In [4]:
```
library("readxl")
read_excel("datahouse.xlsx")
```

A tibble: 21 × 10

Count	price	bedrooms	bathrooms	sqft_living	H Size	Kitchen Size	Bedroom Size	Bathroom Size	Yeai Built
<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
1	3500000	3	2	1180	250	98	140	50	2009
2	7710000	3	3	2570	270	140	140	50	2009
3	2500000	2	2	770	280	60	140	25	2005
4	4500000	4	2	1960	240	100	130	30	2005
5	4000000	3	2	1680	260	100	140	30	2003
6	10000000	4	5	5420	240	240	240	70	2002
7	3200000	3	2	1715	230	98	120	40	2006
8	2800000	3	2	1060	210	99	120	40	2005
9	3200000	3	2	1780	190	120	150	30	2006
10	3500000	3	1	1890	200	130	130	40	2004
11	7000000	3	2	3560	350	200	200	50	2004
12	2900000	2	1	1160	150	80	100	70	1998
13	3500000	3	3	1430	180	90	100	40	2005
14	3300000	3	3	1370	180	70	110	40	2009
15	3800000	3	3	1810	170	85	140	60	2005
16	8500000	4	4	2950	250	160	200	50	2004
17	3500000	2	2	1890	200	140	180	40	1920
18	3000000	2	2	1600	240	140	180	20	1974
19	3000000	2	2	1200	210	120	170	50	1993
20	3200000	3	3	1250	190	120	140	40	1992
21	14000000	4	4	5420	400	240	250	80	2001

In [5]:
```
house <- read_excel("datahouse.xlsx")
```

In [6]:
```
df <- data.frame(house)
```

In [7]:
```
print (df)
```

```
   Count    price bedrooms bathrooms sqft_living H.Size Kitchen.Size
1      1  3500000        3         2        1180    250           98
2      2  7710000        3         3        2570    270          140
3      3  2500000        2         2         770    280           60
4      4  4500000        4         2        1960    240          100
5      5  4000000        3         2        1680    260          100
6      6 10000000        4         5        5420    240          240
7      7  3200000        3         2        1715    230           98
8      8  2800000        3         2        1060    210           99
9      9  3200000        3         2        1780    190          120
10    10  3500000        3         1        1890    200          130
11    11  7000000        3         2        3560    350          200
12    12  2900000        2         1        1160    150           80
13    13  3500000        3         3        1430    180           90
14    14  3300000        3         3        1370    180           70
15    15  3800000        3         3        1810    170           85
16    16  8500000        4         4        2950    250          160
17    17  3500000        2         2        1890    200          140
18    18  3000000        2         2        1600    240          140
19    19  3000000        2         2        1200    210          120
20    20  3200000        3         3        1250    190          120
21    21 14000000        4         4        5420    400          240
   Bedroom.Size Bathroom.Size Year.Built
1           140            50       2009
2           140            50       2009
3           140            25       2005
4           130            30       2005
5           140            30       2003
6           240            70       2002
7           120            40       2006
8           120            40       2005
9           150            30       2006
10          130            40       2004
11          200            50       2004
12          100            70       1998
13          100            40       2005
14          110            40       2009
15          140            60       2005
16          200            50       2004
17          180            40       1920
18          180            20       1974
19          170            50       1993
20          140            40       1992
21          250            80       2001
```

In [8]:
```
data(df)
```

```
Warning message in data(df):
“data set ‘df’ not found”
```

In [9]:
```
data (df)
```

```
Warning message in data(df):
“data set ‘df’ not found”
```

In [10]:
```
data(house)
```

```
Warning message in data(house):
“data set ‘house’ not found”
```

In [11]:
```
data(df)
head(df)
```

```
Warning message in data(df):
“data set ‘df’ not found”
```

A data.frame: 6 × 10

	Count	price	bedrooms	bathrooms	sqft_living	H.Size	Kitchen.Size	Bedroom.Size	Bat
	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	
1	1	3500000	3	2	1180	250	98	140	
2	2	7710000	3	3	2570	270	140	140	
3	3	2500000	2	2	770	280	60	140	
4	4	4500000	4	2	1960	240	100	130	
5	5	4000000	3	2	1680	260	100	140	
6	6	10000000	4	5	5420	240	240	240	

In [12]:
```
install.packages("caret")
```

```
Installing package into ‘/srv/rlibs’
(as ‘lib’ is unspecified)

also installing the dependencies ‘numDeriv’, ‘SQUAREM’, ‘lava’, ‘prodlim’, ‘i
terators’, ‘data.table’, ‘gower’, ‘ipred’, ‘RcppRoll’, ‘timeDate’, ‘foreach’,
‘ModelMetrics’, ‘recipes’
```

In [14]:
```
library(caret)
# Simple linear regression model (lm means linear model)
# model <- train(mpg ~ wt,
#                data = mtcars,
#                method = "lm")

# Multiple linear regression model
model <- train(price ~ .,
               data = df,
               method = "lm")

# Ridge regression model
# model <- train(mpg ~ .,
#                data = mtcars,
#                method = "ridge") # Try using "lasso"
```

In [15]:
```
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,     # number of folds
                           repeats = 10)    # repeated ten times

model.cv <- train(price ~ .,
               data = df,
               method = "lasso",  # now we're using the lasso method
               trControl = fitControl)

model.cv
```

```
1 package is needed for this model and is not installed. (elasticnet). Would
you like to try to install it now?

Error: Required package is missing
Traceback:

1. train(price ~ ., data = df, method = "lasso", trControl = fitControl)
2. train.formula(price ~ ., data = df, method = "lasso", trControl = fitContr
ol)
3. train(x, y, weights = w, ...)
4. train.default(x, y, weights = w, ...)
5. checkInstall(models$library)
6. stop("Required package is missing", call. = FALSE)
```

In [16]:
```
library(caret)
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,     # number of folds
                           repeats = 10)    # repeated ten times

model.cv <- train(mpg ~ .,
                  data = mtcars,
                  method = "lasso",  # now we're using the lasso method
                  trControl = fitControl)

model.cv
```

```
1 package is needed for this model and is not installed. (elasticnet). Would
you like to try to install it now?

Error: Required package is missing
Traceback:

1. train(mpg ~ ., data = mtcars, method = "lasso", trControl = fitControl)
2. train.formula(mpg ~ ., data = mtcars, method = "lasso", trControl = fitCon
trol)
3. train(x, y, weights = w, ...)
4. train.default(x, y, weights = w, ...)
5. checkInstall(models$library)
6. stop("Required package is missing", call. = FALSE)
```

In [17]:
```
install.packages(10-fold CV)
```

```
Error in parse(text = x, srcfile = src): <text>:1:26: unexpected symbol
1: install.packages(10-fold CV
                             ^

Traceback:
```

In [18]:
```
install.packages("10-fold CV")
```

```
Installing package into ‘/srv/rlibs’
(as ‘lib’ is unspecified)

Warning message:
“package ‘10-fold CV’ is not available (for R version 3.6.3)”
```

In [29]:
```
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,     # number of folds
                           repeats = 10)    # repeated ten times

model.cv <- train(price ~ .,
                  data = df,
                  method = "lasso",  # now we're using the lasso method
                  trControl = fitControl)

model.cv
```

```
Warning message in nominalTrainWorkflow(x = x, y = y, wts = weights, info = t
rainInfo, :
“There were missing values in resampled performance measures.”

The lasso

21 samples
 9 predictor

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 10 times)
Summary of sample sizes: 19, 19, 19, 19, 19, 19, ...
Resampling results across tuning parameters:

  fraction  RMSE       Rsquared   MAE
  0.1       2192432.9  0.9808074  2000804.6
  0.5        980962.6  0.9866951   842648.5
  0.9       1115943.9  0.9574710   967956.3

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was fraction = 0.5.
```

In [20]:
```
install.packages("elasticnet")
```

```
Installing package into ‘/srv/rlibs’
(as ‘lib’ is unspecified)

also installing the dependency ‘lars’
```

In [23]:
```
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,     # number of folds
                           repeats = 10)    # repeated ten times

model.cv <- train(price ~ .,
                  data = df,
                  method = "lasso",  # now we're using the lasso method
                  trControl = fitControl)

model.cv
```

```
Warning message in nominalTrainWorkflow(x = x, y = y, wts = weights, info = t
rainInfo, :
“There were missing values in resampled performance measures.”

The lasso

21 samples
 9 predictor

No pre-processing
Resampling: Cross-Validated (10 fold, repeated 10 times)
Summary of sample sizes: 19, 19, 19, 19, 19, 18, ...
Resampling results across tuning parameters:

  fraction  RMSE       Rsquared   MAE
  0.1       2188894.6  0.9659296  1947745.9
  0.5        983546.3  0.9754963   826936.8
  0.9       1134449.7  0.9836662   962393.9

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was fraction = 0.5.
```

In [30]:
```
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,     # number of folds
                           repeats = 10)    # repeated ten times

lambdaGrid <- expand.grid(lambda = 10^seq(10, -2, length=100))

model.cv <- train(price ~ .,
                  data = df,
                  method = "lasso",  # now we're using the lasso method
                  trControl = fitControl,
                  preProcess = c('scale', 'center'),
                  tuneGrid = lambdaGrid,   # Test all the lambda values in the la
mbdaGrid dataframe
                  na.action = na.omit)
model.cv
```

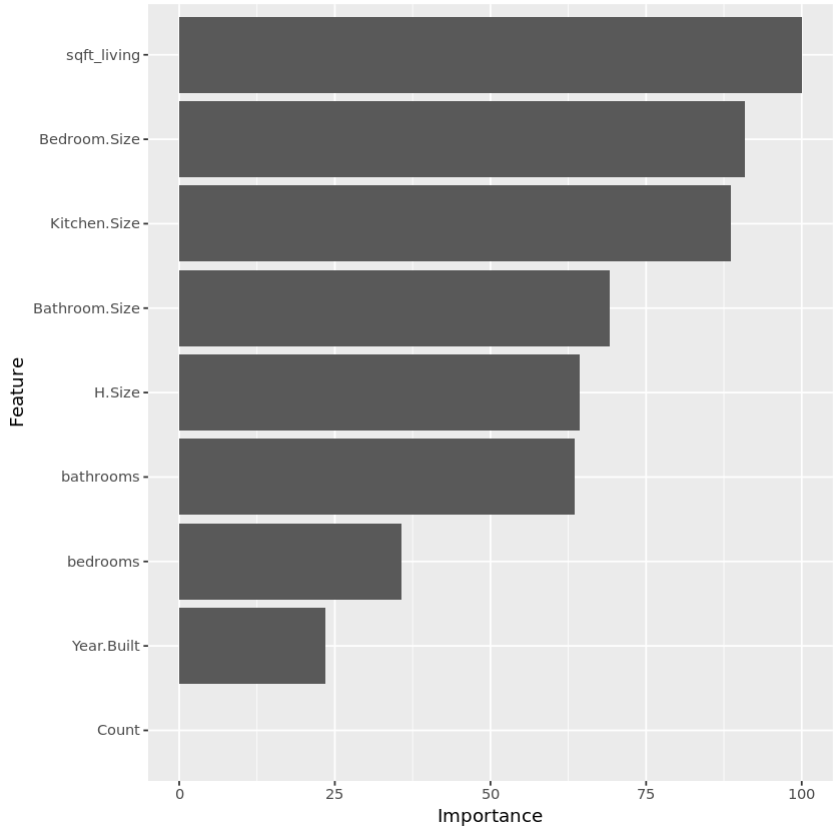
```
Error: The tuning parameter grid should not have columns fraction
Traceback:

1. train(price ~ ., data = df, method = "lasso", trControl = fitControl,
 .     preProcess = c("scale", "center"), tuneGrid = lambdaGrid,
 .     na.action = na.omit)
2. train.formula(price ~ ., data = df, method = "lasso", trControl = fitContr
ol,
 .     preProcess = c("scale", "center"), tuneGrid = lambdaGrid,
 .     na.action = na.omit)
3. train(x, y, weights = w, ...)
4. train.default(x, y, weights = w, ...)
5. stop(paste("The tuning parameter grid should not have columns",
 .     paste(tuneNames, collapse = ", ", sep = "")), call. = FALSE)
```

In [31]:
```
fitControl <- trainControl(## 10-fold CV
                           method = "repeatedcv",
                           number = 10,
                           repeats = 10,
                           search = "random")  # hyper-parameters random searc
h

model.cv <- train(price ~ .,
                  data = df,
                  method = "ridge",
                  trControl = fitControl,
                  preProcess = c('scale', 'center'),
                  na.action = na.omit)

model.cv
```

```
Warning message in nominalTrainWorkflow(x = x, y = y, wts = weights, info = t
rainInfo, :
“There were missing values in resampled performance measures.”

Ridge Regression

21 samples
 9 predictor

Pre-processing: scaled (9), centered (9)
Resampling: Cross-Validated (10 fold, repeated 10 times)
Summary of sample sizes: 19, 19, 19, 19, 18, 19, ...
Resampling results across tuning parameters:

  lambda        RMSE     Rsquared   MAE
  0.0001426452  1183861  0.9688459  1017475.7
  0.0053697883  1151478  0.9695925   987319.6
  0.4047704591  1184047  0.9661215  1016513.7

RMSE was used to select the optimal model using the smallest value.
The final value used for the model was lambda = 0.005369788.
```

In [32]:
```
ggplot(varImp(model.cv))
```

```
Warning message in simpleLoess(y, x, w, span, degree = degree, parametric = p
arametric, :
“pseudoinverse used at 0.98”
Warning message in simpleLoess(y, x, w, span, degree = degree, parametric = p
arametric, :
“neighborhood radius 2.02”
Warning message in simpleLoess(y, x, w, span, degree = degree, parametric = p
arametric, :
“reciprocal condition number  9.8062e-17”
Warning message in simpleLoess(y, x, w, span, degree = degree, parametric = p
arametric, :
“There are other near singularities as well. 1”
Warning message in simpleLoess(y, x, w, span, degree = degree, parametric = p
arametric, :
“pseudoinverse used at 40”
Warning message in simpleLoess(y, x, w, span, degree = degree, parametric = p
arametric, :
“neighborhood radius 10”
Warning message in simpleLoess(y, x, w, span, degree = degree, parametric = p
arametric, :
“reciprocal condition number  0”
```



In [33]:
```
predictions <- predict(model.cv, df)

predictions
```

1: 3994136.09449348 **2:** 6346007.76702737 **3:** 2423526.01604881 **4:** 4445316.44148342 **5:** 3901465.34198125 **6:** 10633012.7955437 **7:** 3970728.98776494 **8:** 3078498.47861112 **9:** 2756294.20971776 **10:** 3104836.59859535 **11:** 8074862.5458977 **12:** 2305749.87071059 **13:** 3663362.83595077 **14:** 3432263.21182053 **15:** 4334705.76022601 **16:** 7459386.1539298 **17:** 3507229.94309307 **18:** 2760844.94613402 **19:** 2996528.92989805 **20:** 3751077.96175402 **21:** 13670165.1093182

In [ ]:
```

```