Optimized Car Price Forecast

Regression vs Machine Learning

MUSTAFA ALMAZERLI

EC Utbildning

April

2024 04

Abstract

Predicting the prices of used cars is a significant challenge in the automotive industry, with implications for both sellers and buyers. In this study, we explore various regression and machine learning techniques to forecast car prices based on their attributes. By analyzing the performance of models on a dataset of car features, we assess their effectiveness in price prediction. Our findings shed light on the suitability of these methods for accurately estimating car prices, offering valuable insights for stakeholders in the automotive market.

Innehållsförteckning

1 Inledning

Att förutsäga priserna på begagnade bilar har blivit allt viktigare inom bilbranschen, med betydande konsekvenser för både säljare och köpare. Med den ökande tillgängligheten av data och avancerade analysmetoder har intresset för att använda regressions- och maskininlärningsmodeller för att förbättra precisionen i prisprognoser ökat markant.

Syftet med denna rapport är att analysera och jämföra effektiviteten hos olika regressions- och maskininlärningsmetoder för att prognostisera bilpriser. Genom att utvärdera olika metoder försöka uppnå rimliga pris på begagnade bilar som tillverkas 2014 och senare med tag på pris och modell. För att uppnå syfte kommer använda en dataset som innehåller information om olika attribut och egenskaper hos nya och begagnade bilar såsom märke, årsmodell, bränsle, körsträcka och växellåda.

för att uppfylla syftet så kommer följande frågeställningar att besvaras:

1. Vilka modell är mest effektiva för att prognostisera begagnade bilpriser baserat på tillgängliga data om märke, årsmodell, bränsle, körsträcka och växellåda?
2. Hur ska vi effektivt hämta bildata från Blocket automatiskt.

2 Teori

2.1 Multipel linjär regression

Multipel linjär regression är en statistisk metod som används för att undersöka sambandet mellan en responsvariabel och flera förklarande variabler. I detta fall använder vi multipel linjär regression för att modellera sambandet mellan priset på en bil (responsvariabel) och olika attribut såsom märke, tillverkningsår, bränsletyp, körsträcka, och växellådstyp.

2.1.1 Modellformulering

Den generella formeln för Multipel linjär regressionsmodell är:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p + \epsilon$$

2.2 Support Vector Machine (SVM)

Support Vector Machine (SVM) är en maskininlärningsalgoritm som används för både klassificering och regression. I detta projekt använder du SVM för att förutsäga priset på bilar baserat på deras attribut.

2.2.1 Modellformulering

$$Y = \beta_0 + \sum i = 1n(\alpha_i \cdot K(x_i, x)) + \epsilon$$

2.2.2 Parametrar och Hyperparametrar

Support Vector Machine (SVM) är en maskininlärningsalgoritm som används för att modellera komplexa sambandet mellan en responsvariabel och flera förklarande variabler. I denna algoritm är det avgörande att välja rätt kernel och justera hyperarmetrar som C och gamma för att optimera modellens prestanda och undvika överanpassning.

För att hitta de bästa parametrarna och hyperparametrarna kan man använda metod som grid search. Nedan är koden som används för att genomföra gird secvh och hitta optimala parametrar för SVM- modellen:

```
# Sök efter de bästa parametrarna och hyperparametrarna
svm_grid <- expand.grid(sigma = c(0.01, 0.1, 1, 10), C = c(0.1, 1, 10, 100))
svm_ctrl <- trainControl(method = "cv", number = 5)
svm_model_tuned <- train(Price ~ ., data = train_data, method = "svmRadial", trControl = svm_ctrl, tuneGrid = svm_grid)
```

3 Metod

För att genomföra denna studie och jämföra effektiviteten hos olika regressions- och maskininlärningsmetoder för att prognostisera bilpriser, samt försöka uppnå rimliga priser på begagnade bilar, följdes en noggrant utformad metodik.

3.1 Datainsamling och förberedelse

Data för denna studie samlades in genom ett grupparbete (egen sida nedan) daten innehåller information om olika attribut och egenskaper hos nya och begagnade bilare såsom märke, årsmodell, bränsle, körsträcka, växellåda och pris.

Innan modellering och analysen påbörjades genomfördes en omfattande förberedelse av datan för att säkerställa kvalitet genom att hantera eventuella brister eller felaktigheter i datan samt att säkerställa att varje rad hade korrekt formaterad information.

3.2 Datamodellering

Data förberedas för modellering genom at delas upp i tre separata delar: träning, test och validering. Mängden är olika från modell till modell. Dt gjordes för att utvärdera modellernas prestanda på oberoende datamängder. Utvärderar modellerna genom att räkna olika prestandamått såsom Root Mean squared error (RMSE), mean absolute error (MAE) och R_squared(R2).

3.3 Utvärdering och jämförelse

Slutligen jämfördes resultaten från SVM och linjär regssion för att bedöma vilken metod som presterande bäst för prediktion av bilpriser.

4 Resultat och Diskussion

4.1 Resultat

Efter tillämpning av linjär regression och Support Vector Machine (SVM) på datasetet för att prognostisera begagnade bilpriser, har vi beräknat Root Mean Squared Error (RMSE) för båda modellerna. Resultaten presenteras nedan:

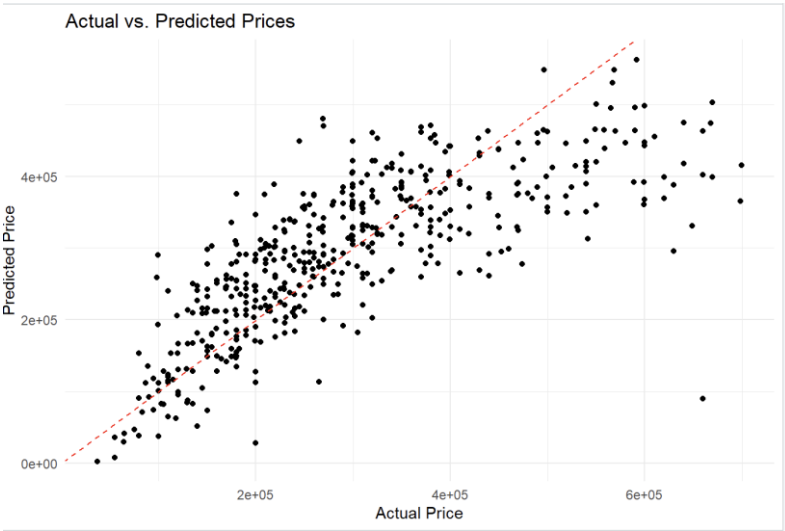
RMSE för olika modeller	
Multipel Linjär Regression	83741.36
Support Vector Machine (SVM)	87439.17
Random Forest	80292.35

4.2 Diskussion

Baserat på RMSE-värdena framgår det att båda multipla linjära regressioner och support vector machine presterade väl för att prognostisera begagnade bilpriser. Medan modellerna tränade på begränsade data såsom årsmodell från 2014 till 2024, bara vissa bilmärken, pris inom intervallet från 50 000 till 700 000. Dessa begränsningar infördes för att förutsäga rimliga bilpris som många har råd att köpa, dessutom valdes dessa begränsning är för att spara tid och vara effektiv i modellträningen.

4.3 testning

vid testningen av modellernas förmåga att prognostiserar bilpriser användes olika utvärderingssäätt, inklusive Root Mean Squared Error (RMSE), R-squared (R^2) och Mean Absolute Error (MAE). Testningen inleddes med att tillämpa modellerna på ett antal bilar som redan fanns i datasetet. Resultaten visade tydliga skillnader mellan de två modellerna. För linjär regression var det vanligt förekommande att modellen prognostiserade priser som var lägre än det verkliga priset på biilen. Å andra sidan hade SVM-modellen ofta prognostiserar priser som var nära det verkliga priset eller hade en mindre avvikelse.



4.4 hämta data automatisk

för att automatisera hämtningen av data om bilar från Blocket-sidan utforskade jag olika källor och testade flera metoder på egen hand, sen [redacted] var med på utforskning. Efter flera försök lyckades jag med att ladda ner data från Blocket-sidan, men upptäckte att de nedladdade sidorna var tomma, vilket var en besvikelse och ett hinder för mitt projekt. Detta ledde mig att fortsätta undersöka och experimentera för att hitta rätt metod för att hämta data på ett tillförlitligt sätt. Trots att jag gjorde flera försök och testade olika tillvägagångssätt lyckades jag aldrig hitta en fullständig och tillförlitlig lösning för att hämta data från Blocket-sidan. Detta visar på utmaningarna med att automatisera hämtningen av data från webbsidor och betonar vikten av att vara beredd på att möta hinder och utforska olika alternativ för att uppnå önskade resultat. Trots att jag inte lyckades med mitt försök lärde jag mig värdefulla läxor om webbskrapning och datahantering som jag kan tillämpa i framtida projekt.

skript nedan laddade ner tomma sidor från Blocket, sen under grupparbete lyckas vi gör webbskrapning

```
2
3  f = open('used_cars.csv', 'w')
4  f.write('date,title,distance,price,fuel,gear,location\n')
5
6  for i in range(1000):
7    my_url = 'https://www.blocket.se/bilar?cg=1020&st=s&l=0&f=p&w=1&o=' + str(i)
8  client = requests.get(my_url).text
9  soup = BeautifulSoup(client, "html.parser")
10
11 try:
12   stop = soup.find_all('div', {'class': 'ads_not_found col-xs-12'})[0].h3.text
13 except:
14   print('Page number: ' + str(i))
15 continue
16
17 cars = soup.find_all('div', {'id': 'item_list'})[0].find_all('article')
18
19 for car in cars:
20   info = car
21
22 try:
23   date = info.div.time.get('datetime')
24 title = info.div.find('a', {'class': 'item_link'}).text.replace(',', '|')
25
26 splited_info = info.div.p.text.split(' | ')
27
28 fuel = splited_info[0]
29 gear = splited_info[1]
30 dist = splited_info[2]
31 price = info.div.find('p', {'class': 'list_price font-large'}).text
32 location = info.div.div.text
33
34 f.write(date + "," + title + "," + dist + "," + price + "," + fuel + "," + gear + "," + location + "\n")
35 except:
36   print('Failed to fetch information for car on page number: ' + str(i))
37 continue
38
39 f.close()
```

sedan försökte med API har läst dokument, har mål att ladda ner ”totalt antal bilar per 1000

invånare” men misslyckade här nedan koden:

```r
library(httr)
library(jsonlite)
library(openxlsx)
# API-länken
url <- "https://api.scb.se/OV0104/v1/doris/sv/ssd/START/TK/TK1001/TK1001A/PersBilarA"
# JSON-frågan
query_json <- toJSON(list(
  query = list(
    list(
      code = "Region",
      selection = list(
        filter = "item",
        values = list("1980")
      )
    ),
    list(
      code = "Agarkategori",
      selection = list(
        filter = "item",
        values = list("060")
      )
    )
  ),
  response = list(
    format = "json"
  )
))
# POST-anropet till API:et
response <- httr::POST(url, body = query_json, encode = "json", verbose())
# Kontrollera svaret
if (httr::http_type(response) == "application/json") {
  # Konvertera JSON-svaret till en data.frame
  data <- jsonlite::fromJSON(content(response, as = "text"), simplifyDataFrame = TRUE)
  # Visa strukturen på datan
  str(data)
  # Spara datan till Excel-fil
  openxlsx::write.xlsx(data, "output.xlsx")
} else {
  # Visa svaret från servern om det inte är JSON
  print(content(response, as = "text"))
}

```

5 Slutsatser

I denna studie utforskade vi effektiviteten hos linjär regression och Support Vector Machine (SVM) för att prognostisera begagnade bilpriser baserat på tillgängliga attribut såsom märke, årsmodell, bränsle, körsträcka och växellådstyp. Genom att tillämpa dessa modeller på en dataset kunde vi dra följande slutsatser:

5.1 Effektivitet av modeller

båda linjära regressioner och SVM visade sig vara effektiva verktyg för att prognostisera begagnade bilpriser. Deras förmåga att göra rimliga och tillförlitliga prisprognoser baserat på tillgängliga attribut är betydande och erbjuder ett värdefullt verktyg för bilbranschen. Det är dock viktigt att notera att denna studie hade vissa begränsningar, inklusive begränsad tillgänglighet av attribut för bilar. Trots att båda modellerna presterade väl, kan det finnas variation i deras prestanda beroende på den specifika dataseten och problemets komplexitet. Noggrann utvärdering och jämförelse av olika modeller är därför avgörande för att välja den mest lämpliga modell.

5.2 Testning

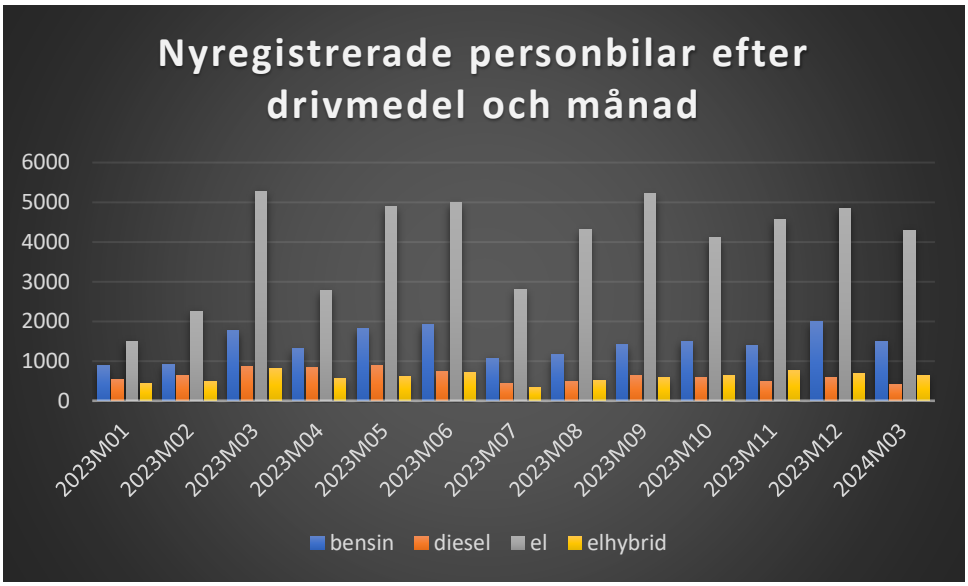
Började testning med att testa min SVM modell innan och efter justera parametrarna och hyperparametrarna med s grid search. Exempel visar samma bil med samma attribut fast olika modeller.

```
> new_car <- data.frame(Brand = "Toyota", Year = 2014, Fuel = "Hybrid", Mileage = 1712
1, Gearbox = "Automat")
> new_car_prediction <- predict(svm_model, newdata = new_car)
> print(new_car_prediction)
       1
132190.8
> new_car <- data.frame(Brand = "Toyota", Year = 2014, Fuel = "Hybrid", Mileage = 1712
1, Gearbox = "Automat")
> new_car_prediction <- predict(best_svm_model, newdata = new_car)
> print(new_car_prediction)
       1
143698.3
```

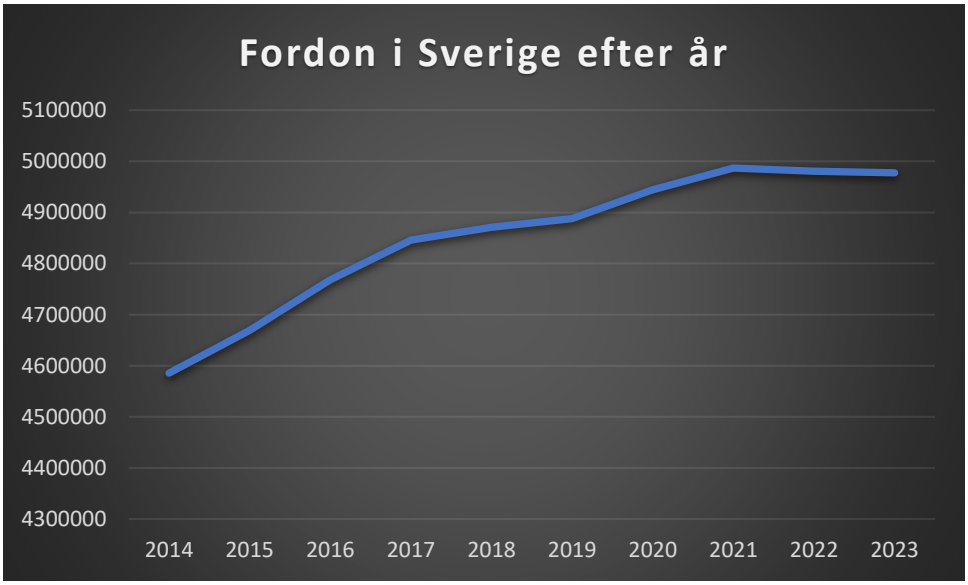
Här testa jag samma bil med Linjär Regression modell

```
> new_car <- data.frame(Brand = "Toyota", Year = 2014, Fuel = "Hybrid", Mileage = 1712
1, Gearbox = "Automat")
> new_car_prediction <- predict(lm_model, newdata = new_car)
> print(new_car_prediction)
       1
103111.9
```

Sammanfattningsvis ger denna studie värdefulla insikter för intressenter inom bilbranschen, inklusive säljare, köpare och återförsäljare, genom att erbjuda en bättre förståelse för de metoder som kan användas för att prognostisera begagnade bilpriser, men modellen behöver mer data och attribut för att kunna prognoserar rimliga priser. diagrammen nedan visar antal nyregistrerande bilar i Sverige efter drivmedel vilken visar att för gör bättre modeller behöver mer data om bl.a. elbilar.



Nedan ser vi en statistik som visar antal bilar i trafik i Sverige, Y axis summa X axis år, diagram visar tydligt att antal bilar i trafik ökar vilket gör att den typ av programmering kommer hjälpa båda köpare och säljare.



6 Appendix A

```
# Läs in nödvändiga paket
library(readxl)
library(dplyr)
library(ggplot2)
library(caret)


# Läs in data från Excel-filen
file <- "C:/Users/musta/OneDrive/Skrivbord/Examinationsuppgift/data_bil.xlsx"
data_bil <- read_excel(file)

# Inspektera data
str(data_bil)
head(data_bil)
summary(data_bil)
View(data_bil)

# Skapa en scatterplot för att visualisera data (t.ex. Price vs. Year)
ggplot(data = data_bil, aes(x = Year, y = Price, color = Fuel)) +
  geom_point(size = 2) +
  labs(x = "Year", y = "Price", title = "Scatterplot of Price vs Year") +
  theme_minimal()

# Dela upp data
set.seed(123)
spec <- c(train = .6, test = .2, validate = .2)

g <- sample(cut(
  seq(nrow(data_bil)),
  nrow(data_bil) * cumsum(c(0, spec)),
  labels = names(spec)
))

res <- split(data_bil, g)
train_data <- res$train
validate_data <- res$validate

# Variabelhantering
train_data$Brand <- as.factor(train_data$Brand)
train_data$Fuel <- as.factor(train_data$Fuel)
train_data$Gearbox <- as.factor(train_data$Gearbox)
```

```r
#data visualisering
ggplot(data = train_data, aes(x = Mileage, y = Price, color = Fuel)) +
  geom_point(size = 2) +
  labs(x = "Mileage", y = "Price", title = "Scatterplot of Price vs Mileage by B
  theme_minimal()

# Skapa en vektor med alla unika nivåer av Brand från både tränings- och valider
all_levels <- union(levels(train_data$Brand), levels(validate_data$Brand))

# Uppdatera faktorn Brand så att den innehåller alla unika nivåer från både trän
train_data$Brand <- factor(train_data$Brand, levels = all_levels)
validate_data$Brand <- factor(validate_data$Brand, levels = all_levels)

# Bygg en linjär regressionsmodell med träningsdata
lm_model <- lm(Price ~ Brand + Year + Fuel + Mileage + Gearbox, data = train_dat
predictions <- predict(lm_model, newdata = validate_data)

summary(lm_model)

par(mfrow = c(2, 2))
plot(lm_model)
geom_point(lm_model)

vif(lm_model)
# Gör förutsägelser igen efter att NA-värden har tagits bort
predictions <- predict(lm_model, newdata = validate_data)
# Beräkna RMSE
validation_rmse <- sqrt(mean((validate_data$Price - predictions)^2))
validation_rmse

# Bygg linjär regressionsmodell med korsvalidering
ctrl <- trainControl(method = "cv", number = 5)
lm_model_cv <- train(Price ~ Brand + Year + Fuel + Mileage + Gearbox,
                     data = train_data,
                     method = "lm",
                     trControl = ctrl)


# Kontrollera vilka variabler som är numeriska
numeric_variables <- sapply(train_data, is.numeric)
print(numeric_variables)
```

Källförteckning

(Schauberger, u.d., s. 1)

(statistikdatabasen, u.d.)

(htt)

Lätt att börja. Svårt att sluta. (youtube.com)

beautifulsoup4 · PyPI

Multiple Linear Regression Using R (youtube.com)