

به نام خدا

فاز دوم پروژه

امیرمحمد رنجبر پازکی ۸۱۰۱۹۵۴۰۲

دکتر بهرک  
استباط آماری

دانشکده مهندسی برق و کامپیووتر دانشگاه تهران

تابستان ۹۹

## تمیز کردن داده ها.

ابتدا با استفاده از دستور زیر تعداد مقادیر گمشده هر ویژگی را به دست می آوریم. تکه کد و خروجی در پایین آمده است.

```
print(sapply(usedCar, function(x) sum(is.na(x))))
```

rownum	price	acquisition_date	badge	body_type
0	3	0	0	0
category	colour	cylinders	economy	fuel
0	0	2488	3920	0
last_updated	litres	location	make	model
0	2488	0	0	0
odometer	transmission	year		
1550	0	0		

همان طور که مشاهده می شود، ویژگی های `liters`, `economy`, `cylinders`, `price` دارای داده های گمشده هستند.

Price یکی از مهمترین ویژگی های این مجموعه داده است و تعداد داده های گمشده آن بسیار کم است. اگر آن مقادیر را پر کنیم، ممکن است تاثیرگذار در بررسی ها باشد. به این دو دلیل، این داده ها را حذف می کنیم.

```
usedCars <- usedCar[!is.na(usedCar$price),]
```

در بقیه ویژگی ها، تعداد داده های دارای مقادیر گمشده قابل توجه هستند. به همین دلیل، این داده ها را حذف نمی کنیم و با مقادیر میانی جایگزین می کنیم. برای مقادیر عددی با میانگین و برای مقادیر کیفی با میانه مقادیر گمشده را پر می کنیم. (البته می توانیم از `regression` برای پر کردن این مقادیر استفاده کنیم).

مقادیر عددی ویژگی های `odometer` و `economy` و دسته بندی شده ویژگی های `liters` و `cylinders` هستند.

```
usedCars$economy <- ifelse(is.na(usedCars$economy), mean(usedCars$economy, na.rm=TRUE), usedCars$economy)
usedCars$odometer <- ifelse(is.na(usedCars$odometer), mean(usedCars$odometer, na.rm=TRUE), usedCars$odometer)
usedCars$litres <- ifelse(is.na(usedCars$litres), median(usedCars$litres, na.rm=TRUE), usedCars$litres)
usedCars$cylinders <- ifelse(is.na(usedCars$cylinders), median(usedCars$cylinders, na.rm=TRUE), usedCars$cylinders)
```

سوال 1. دو متغیر `make` و `body_type` انتخاب شده متغیر های `categorical` هستند. برای بررسی تعداد سطح های این دو متغیر از دستور زیر استفاده می کنیم. خروجی نیز آمده است.

```
print(levels(usedCars$body_type))
print(levels(usedCars$make))
```

""	"Conv"	"Convertible"	"Coup"	"Hatch"	"Sedan"	"Suv"	"SUV"
"Wagon"							
"Subaru"							
"Toyota"							

همانطور که مشاهده می‌کنید، متغیر `body_type` بیش از دو سطح دارد؛ پس انتخاب‌های مناسبی هستند. تنها نکته‌ای که وجود دارد، وجود سطح بی‌نام در هر متغیر `body_type` است که مربوط به سطر پر نشده است. تعداد این داده‌ها ۲ عدد است که در مقایسه با تعداد کل داده‌ها ناچیز است؛ پس، این داده‌ها را حذف می‌کنیم. تکه کد مربوط به این بخش و تعداد داده‌های خالی قبل و پس از انجام این کار در زیر آمده است.

```
print(nrow(usedCars[usedCars$body_type == "",])) [1] 2
usedCars <- usedCars[usedCars$body_type != "",]
print(nrow(usedCars[usedCars$body_type == "",])) [1] 0
```

a. سطح درنظر گرفته شده برای متغیر `Sedan` و برای متغیر `Subaru` است. در این سوال باید بازه اطمینانی برای اختلاف نسبت این دو دربیاوریم. (با توجه به پاسخ تی ای) شرایط نیز برقرار است.

نمونه‌برداری به صورت تصادفی انجام شده است. سایز نمونه کمتر از ۱۰ درصد جامعه است. همچنین، در هر گروه ۱۰ نمونه موفقیت (سطح‌های فرض شده) و ۱۰ نمونه از سایر سطوح وجود دارد.

## point estimate $\pm$ margin of error

$$(\hat{p}_1 - \hat{p}_2) \pm z^* SE_{(\hat{p}_1 - \hat{p}_2)}$$

$$SE = \sqrt{\frac{\hat{p}_1(1 - \hat{p}_1)}{n_1} + \frac{\hat{p}_2(1 - \hat{p}_2)}{n_2}}$$

برای این کار از فرمول بالا استفاده می‌کنیم و این مقادیر را با استفاده از R محاسبه می‌کنیم. تکه کد و نتایج در زیر آمده است.

```
## a ##
subaru_count <- nrow(usedCars[usedCars$make == "Subaru",])
subaru_pe <- subaru_count / nrow(usedCars)
sedan_count <- nrow(usedCars[usedCars$body_type == "Sedan",])
sedan_pe <- sedan_count / nrow(usedCars)
point_estimate <- subaru_pe - sedan_pe
standard_error <- sqrt((subaru_pe * (1-subaru_pe)/subaru_count)+(sedan_pe * (1-sedan_pe)/sedan_count))
z_limit <- 0.05 / 2
z_star <- -qnorm(z_limit)
margin_of_error <- standard_error * z_star
lower_bound <- point_estimate - margin_of_error
upper_bound <- point_estimate + margin_of_error
print(paste("95% CI: (", lower_bound, ", ", upper_bound, ")"))
"95% CI: ( 0.533584969888772 , 0.555217918347897 )"
```

۹۵ درصد اطمینان داریم که درصد ماشین‌هایی که برند **Subaru** ساخته‌است، بین ۵۳.۳۵ تا ۵۵.۵۲ درصد بیشتر از درصد ماشین‌هایی با نوع **Sedan** است.

**b**. برای تست استقلال دو متغیر باید از آزمون استقلال **chi\_squared** استفاده کنیم. به این منظور باید ابتدا جدول **observation** را از داده‌ها تشکیل دهیم و سپس، از روی این جدول، جدول **expected** را به دست آوریم. تکه کد و نتایج این بخش در زیر آمده است.

```
observation_table <- table(usedCars$make, usedCars$body_type)
expected_table <- as.array(margin.table(observation_table, 1)) %*% t(as.array(margin.table(observation_table, 2))) / margin.table(observation_table)
print(observation_table)
print(expected_table)
```

	Conv	Convertible	Coup	Hatch	Sedan	Suv	SUV	Wagon
Subaru	0	0	0	6870	5448	578	13177	335
Toyota	0	2	2	8	0	500	11312	269

	Conv	Convertible	Coup	Hatch	Sedan	Suv	SUV	Wagon	
Subaru	0	1.3718085	1.3718085	5.487234	4712.162	3736.806	739.4048	16797.109	414.2862
Toyota	0	0.6281915	0.6281915	2.512766	2157.838	1711.194	338.5952	7691.891	189.7138

شرایط آزمون نیز برقرار است. Observation‌ها از یکدیگر مستقلند چراکه به صورت تصادفی نمونه‌برداری شده است. تعداد نمونه‌برداری کمتر از ۱۰ درصد جامعه است. هر مورد نیز صرفا در یک خانه قرار می‌گیرد. از طرفی در هر خانه جدول ۵ مورد تقریباً داریم. (به جز دو خانه که می‌توانیم ستون اول و دوم رو با هم ترکیب کنیم. این کار کمی دقت را بالاتر می‌برد). حال به محاسبه آماره آزمون و درجه آزادی می‌پردازیم. (نحوه محاسبه در زیر آمده است). سپس، با استفاده از این موارد به محاسبه **p-value** می‌پردازیم.

$$\chi^2 = \sum_{i=1}^k \frac{(O - E)^2}{E}$$

$$df = (R - 1) \times (C - 1)$$

تکه کد و نتیجه در زیر آمده است.

```
chi_stat <- sum((expected_table - as.array(observation_table))^2 / expected_table)
df <- (length(levels(usedCars$make))-1) * (length(levels(usedCars$body_type))-1)
p_value <- pchisq(chi_stat, df = df, lower.tail = FALSE)
print(paste("P-value: ", p_value))
[1] "P-value: 0"
```

$P\text{-value}$  صفر شد و به همین دلیل، فرض صفر رد می‌شود. پس، دو متغیر `make` و `body_type` به یکدیگر وابسته هستند و از هم مستقل نیستند.

**سؤال ۲.** متغیر انتخابی متغیر `transmission` است که دو سطح `manual` و `automatic` دارد و `binary` است. البته لازم به ذکر است این متغیر نیز ۷ داده واردنده دارد که می‌توان به دلیل کم بودن آن‌ها را حذف کرد. تکه کد زیر این کار را انجام می‌دهد.

```
print(nrow(usedCars[usedCars$transmission=="",])) [1] 7
usedCars <- usedCars[usedCars$transmission != "",] [1] 0
print(nrow(usedCars[usedCars$transmission=="",]))
```

ابتدا نمونه کوچکی به اندازه ۱۰ برمی‌داریم. موفقیت را دستی بودن ماشین در نظر می‌گیریم. این نسبت را محاسبه می‌کنیم که  $\hat{p}$  مسئله است. آزمون فرض به صورت زیر می‌تواند بیان شود.

فرض صفر: دستی بودن یا نبودن ماشین‌ها تصادفی است.  $p=0.5$

فرض مقابل: دستی بودن یا نبودن ماشین‌ها تصادفی نیست و درصد ماشین‌های دستی بیشتر است.  $p>0.5$

به این دلیل که سایز نمونه ما بسیار کوچک است، شرط دوم استفاده از قضیه حد مرکزی برقرار نیست. پس به سراغ `simulation` می‌رویم. فرض می‌کنیم یک سکه `fair` داریم که شانس رو و پشت امدن آن برابر است. در هر آزمایش ۱۰ بار آن را پرتتاب می‌کنیم و نسبت رو آمدن را حساب می‌کند. این آزمایش را ۱۰۰۰ بار تکرار می‌کنیم و سپس، با توجه به نتایج `p-value` را محاسبه می‌کنیم و نتیجه‌گیری می‌کنیم. تکه کد و `p-value` در زیر قابل مشاهده است.

```
first_sample <- sample(usedCars$transmission, size = 10)
manual_count <- length(first_sample[first_sample == "Manual"])
p_hat <- manual_count / 10
proportions <- vector()
for (n in 1:1000) {
  success_count <- 0
  for (i in 1:10) {
    random_number <- runif(1)
    if (random_number >= 0.5) {
      success_count <- success_count + 1
    }
  }
  new_prop <- success_count / 10
  proportions <- c(proportions, new_prop)
}
extreme_result <- 0
for (n in 1:1000) {
  if (proportions[n] >= p_hat) {
    extreme_result <- extreme_result + 1
  }
}
```

```

p_value <- extreme_result / 1000
print(paste("P-value:", p_value))

```

"P-value: 0.636"

مقدار **p-value** بسیار زیاد است و بیشتر از ۰.۵ است. در نتیجه، فرض صفر را نمی‌توان رد کرد. در نتیجه نسبت دستی و اتوماتیک بودن ۰.۵ است و شواهد کافی برای رد آن وجود ندارد.

**سوال ۳.a.** متغیر انتخاب شده متغیر **body\_type** است. توزیع احتمال دسته‌های مختلف با استفاده از تکه کد زیر محاسبه شده است. توزیع احتمالاتی در ادامه آمده است.

```

for (body_type in levels(usedCars$body_type)) {
  body_type_count <- nrow(usedCars[usedCars$body_type == body_type,])
  body_type_prob <- body_type_count / nrow(usedCars)
  print(paste(body_type, ":", body_type_prob))
}

```

"Conv : 5.19561490102354e-05"  
 "Convertible : 5.19561490102354e-05"  
 "Coup : 0.000207824596040941"  
 "Hatch : 0.178443393775653"  
 "Sedan : 0.141424637605861"  
 "Suv : 0.0279524081675066"  
 "SUV : 0.636177066555827"  
 "Wagon : 0.0156907570010911"

سپس دو نمونه یکی به صورت تصادفی و یکی به صورت **biased** انتخاب می‌کنیم. نمونه صرفا شامل **SUV** و **Sedan** می‌باشد.

```

random_sample <- usedCars[sample(nrow(usedCars), size = 100),]
sedan_suv_cars <- usedCars[usedCars$body_type == "SUV" | usedCars$body_type == "Sedan",]
biased_sample <- sedan_suv_cars[sample(nrow(sedan_suv_cars), size = 100),]

```

حال با اجرای **GoF** تطابق نمونه‌ها با توزیع اصلی را بررسی می‌کنیم. تکه کد و **p-value**‌ها در زیر آمده است.

```

random_sample <- usedCars[sample(nrow(usedCars), size = 100),]
sedan_suv_cars <- usedCars[usedCars$body_type == "SUV" | usedCars$body_type == "Sedan",]
biased_sample <- sedan_suv_cars[sample(nrow(sedan_suv_cars), size = 100),]
random_count <- vector()
biased_count <- vector()
for (body_type in levels(usedCars$body_type)) {
  body_type_count <- nrow(random_sample[random_sample$body_type == body_type,])
  random_count <- c(random_count, body_type_count)
  body_type_count <- nrow(biased_sample[biased_sample$body_type == body_type,])
  biased_count <- c(biased_count, body_type_count)
}
random_expected <- probs * 100
biased_expected <- probs * 100
random_chi <- sum((random_count - random_expected)^2 / random_expected)
biased_chi <- sum((biased_count - biased_expected)^2 / biased_expected)
df <- length(random_count)
random_p_value <- pchisq(random_chi, df = df, lower.tail = FALSE)
biased_p_value <- pchisq(biased_chi, df = df, lower.tail = FALSE)

```

```

print(paste("Random P-value: ", random_p_value))
print(paste("Biased P-value:", biased_p_value))

"Random P-value: 0.88910208470535"
"Biased P-value: 0.000144820112111439"

```

همانطور که مشاهده می‌شود، مقدار **p-value** برای نمونه تصادفی خیلی بزرگ است و در نتیجه، نمی‌توان فرض صفر را رد کرد و تطابق با توزیع اصلی وجود دارد.

مقدار **p-value** برای نمونه با سوگیری بسیار کم است و در نتیجه، فرض صفر رد می‌شود. پس، این نمونه با توزیع اصلی تطابق ندارد.

**b**. متغیر انتخاب شده متغیر **make** است. برای تست استقلال دو متغیر باید از آزمون استقلال استفاده کنیم. به این منظور باید ابتدا جدول **observation** را از داده‌ها تشکیل دهیم و سپس، از روی این جدول، جدول **expected** را به دست آوریم. تکه کد و نتایج این بخش در زیر آمده است.

```

observation_table <- table(usedCars$make, usedCars$body_type)
expected_table <- as.array(margin.table(observation_table, 1)) %*% t(as.array(margin.table(observation_table, 2))) / margin.table(observation_table)
print(observation_table)

```

	Conv	Convertible	Coup	Hatch	Sedan	Suv	SUV	Wagon	
Subaru	0	0	0	0	6870	5448	578	13177	335
Toyota	0	2	2	8	0	0	500	11312	269

	Conv	Convertible	Coup	Hatch	Sedan	Suv	SUV	Wagon	
Subaru	0	1.3718085	1.3718085	5.487234	4712.162	3736.806	739.4048	16797.109	414.2862
Toyota	0	0.6281915	0.6281915	2.512766	2157.838	1711.194	338.5952	7691.891	189.7138

شرایط آزمون نیز برقرار است. Observation‌ها از یکدیگر مستقلند چراکه به صورت تصادفی نمونه‌برداری شده است. تعداد نمونه‌برداری کمتر از ۱۰ درصد جامعه است. هر مورد نیز صرفا در یک خانه قرار می‌گیرد. از طرفی در هر خانه جدول ۵ **expected** مورد تقریباً داریم. (به جز دو خانه که می‌توانیم ستون اول و دوم رو با هم ترکیب کنیم. این کار کمی دقیق‌تر را بالاتر می‌برد.) حال به محاسبه آماره آزمون و درجه آزادی می‌پردازیم. (نحوه محاسبه در زیر آمده است). سپس، با استفاده از این موارد به محاسبه **p-value** می‌پردازیم.

$$\chi^2 = \sum_{i=1}^k \frac{(O - E)^2}{E}$$

$$df = (R - 1) \times (C - 1)$$

تکه کد و نتیجه در زیر آمده است.

```

chi_stat <- sum((expected_table - as.array(observation_table))^2/expected_table)
df <- (length(levels(usedCars$make))-1) * (length(levels(usedCars$body_type))-1)
p_value <- pchisq(chi_stat, df = df, lower.tail = FALSE)
print(paste("P-value: ", p_value))
[1] "P-value: 0"

```

$P\text{-value}$  صفر شد و به همین دلیل، فرض صفر رد می‌شود. پس، دو متغیر `make` و `body_type` به یکدیگر وابسته هستند و از هم مستقل نیستند.

سوال ۴. متغیر `response` انتخابی متغیر `price` است. یکی از بهترین تخمین‌گرهای این متغیر `odometer` می‌تواند باشد چراکه هرچقدر کیلومتر ماشین بیشتر شود، قیمت آن کمتر می‌شود.

.a. کد و نتیجه در زیر آمده است.

```
price_lr <- lm(price ~ odometer, data = usedCars)
print(summary(price_lr))
```

Residuals:

Min	1Q	Median	3Q	Max
-32416	-4922	-1080	2925	12313335

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	3.248e+04	5.148e+02	63.10	<2e-16 ***
odometer	-1.122e-01	4.290e-03	-26.15	<2e-16 ***
---				
Signif. codes:	0 ‘***’	0.001 ‘**’	0.01 ‘*’	0.05 ‘.’
	0.1 ‘ ’	1		

Residual standard error: 64900 on 38492 degrees of freedom

Multiple R-squared: 0.01746, Adjusted R-squared: 0.01743

F-statistic: 683.9 on 1 and 38492 DF, p-value: < 2.2e-16

.b

$$\hat{price} = 32480 - 0.1122 \times odometer$$

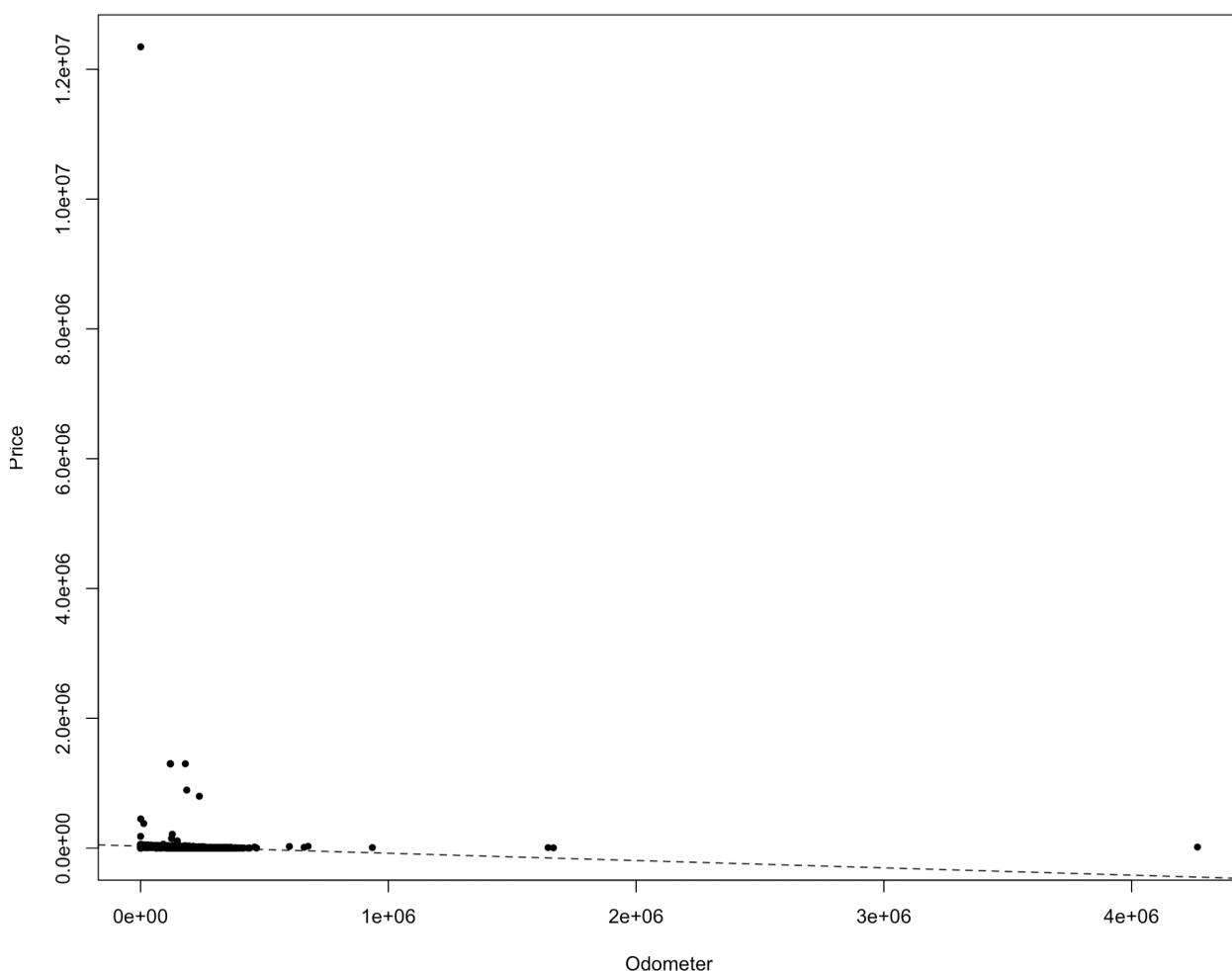
عرض از مبدا: ماشینی با کیلومتر صفر قیمت تخمینی برابر ۳۲۴۸۰ دلار دارد.

شیب: به ازای هر کیلومتر اضافه کارکرد، قیمت ماشین به طور میانگین ۰.۱۱۲۲ دلار کاهش می‌یابد.

.c. کد و نمودار مربوط در زیر قابل مشاهده‌اند.

```
plot(usedCars$odometer, usedCars$price, main="Price vs. Odometer", xlab = "Odometer", ylab = "Price", pch = 20)
abline(price_lr, lty = 2)
```

Price vs. Odometer



**d.** آزمون فرض برای این تست کردن شاخص بودن تخمین‌گر به صورت زیر است.

فرض صفر: شیب خط رگرسیون = 0 است و تخمین‌گر رابطه خطی با پاسخ ندارد.

فرض مقابل: شیب خط رگرسیون ≠ 0 نیست و تخمین‌گر رابطه خطی با پاسخ دارد.

کد برای ساخت مدل و آزمون فرض در زیر آمده است. نتیجه نیز در قالب جدولی نمایش داده شده است.

```
price_little_lr <- lm(price ~ odometer, data = random_sample)
print(summary(price_little_lr))
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	3.253e+04	2.509e+03	12.964	1.35e-12 ***
odometer	-1.213e-01	2.289e-02	-5.298	1.73e-05 ***
---				

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

همانطور که در جدول بالا مشاهده می‌شود، این متغیر  $p\_value$  برابر  $1.73 \times 10^{-5}$  دارد که مقدار بسیار کمی است. پس، فرض صفر رد می‌شود و فرض مقابل پذیرفته می‌شود. پس، این متغیر تخمین‌گر خوبی است.

۲. کد محاسبه بازه اطمینان و بازه اطمینان ۹۵ درصدی در زیر آمده است.

```
lr_summary <- summary(price_little_lr)
lr_coeficients <- lr_summary$coefficients
point_estimate <- lr_coeficients["odometer", "Estimate"]
df <- 25
t_star <- -qt(0.025, df=df)
standard_error <- lr_coeficients["odometer", "Std. Error"]
margin_of_error <- t_star * standard_error
lower_bound <- point_estimate - margin_of_error
upper_bound <- point_estimate + margin_of_error
print(paste("95% CI: (", lower_bound, ", ", upper_bound, ")"))
"95% CI: (-0.158421739810758 , -0.068042698509586 )"
```

۹۵ درصد اطمینان داریم که به ازای هر کیلومتر اضافه کار کردن، قیمت ماشین به طور میانگین از ۰۶۸۰ تا ۱۵۸۴ دلار کاهش پیدا می کند.

۳. متغیر انتخابی دیگر متغیر **transmission** است که بیانگر دستی یا اتوماتیک بودن ماشین هاست. کد و مدل ساخته شده در زیر آمده است.

```
price_bigger_lr <- lm(price ~ odometer + transmission, data=random_sample)
print(summary(price_bigger_lr))
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t )		
(Intercept)	3.931e+04	4.572e+03	8.597	8.62e-09 ***		
odometer	-1.632e-01	4.038e-02	-4.041	0.000475 ***		
transmissionManual	3.658e+03	4.813e+03	0.760	0.454609		
---						
Signif. codes:	0 ‘***’	0.001 ‘**’	0.01 ‘*’	0.05 ‘.’	0.1 ‘ ’	1

Residual standard error: 11070 on 24 degrees of freedom

Multiple R-squared: 0.4114, Adjusted R-squared: 0.3624

F-statistic: 8.389 on 2 and 24 DF, p-value: 0.001728

Adjusted R-squared در مدل قبلی ۰.۳۷۳۲ و در مدل جدید ۰.۳۶۲۴ است و این نشان دهنده این است که متغیر جدید به اندازه کافی مفید نبوده است.

Residual standard error: 10980 on 25 degrees of freedom	Residual standard error: 11070 on 24 degrees of freedom
Multiple R-squared: 0.3973,	Adjusted R-squared: 0.3732
F-statistic: 16.48 on 1 and 25 DF, p-value: 0.0004254	Multiple R-squared: 0.4114,
print(anova(price_little_lr))	Adjusted R-squared: 0.3624
print(anova(price_bigger_lr))	F-statistic: 8.389 on 2 and 24 DF, p-value: 0.001728

```
Response: price
  Df Sum Sq Mean Sq F value Pr(>F)
odometer  1 2681263423 2681263423 129.67 2.19e-11 ***
Residuals 25 516935359 20677414
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
Response: price
  Df Sum Sq Mean Sq F value Pr(>F)
odometer  1 2681263423 2681263423 132.358 2.976e-11 ***
transmission 1 30751189 30751189 1.518 0.2299
Residuals 24 486184170 20257674
```

کد و خروجی جدول ANOVA برای این دو مدل در بالا قابل مشاهده است. از تقسیم مجموع **sum sq** های متغیرها مدل بر مجموع این متغیرها با **R-squared** مقدار **residuals** به دست می آید. بر این مبنای مدل با متغیر بیشتر بهتر است چراکه تعداد متغیرهای آن بالا رفته است و به عبارتی، میزان **explained variability** افزایش یافته است. در نتیجه، مدل پیشرفت کرده است اما خطر **overfitting** وجود دارد.

**سوال ۵.** a. برای این قسمت از معیار **p-value** استفاده می کنیم. در **backward Elimination** ابتدا مدل کامل را می سازیم و سپس، متغیر با بالاترین مقدار **p-value** را حذف می کنیم و آنقدر این کار را ادامه می دهیم تا تمام متغیرهایمان **significant** شوند. به دلیل، تعداد بالای متغیرها روند انتخاب در کد آمده است ولی نتایج مدل نهایی و کد مربوطه در زیر قابل مشاهده است. به دلیل تعداد بالای متغیرها بعضی متغیرها بی ربط دستی حذف شده اند. به ترتیب این متغیرها حذف می شوند.

Make  
Model  
Body\_type  
Litres  
Fuel  
Cylinders  
Location  
Transmion

```
first_step_model <- lm(price ~ body_type + category + colour + cylinders + economy + fuel + litres + location + make + model + odometer+ transmission + year, data = usedCars)
print(summary(first_step_model))
second_step_model <- lm(price ~ body_type + category + colour + cylinders + economy + fuel + litres + location + model + odometer + transmission + year, data = usedCars)
print(summary(second_step_model))
third_step_model <- lm(price ~ body_type + category + colour + cylinders + economy + fuel + litres + location + odometer + transmission + year, data = usedCars)
print(summary(third_step_model))
fourth_step_model <- lm(price ~ category + colour + cylinders + economy + fuel + litres + location + odometer + transmission + year, data = usedCars)
print(summary(fourth_step_model))
fifth_step_model <- lm(price ~ category + colour + cylinders + economy + fuel + location + odometer + transmission + year, data = usedCars)
print(summary(fifth_step_model))
sixth_step_model <- lm(price ~ category + colour + cylinders + economy + location + odometer + transmission + year, data = usedCars)
print(summary(sixth_step_model))
seventh_step_model <- lm(price ~ category + colour + economy + location + odometer + transmission + year, data = usedCars)
print(summary(seventh_step_model))
eighth_step_model <- lm(price ~ category + colour + economy + odometer + transmission + year, data = usedCars)
print(summary(eighth_step_model))
ninth_step_model <- lm(price ~ category + colour + economy + odometer + year, data = usedCars)
print(summary(ninth_step_model))
```

بخش بعدی این قسمت مربوط به ساخت مدل با استفاده از **forward selection** است. این مدلسازی به این صورت است که از مدل خالی شروع کرده و به ازای تمام متغیرها مدل می سازیم. هر کدام **p-value** کمتری از بقیه داشت، **Significant** تر است و برای مدل انتخاب می شود و دوباره همین روند تکرار می شود تا زمانی که هیچ یک از متغیرها **significant** نشوند. با توجه به بخش قبل و داشتن ۱۶ متغیر به نظر می رسد که برای این قسمت باید دست کم ۷۰ مدل ساخته شود تا به جواب قبلی بررسیم که این عدد امکان پذیر نیست.

Coefficients:		Estimate	Std. Error	t value	Pr(> t )
(Intercept)	-1.240e+06	2.788e+05	-4.449	8.66e-06	***
categoryDemo	-8.872e+02	2.233e+03	-0.397	0.691084	
categoryOther	1.794e+02	1.883e+03	0.095	0.924090	
categoryOther2	-2.774e+04	3.770e+03	-7.359	1.88e-13	***
categoryPrivate	-6.845e+03	1.513e+03	-4.525	6.05e-06	***
categoryUsed	-5.708e+03	1.477e+03	-3.865	0.000111	***
colour/cloth	-4.434e+04	6.475e+04	-0.685	0.493520	
colourBeige	-3.807e+04	1.385e+04	-2.750	0.005970	**
colourBlack	-3.603e+04	3.274e+03	-11.005	< 2e-16	***
colourBlue	-3.625e+04	3.250e+03	-11.152	< 2e-16	***
colourBrown	-3.464e+04	9.580e+03	-3.616	0.000299	***
colourBurgundy	-4.082e+04	7.657e+03	-5.331	9.82e-08	***
colourCrystal Pearl	-3.250e+04	5.300e+03	-6.132	8.78e-10	***
colourDeep Cherry	-3.925e+04	3.748e+04	-1.047	0.295078	
colourDune	-3.849e+04	6.475e+04	-0.594	0.552245	
colourEbony	-3.726e+04	4.979e+03	-7.484	7.38e-14	***
colourEnvy	-3.499e+04	4.585e+04	-0.763	0.445355	
colourGlacier	-4.364e+04	6.476e+04	-0.674	0.500337	
colourGold	-3.818e+04	4.357e+03	-8.763	< 2e-16	***
colourGraphite	-3.574e+04	5.115e+03	-6.987	2.86e-12	***
colourGreen	-3.703e+04	4.292e+03	-8.629	< 2e-16	***
colourGrey	-3.600e+04	3.225e+03	-11.162	< 2e-16	***
colourHazel	-2.789e+04	4.585e+04	-0.608	0.543052	
colourInferno	-3.148e+04	1.895e+04	-1.661	0.096779	.
colourInk	-3.193e+04	8.198e+03	-3.894	9.86e-05	***
colourMagenta	-3.823e+04	4.584e+04	-0.834	0.404341	
colourMetal Storm	-3.883e+04	3.250e+04	-1.195	0.232159	
colourOrange	-3.012e+04	6.487e+03	-4.642	3.45e-06	***
colourOther	-4.140e+04	8.263e+03	-5.011	5.44e-07	***
colourPurple	-3.342e+04	1.305e+04	-2.560	0.010474	*
colourRed	-3.730e+04	3.376e+03	-11.049	< 2e-16	***
colourSandstone	-4.025e+04	2.070e+04	-1.944	0.051874	.
colourSilver	-3.686e+04	3.178e+03	-11.599	< 2e-16	***
colourWhite	-3.601e+04	3.171e+03	-11.354	< 2e-16	***
colourWildfire	-3.553e+04	6.807e+03	-5.220	1.80e-07	***
colourYellow	-3.536e+04	1.447e+04	-2.444	0.014526	*
economy	9.886e+02	3.435e+02	2.878	0.004008	**
odometer	-7.633e-02	7.301e-03	-10.454	< 2e-16	***
year	6.468e+02	1.381e+02	4.683	2.84e-06	***
---					

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

**b.** در این قسمت با استفاده از **caret package** مجموعه داده را به پنج قسمت تقسیم کردیم و با چهار قسمت آموزش دادیم و روی قسمت پنجم تست کردیم. **RMSE** میانگین این مقادیر به عنوان مقدار مدل نهایی به دست می‌آید. پس مدل ما به طور میانگین ۴۰۳۷۷.۴۲ خطای دارد. کد و نتیجه در زیر قابل مشاهده است.

### Linear Regression

38494 samples  
5 predictor

No pre-processing

Resampling: Cross-Validated (5 fold)

Summary of sample sizes: 30796, 30794, 30795, 30796, 30795

Resampling results:

RMSE	Rsquared	MAE
40377.42	0.2408789	5277.949

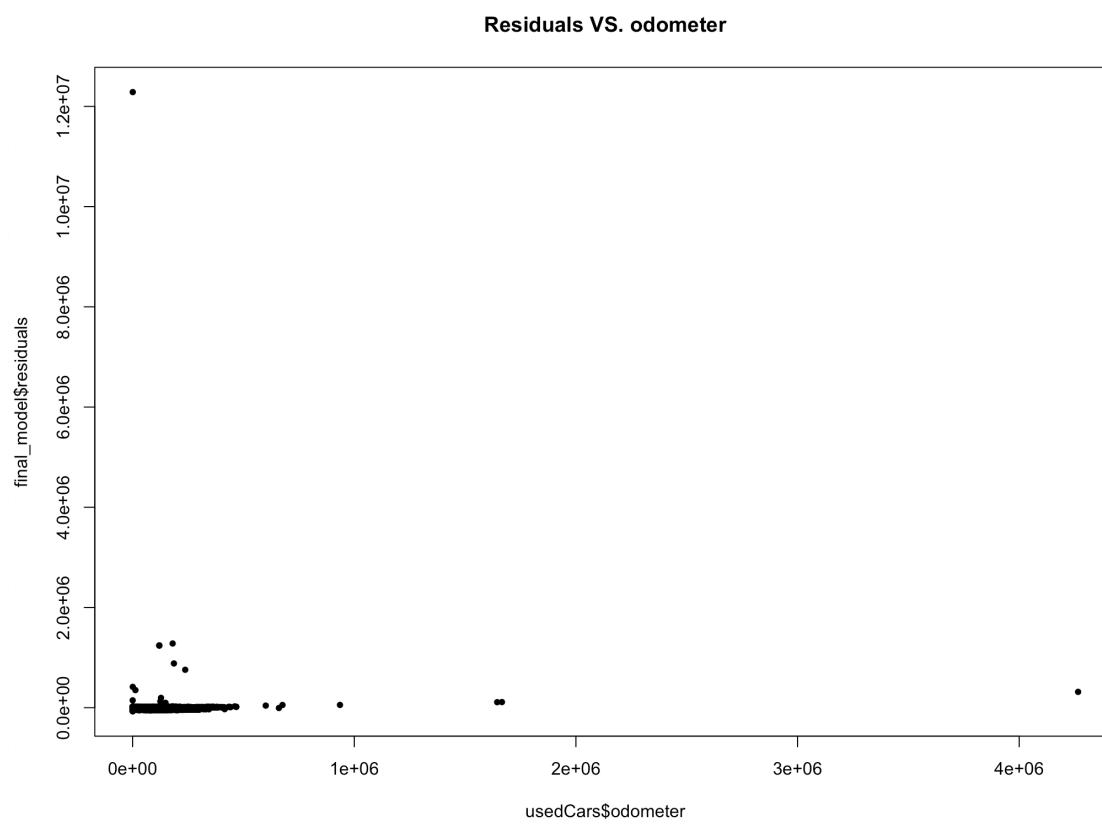
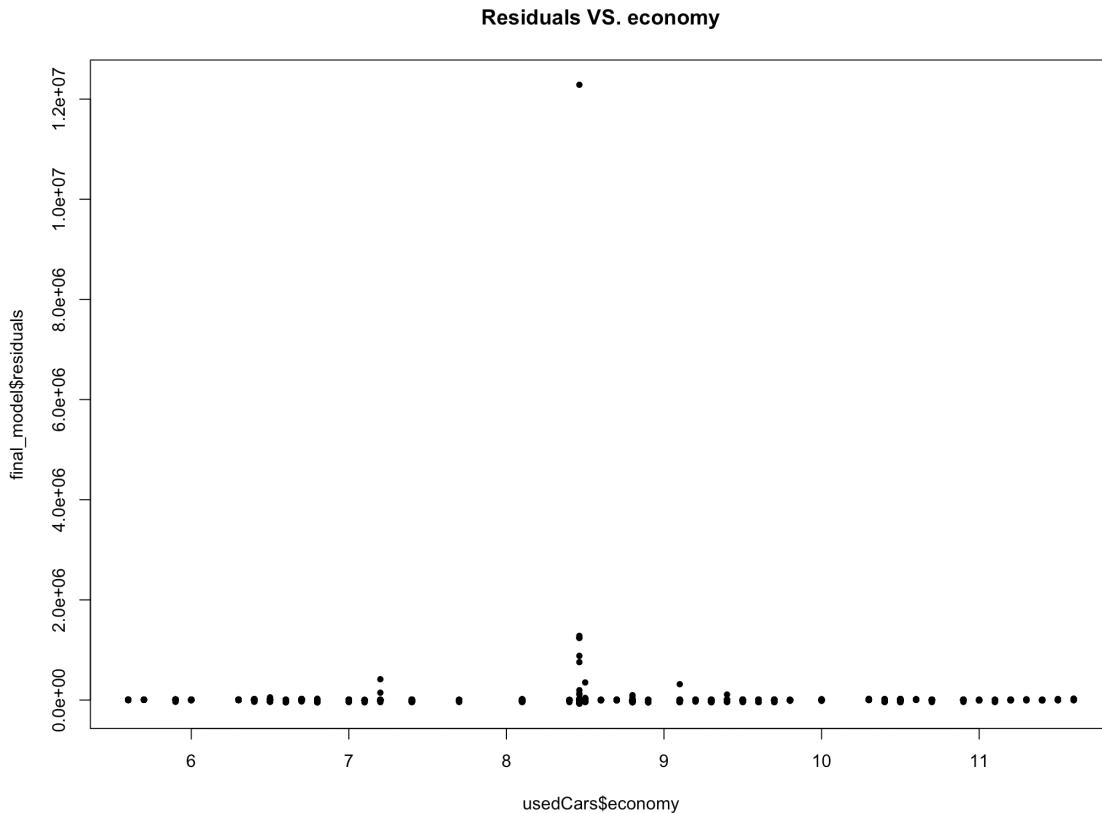
Tuning parameter 'intercept' was held constant at a value of TRUE

```

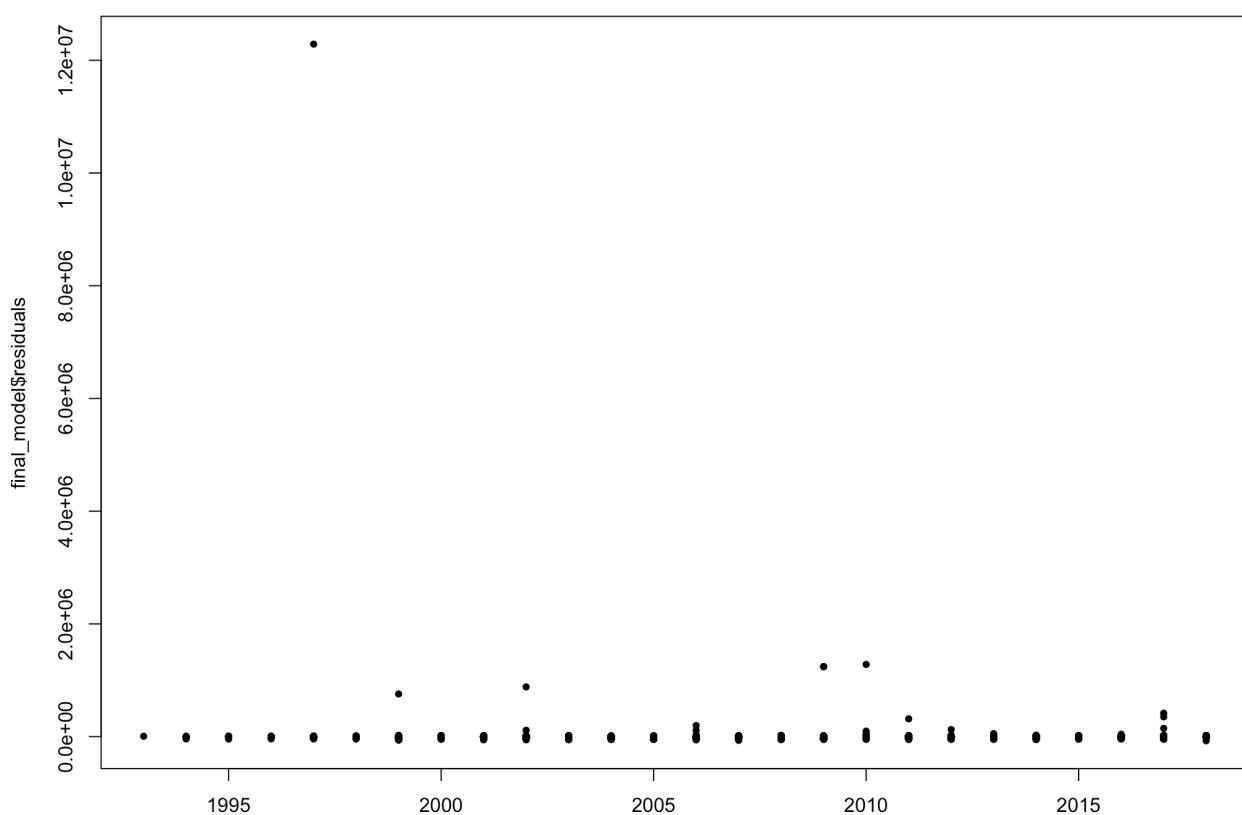
library(caret)
# Define training control
set.seed(123)
train.control <- trainControl(method = "cv", number = 5)
# Train the model
final_model <- train(price ~ category + colour + economy + odometer + year, data = usedCars, method = "lm",
                      trControl = train.control)
# Summarize the results
print(final_model)

```

C. شرط اول مربوط به رابطه خطی بین متغیرهای عددی و متغیر پاسخ است. با رسم نمودار باقیماندها بر حسب متغیر عددی می‌توان این موضوع را بررسی کرد. نمودارها و کد در زیر آمده است.



### Residuals VS. year



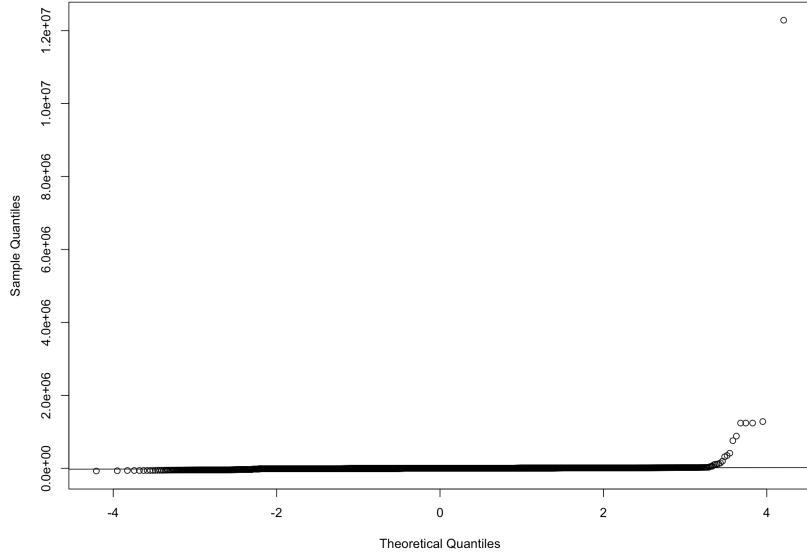
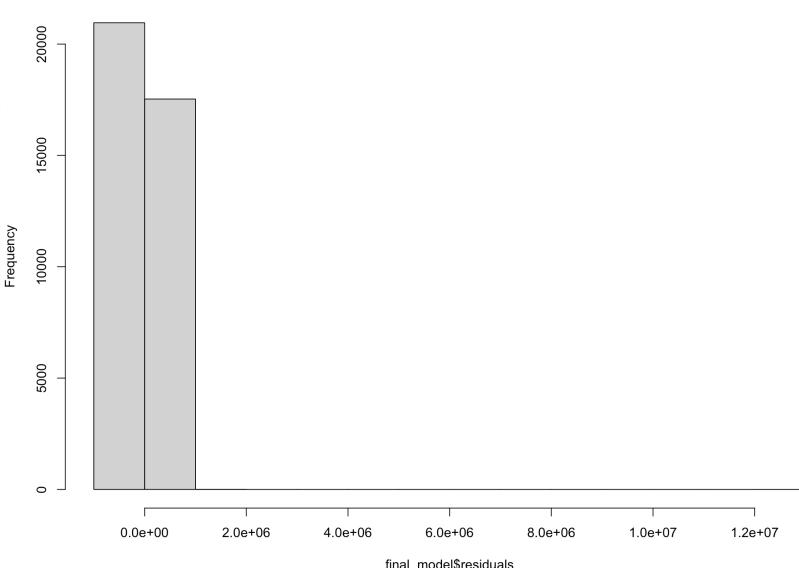
```
final_model <- ninth_step_model
plot(final_model$residuals ~ usedCars$economy, main= "Residuals VS. economy", pch=20)
plot(final_model$residuals ~ usedCars$odometer, main= "Residuals VS. odometer", pch = 20)
plot(final_model$residuals ~ usedCars$year, main= "Residuals VS. year", pch = 20)
```

همانطور که در نمودارهای بالا دیده می شود، به طور کلی حول صفر قرار گرفته اند و به صورت تصادفی پخش شده اند. البته مقادیر مثبت بیش از منفی هاست و این تعادل نرمال بودن را به هم می زند و اعتبار مدل را تا حدی زیر سوال می برد.

شرط دوم توزیع نرمال باقیمانده هاست. این شرط با رسم **qqplot** با توزیع نرمال و هیستوگرام بررسی می شود. نمودار و تکه کد در زیر قابل مشاهده است.

Histogram of final\_model\$residuals

Normal Q-Q Plot

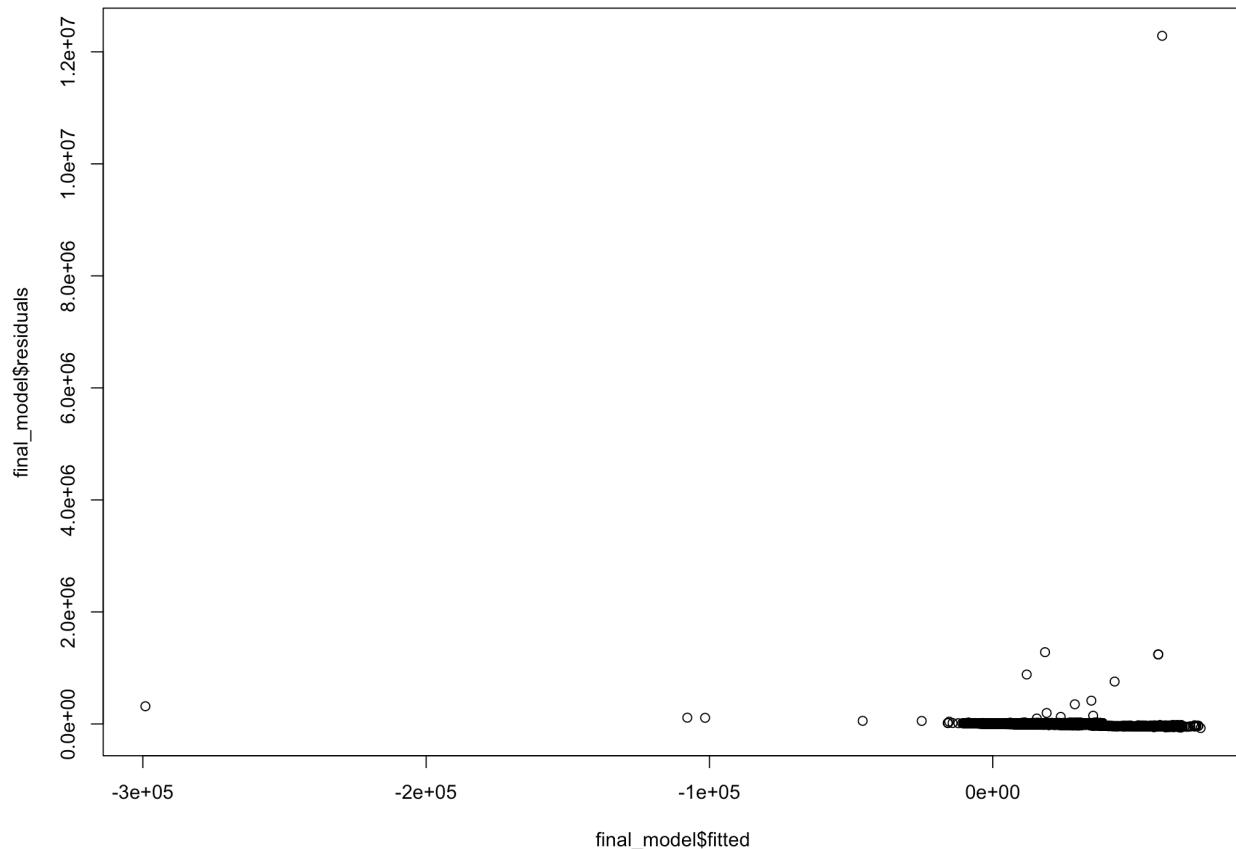


```
hist(final_model$residuals)
qqnorm(final_model$residuals)
qqline(final_model$residuals)
```

همانطور که در نمودارهای بالا دیده می شود، باقیمانده ها از توزیع نرمال پیروی نمی کنند. **Qqplot** به دلیل عرض باریک نمودار این گونه شده است اما همانطور که دیده می شود، دم آن خم شده است.

شرط سوم **variability** تقریباً یکسان باقیمانده هاست. برای بررسی این موضوع، باید نمودار باقیمانده ها بر حسب متغیر تخمین زده شده رسم شود. نمودار و تکه کد در زیر قابل مشاهده است.

```
plot(final_model$residuals ~ final_model$fitted)
```

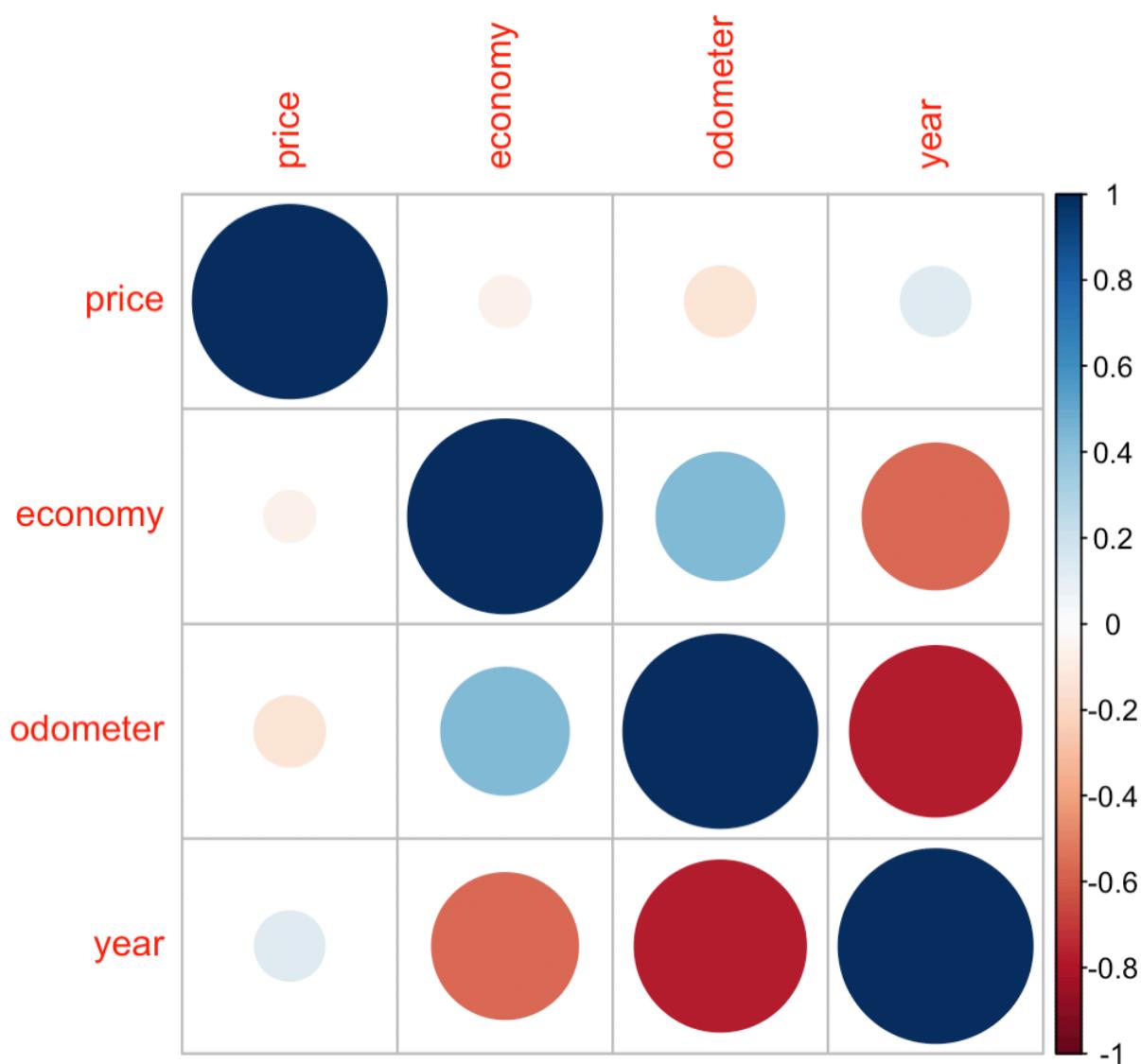


همانطور که در نمودار بالا دیده می شود، مقادیر مثبت خیلی زیاد به چشم می خورد. در صورتی که در منفی ها به این شکل نیست.

هر سه شرط تا حد خوبی برقرار نبودند. پس این مدل قابل اعتماد نیست چراکه شروط اولیه **linear regression** را ندارد.

**d**. کد و نمودار برای رسم **correlogram** به صورت زیر است.

```
library(corrplot)
model_mat <- usedCars[, c(1, 8, 15, 17)]
cor_mat <- cor(model_mat)
corrplot(cor_mat, method="circle")
```



همانطور که در نمودار بالا مشاهده می‌شود، **odometer** و **year** با یکدیگر رابطه معکوس و خطی زیادی دارند. دلیل آن هم به وضوح کارکرد بیشتر ماشین‌ها قدیمی‌تر است. سپس، **year** و **odometer** رابطه معکوس متعادل دارند. **Economy** و **odometer** رابطه مستقیم ملایمی با یکدیگر دارند.

با توجه به نمودار بالا، **odometer** و **year** تأثیر بیشتری در قیمت داشته‌اند. **E**. با دیدن خلاصه مدل، می‌توانیم میزان **R-squared** را ببینیم که دقیقاً بیانگر این است که چه میزان از **variability** متغیر پاسخ توسط مدل ما توجیه می‌شود. این مقدار برابر با ۰.۰۲۵۲۹ است. **F**.

Residual standard error: 64670 on 38455 degrees of freedom  
 Multiple R-squared: 0.02529, Adjusted R-squared: 0.02433  
 F-statistic: 26.26 on 38 and 38455 DF, p-value: < 2.2e-16

با توجه به بخش‌های قبل، این مدل دقیق ندارد. همچنین، شروط لازم برای **linear regression** را نیز دارا نیست. پس، این مدل مدل مناسبی نیست.

**سوال ۶.** متغیر **categorial** انتخابی متغیر **make** است. متغیرهای **make** انتخاب شده متغیر قیمت و تعداد سیلندرها هستند.  
**a.** کد ساخت مدل و نتیجه آن در ادامه آمده است.

```
make_glm <- glm(make ~ price + cylinders, data = usedCars)
print(summary(make_glm))
```

Coefficients:

	Estimate	Std. Error	z value	Pr(> z )
(Intercept)	-7.542e-01	2.178e-02	-34.636	< 2e-16 ***
price	-2.235e-06	8.673e-07	-2.577	0.00996 **
cylinders6	1.535e+01	5.464e+01	0.281	0.77870
---				
Signif. codes:	0 ‘***’	0.001 ‘**’	0.01 ‘*’	0.05 ‘.’
	0.1 ‘ ’	1		

(Dispersion parameter for binomial family taken to be 1)

```
Null deviance: 47913 on 38493 degrees of freedom
Residual deviance: 47295 on 38491 degrees of freedom
AIC: 47301
```

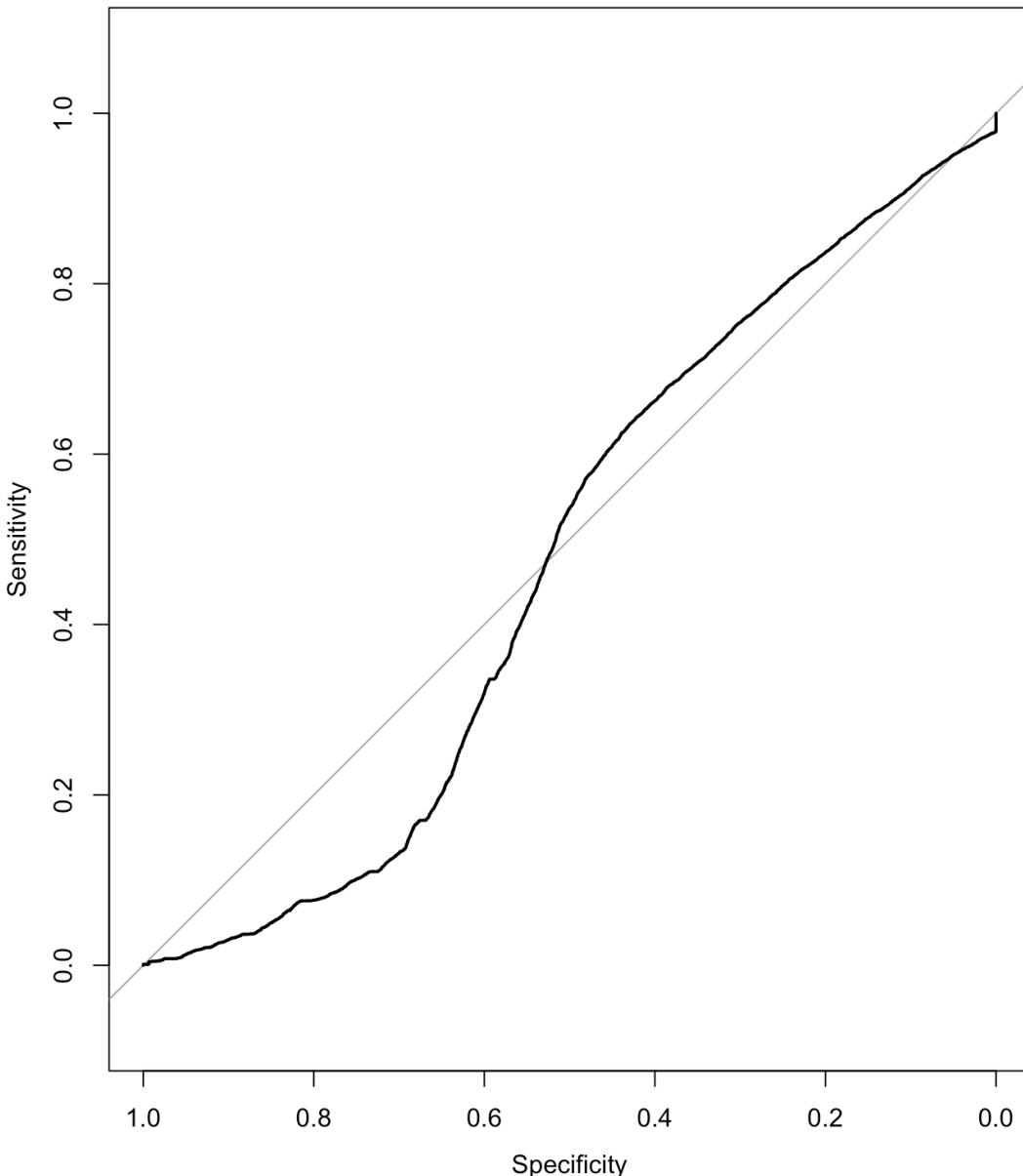
یک مدل ساخته شده است که در حقیقت مقدار لگاریتم شانس را به ما می دهد.  
**b.** عرض از مبدا: این مقدار برابر لگاریتم شانس تویاتو بودن ماشینی با قیمت صفر و ۴ سیلندر است.  
شیب قیمت: به ازای هر دلار افزایش قیمت ماشین، با فرض ثابت ماندن باقی متغیرها، مقدار لگاریتم شانس تویوتا بودن به میزان  $2.235 \times 10^{-4}$  کاهش می یابد.  
شیب تعداد سیلندرها: با فرض ثابت ماندن باقی متغیرها، مقدار لگاریتم شانس تویوتا بودن ماشین های شش سیلندر ۱۵.۳۵ بیشتر از ماشین های چهار سیلندری است.  
**c.** نمودار ROC و مقدار AUC با استفاده از pROC package محاسبه شد. تکه کد محاسبه، نمودار و مقدار AUC در زیر قابل مشاهده است.

```
prediction <- predict.glm(make_glm, usedCars)
library(pROC)
roc_obj <- roc(usedCars$make, prediction)
plot(roc_obj)
print(auc(roc_obj))
```

Setting levels: control = Subaru, case = Toyota

Setting direction: controls > cases

Area under the curve: 0.4738



این مدل **AUC** بسیار پایینی دارد و به همین علت، مدل اصلا خوبی نیست. مدل‌هایی با دقت بالای ۹۰ درصد عالی و بالی ۸۰ درصد قابل قبول محسوب می‌شوند. این مدل حتی از تشخیص تصادفی که **AUC** پنجاه درصد دارد، هم بدتر است. این سطح زیر نمودار در حقیقت بیانگر این است که تا چه حد می‌توانیم **True positive** داشته باشیم در عین اینکه **False positive** نداریم و بیانگر عملکرد خوب مدل است.

برای این منظور باید از جدول **coefficients** مدل استفاده کنیم و با استفاده از **d** **estimate**ها و استاندارد ارورها این بازه‌های اطمینان را محاسبه کنیم. تکه کد و مقادیر این بازه‌های اطمینان در زیر آمده است. تنها نکته این سوال این است که ابتدا بازه اطمینان لگاریتم به دست می‌آید و با **e** به توان بازه می‌توان بازه اطمینان شانس را به دست آورد.

```
[1] "Intercept:"
[1] "98% CI: ( 0.447153008180097 , 0.494829953213589 )"
[1] "Price slope:"
[1] "98% CI: ( 0.999995747308714 , 0.999999782418787 )"
[1] "Cylinder slope:"
[1] "98% CI: ( 2.91307050488764e-49 , 7.45698600143775e+61 )"
```

```

odd_ratio_98_ci <- function(point_estimate, standard_error) {
  z_star <- -qnorm(0.01)
  margin_of_error <- z_star * standard_error
  lower_bound <- exp(point_estimate - margin_of_error)
  upper_bound <- exp(point_estimate + margin_of_error)
  print(paste("98% CI: (", lower_bound, ", ", upper_bound, ")"))
}

coefs <- summary(make_glm)$coefficients
print("Intercept:")
odd_ratio_98_ci(coefs[["Intercept"]], coefs[["Intercept"]], "Std. Error"])
print("Price slope:")
odd_ratio_98_ci(coefs[["price"]], "Estimate"], coefs[["price"]], "Std. Error"])
print("Cylinder slope:")
odd_ratio_98_ci(coefs[["cylinders6"]], "Estimate"], coefs[["cylinders6"]], "Std. Error"])

```

**سوال ۷.** a. با توجه به مدل به دست آمده داریم.

	Estimate	Std. Error	z value	Pr(> z )	
(Intercept)	-7.542e-01	2.178e-02	-34.636	< 2e-16	***
price	-2.235e-06	8.673e-07	-2.577	0.00996	**
cylinders6	1.535e+01	5.464e+01	0.281	0.77870	
---					

همانطور که در تصویر بالا مشخص است، مقدار **p-value** قیمت خیلی پایین‌تر است. در نتیجه، از نظر آماری **significant** تر است. تعداد سیلندر تخمین‌گر خوبی نیست چراکه از نظر آماری **p-value** بسیار بالایی دارد.

**b.** متغیر انتخابی برای این بخش تعداد سیلندر هاست. کد مولد نمودار و نمودار **OR** در زیر قابل مشاهده است. برای این بهشت ابتدا میزان **odds\_ratio** این متغیر محاسبه شد. سپس، با استفاده از این مقدار و فرض مقادیر مختلف برای احتمال تویوتا بودن سازنده به شرط چهار سیلندره بودن مقدار دیگری به دست آمد.

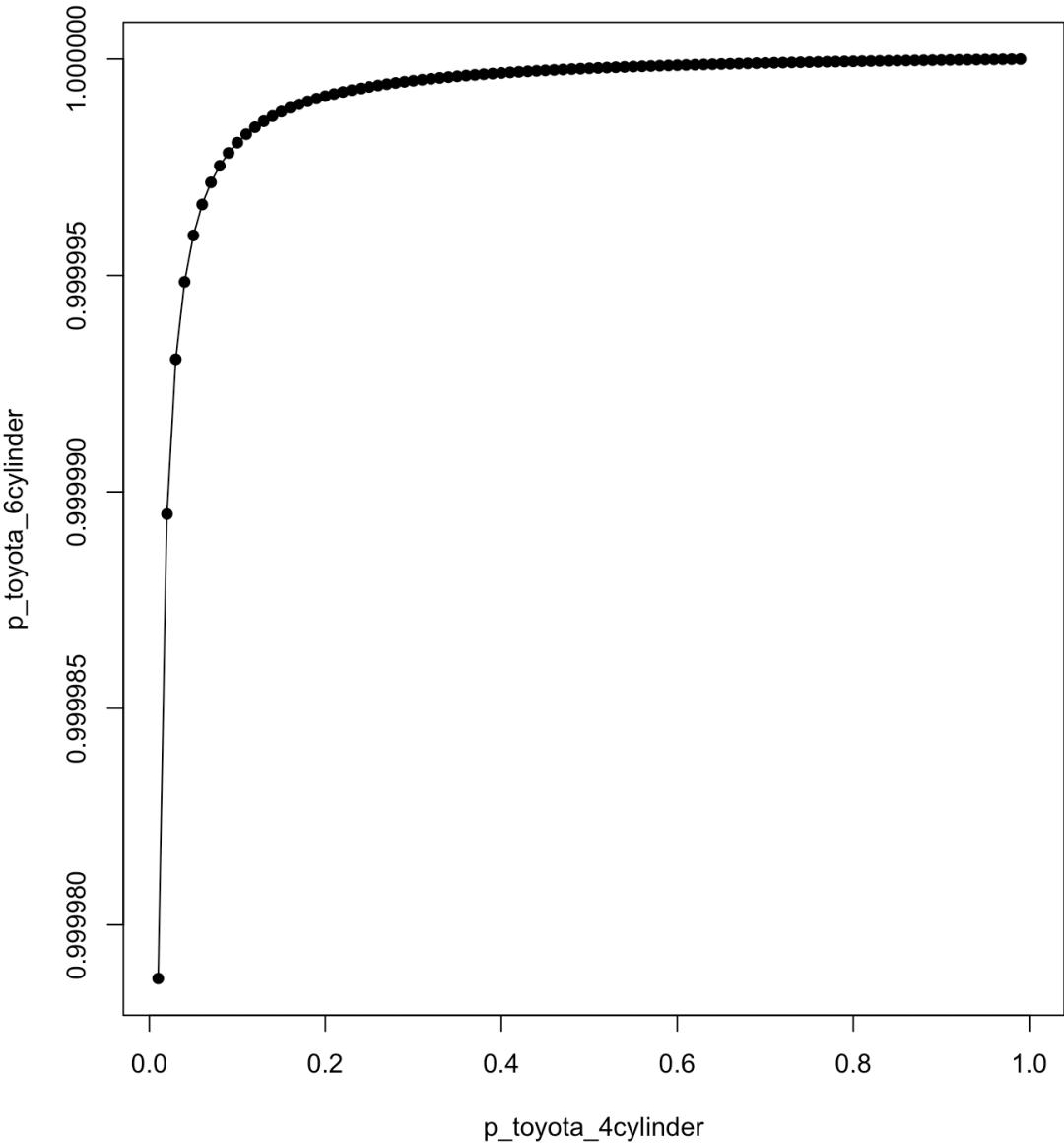
همانطور که مشاهده می‌کنید، احتمال تویوتا بودن به شرط شش سیلندره بودن خیلی بیشتر از احتمال تویوتا بودن به شرط چهار سیلندره بودن است. در حقیقت همانطور که نمودار نشان می‌دهد، احتمال اول از همان ابتدا در نزدیکی یک قرار دارد و رفتہ رفته بیشتر به یک میل می‌کند. این **RR** بسیار بالا به دلیل بالا بودن **odds ratio** می‌باشد.

```

ds_ratio <- exp(coefs[["cylinders6"]], "Estimate"])
toyota_4cylinder <- seq(0.01, 1, 0.01)
toyota_6cylinder <- vector()
r (p in p_toyota_4cylinder) {
x <- (p / (1-p)) * odds_ratio
new_prob <- x / (x+1)
p_toyota_6cylinder <- c(p_toyota_6cylinder, new_prob)

ot(p_toyota_4cylinder, p_toyota_6cylinder, pch=16)
nes(p_toyota_4cylinder[order(p_toyota_4cylinder)], p_toyota_6cylinder[order(p_toyota_4cylinder)], xlim=range(p_toyota_4cylinder)

```

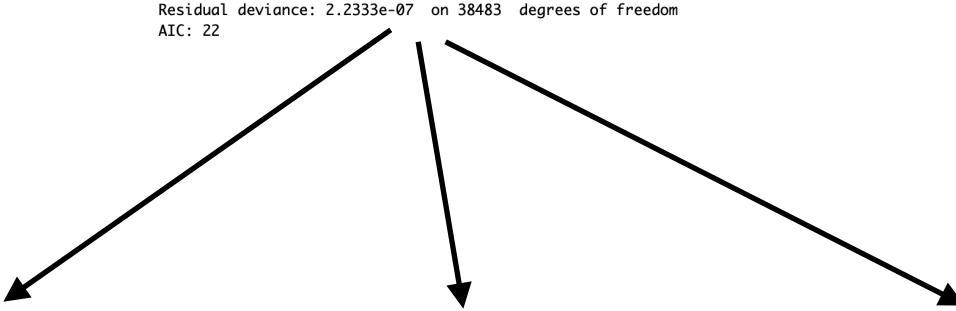


**C.** ابتدا از مدل کامل شروع می‌کنیم و رفته رفته متغیرهای را حذف می‌کنیم و هر کدام **AIC** کمتری داشت را برابر می‌داریم. **AIC** بیانگر خطای تخمین خارج از نمونه است. آنقدر این کار را تکرار می‌کنیم تا دیگر **AIC** کاهش پیدا نکند. مدل نهایی شامل متغیر مدل است. در زیر کد و خروجی مدل نهایی را می‌توان دید.

Coefficients:					
	Estimate	Std. Error	z value	Pr(> z )	
(Intercept)	-2.657e+01	2.519e+05	0.000	1.00	
price	2.104e-19	2.774e-02	0.000	1.00	
body_typeConvertible	4.577e-06	3.561e+05	0.000	1.00	
body_typeCoup	4.588e-06	2.815e+05	0.000	1.00	
body_typeHatch	4.560e-06	4.364e+05	0.000	1.00	
body_typeSedan	4.560e-06	4.364e+05	0.000	1.00	
body_typeSuv	4.560e-06	2.521e+05	0.000	1.00	
body_typeSUV	4.560e-06	2.518e+05	0.000	1.00	
body_typeWagon	4.560e-06	2.522e+05	0.000	1.00	
modelImpreza	-1.478e-11	3.564e+05	0.000	1.00	
modelRAV4	5.313e+01	4.417e+03	0.012	0.99	

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 4.7913e+04 on 38493 degrees of freedom  
 Residual deviance: 2.2333e-07 on 38483 degrees of freedom  
 AIC: 22



```

Coefficients:
Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.657e+01 3.078e+03 -0.009 0.993
price -0.936e-19 2.774e-02 0.000 1.000
modelImpreza -4.355e-13 4.395e+03 0.000 1.000
modelRAV4 5.313e+01 4.416e+03 0.012 0.990

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 4.7913e+04 on 38493 degrees of freedom
Residual deviance: 2.2333e-07 on 38490 degrees of freedom
AIC: 8

```

	Estimate	Std. Error	z value	Pr(> z )
(Intercept)	1.962e+01	7.604e+03	0.003	0.998
price	-2.195e-05	1.093e-06	-20.082	<2e-16 ***
body_typeConvertible	3.959e-02	1.075e+04	0.000	1.000
body_typeCoup	-1.180e-02	8.502e+03	0.000	1.000
body_typeHatch	-3.879e+01	7.605e+03	-0.005	0.996
body_typeSedan	-3.878e+01	7.606e+03	-0.005	0.996
body_typeSuv	-1.938e+01	7.604e+03	-0.003	0.998
body_typeSUV	-1.926e+01	7.604e+03	-0.003	0.998
body_typeWagon	-1.956e+01	7.604e+03	-0.003	0.998
---				
Signif. codes:	0 '***'	0.001 '**'	0.01 '*'	0.05 '.'
	0.1 ' '	1		

```

Estimate Std. Error z value Pr(>|z|)
(Intercept) 1.962e+01 7.604e+03 0.003 0.998
price -2.195e-05 1.093e-06 -20.082 <2e-16 ***
body_typeConvertible 3.959e-02 1.075e+04 0.000 1.000
body_typeCoup -1.180e-02 8.502e+03 0.000 1.000
body_typeHatch -3.879e+01 7.605e+03 -0.005 0.996
body_typeSedan -3.878e+01 7.606e+03 -0.005 0.996
body_typeSuv -1.938e+01 7.604e+03 -0.003 0.998
body_typeSUV -1.926e+01 7.604e+03 -0.003 0.998
body_typeWagon -1.956e+01 7.604e+03 -0.003 0.998
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' '

```

```

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 4.7913e+04 on 38493 degrees of freedom
Residual deviance: 2.2333e-07 on 38484 degrees of freedom
AIC: 20

```

	Estimate	Std. Error	z value	Pr(> z )
(Intercept)	-2.657e+01	2.519e+05	0.000	
body_typeConvertible	4.498e-06	3.561e+05	0.000	
body_typeCoup	4.516e-06	2.815e+05	0.000	
body_typeHatch	-3.626e-08	4.364e+05	0.000	
body_typeSedan	-3.621e-08	4.364e+05	0.000	
body_typeSuv	-3.639e-08	2.521e+05	0.000	
body_typeSUV	-3.620e-08	2.518e+05	0.000	
body_typeWagon	-3.631e-08	2.522e+05	0.000	
modelImpreza	-3.043e-12	3.564e+05	0.000	
modelRAV4	5.313e+01	4.416e+03	0.012	0.990

Coefficients:

	Estimate	Std. Error	z value	Pr(> z )
(Intercept)	-7.134e-01	2.262e-02	-31.542	< 2e-16 ***
price	-3.120e-06	9.186e-07	-3.396	0.000683 ***
---				
Signif. codes:	0 '***'	0.001 '**'	0.01 '*'	0.05 '.'
	0.1 ' '	1		

(Dispersion parameter for binomial family taken to be 1)

Null deviance: 47913 on 38493 degrees of freedom
Residual deviance: 47898 on 38492 degrees of freedom
AIC: 47902

```

make_glm <- glm(make ~ price + body_type + model, family = binomial, data = usedCars)
print(summary(make_glm))
make_glm <- glm(make ~ body_type + model, family = binomial, data = usedCars)
print(summary(make_glm))
make_glm <- glm(make ~ price + model, family = binomial, data = usedCars)
print(summary(make_glm))
make_glm <- glm(make ~ price + body_type, family = binomial, data = usedCars)
print(summary(make_glm))
make_glm <- glm(make ~ price, family = binomial, data = usedCars)
print(summary(make_glm))
make_glm <- glm(make ~ model, family = binomial, data = usedCars)
print(summary(make_glm))

```

همانطور که دیدید، مدل به تنها یک بهترین تخمینگر برای متغیر سازنده است و این کاملاً نمایان است که از مدل یک ماشین می‌توان به سازنده آن پی برد. به همین دلیل، این مدل کمتر AIC یا همان خطرا دارد.

d. برای این کار بیشینه مجموع specificity و sensitivity را بر حسب مقادیر مختلف threshold به دست می‌آوریم و به عنوان مرز بر می‌داریم. کد در زیر قابل مشاهده است.

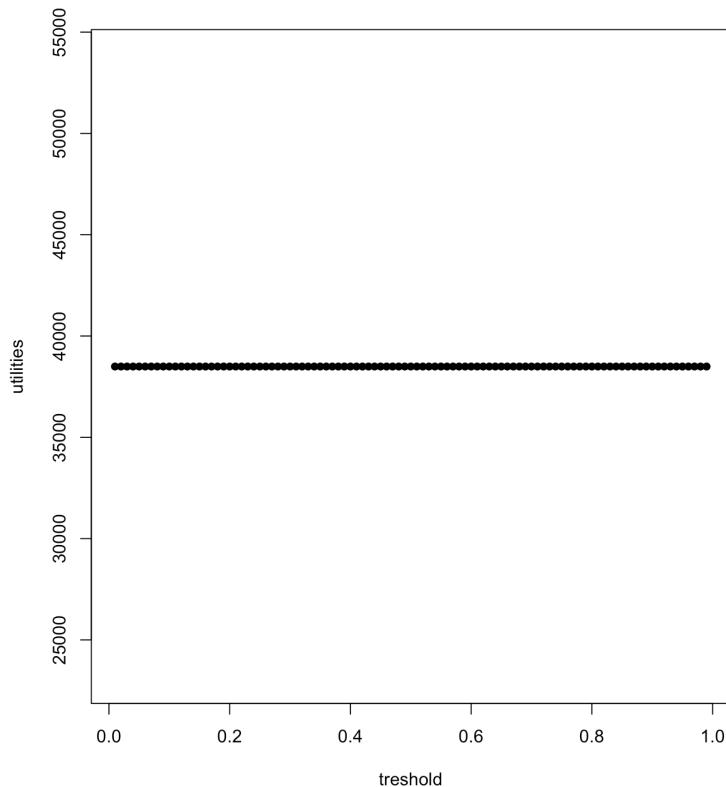
```

threshold <- seq(0.01, 0.99, 0.01)
actual <- usedCars$make
prediction <- predict.glm(make_glm, usedCars)
predicted <- 1/ (1 + exp(-prediction))
max_tresh <- 0
max_score <- 0
for (tresh in threshold) {
  class_prediction <-
    ifelse(predicted > tresh,
      "Toyota",
      "Subaru")
  class_prediction <- as.factor(class_prediction)
  conf_mat <- confusionMatrix(class_prediction, actual)
  score <- conf_mat$byClass[["Sensitivity"]] + 1 - conf_mat$byClass[["Specificity"]]
  if (score > max_score) {
    max_score <- score
    max_tresh <- tresh
  }
}
print(paste("Best threshold:", max_tresh))

```

مقدار بهینه مرز ۱۰۰ به دست آمد و دلیل آن این است که از روی مدل سازنده کاملاً مشخص می‌شود و در نتیجه، مرزی از یک حد بزرگتر همه را کاملاً درست تشخیص می‌دهد. البته این کد به صورت عام کار می‌کند.

e. این بخش نیز کاملاً مانند بخش قبل است، با این تفاوت که امتیاز در اینجا با استفاده تابع utility مشخص می‌شود که بستگی به کاربرد می‌توان به هریک از چهار عامل وزن خاصی دارد. در اینجا به عنوان مثال، به **false negative** و **true positive** وزن ۱، به **false positive** وزن ۵- و به **true negative** وزن منفی ۱۰ داده شده است. باز هم مرز بهینه ۱۰ درآمد. تکه کد و نمودار در زیر قابل مشاهده است.



```

threshold <- seq(0.01, 0.99, 0.01)
actual <- usedCars$make
prediction <- predict.glm(make_glm, usedCars)
predicted <- 1/ (1 + exp(-prediction))
max_tresh <- 0
max_score <- 0
utilities <- vector()
for (tresh in threshold) {
  class_prediction <-
    ifelse(predicted > tresh,
      "Toyota",
      "Subaru")
  class_prediction <- as.factor(class_prediction)
  conf_mat <- confusionMatrix(class_prediction, actual)
  tp <- conf_mat$table["Toyota", "Toyota"]
  tn <- conf_mat$table["Subaru", "Subaru"]
  fp <- conf_mat$table["Toyota", "Subaru"]
  fn <- conf_mat$table["Subaru", "Toyota"]
  utility <- tp + tn - 5 * fp - 10 * fn
  utilities <- c(utilities, utility)
  if (utility > max_score) {
    max_score <- utility
    max_tresh <- tresh
  }
}
print(paste("Best threshold:", max_tresh))
plot(threshold, utilities, pch=16)
lines(threshold[order(threshold)], utilities[order(threshold)], xlim=range(threshold), ylim=range(utilities), pch=16)

```

مرز دو قسمت یکسان به دست آمدند. البته دلیل این موضوع، این است که از روی مدل سازنده کاملاً مشخص می‌شود و در نتیجه، مرزی از یک حد بزرگتر همه را کاملاً درست تشخیص می‌دهد. البته این کد به صورت عام کار می‌کند و در حالت کلی، با توجه به وزن دادن به چهار معیار مرز می‌تواند جابجا شود.

**سوال ۸.** متغیر پاسخ مصرف سوخت(explanatory economy) است. متغیرهای transmission fuel به نظر تعداد سیلندرها، نوع سوخت(fuel)، حجم موتور، میزان کارکرد، نوع و سال ساخت هستند. برای انتخاب مدل بهینه از backward elimination استفاده می‌کنیم. معیار برای این کار نیز p-value می‌باشد. مراحل انجام کار به همراه نتیجه و کد در زیر آمده است.

#### Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	2.186e+02	2.537e+00	86.144	<2e-16 ***
cylinders6	-1.174e+00	4.645e-02	-25.264	<2e-16 ***
fuelUnleaded	1.479e+00	1.463e-02	101.112	<2e-16 ***
litres	2.418e+00	1.550e-02	156.042	<2e-16 ***
odometer	5.575e-08	7.245e-08	0.769	0.442
transmissionManual	3.897e-01	8.899e-03	43.797	<2e-16 ***
year	-1.079e-01	1.257e-03	-85.864	<2e-16 ***
<hr/>				
Signif. codes:	0 ‘***’	0.001 ‘**’	0.01 ‘*’	0.05 ‘.’
	0.1 ‘ ’	1		

مدل کامل بررسی شد. تنها متغیری که significant نیست، آن odometer است. پس، آن را حذف می‌کنیم.

#### Coefficients:

	Estimate	Std. Error	t value	Pr(> t )
(Intercept)	2.200e+02	1.718e+00	128.04	<2e-16 ***
cylinders6	-1.174e+00	4.645e-02	-25.27	<2e-16 ***
fuelUnleaded	1.478e+00	1.455e-02	101.56	<2e-16 ***
litres	2.418e+00	1.548e-02	156.19	<2e-16 ***
transmissionManual	3.898e-01	8.898e-03	43.81	<2e-16 ***
year	-1.086e-01	8.515e-04	-127.59	<2e-16 ***

حال تمام متغیرها significant هستند و مدل نهایی است.

```
usedCars <- usedCars[usedCars$fuel != "",]
usedCars$fuel <- as.factor(usedCars$fuel)
droplevels(usedCars$fuel)
usedCars$transmission <- as.factor(usedCars$transmission)
first_step_lm <- lm(economy ~ cylinders + fuel + litres + odometer + transmission + year, data = usedCars)
print(summary(first_step_lm))
second_step_lm <- lm(economy ~ cylinders + fuel + litres + transmission + year, data = usedCars)
print(summary(second_step_lm))
```

بیشترین تاثیر در مصرف سوخت از نظر مقداری مربوط به بیشترین شبیه می‌شود. بنابراین، بیشترین تاثیر را حجم موتور یا همان litres دارد. البته متغیرها به یکدیگر نزدیک هستند.