Project Summary

Ethan Tran

Introduction

This report presents a comprehensive analysis of German Mazda car listings. The goal of this project is to explore the relationships between various car attributes, apply regression models, and implement advanced analytical techniques to derive meaningful insights. The analysis includes data description, statistical summaries, model fitting, and advanced methodologies such as shrinkage methods and bootstrapping.

Data Description and Initial Analysis

The dataset comprises Mazda car listings from Germany, detailing attributes such as price, mileage, year of manufacture, fuel type, and more. After cleaning and filtering, we selected 600 cars for our study.

Summary Statistics:

```
data <- read.csv("gcar_data.csv")
data[data == ""] <- NA
data1 <- na.omit(subset(data, data$brand == 'mazda'))
data1[, c("power_ps", "price_in_euro", "mileage_in_km")] <- lapply(data1[, c("power_ps", "price_in_euro", "mileage_in_km")], as.integer)
cleaned_df <- na.omit(data1)

cleaned_df[, "fuel_consumption_l_100km"] <- sapply(cleaned_df[, "fuel_consumption_l_100km"], function(x) as.numeric(gsub(",", ".", strsplit(x, " l/100 km")[[1]])))
cleaned_df <- cleaned_df[cleaned_df$transmission_type != "Unknown", ]
cleaned_df <- cleaned_df[complete.cases(cleaned_df), ]

# Sample 600 cars randomly
set.seed(123) # Reproducibility
want_rows <- 600
random_rows <- sample(1:nrow(cleaned_df), want_rows, replace = FALSE)
project_df <- cleaned_df[random_rows, c(-1, -2, -8, -13)]

summary(project_df)
```

```
##     model              color           registration_date      year
##  Length:600         Length:600         Length:600         Length:600
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##  price_in_euro      power_ps   transmission_type   fuel_type
##  Min.   :  390   Min.   : 68   Length:600         Length:600
##  1st Qu.:12980   1st Qu.:120   Class :character   Class :character
##  Median :17700   Median :150   Mode  :character   Mode  :character
##  Mean   :17223   Mean   :143
##  3rd Qu.:22088   3rd Qu.:165
##  Max.   :32790   Max.   :260
##  fuel_consumption_l_100km mileage_in_km    offer_description
##  Min.   : 3.40            Min.   :     1   Length:600
##  1st Qu.: 5.10            1st Qu.: 45735   Class :character
##  Median : 5.80            Median : 69995   Mode  :character
##  Mean   : 5.87            Mean   : 84483
##  3rd Qu.: 6.40            3rd Qu.:110354
##  Max.   :10.40            Max.   :350000
```

```
# Set aside some data for testing
sample_index <- createDataPartition(project_df$price_in_euro, p = 0.8, list = FALSE)
train_data <- project_df[sample_index, ]
test_data <- project_df[-sample_index, ]

train_data_t <- train_data %>%
  mutate(log_price = log(price_in_euro),
         power_mileage_interaction = power_ps * mileage_in_km,
         mileage_squared = mileage_in_km^2,
         log_fuel_consumption = log(fuel_consumption_l_100km),
         transmission_automatic = ifelse(transmission_type == "Automatic", 1, 0)
         )

new_data <- test_data %>%
  mutate(log_price = log(price_in_euro),
         power_mileage_interaction = power_ps * mileage_in_km,
         mileage_squared = mileage_in_km^2,
         log_fuel_consumption = log(fuel_consumption_l_100km),
         transmission_automatic = ifelse(transmission_type == "Automatic", 1, 0)
         )
```

Key observations include:

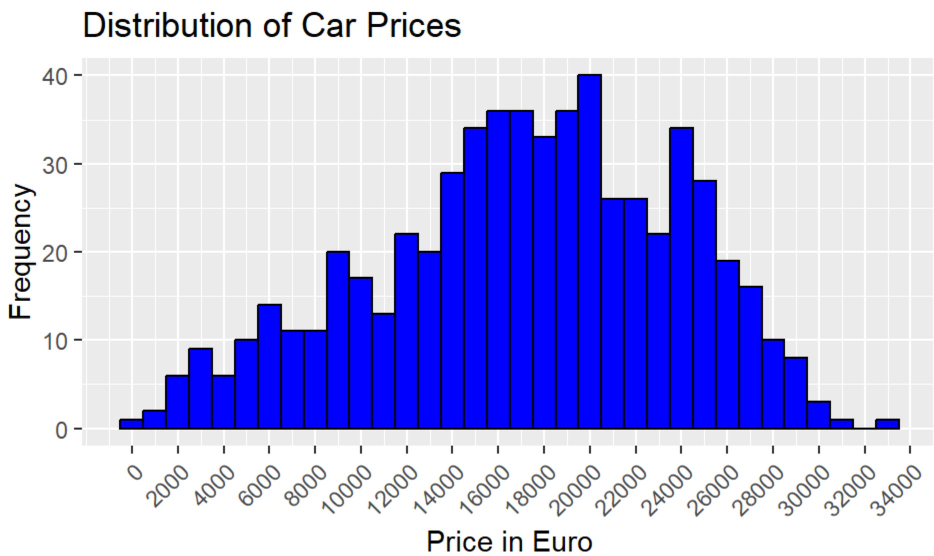
- The dataset contains 600 Mazda car listings after cleaning and filtering.
- The price and mileage of the cars are negatively correlated.
- No Mazdas were manufactured past 2019, and there are no electric Mazda models in the dataset.

Exploratory Data Analysis

Independent Quantitative Variables

Car Prices

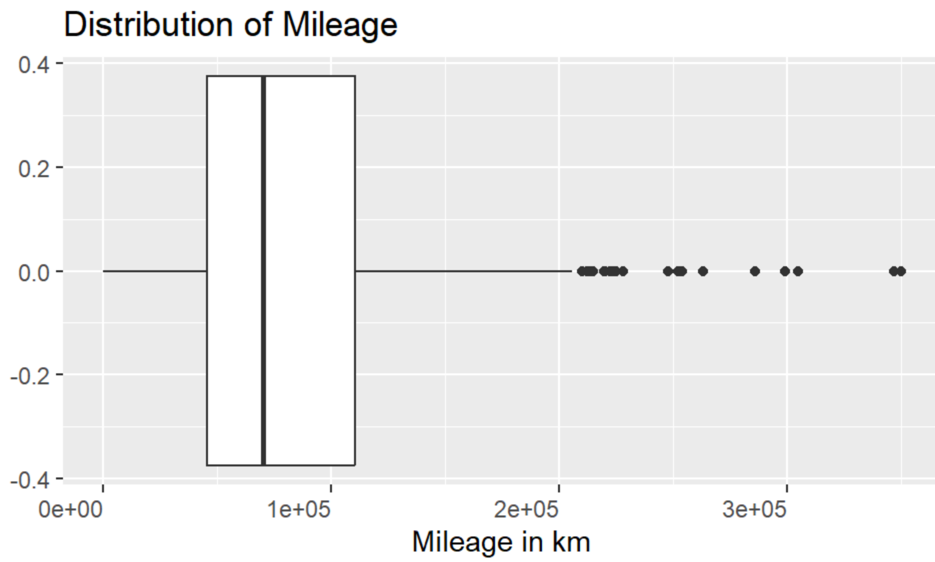
```
ggplot(project_df, aes(x = price_in_euro)) +
  geom_histogram(binwidth = 1000, fill = "blue", color = "black") +
  labs(x = "Price in Euro", y = "Frequency", title = "Distribution of Car Prices") +
  scale_x_continuous(breaks = seq(0, 40000, by = 2000)) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



The distribution of car prices is slightly skewed to the right, with a peak frequency at the 19,000-19,999 euros price range. There are no outliers, and only a few Mazdas in the dataset were priced at over 30,000 euros.

Mileage

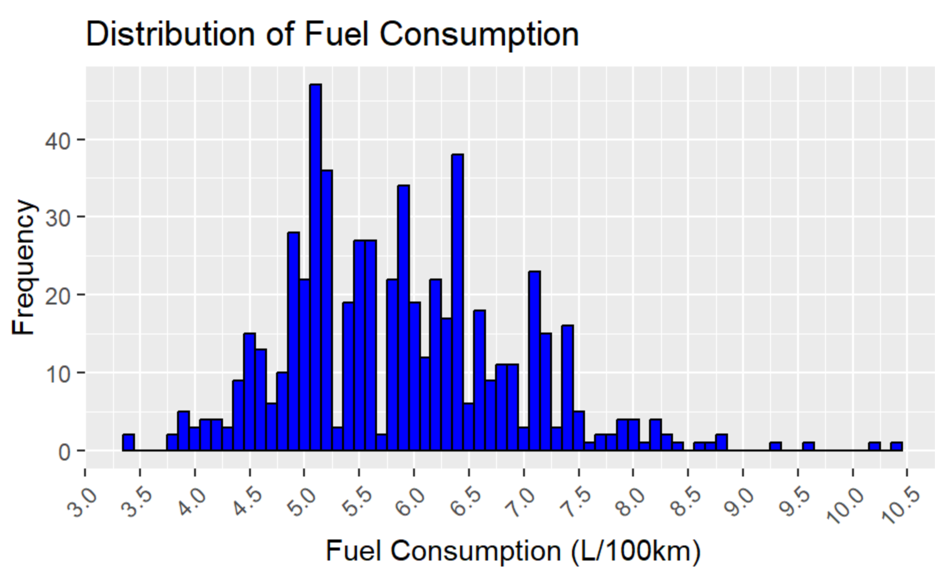
```
ggplot(project_df, aes(x = mileage_in_km)) +
  geom_boxplot() +
  labs(x = "Mileage in km", title = "Distribution of Mileage") +
  scale_x_continuous(breaks = seq(0, 400000, by = 100000)) +
  theme(axis.text.x = element_text(angle = 0, hjust = 1))
```



The mileage of the listed cars is heavily concentrated around the 70,000 kilometer range, with many outliers past the 200,000 kilometer range. Half of the data falls within the 50,000 to 100,200 kilometers in mileage range.

Fuel Consumption

```
ggplot(project_df, aes(x = fuel_consumption_l_100km)) +
  geom_histogram(binwidth = 0.1, fill = "blue", color = "black") +
  labs(x = "Fuel Consumption (L/100km)", y = "Frequency", title = "Distribution of Fuel Consumpt
ion") +
  scale_x_continuous(breaks = seq(0, 20, by = 0.5)) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

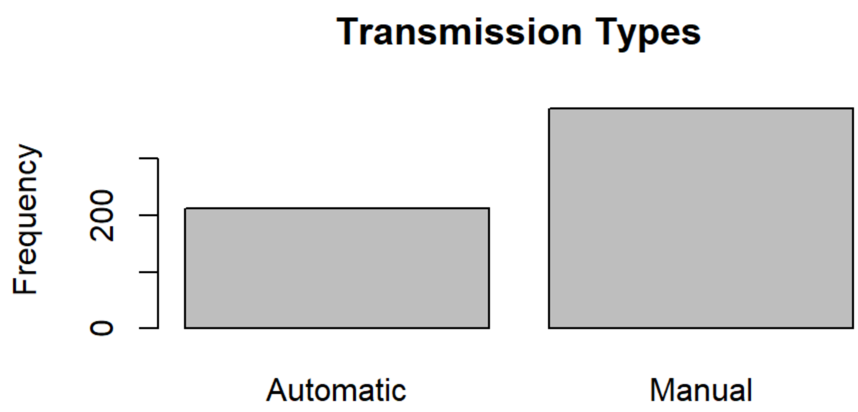


The distribution of fuel consumption has a slight left skew, with numerous spikes present in the 5.0 - 7.5 liter per kilometer range. There is a vague bell-shaped curve in this distribution.

Independent Categorical Variables

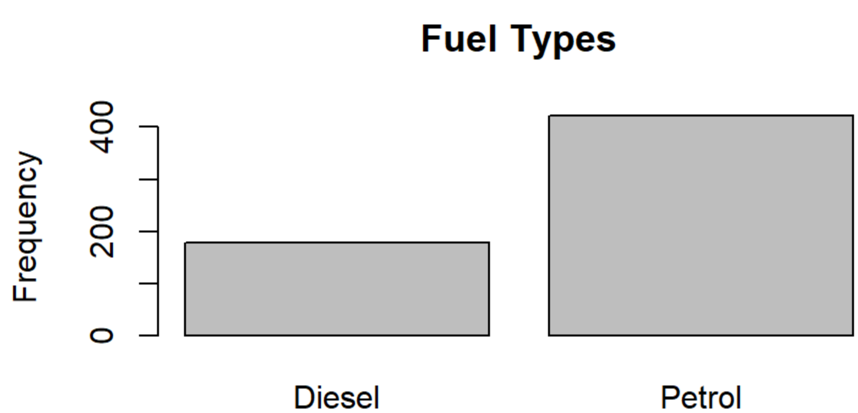
Transmission Type

```
barplot(table(project_df$transmission_type), ylab = "Frequency", main = "Transmission Types")
```



Fuel Type

```
barplot(table(project_df$fuel_type), ylab = "Frequency", main = "Fuel Types")
```



The Mazda cars from the dataset are split into two different transmission types, with significantly more manual than automatic. There were 4 fuel types in the dataset: diesel, petrol, hybrid, and LPG. However, hybrid and LPG fuel types are outliers and they rarely occur after randomly selecting 600 Mazdas. There is a clear majority in the Petrol type, with diesel taking up most of the rest of the data.

Regression Analysis

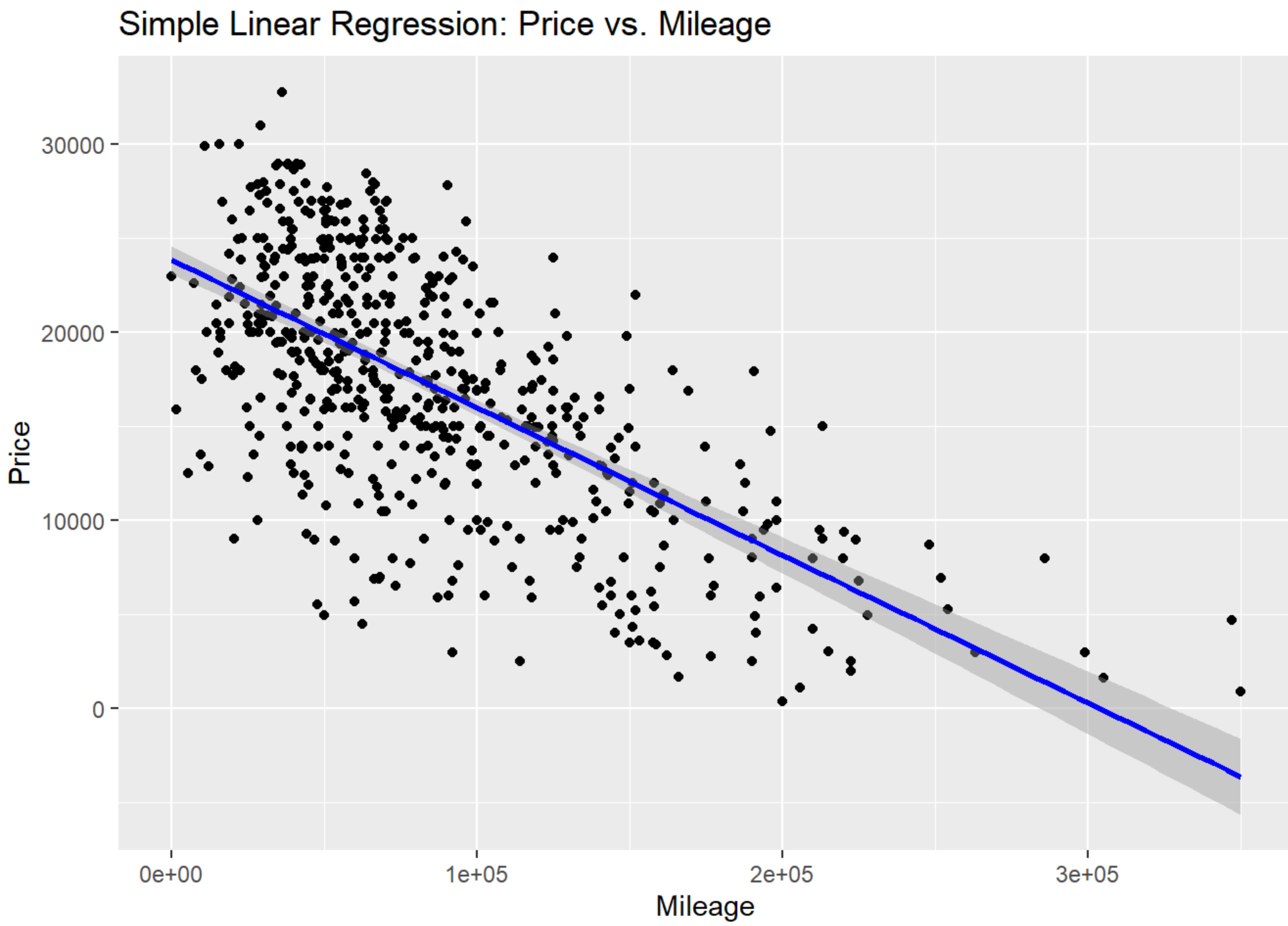
Simple Linear Regression

We first examined the relationship between car price and mileage using simple linear regression.

```
# Simple Linear Regression
slr_model <- lm(price_in_euro ~ mileage_in_km, data = project_df)
summary(slr_model)
```

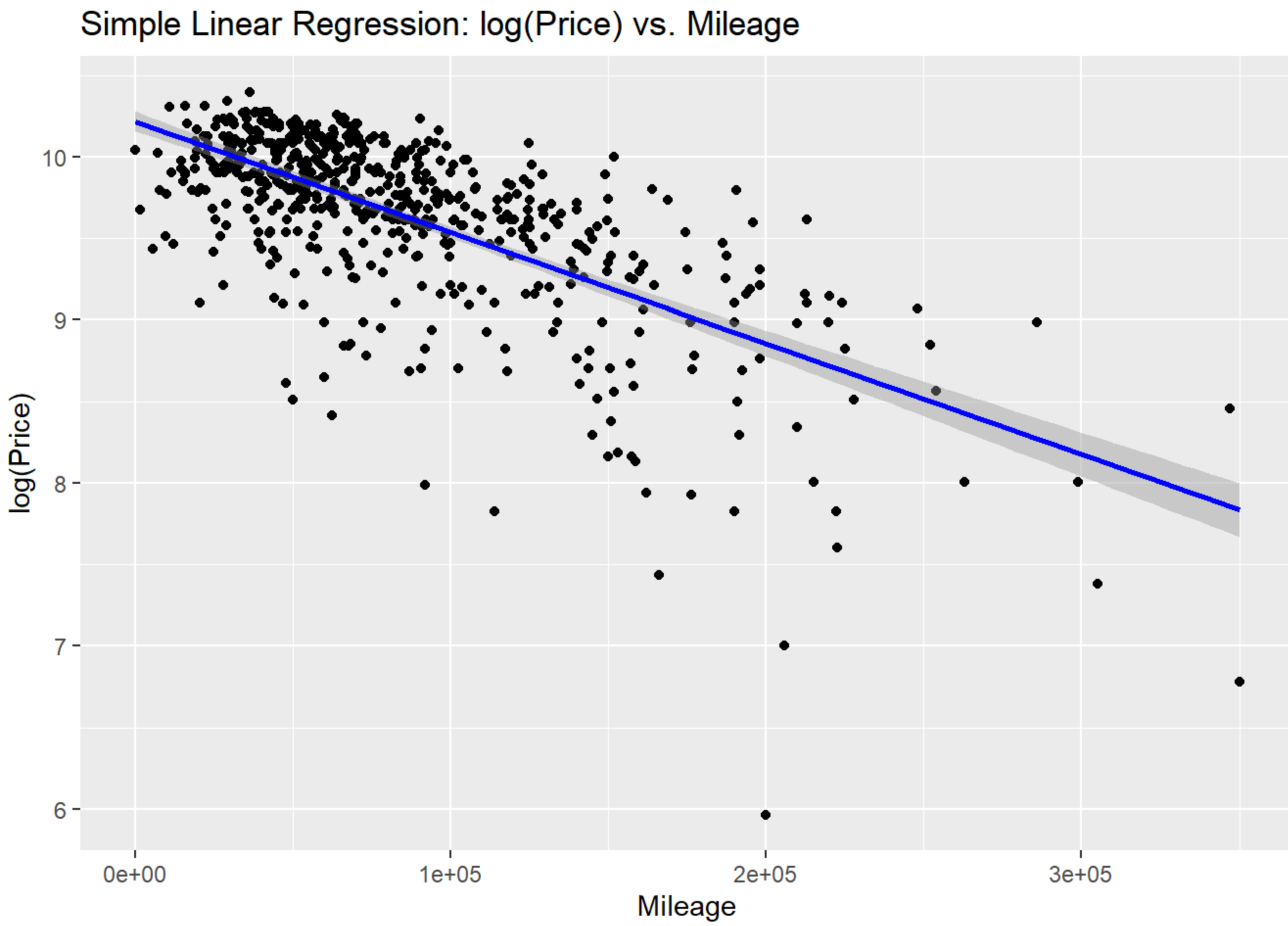
```
##
## Call:
## lm(formula = price_in_euro ~ mileage_in_km, data = project_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14981.4  -2965.0    152.6   3719.9  11782.2
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)    2.386e+04  3.812e+02   62.58   <2e-16 ***
## mileage_in_km -7.853e-02  3.800e-03  -20.67   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5036 on 598 degrees of freedom
## Multiple R-squared:  0.4167, Adjusted R-squared:  0.4157
## F-statistic: 427.1 on 1 and 598 DF,  p-value: < 2.2e-16
```

```
# Plotting the regression line
ggplot(project_df, aes(x = mileage_in_km, y = price_in_euro)) +
  geom_point() +
  geom_smooth(method = "lm", col = "blue") +
  labs(title = "Simple Linear Regression: Price vs. Mileage", x = "Mileage", y = "Price")
```



As you can see from the points on the graph it doesn't quite fit all the model assumptions perfectly, specifically constant variance. We can fix this with a log transformation on price.

```
##
## Call:
## lm(formula = log_price_in_euro ~ mileage_in_km, data = project_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.88979 -0.14341  0.08126  0.26579  0.87184
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    1.022e+01  3.108e-02  328.73   <2e-16 ***
## mileage_in_km -6.809e-06  3.098e-07  -21.98   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4106 on 598 degrees of freedom
## Multiple R-squared:  0.4468, Adjusted R-squared:  0.4459
## F-statistic:   483 on 1 and 598 DF,  p-value: < 2.2e-16
```



Now

after the log transformation you can tell that the points on the graph fit the model assumptions much better. In addition the findings indicated a significant negative relationship between mileage and price, and transforming price via log resolved some issues with non-constant variance.

Multiple Linear Regression

We extended the analysis to include additional variables such as power, fuel consumption, and transmission type.

```
# Multiple Linear Regression
mlr_model <- lm(log_price ~ power_ps + mileage_in_km + fuel_consumption_l_100km + log_fuel_consu
mption, data = train_data_t)
summary(mlr_model)
```

```
##
## Call:
## lm(formula = log_price ~ power_ps + mileage_in_km + fuel_consumption_l_100km +
##     log_fuel_consumption, data = train_data_t)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.34780 -0.11385  0.02778  0.13569  0.94094
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)               6.506e+00  4.928e-01  13.202  < 2e-16 ***
## power_ps                  9.235e-03  4.107e-04  22.483  < 2e-16 ***
## mileage_in_km            -6.575e-06  2.474e-07 -26.579  < 2e-16 ***
## fuel_consumption_l_100km -8.588e-01  1.036e-01  -8.288 1.18e-15 ***
## log_fuel_consumption      4.221e+00  6.162e-01   6.850 2.29e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2752 on 476 degrees of freedom
## Multiple R-squared:  0.7468, Adjusted R-squared:  0.7446
## F-statistic: 350.9 on 4 and 476 DF,  p-value: < 2.2e-16
```

Including more variables improved the model’s performance, with newer cars and those with higher power and specific fuel types priced higher.

Advanced Analysis

Shrinkage Methods

Ridge Regression

```
x_train <- model.matrix(log_price ~ power_ps + mileage_in_km + fuel_consumption_l_100km + log_fu
el_consumption, train_data_t)[, -1]
y_train <- train_data_t$log_price

# Cross-validation for Ridge Regression
set.seed(1)
cv_ridge <- cv.glmnet(x_train, y_train, alpha = 0)
best_lambda_ridge <- cv_ridge$lambda.min
best_lambda_ridge
```

```
## [1] 0.0366351
```

```
# Fit Ridge Regression
ridge_model <- glmnet(x_train, y_train, alpha = 0, lambda = best_lambda_ridge)
show(ridge_model)
```

```
##
## Call:  glmnet(x = x_train, y = y_train, alpha = 0, lambda = best_lambda_ridge)
##
##   Df  %Dev  Lambda
## 1  4 71.79 0.03664
```

LASSO Regression

```
# Cross-validation for LASSO
set.seed(1)
cv_lasso <- cv.glmnet(x_train, y_train, alpha = 1)
best_lambda_lasso <- cv_lasso$lambda.min
best_lambda_lasso
```

```
## [1] 0.0001478966
```

```
# Fit LASSO Regression
lasso_model <- glmnet(x_train, y_train, alpha = 1, lambda = best_lambda_lasso)
lasso_model
```

```
##
## Call:  glmnet(x = x_train, y = y_train, alpha = 1, lambda = best_lambda_lasso)
##
##   Df  %Dev    Lambda
## 1  4 74.67 0.0001479
```

Model Comparison

```
mlr_coef <- coef(mlr_model)
ridge_coef <- as.matrix(coef(ridge_model, s = best_lambda_ridge))
lasso_coef <- as.matrix(coef(lasso_model, s = best_lambda_lasso))

coef_comparison <- data.frame(
  Predictor = rownames(coef(ridge_model)),
  MLR = mlr_coef[match(rownames(coef(ridge_model)), names(mlr_coef))],
  Ridge = ridge_coef,
  LASSO = lasso_coef
)
coef_comparison
```

```
##                                             Predictor           MLR            s1
## (Intercept)                             (Intercept)  6.506262e+00  9.749029e+00
## power_ps                                   power_ps  9.234815e-03  8.289744e-03
## mileage_in_km                         mileage_in_km -6.574502e-06 -6.613049e-06
## fuel_consumption_l_100km fuel_consumption_l_100km -8.587624e-01 -1.420301e-01
## log_fuel_consumption         log_fuel_consumption  4.220797e+00  5.313391e-02
##                                  s1.1
## (Intercept)               6.656574e+00
## power_ps                  9.219872e-03
## mileage_in_km            -6.594964e-06
## fuel_consumption_l_100km -8.262654e-01
## log_fuel_consumption      4.028580e+00
```

Interpretation: The comparison of coefficients across these models shows that both ridge regression and LASSO shrink the coefficients towards zero, addressing multicollinearity and reducing the potential for overfitting. The intercepts in both RR and LASSO are higher than in MLR, indicating a baseline adjustment due to regularization. The coefficients for power_ps and other predictors are closer to each other in ridge regression and LASSO compared to MLR, suggesting a more balanced influence of these variables in the regularized models.

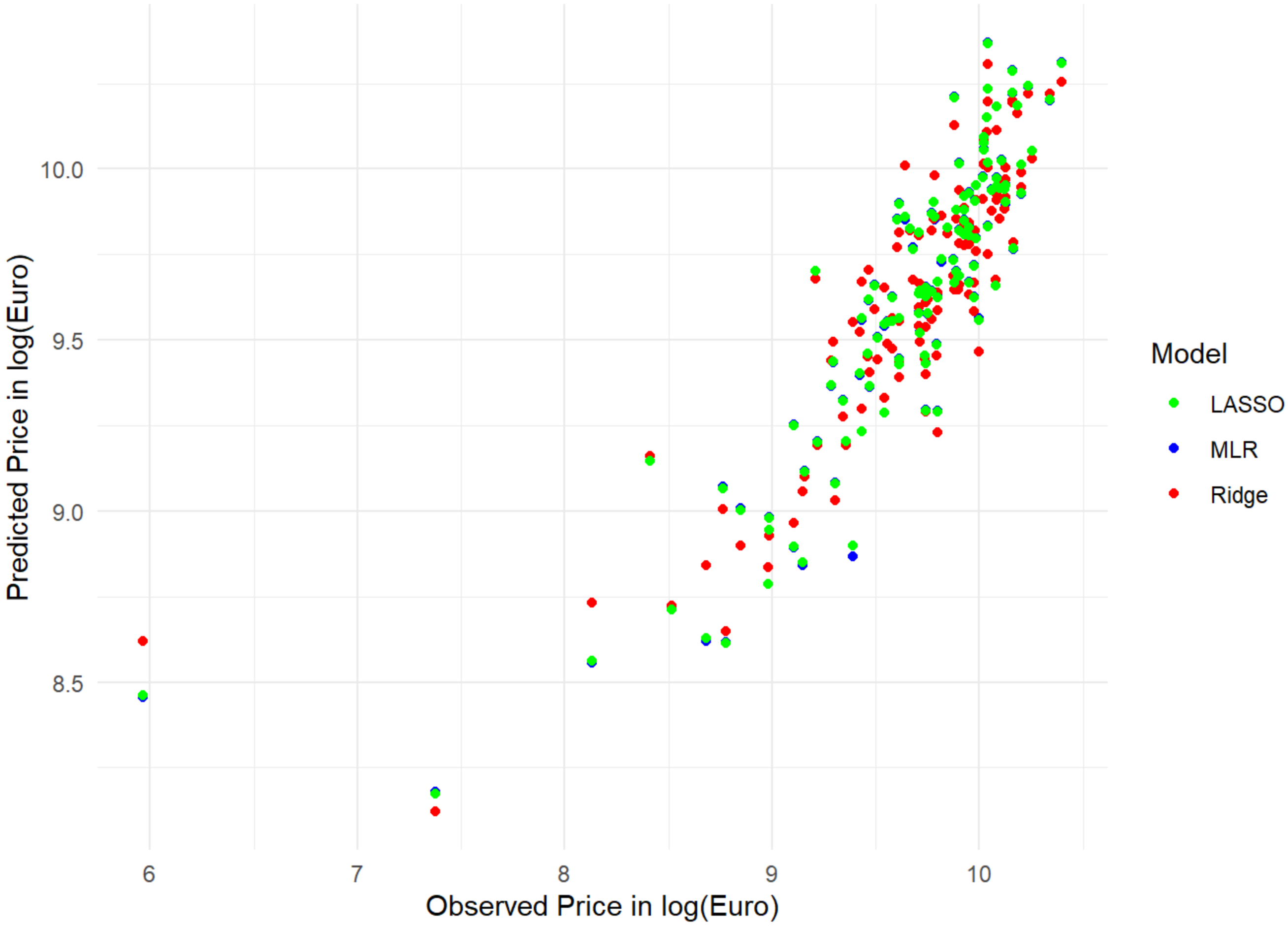
```
x_test <- model.matrix(log_price ~ power_ps + mileage_in_km + fuel_consumption_l_100km + log_fue
l_consumption, new_data)[, -1]
y_test <- new_data$log_price



mlr_pred <- predict(mlr_model, new_data)
ridge_pred <- predict(ridge_model, newx = x_test, s = best_lambda_ridge)
lasso_pred <- predict(lasso_model, newx = x_test, s = best_lambda_lasso)

predictions <- data.frame(
  Observed <- y_test,
  MLR <- mlr_pred,
  Ridge <- ridge_pred,
  LASSO <- lasso_pred
)

ggplot(predictions, aes(x = Observed)) +
  geom_point(aes(y = MLR, color = "MLR"), size=1.5) +
  geom_point(aes(y = Ridge, color = "Ridge"), size=1.5) +
  geom_point(aes(y = LASSO, color = "LASSO"), size=1.5) +
  labs(y = "Predicted Price in log(Euro)", x = "Observed Price in log(Euro)") +
  scale_color_manual(values = c("MLR" = "blue", "Ridge" = "red", "LASSO" = "green"), name = "Mod
el") +
  theme_minimal()
```



The results demonstrated that Ridge and LASSO regression models outperformed the MLR model in terms of stability and robustness.

Innovation: Bootstrapping

We explored bootstrapping to address potential issues with variance.

```
train_data_no_influential <- train_data_t[-c(54887, 55259), ]
model_no_influential <- lm(log_price ~ power_ps + mileage_in_km + fuel_consumption_l_100km + log
_fuel_consumption, data = train_data_no_influential)

sample_d <- train_data_no_influential[sample(nrow(train_data_no_influential), 20, replace = TRU
E), ]
sample_m <- lm(log_price ~ power_ps + mileage_in_km + fuel_consumption_l_100km + log_fuel_consum
ption, data = sample_d)

#set all the coefs to null to start the bootstrap
coef_intercept <- NULL
coef_ps <- NULL
coef_km <- NULL
coef_lkm <- NULL
coef_lfc <- NULL
rsq <- NULL

#raise the living hell out of this number to get a more accurate bootstrap but it will take more
time to compile
for(i in 1:10000){

train_s = sample_d[sample(1:nrow(sample_d), nrow(sample_d), replace = TRUE), ]

  model_bootstrap <- lm(log_price ~ power_ps + mileage_in_km + fuel_consumption_l_100km + log_fu
el_consumption, data = train_s)

  rsq <- c(rsq, summary(model_bootstrap)$r.squared)

  coef_intercept <-
    c(coef_intercept, model_bootstrap$coefficients[1])

  coef_ps <-
    c(coef_ps, model_bootstrap$coefficients[2])

  coef_km <-
    c(coef_km, model_bootstrap$coefficients[3])

  coef_lkm <-
    c(coef_lkm, model_bootstrap$coefficients[4])

  coef_lfc <-
    c(coef_lfc, model_bootstrap$coefficients[5])
}

coefs <- rbind(coef_intercept, coef_ps, coef_km, coef_lkm, coef_lfc)

means_coef_boot = c(mean(coef_intercept), mean(coef_ps), mean(coef_km), mean(coef_lkm), mean(coe
f_lfc))
knitr::kable(round(
  cbind(
    train_model = coef(summary(model_no_influential))[, 1],
```

```
    bootstrap_model = means_coef_boot),6),
  "simple", caption = "Coefficients in different models")
```

Coefficients in different models

	train_model	bootstrap_model
(Intercept)	6.506262	2.754528
power_ps	0.009235	0.013375
mileage_in_km	-0.000007	-0.000007
fuel_consumption_l_100km	-0.858762	-1.662457
log_fuel_consumption	4.220797	8.686412

```r
#all the CI for the different coeffiecients

a <-
  cbind(
    quantile(coef_intercept, prob = 0.025),
    quantile(coef_intercept, prob = 0.975))
b <-
  cbind(
    quantile(coef_ps, prob = 0.025),
    quantile(coef_ps, prob = 0.975))
c <-
  cbind(
    quantile(coef_km, prob = 0.025),
    quantile(coef_km, prob = 0.975))
d <-
  cbind(
    quantile(coef_lkm, prob = 0.025),
    quantile(coef_lkm, prob = 0.975))
e <-
  cbind(
    quantile(coef_lfc, prob = 0.025),
    quantile(coef_lfc, prob = 0.975))



f <-
  round(cbind(
    population = confint(model_no_influential),
    #sample = confint(sample_m),
    boot = rbind(a, b, c, d, e)), 6)
colnames(f) <- c("2.5 %", "97.5 %",
                 "2.5 %", "97.5 %")
knitr::kable(rbind(
  c('train data',
    'train data',
    'bootstrap',
    'bootstrap'),f), caption = "Confidence Intervals from the two models")
```

Confidence Intervals from the two models

	2.5 %	**97.5 %**	**2.5 %**	**97.5 %**
	train data	train data	bootstrap	bootstrap
(Intercept)	5.537887	7.474637	-3.778139	7.75632
power_ps	0.008428	0.010042	0.007816	0.021729
mileage_in_km	-7e-06	-6e-06	-1e-05	-4e-06
fuel_consumption_l_100km	-1.062351	-0.655174	-3.176411	-0.668864
log_fuel_consumption	3.010058	5.431535	2.581736	17.552092

```r
#R squared bootstrap:

g <-
  cbind(
    mean(rsq),
    quantile(rsq, prob = 0.025),
    quantile(rsq, prob =0.975)
    )
h <-
  cbind(summary(model_no_influential)$r.squared, NA, NA)

colnames(g) <- c("mean", "2.5 %", "97.5 %")
rownames(g) <- "Bootstrap Model"
rownames(h) <- "Normal Model"
knitr::kable(rbind(round(rbind(g, h), 3)), caption = "R^2 Results for both models")
```

R^2 Results for both models

	mean	2.5 %	97.5 %
Bootstrap Model	0.853	0.698	0.971
Normal Model	0.747	NA	NA

The bootstrapping results showed that the average R^2 value across the models was better than the R^2 value from our initial model, indicating an improvement.

Summary

This comprehensive analysis of German Mazda car listings involved several steps, including data cleaning, exploratory data analysis, regression modeling, and advanced techniques such as shrinkage methods and bootstrapping. Key findings include the significant negative relationship between mileage and price, the improved model performance with additional variables, and the effectiveness of Ridge and LASSO regressions in providing robust predictions. The bootstrapping technique further validated the model's reliability, ensuring that the assumptions of linear regression were met.

This project not only provided valuable insights into the factors affecting car prices but also demonstrated the application of various statistical methods to achieve accurate and reliable predictions. The combined approach of traditional regression and modern shrinkage techniques offers a comprehensive framework for analyzing complex datasets, ensuring robust and generalizable results.