*len(hoyde_subset_transpose.columns)


# Save the DataFrame as csv
hoyde_subset_transpose.to_csv("oblig_1b/hoyde_subset_transpose.csv", index=False)

# Create a new DataFrame with columns "1" and "0"
lengde_subset = pa.DataFrame({
    "0": p1["Lengde"].iloc[0:5]
})
```

```
# Remove name of columns
lengde_subset.columns = [None]*len(lengde_subset.columns)
# Save the DataFrame as csv
lengde_subset.to_csv("oblig_1b/lengde_subset.csv", index=False)


lengde_subset_transpose = lengde_subset.transpose()
# Remove name of columns
lengde_subset_transpose.columns = [None]*len(lengde_subset_transpose.columns)

# Save the DataFrame as csv
lengde_subset_transpose.to_csv("oblig_1b/lengde_subset_transpose.csv", index=False)

# Convert the DataFrames to numpy arrays
hoyde_subset_np = hoyde_subset.values
hoyde_subset_transpose_np = hoyde_subset_transpose.values
lengde_subset_np = lengde_subset.values
lengde_subset_transpose_np = lengde_subset_transpose.values

# Calculating regression line
# Calculate the dot product of the matrix and its transpose
XtX = np.dot(hoyde_subset_transpose_np, hoyde_subset_np)
XtX_inv = np.linalg.inv(XtX)


Xty = np.dot(hoyde_subset_transpose_np, lengde_subset_np)

beta = np.dot(XtX_inv, Xty)
alpha = beta[0]

#regression_line = alpha + beta[1]*hoyde_subset_np
regression_line = alpha + beta[1]*hoyde_subset_np




# Plot p1 as scatter plot
plt.scatter(p1.Hoyde[0:5], p1.Lengde[0:5], color="pink", marker="o")
# Plot regression line from x = 0 to end of p1
plt.plot(hoyde_subset_np, regression_line, color="blue", linewidth=3)

# Add labels and title
plt.xlabel('Hoyde')
plt.ylabel('Lengde')
plt.title('Scatter Plot')
#save the plot
plt.savefig("oblig_1b/plot1.png")
# Show the plot
plt.show()



##Oppgave C
```

```
# Importing csv files
p1 = pa.read_csv("oblig_1b/Terning_20.csv")

# Create a new DataFrame with columns "1" and "0"
hoyde_subset_all = pa.DataFrame({
    "0": np.ones(30),
    "1": p1["Hoyde"].iloc[0:30]
})
# Remove name of columns
hoyde_subset_all.columns = [None]*len(hoyde_subset_all.columns)


# Save the DataFrame as csv
hoyde_subset_all.to_csv("oblig_1b/hoyde_subset_all.csv", index=False)

hoyde_subset_all_transpose = hoyde_subset_all.transpose()  # Define "hoyde_subset_all_transpose" by tra

hoyde_subset_all_transpose.columns = [None]*len(hoyde_subset_all_transpose.columns)


# Save the DataFrame as csv
hoyde_subset_all_transpose.to_csv("oblig_1b/hoyde_subset_all_transpose.csv", index=False)

# Create a new DataFrame with columns "1" and "0"
lengde_subset_all = pa.DataFrame({
    "0": p1["Lengde"].iloc[0:30]
})

# Remove name of columns
lengde_subset_all.columns = [None]*len(lengde_subset_all.columns)
# Save the DataFrame as csv
lengde_subset_all.to_csv("oblig_1b/lengde_subset_all.csv", index=False)


lengde_subset_all_transpose = lengde_subset_all.transpose()
# Remove name of columns
lengde_subset_all_transpose.columns = [None]*len(lengde_subset_all_transpose.columns)

# Save the DataFrame as csv
lengde_subset_all_transpose.to_csv("oblig_1b/lengde_subset_all_transpose.csv", index=False)

# Convert the DataFrames to numpy arrays
hoyde_subset_all_np = hoyde_subset_all.values
hoyde_subset_all_transpose_np = hoyde_subset_all_transpose.values
lengde_subset_all_np = lengde_subset_all.values
lengde_subset_all_transpose_np = lengde_subset_all_transpose.values

# Calculating regression line
# Calculate the dot product of the matrix and its transpose
XtX = np.dot(hoyde_subset_all_transpose_np, hoyde_subset_all_np)
XtX_inv = np.linalg.inv(XtX)
```

```
Xty = np.dot(hoyde_subset_all_transpose_np, lengde_subset_all_np)

beta = np.dot(XtX_inv, Xty)
alpha = beta[0]
print(alpha)
print(beta[1])






#regression_line = alpha + beta[1]*hoyde_subset_all_np
regression_line = alpha + beta[1]*hoyde_subset_all_np
residuals = lengde_subset_all_np - regression_line
#print(diviance)



# Plot p1 as scatter plot
plt.scatter(p1.Hoyde[0:30], p1.Lengde[0:30], color="pink", marker="o")
# Plot regression line from x = 0 to end of p1
plt.plot(hoyde_subset_all_np, regression_line, color="blue", linewidth=3)

# Add labels and title
plt.xlabel('Hoyde')
plt.ylabel('Lengde')
plt.title('Scatter Plot')
#save the plot
plt.savefig("oblig_1b/plot2.png")
plt.show()
```

Including Plots

Poenget med oppgaven er vel å vise at eg kan lage en PDF i R studio. Siden oppgave 1b er gjort i python så du en gennerert kode fra gemini.

```
# Libraries
library(dplyr)
library(ggplot2)

# Read data
data <- read.csv("oblig_1b/Terning_20.csv")

# Subset first 5 rows
height_subset <- data %>%
  slice(1:5) %>%
  mutate(Intercept = 1)

# Subset first 5 rows (transposed)
height_subset_t <- t(height_subset)

# Save subsets
write.csv(height_subset, "oblig_1b/hoyde_subset.csv", row.names = FALSE)
```

```r
write.csv(height_subset_t, "oblig_1b/hoyde_subset_transpose.csv", row.names = FALSE)

# Subset first 5 rows for length
length_subset <- data %>%
  slice(1:5) %>%
  select(Lengde)

# Subset first 5 rows (transposed) for length
length_subset_t <- t(length_subset)

# Save subsets
write.csv(length_subset, "oblig_1b/lengde_subset.csv", row.names = FALSE)
write.csv(length_subset_t, "oblig_1b/lengde_subset_transpose.csv", row.names = FALSE)

# Convert DataFrames to matrices
height_subset_matrix <- as.matrix(height_subset)
height_subset_t_matrix <- t(height_subset_matrix)
length_subset_matrix <- as.matrix(length_subset)
length_subset_t_matrix <- t(length_subset_matrix)

# Calculate regression coefficients
XtX <- t(height_subset_t_matrix) %*% height_subset_matrix
XtX_inv <- solve(XtX)
Xty <- t(height_subset_t_matrix) %*% length_subset_matrix
beta <- Xty %*% XtX_inv
alpha <- beta[1]

# Calculate regression line
regression_line <- alpha + beta[2] * height_subset_matrix[, 2]

# Plot data and regression line
ggplot(data, aes(x = Hoyde, y = Lengde)) +
  geom_point(color = "pink", size = 3) +
  geom_line(aes(y = regression_line), color = "blue", size = 1.5) +
  labs(x = "Hoyde", y = "Lengde", title = "Scatter Plot") +
  theme_classic()
ggsave("oblig_1b/plot1.png")

# ----- Oppgave C -----

# Subset all rows
height_subset_all <- data %>%
  mutate(Intercept = 1)

# Subset all rows (transposed)
height_subset_t_all <- t(height_subset_all)

# Save subsets
write.csv(height_subset_all, "oblig_1b/hoyde_subset_all.csv", row.names = FALSE)
write.csv(height_subset_t_all, "oblig_1b/hoyde_subset_all_transpose.csv", row.names = FALSE)

# Subset all rows for length
length_subset_all <- data %>%
  select(Lengde)

# Subset all rows (transposed) for length
length_subset_t_all <- t(length_subset_all)

# Save subsets
write.csv(length_subset_all, "oblig_1b/lengde_subset_all.csv", row.names = FALSE)
write.csv(length_subset_t_all, "oblig_1b/lengde_subset_all_transpose.csv", row.names = FALSE)

# Convert DataFrames to matrices
height_subset_all_matrix <- as.matrix(height_subset_all)
height_subset_t_all_matrix <- t(height_subset_all_matrix)
length_subset_all_matrix <- as.matrix(length_subset_all)
length_subset_t_all_matrix <- t(length_subset_all_matrix)

# Calculate regression coefficients
XtX <- t(height_subset_t_all_matrix) %*% height_subset_all_matrix
XtX_inv <- solve(XtX)
Xty <- t(height_subset_t_all_matrix) %*% length_subset_all_matrix
beta <- Xty %*% XtX_inv
alpha <- beta[1]
print(alpha)
```