Question 1

Question1. (A)

We first get an insight into the data by looking at our base model including all the features, looking at the corelation matrix, OOS R^2, and identifying which features are significant, highly corelated, etc.

```r
library(tidyverse)
library(dplyr)

train <- read.csv("laptop_train.csv")
test  <- read.csv("laptop_test.csv")

train$Company  <- factor(train$Company)
train$TypeName <- factor(train$TypeName)
train$GPU      <- factor(train$GPU)

# These should stay numeric
num_vars <- c("Screen","Memory","Weight","Rating","Price")
train[num_vars] <- lapply(train[num_vars], as.numeric)

# correlation matrix
cor(train[, num_vars], use = "complete.obs")
```

```
##             Screen     Memory      Weight      Rating      Price
## Screen  1.00000000 0.09940796  0.81150314 -0.03097351 -0.1306660
## Memory  0.09940796 1.00000000  0.29239722  0.02431015  0.7224164
## Weight  0.81150314 0.29239722  1.00000000 -0.01924711  0.1167789
## Rating -0.03097351 0.02431015 -0.01924711  1.00000000 -0.0290045
## Price  -0.13066597 0.72241635  0.11677891 -0.02900450  1.0000000
```

```r
# Base Model
model1 <- lm(Price ~ InventoryID + Screen + Memory + Weight + Rating + Company + TypeName + GPU, data =
summary(model1)
```

```
##
## Call:
## lm(formula = Price ~ InventoryID + Screen + Memory + Weight +
##     Rating + Company + TypeName + GPU, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -975.88 -188.34  -35.98  161.98 1523.80
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)        1421.95846  289.06648   4.919 1.10e-06 ***
## InventoryID           0.06703    0.08057   0.832  0.40572
## Screen             -120.38632   21.81619  -5.518 4.94e-08 ***
## Memory               87.25133    4.40165  19.822  < 2e-16 ***
```

```
## Weight              270.53619   49.45684   5.470 6.41e-08 ***
## Rating              -11.66999    4.61720  -2.528  0.01172 *
## CompanyDell          77.66116   44.30153   1.753  0.08007 .
## CompanyHP           147.04460   51.77867   2.840  0.00465 **
## CompanyLenovo        -1.75199   61.33453  -0.029  0.97722
## TypeNameNotebook    -81.93259   59.12440  -1.386  0.16629
## TypeNameUltrabook   405.48355   76.00576   5.335 1.32e-07 ***
## GPUIntel            164.06650   39.70629   4.132 4.06e-05 ***
## GPUNvidia           217.76478   46.46989   4.686 3.39e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 337.7 on 652 degrees of freedom
## Multiple R-squared:  0.6601, Adjusted R-squared:  0.6538
## F-statistic: 105.5 on 12 and 652 DF,  p-value: < 2.2e-16
```

```r
library(olsrr)
ols_step_backward_p(model1, p_val = 0.05, progress = TRUE)
```

```
## Backward Elimination Method
## ---------------------------
##
## Candidate Terms:
##
## 1. InventoryID
## 2. Screen
## 3. Memory
## 4. Weight
## 5. Rating
## 6. Company
## 7. TypeName
## 8. GPU
##
##
## Variables Removed:
##
## => InventoryID
##
## No more variables to be removed.

##
##
##                               Stepwise Summary
## -----------------------------------------------------------------------------
## Step    Variable         AIC         SBC         SBIC        R2       Adj. R2
## -----------------------------------------------------------------------------
##  0      Full Model     9645.396    9708.393    7750.566    0.66009    0.65383
##  1      InventoryID    9644.102    9702.599    7749.215    0.65973    0.65399
## -----------------------------------------------------------------------------
##
## Final Model Output
## ------------------
##
##                           Model Summary
## --------------------------------------------------------------------
```

```
## R                         0.812       RMSE                 334.527
## R-Squared                 0.660       MSE               111908.165
## Adj. R-Squared            0.654       Coef. Var             32.882
## Pred R-Squared            0.644       AIC                 9644.102
## MAE                     247.405       SBC                 9702.599
## ------------------------------------------------------------------
##  RMSE: Root Mean Square Error
##  MSE: Mean Square Error
##  MAE: Mean Absolute Error
##  AIC: Akaike Information Criteria
##  SBC: Schwarz Bayesian Criteria
##
##                                  ANOVA
## --------------------------------------------------------------------------
##                     Sum of
##                    Squares         DF     Mean Square       F          Sig.
## --------------------------------------------------------------------------
## Regression    144284374.791         11    13116761.345    115.095    0.0000
## Residual       74418929.460        653      113964.670
## Total         218703304.251        664
## --------------------------------------------------------------------------
##
##                                       Parameter Estimates
## ----------------------------------------------------------------------------------------------------
##             model        Beta    Std. Error    Std. Beta       t        Sig        lower       upper
## ----------------------------------------------------------------------------------------------------
##       (Intercept)    1434.247       288.621                  4.969    0.000      867.510    2000.984
##            Screen    -120.932        21.801       -0.249    -5.547    0.000     -163.741     -78.123
##            Memory      87.373         4.398        0.574    19.866    0.000       78.736      96.009
##            Weight     271.099        49.441        0.287     5.483    0.000      174.017     368.180
##            Rating     -11.821         4.613       -0.059    -2.563    0.011      -20.878      -2.763
##       CompanyDell      86.725        42.931        0.069     2.020    0.044        2.426     171.025
##         CompanyHP     170.395        43.502        0.131     3.917    0.000       84.974     255.816
##     CompanyLenovo      35.328        42.129        0.028     0.839    0.402      -47.396     118.052
##  TypeNameNotebook     -76.295        58.721       -0.061    -1.299    0.194     -191.600      39.009
## TypeNameUltrabook     416.394        74.848        0.265     5.563    0.000      269.421     563.366
##          GPUIntel     165.430        39.663        0.144     4.171    0.000       87.547     243.312
##         GPUNvidia     218.139        46.457        0.173     4.696    0.000      126.916     309.361
## ----------------------------------------------------------------------------------------------------
```

```r
# out-of-sample R² for base model
pred_test <- predict(model1, newdata = test)
sse <- sum((test$Price - pred_test)^2)
sst <- sum((test$Price - mean(test$Price))^2)
R2_out <- 1 - sse/sst
R2_out
```

```
## [1] 0.5506981
```

Looking at the correlation matrix, I observed that Screen and Weight are highly correlated (0.81). This high collinearity inflates variances of coefficient estimates and makes interpretation unstable. Indeed, in the full regression output, both Screen and Weight appeared significant, but their strong correlation makes it difficult to separate their individual contributions. To avoid redundancy and multicollinearity, I decided to drop Weight and retain Screen, which has a closer correlation to our dependent variable (price), low p-value (statistically significant), and which is more directly interpretable as a feature consumers see when buying laptops. Next, I examined the coefficient for InventoryID. This variable is simply an internal stock identifier and has no managerial meaning for pricing. Its coefficient was very small (0.067), with a p-value of 0.406, confirming it was not statistically significant. Including InventoryID risks overfitting without adding explanatory value. Therefore, I chose to exclude InventoryID from the second model.

```r
# 2nd Model to compare with
model2 <- lm(Price ~ Screen + Memory + Rating + Company + TypeName + GPU, data = train)
summary(model2)
```

```
##
## Call:
## lm(formula = Price ~ Screen + Memory + Rating + Company + TypeName +
##     GPU, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1040.73  -194.38   -36.05   166.39  1559.87
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        823.027    272.079   3.025  0.00258 **
## Screen             -36.598     15.791  -2.318  0.02077 *
## Memory              90.206      4.464  20.208  < 2e-16 ***
## Rating             -12.815      4.710  -2.721  0.00669 **
## CompanyDell        124.905     43.294   2.885  0.00404 **
## CompanyHP          175.313     44.449   3.944 8.87e-05 ***
## CompanyLenovo       60.894     42.790   1.423  0.15519
## TypeNameNotebook  -234.455     52.273  -4.485 8.60e-06 ***
## TypeNameUltrabook  203.684     65.418   3.114  0.00193 **
## GPUIntel           163.638     40.534   4.037 6.05e-05 ***
## GPUNvidia          227.598     47.445   4.797 2.00e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 345 on 654 degrees of freedom
## Multiple R-squared:  0.6441, Adjusted R-squared:  0.6386
## F-statistic: 118.3 on 10 and 654 DF,  p-value: < 2.2e-16
```

```r
# out-of-sample R² for model2
pred_test <- predict(model2, newdata = test)
sse <- sum((test$Price - pred_test)^2)
sst <- sum((test$Price - mean(test$Price))^2)
R2_out <- 1 - sse/sst
R2_out
```

```
## [1] 0.5216836
```

Here we notice that our OOS R^2 value and Multiple R-squared values have dropped. Although removing InventoryID and Weight reduces the OOS R^2 of our model (from 0.5506981 to 0.5216836), InventoryID was removed for the above mentioned reasons as it was not significant and also has no managerial insight or impact on the model. However, removing Weight was a tradeoff between predictive power and interpretability given that it is highly corelated with Screen. If our emphasis was purely on predictive power, one could make a case to include it in the model.

```r
# 3rd Model to compare with
model3 <- lm(Price ~ Screen + Memory + Company + TypeName + GPU, data = train)
summary(model3)
```

```
##
## Call:
## lm(formula = Price ~ Screen + Memory + Company + TypeName + GPU,
##     data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1094.55  -205.66   -29.65   159.11  1543.68
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        716.852    270.578   2.649  0.00826 **
## Screen             -34.580     15.850  -2.182  0.02949 *
## Memory              90.349      4.485  20.144  < 2e-16 ***
## CompanyDell        127.121     43.497   2.923  0.00359 **
## CompanyHP          177.134     44.660   3.966 8.11e-05 ***
## CompanyLenovo       64.252     42.981   1.495  0.13542
## TypeNameNotebook  -228.566     52.483  -4.355 1.54e-05 ***
## TypeNameUltrabook  209.510     65.702   3.189  0.00150 **
## GPUIntel           162.281     40.728   3.985 7.52e-05 ***
## GPUNvidia          223.443     47.652   4.689 3.34e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 346.7 on 655 degrees of freedom
## Multiple R-squared:   0.64,  Adjusted R-squared:  0.6351
## F-statistic: 129.4 on 9 and 655 DF,  p-value: < 2.2e-16
```

```r
# out-of-sample R² for model3
pred_test <- predict(model3, newdata = test)
sse <- sum((test$Price - pred_test)^2)
sst <- sum((test$Price - mean(test$Price))^2)
R2_out <- 1 - sse/sst
R2_out
```

```
## [1] 0.5225497
```

Here I noticed that in the 3rd model, removing rating improves our OOS R^2 but worsens our Multiple R-squared. But I still chose to include it as it remains statistically significant in our model and also provides managerial impact and is intuitively a factor that consumers consider when thinking about price.

After considering the models, I decided to go ahead with Model2 (i.e dropping InventoryID and Weight, but not dropping Rating) as this configuration provides much more interpretability at a slightly low prediction power. It's important to know the requirements of our client so we can decide the tradeoff between predictive power and explainability/managerial sense.

```r
model2 <- lm(Price ~ Screen + Memory + Rating + Company + TypeName + GPU, data = train)
summary(model2)
```

```
##
## Call:
```

```
## lm(formula = Price ~ Screen + Memory + Rating + Company + TypeName +
##     GPU, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1040.73  -194.38   -36.05   166.39  1559.87
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        823.027    272.079   3.025  0.00258 **
## Screen             -36.598     15.791  -2.318  0.02077 *
## Memory              90.206      4.464  20.208  < 2e-16 ***
## Rating             -12.815      4.710  -2.721  0.00669 **
## CompanyDell        124.905     43.294   2.885  0.00404 **
## CompanyHP          175.313     44.449   3.944 8.87e-05 ***
## CompanyLenovo       60.894     42.790   1.423  0.15519
## TypeNameNotebook  -234.455     52.273  -4.485 8.60e-06 ***
## TypeNameUltrabook  203.684     65.418   3.114  0.00193 **
## GPUIntel           163.638     40.534   4.037 6.05e-05 ***
## GPUNvidia          227.598     47.445   4.797 2.00e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 345 on 654 degrees of freedom
## Multiple R-squared:  0.6441, Adjusted R-squared:  0.6386
## F-statistic: 118.3 on 10 and 654 DF,  p-value: < 2.2e-16
```

(B)

In the final model, the effects of Screen, Memory, Company, Rating, Ultrabook type, and GPU are managerially sensible, as they align with expectations that screen size, RAM, premium designs, brand value, GPUs have a strong impact on laptop prices. However, the negative effect of Screen size and Rating seems to be conterintuitive. So it is imperative to investigate those features further in depth and look carefully at the feature data provided. But the other variables included do follow expectations based on their estimates and their impact on price.

(C)

Based on the model, HP has the highest effect on laptop price adding 175.313 more than Asus holding other factors constant, commanding the largest premium over the other brands. Even though Lenovo's coefficients are the lowest among the listed coefficients, one can argue that Asus has the smallest effect, since it is the baseline and all other manufacturers have higher estimated coefficients. While Lenovo's effect is positive (60.894), it is not statistically significant, which suggests Lenovo laptops are priced similarly to Asus on average.

(D)

```r
pred_test <- predict(model2, newdata = test)
sse <- sum((test$Price - pred_test)^2)
sst <- sum((test$Price - mean(test$Price))^2)
R2_out <- 1 - sse/sst
R2_out  # out-of-sample R²
```

```
## [1] 0.5216836
```

Interpretation: This means that when predicting laptop prices on unseen test data,the model explains about 52.16836% of the variation in prices compared to simply predicting the mean price for all laptops.

Formally, out-of-sample R^2 is defined as $R^2 = 1 - (SSE/SST)$, where SSE is the sum of squared prediction errors on the test set and SST is the total sum of squared deviations of the test set prices from their mean. An out-of-sample R^2 of 0.5216836 indicates that the model explains 52.16836% of the variation in laptop prices in the test data, compared to a baseline model that always predicts the average price.

(E)

```
newlap <- data.frame(
  InventoryID = 950,
  Screen = 15.6,
  Memory = 6,
  Weight = 3,
  Rating = 8,
  Company = factor("Asus", levels = levels(train$Company)),
  TypeName = factor("Ultrabook", levels = levels(train$TypeName)),
  GPU = factor("Intel", levels = levels(train$GPU))
)

# Prediction with prediction interval (includes error variance)
pred <- predict(model2, newdata = newlap, interval = "prediction", level = 0.95)
pred
```

```
##        fit      lwr      upr
## 1 1058.133 371.3338 1744.931
```

```
p <- predict(model2, newdata = newlap, se.fit = TRUE)
sigma2 <- summary(model2)$sigma^2
se_pred <- sqrt(p$se.fit^2 + sigma2)
df <- model2$df.residual

t_stat <- (1100 - p$fit) / se_pred
prob_gt_1100 <- 1 - pt(t_stat, df = df)
prob_gt_1100
```

```
##         1
## 0.4523782
```

For this laptop, the model predicts an average price of 1,058.133 euros, with a 95% prediction interval ranging from 371.3338 euros to 1,744.931 euros. The probability that the actual price exceeds €1,100 is 45.23782%. This calculation is based on the assumptions that regression errors are normally distributed (so the t-distribution approximation for the probability is valid), homoscedastic, and independent, and that the model is correctly specified.

(F)

```
model_mem_gpu <- lm(Price ~ Screen + Memory + Rating + TypeName + Company +
                      GPU + Memory:GPU, data = train)
summary(model_mem_gpu)
```

```
##
## Call:
## lm(formula = Price ~ Screen + Memory + Rating + TypeName + Company +
##     GPU + Memory:GPU, data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1086.71  -194.71   -30.79   158.38  1569.31
```

```
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)    
## (Intercept)        1044.932    277.494   3.766 0.000181 ***
## Screen              -38.107     15.765  -2.417 0.015915 *  
## Memory               60.243      9.964   6.046  2.5e-09 ***
## Rating              -13.055      4.675  -2.792 0.005384 ** 
## TypeNameNotebook   -211.650     54.205  -3.905 0.000104 ***
## TypeNameUltrabook   213.135     69.884   3.050 0.002382 ** 
## CompanyDell         121.588     42.982   2.829 0.004816 ** 
## CompanyHP           165.627     44.206   3.747 0.000195 ***
## CompanyLenovo        61.037     42.503   1.436 0.151466    
## GPUIntel            -70.059     90.786  -0.772 0.440573    
## GPUNvidia           -78.966    101.655  -0.777 0.437557    
## Memory:GPUIntel      32.854     11.941   2.751 0.006102 ** 
## Memory:GPUNvidia     39.956     11.632   3.435 0.000631 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 342.4 on 652 degrees of freedom
## Multiple R-squared:  0.6505, Adjusted R-squared:  0.6441 
## F-statistic: 101.1 on 12 and 652 DF,  p-value: < 2.2e-16
```

```r
pred_test <- predict(model_mem_gpu, newdata = test)
sse <- sum((test$Price - pred_test)^2)
sst <- sum((test$Price - mean(test$Price))^2)
R2_out <- 1 - sse/sst
R2_out  # out-of-sample R²
```

```
## [1] 0.5180113
```

The interaction terms show that the effect of Memory depends on the GPU type installed. For AMD laptops, each extra unit of memory adds about 60.243 euros to price. For Intel and Nvidia laptops, the memory premium is much higher, about 93.097 euros and 100.199 euros per memory unit, respectively. This suggests that additional RAM is valued more when paired with stronger GPUs. Compared to the model in part (a), this specification highlights brand-technology complementarities, and shows that consumers value memory more when paired with GPUs, though overall model fit (adjusted R^2) is slightly lower. When I added the Memory × GPU interaction, the adjusted R^2 increased slightly (0.639 to 0.644), but the out-of-sample R^2 decreased marginally (0.522 to 0.518) which suggests no predictive gain, but the model provides richer interpretation: the price premium for additional RAM depends on GPU type.

(G)

```r
train$Company  <- factor(train$Company,  levels = c("Asus","Dell","HP","Lenovo"))
train$TypeName <- factor(train$TypeName, levels = c("Gaming","Notebook","Ultrabook"))
train$GPU      <- factor(train$GPU,      levels = c("AMD","Intel","Nvidia"))

# Model with GPU × Company interaction
model_gpu_company <- lm(
  Price ~ Screen + Memory + Rating + TypeName + GPU + Company + GPU:Company,
  data = train
)
summary(model_gpu_company)
```

```
##
```

```
## Call:
## lm(formula = Price ~ Screen + Memory + Rating + TypeName + GPU +
##     Company + GPU:Company, data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1162.7  -201.0   -16.2   176.6  1515.5
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                1101.353    293.294   3.755 0.000189 ***
## Screen                      -36.089     15.640  -2.307 0.021343 *
## Memory                       88.693      4.388  20.212  < 2e-16 ***
## Rating                      -12.318      4.623  -2.665 0.007901 **
## TypeNameNotebook           -163.473     53.046  -3.082 0.002145 **
## TypeNameUltrabook           295.321     66.385   4.449 1.02e-05 ***
## GPUIntel                   -270.926    141.486  -1.915 0.055950 .
## GPUNvidia                   -56.560    135.229  -0.418 0.675900
## CompanyDell                -253.432    138.346  -1.832 0.067429 .
## CompanyHP                  -186.109    143.655  -1.296 0.195600
## CompanyLenovo              -297.136    151.692  -1.959 0.050563 .
## GPUIntel:CompanyDell        394.962    151.783   2.602 0.009476 **
## GPUNvidia:CompanyDell       517.452    153.830   3.364 0.000814 ***
## GPUIntel:CompanyHP          472.646    155.538   3.039 0.002471 **
## GPUNvidia:CompanyHP         178.070    162.639   1.095 0.273978
## GPUIntel:CompanyLenovo      490.746    163.354   3.004 0.002766 **
## GPUNvidia:CompanyLenovo     242.211    162.478   1.491 0.136519
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 337.5 on 648 degrees of freedom
## Multiple R-squared:  0.6625, Adjusted R-squared:  0.6541
## F-statistic: 79.49 on 16 and 648 DF,  p-value: < 2.2e-16
```

```r
pred_test <- predict(model_gpu_company, newdata = test)
sse <- sum((test$Price - pred_test)^2)
sst <- sum((test$Price - mean(test$Price))^2)
R2_out <- 1 - sse/sst
R2_out  # out-of-sample R²
```

```
## [1] 0.5331728
```

The GPU * Company interaction terms show that the price impact of GPUs varies by manufacturer. For Asus (baseline), Intel and Nvidia GPUs do not add value, but for Dell, HP, and Lenovo, Intel GPUs command strong positive premiums of about 394-491 euros. Nvidia GPUs also add large premiums at Dell but not at HP or Lenovo. Compared to the model in part (a), this specification provides richer insight into brand-specific GPU pricing and slightly improves both adjusted R^2 (0.639 to 0.654) and out-of-sample R^2 (0.522 to 0.533).