Price and Performance in Central Processing Units

Alan Liu

11/19/2020

## Introduction

# Load library  
library(tidyverse)

## -- Attaching packages --------------------------------------------------------------------------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2 v purrr 0.3.4  
## v tibble 3.0.3 v dplyr 1.0.2  
## v tidyr 1.1.1 v stringr 1.4.0  
## v readr 1.3.1 v forcats 0.5.0

## Warning: package 'ggplot2' was built under R version 4.0.3

## -- Conflicts ------------------------------------------------------------------------------------------------------------ tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

# install.packages(readr)  
library(readr)  
  
# install.packages(car) on a separate R script  
library(car)

## Warning: package 'car' was built under R version 4.0.3

## Loading required package: carData

## Warning: package 'carData' was built under R version 4.0.3

##   
## Attaching package: 'car'

## The following object is masked from 'package:dplyr':  
##   
## recode

## The following object is masked from 'package:purrr':  
##   
## some

# install.packages(psych) on a separate R script  
library(psych)

##   
## Attaching package: 'psych'

## The following object is masked from 'package:car':  
##   
## logit

## The following objects are masked from 'package:ggplot2':  
##   
## %+%, alpha

# install.packages('readr'leaps')  
library(leaps)  
  
# Import data  
cpudata <- read\_csv("researchproject\_data.csv")

## Parsed with column specification:  
## cols(  
## MSRP = col\_double(),  
## Year = col\_double(),  
## Benchmark\_Result = col\_double(),  
## Brand = col\_character(),  
## Processor = col\_character(),  
## Chipset = col\_character()  
## )

## Variables

MSRP - Currency, numerical variable that displays the original price a chip was marketed for

Year - Date, numerical variable that is the original year of release for the chip

Benchmark\_Result - Rating, numerical variable that calculates the performance of a chip

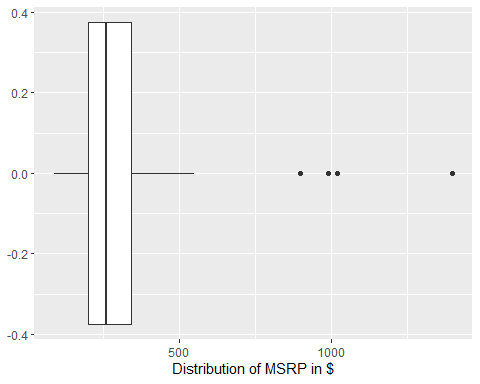
Brand - Name, categorical variable that displays the branding of a chip

Processor - Name, categorical variable that displays the associated chip

Chipset - Name, categorical variable that displays the chipset the processor was built on

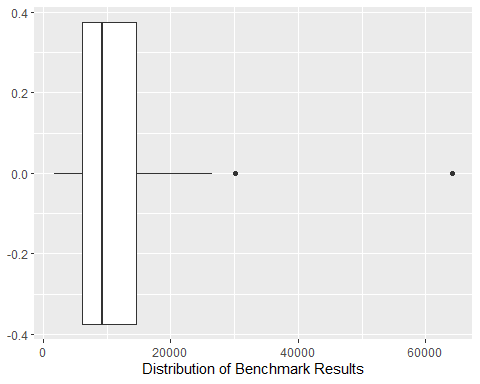
## Variable Regression Analysis

# Visualize the predictor variable  
# Creates the box plot for MSRP  
ggplot(cpudata, aes(y=MSRP)) +  
 geom\_boxplot() + coord\_flip() +  
 labs(y = "Distribution of MSRP in $")



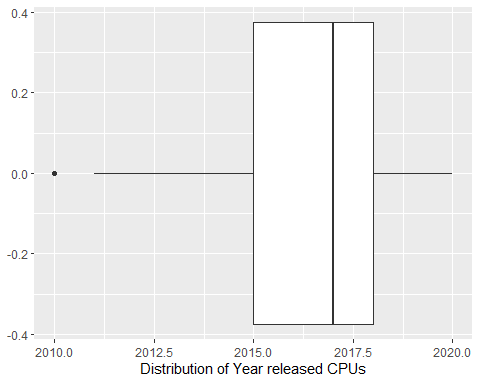
This box plot shows a distribution of the MSRP across the 60 CPUs in the data.

# Visualize the predictor variable  
# Creates the box plot for Benchmark\_Results  
ggplot(cpudata, aes(y=Benchmark\_Result)) + geom\_boxplot() + coord\_flip() +  
 labs(y = "Distribution of Benchmark Results")



This box plot shows a distribution of the Benchmark Results for all the CPUs. The higher the score, the better performance the CPU has.

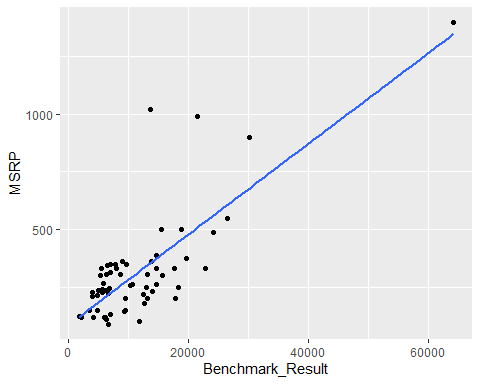
# Visualize the predictor variable  
# Creates the box plot for Year  
ggplot(cpudata, aes(y=Year)) + geom\_boxplot() + coord\_flip() +  
 labs(y = "Distribution of Year released CPUs")



This box plot shows a distribution of the years released for all the CPUs.

# Visualize the data with the regression line  
# Creates the scatterplot for MSRP  
ggplot(cpudata, aes(x=Benchmark\_Result, y=MSRP)) +  
 geom\_point() +  
 geom\_smooth(method='lm', se = FALSE)

## `geom\_smooth()` using formula 'y ~ x'

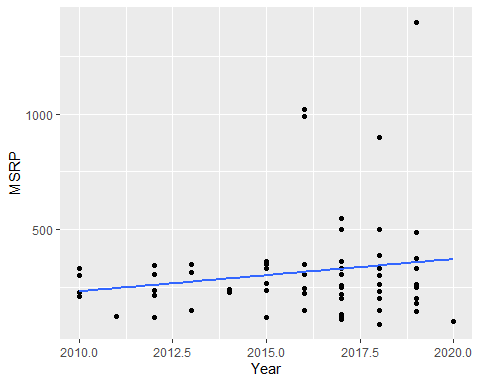


# Summary statistics : Correlation coefficient  
# Calculates the correlation coefficient for Air Permeability  
cor(cpudata$MSRP,cpudata$Benchmark\_Result)

## [1] 0.7791367

# Visualize the data with the regression line  
# Creates the scatterplot for Year  
ggplot(cpudata, aes(x=Year, y=MSRP)) +  
 geom\_point() +  
 geom\_smooth(method='lm', se = FALSE)

## `geom\_smooth()` using formula 'y ~ x'

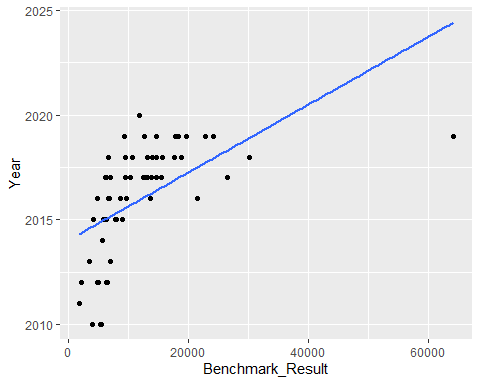


# Summary statistics : Correlation coefficient  
# Calculates the correlation coefficient for Air Permeability  
cor(cpudata$MSRP,cpudata$Year)

## [1] 0.1625529

# Visualize the data with the regression line  
# Creates the scatterplot for Predictors  
ggplot(cpudata, aes(x=Benchmark\_Result, y=Year)) +  
 geom\_point() +  
 geom\_smooth(method='lm', se = FALSE)

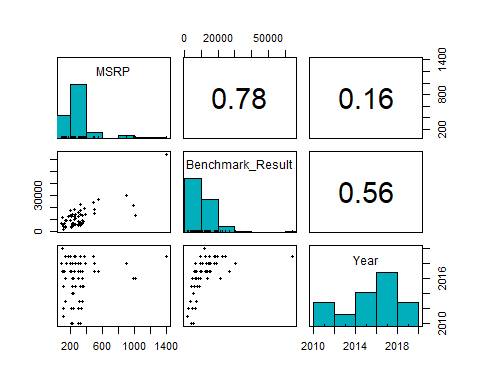
## `geom\_smooth()` using formula 'y ~ x'



# Summary statistics : Correlation coefficient  
# Calculates the correlation coefficient for Air Permeability  
cor(cpudata$Year,cpudata$Benchmark\_Result)

## [1] 0.5551998

# A fancy scatterplot matrix  
pairs.panels(cpudata[c("MSRP","Benchmark\_Result","Year")],  
method = "pearson", # correlation method  
hist.col = "#00AFBB", # color of histogram  
smooth = FALSE, density = FALSE, ellipses = FALSE)



Graphs indicate the correlation coefficients among all three variables and provide various plots that indicate point distributions (via histograms and scatterplots).

## Model Building Strategy

# Fit the regression model with 1 predictor, Benchmark Results  
reg <- lm(MSRP ~ Benchmark\_Result + Year, cpudata)  
  
# Display the summary table for the regression model  
summary(reg)

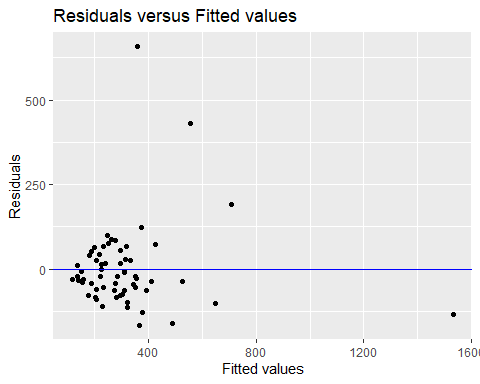
##   
## Call:  
## lm(formula = MSRP ~ Benchmark\_Result + Year, data = cpudata)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -166.93 -63.79 -23.16 40.69 658.42   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.805e+04 1.474e+04 4.617 2.21e-05 \*\*\*  
## Benchmark\_Result 2.517e-02 2.140e-03 11.763 < 2e-16 \*\*\*  
## Year -3.374e+01 7.318e+00 -4.611 2.25e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 127.6 on 58 degrees of freedom  
## Multiple R-squared: 0.7125, Adjusted R-squared: 0.7025   
## F-statistic: 71.85 on 2 and 58 DF, p-value: < 2.2e-16

# Display the correlation coefficient  
coefficients(reg)

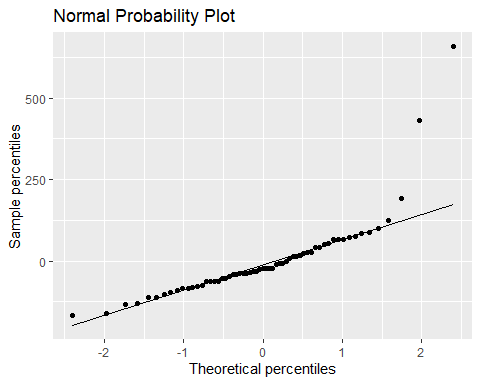
## (Intercept) Benchmark\_Result Year   
## 6.804720e+04 2.517316e-02 -3.374488e+01

## Assumptions Check

# Residuals versus Fitted values  
cpudata$resids <- residuals(reg)  
cpudata$predicted <- predict(reg)  
ggplot(cpudata, aes(x=predicted, y=resids)) + geom\_point() + geom\_hline(yintercept=0, color = "blue") +  
labs(title ="Residuals versus Fitted values", x = "Fitted values", y ="Residuals")



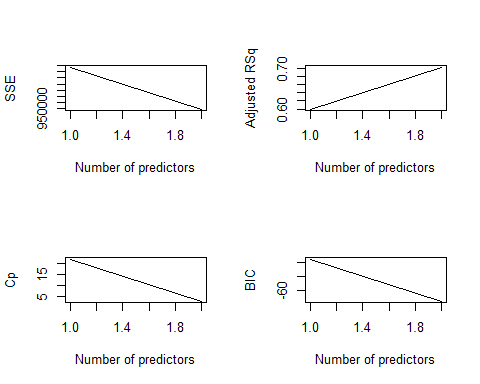
# Normal probability plot  
ggplot(cpudata, aes(sample = resids)) + stat\_qq() + stat\_qq\_line() +  
labs(title ="Normal Probability Plot", x = "Theoretical percentiles", y = "Sample percentiles")



Will have to ejected some outliers in order to make the assumptions checks pass. Too many outliers that could lead to incorrect assumptions.

## Deciding on the Best Model

# Find the best model for each number of predictors (with 3 predictors maximum)  
models <- regsubsets(MSRP ~ Benchmark\_Result + Year, cpudata, nvmax = 3)  
models.sum <- summary(models)  
# Create four plots within a 2x2 frame to compare the different criteria  
par(mfrow = c(2,2))  
# SSE  
plot(models.sum$rss, xlab = "Number of predictors", ylab = "SSE", type = "l")  
# R2  
plot(models.sum$adjr2, xlab = "Number of predictors", ylab = "Adjusted RSq", type = "l")  
# Mallow's Cp  
plot(models.sum$cp, xlab = "Number of predictors", ylab = "Cp", type = "l")  
# BIC  
plot(models.sum$bic, xlab = "Number of predictors", ylab = "BIC", type = "l")



# Calculate the squared predictor variables to include in the model and the interaction term:  
cpudata <- cpudata %>%  
mutate(bench2 = Benchmark\_Result^2,  
bench.year = Benchmark\_Result\*Year)  
# Fit the polynomial regression model  
reg2 <- lm(MSRP ~ Year + Benchmark\_Result + bench2 + bench.year, cpudata)  
# Display the summary table for the regression model  
summary(reg2)

##   
## Call:  
## lm(formula = MSRP ~ Year + Benchmark\_Result + bench2 + bench.year,   
## data = cpudata)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -305.43 -43.84 -6.61 32.58 529.29   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) -1.458e+04 2.344e+04 -0.622 0.537   
## Year 7.194e+00 1.164e+01 0.618 0.539   
## Benchmark\_Result 1.582e+01 2.871e+00 5.510 9.40e-07 \*\*\*  
## bench2 9.386e-08 8.796e-08 1.067 0.291   
## bench.year -7.826e-03 1.424e-03 -5.497 9.85e-07 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 100.9 on 56 degrees of freedom  
## Multiple R-squared: 0.8263, Adjusted R-squared: 0.8139   
## F-statistic: 66.59 on 4 and 56 DF, p-value: < 2.2e-16

Choose the best model.

# Display the best model (selected predictors are indicated by \*) for each number of predictors  
models.sum$outmat

## Benchmark\_Result Year  
## 1 ( 1 ) "\*" " "   
## 2 ( 1 ) "\*" "\*"

Creating the final model

# Printing the final model with the best number of predictor variables  
final\_model <- lm(MSRP ~ Benchmark\_Result + Year, cpudata)  
summary(final\_model)

##   
## Call:  
## lm(formula = MSRP ~ Benchmark\_Result + Year, data = cpudata)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -166.93 -63.79 -23.16 40.69 658.42   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)   
## (Intercept) 6.805e+04 1.474e+04 4.617 2.21e-05 \*\*\*  
## Benchmark\_Result 2.517e-02 2.140e-03 11.763 < 2e-16 \*\*\*  
## Year -3.374e+01 7.318e+00 -4.611 2.25e-05 \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 127.6 on 58 degrees of freedom  
## Multiple R-squared: 0.7125, Adjusted R-squared: 0.7025   
## F-statistic: 71.85 on 2 and 58 DF, p-value: < 2.2e-16

The equation for the final model is without outliers removed:

MSRP = 6805 + 2.517e^-2(Benchmark\_Result) - 33.74(Year)

The coefficient of determination is 0.7125.