Pontificia Universidad Católica de Chile
Escuela de Administración
Machine Learning Para Negocios (EAA3707-1)
Profesora: María Ignacia Vicuña

Tarea 1:
Machine Learning Para Negocios

Nombres:
Vicente Jaramillo
José Vilchez

Fecha de entrega:
23 de Septiembre, 2022

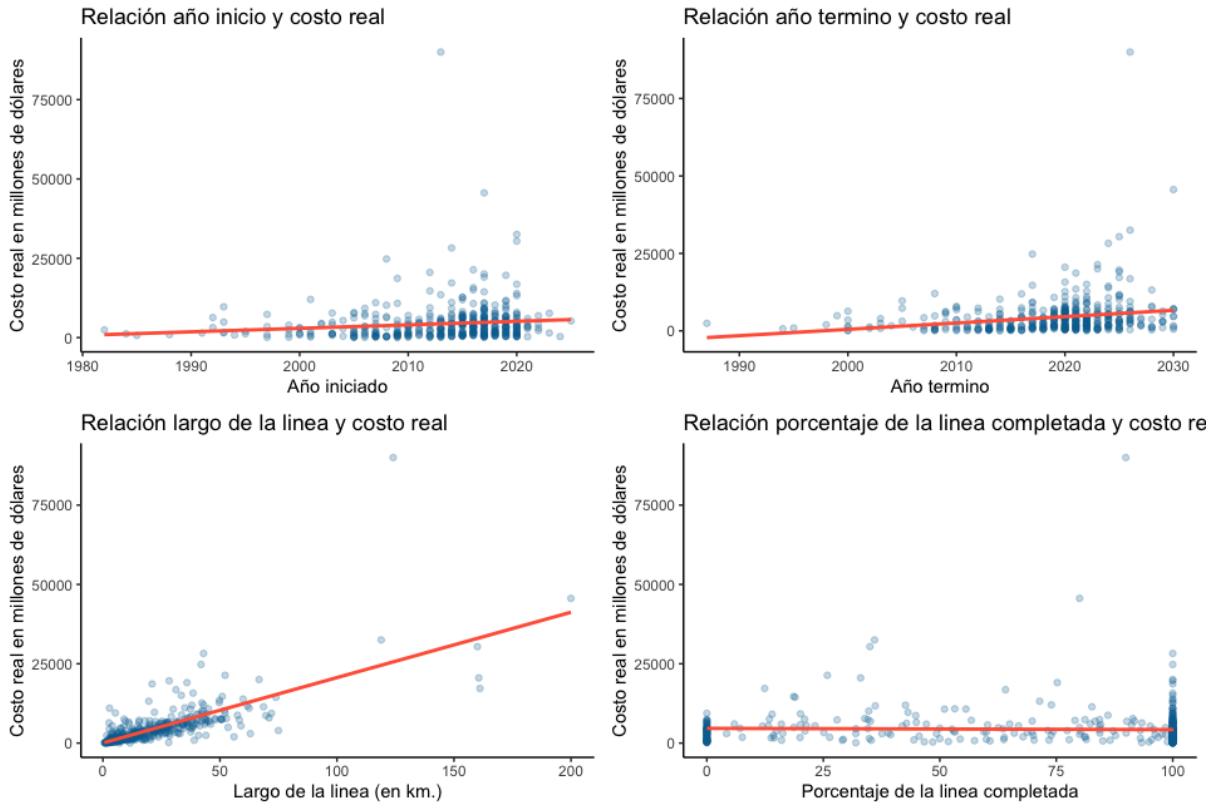
Pontificia Universidad Católica de Chile
Escuela de Administración
Machine Learning Para Negocios (EAA3707-1)
Profesora: María Ignacia Vicuña

Pregunta 1

(a) Realice un análisis bivariado de las variable real_cost con todas las variables independientes. Este análisis debe contener los gráficos de dispersión de cada variable independiente con la variable objetivo y además sus correlaciones de pearson. Interprete los resultados.

Inicialmente se procedió a realizar la limpieza de la base de datos *transit_cost* en donde todas aquellas variables que presentaban valores nulos fueron cambiados a cero con el fin de no generar problemas respecto al estudio de las correlaciones. Así también se realizó un volvio a calcular la variable *tunnel_per* dado que los porcentajes mostrados se encontraban en formato no numérico.
Se presenta a continuación los gráficos de dispersión entre las variables independientes y la variable *real_cost* con el fin de comprender visualmente su comportamiento. Para esto se separaron aquellas variables continuas de las categorías.

Variables continuas

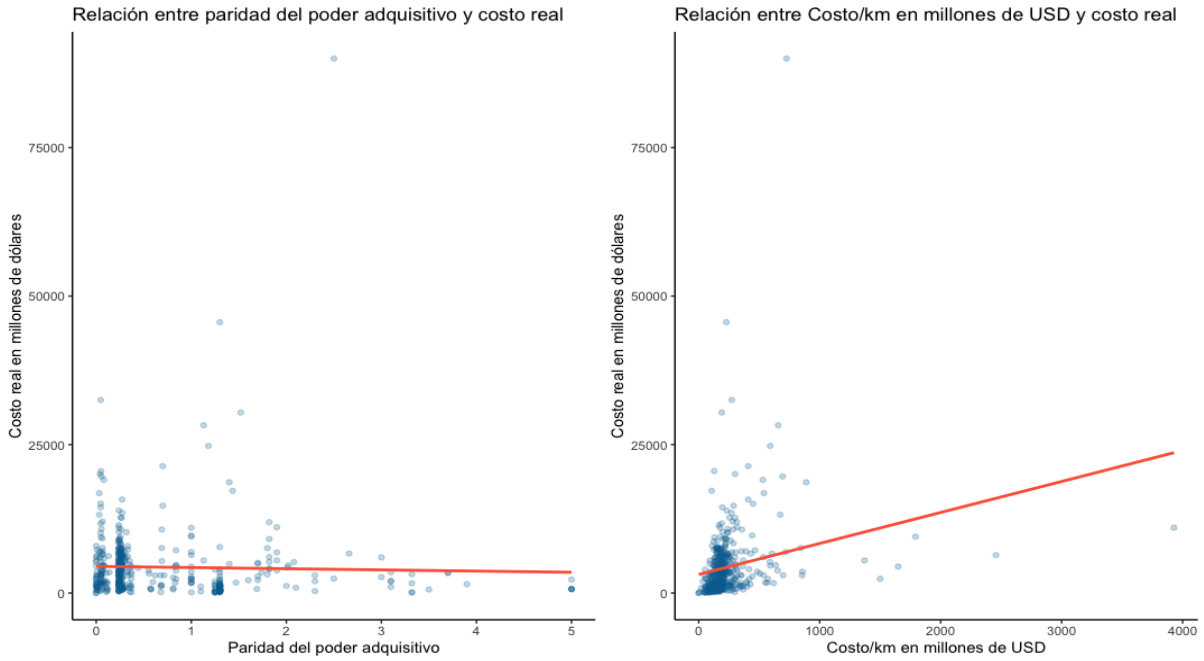


Costo real y año iniciado (star_year): Se observa una correlación baja y positiva entre ambas variables de 12%.

Costo real y año término (end_year): Se observa una correlación mediana-baja y positiva entre ambas variables de 21,1%.

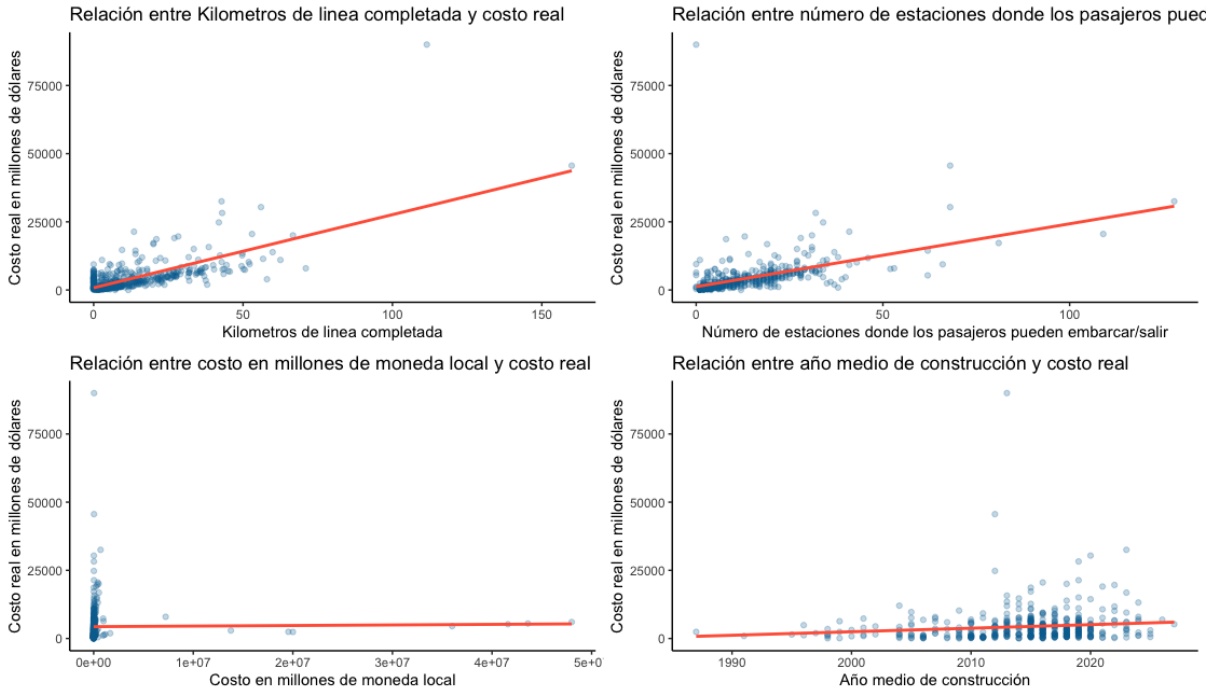
Costo real y largo de la línea (en km.) (length): Se observa una correlación alta y positiva, de 73,4% (sobre el 30%).

Costo real y porcentaje de la línea completada (tunnel_per): Se observa una correlación baja y negativa de -2,53%.



Costo real y paridad del poder adquisitivo (ppp_rate): Se observa una correlación baja y negativa de -2,78%.

Costo real y costo/km en millones de USD (cost_km_millions): Se observa una correlación mediana-baja y positiva de 23%.



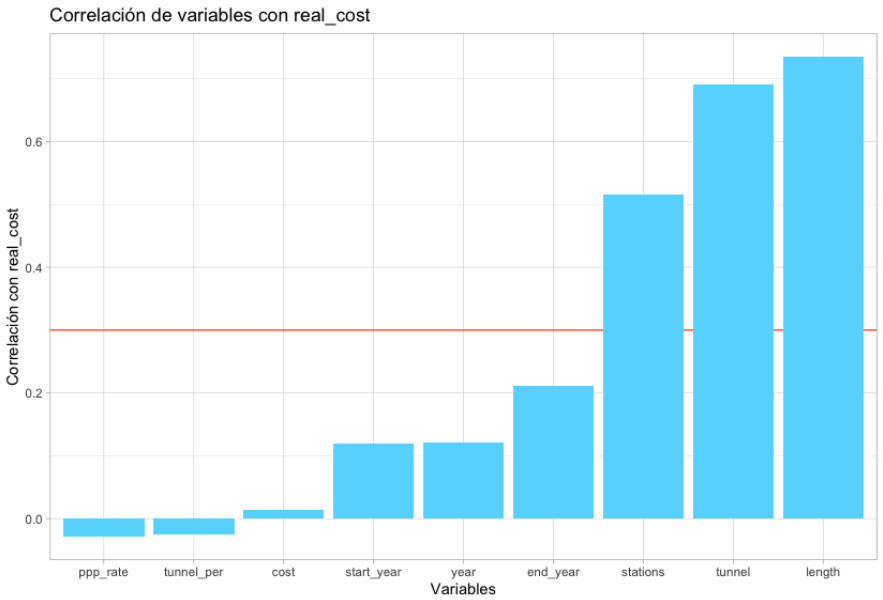
Costo real y kilómetros de línea completada (tunnel): Se observa una correlación alta y positiva de 69,1% (sobre el 30%).

Costo real y número de estaciones donde los pasajeros pueden embarcar/salir (stations): Se observa una correlación medianamente alta y positiva de 51,6% (sobre 30%).

Costo real y costo en millones de moneda local (cost): Se observa una correlación baja y positiva de 1,41%.

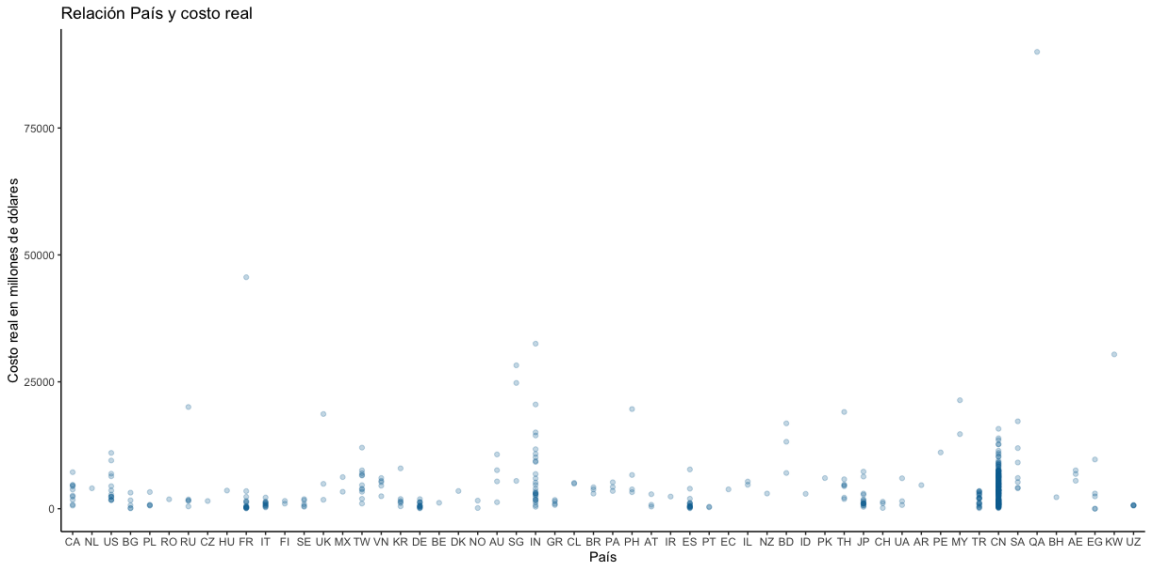
Costo real y año medio de construcción (year): Se observa una correlación positiva y baja de 12,1%.

A continuación se analizan las correlaciones de las variables con *real_cost:*



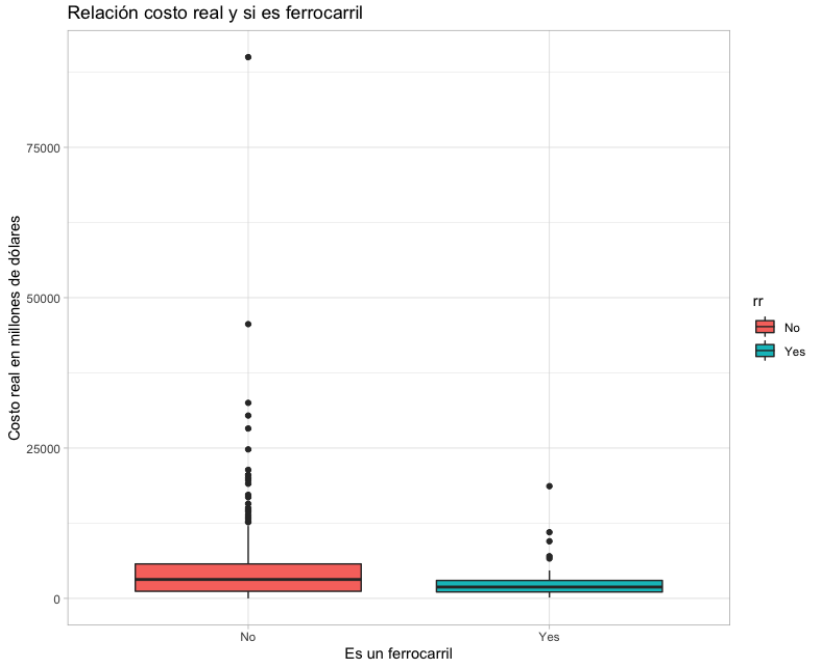
Gráficos bivariados de variables categóricas:

Relación país y costo real:

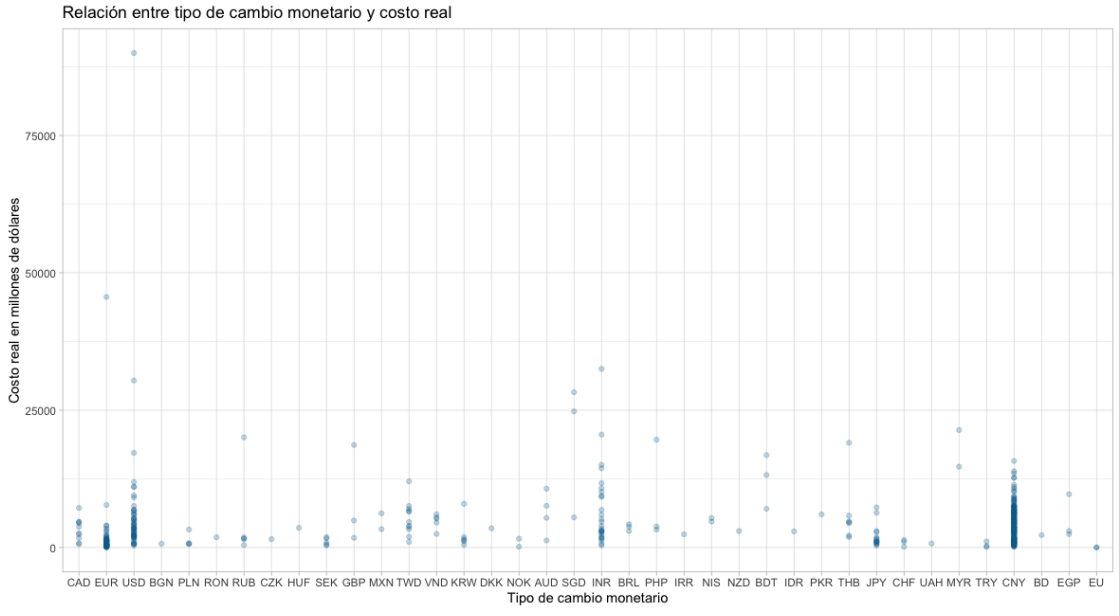


Pontificia Universidad Católica de Chile
Escuela de Administración
Machine Learning Para Negocios (EAA3707-1)
Profesora: María Ignacia Vicuña

Relación costo real y si es ferrocarril:



Relación entre tipo de cambio monetario y costo real:



Con lo que respecta a la correlación entre las variables categóricas se realizó una correlación policórica lo que se presentó los siguientes resultados:

Variable	**Correlación con *real_cost***
Country	0.1669
City	0.1679
RR	-0.1318
Line	0.0198
Currency	0.2317

(b) A partir del análisis bivariado realizado en (a), seleccione aquellas variables cuya correlación con la variable objetivo sea superior a 0.3.

En consecuencia a las observaciones del inciso (a) se encontró que aquellas variables con una correlación con la variable objetivo superior a 0.3 son: ***Stations, Tunnel y Length***.

(c) Con la librería Tidymodels de R ajuste un modelo de regresión lineal múltiple con las variables seleccionadas en el paso anterior. Para ello realice lo siguiente:

- Utilice la semilla set.sed(3707) y realice una partición al 75 % para los datos de entrenamiento.
- Ajuste un modelo de regresión lineal múltiple y reporte los coeficientes estimados del modelo. Con un nivel de significancia al 5 % ¿Qué variables predictoras son significativas?
- Con la muestra de validación, reporte las medidas de bondad de ajuste MAPE, R2.
- Realice un gráfico de dispersión entre el costo real del proyecto y su valor predicho.
- Ajuste nuevamente la regresión utilizando 10-fold cross validación, y reporte las medidas de bondad de ajuste MAPE, R2. Comente y compare con lo obtenido sin remuestreo.

Se formuló dadas las variables del inciso anterior el siguiente modelo de regresión lineal múltiple:

$$real\_cost = \beta_0 + \beta_1 \cdot stations + \beta_2 \cdot tunnel + \beta_3 \cdot length$$

Respecto a la partición al 75% se realizó con una estratificación con respecto a la variable *stations* debido a que sin este se generaba una gran diferencia entre la media y la desviación estándar de los datos de testeo y la de práctica.

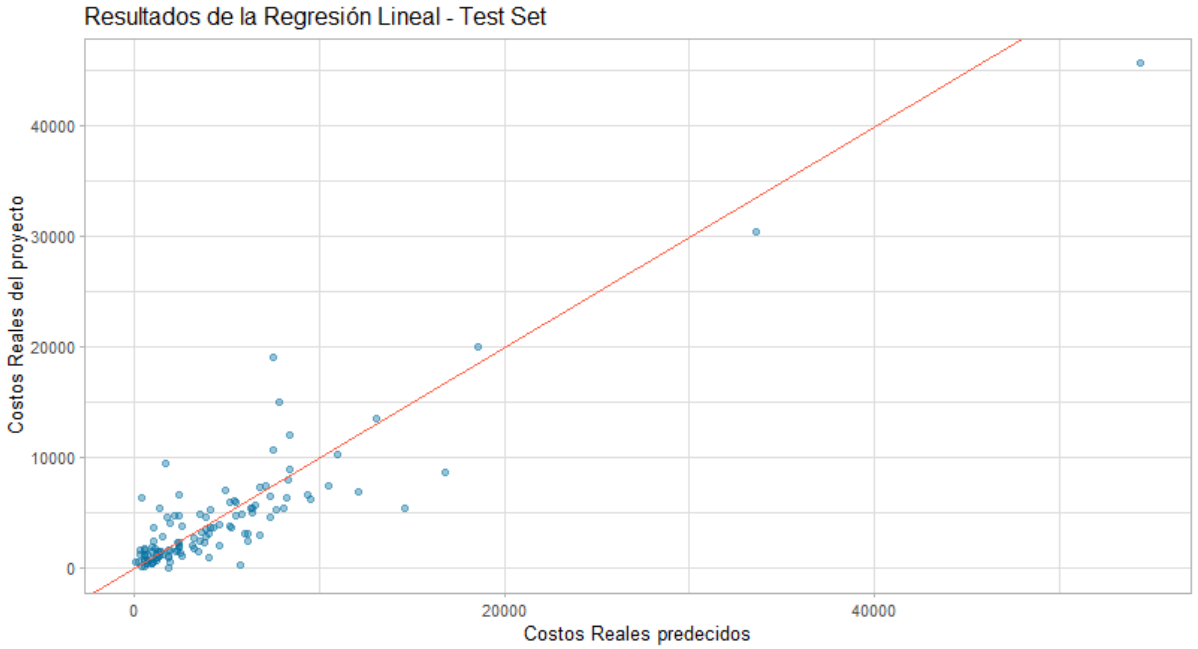
Ante los códigos que se encuentran en el anexo se presenta la siguiente tabla con los coeficientes estimados del modelo:

Variable	Estimación	Error estándar	p-value
Intercepto	4338	233	5.9e-54
stations	-1758	431	5.62e-5
tunnel	1696	328	3.98e-7
length	4637	520	2.89e-17

Como se puede apreciar el valor de los *p-value* son todas por debajo del 5% lo que señala que todas las variables son significativas. Ahora bien, dado que en en el inciso (a) se mostró que ***stations*** presentaba una correlación positiva, pero al realizar la regresión su coeficiente estimado es negativo da señales de existencia de colinealidad entre las variables.

Con la muestra de validación se reporta que la bondad de ajuste MAPE da como resultado infinito a causa de que uno de los valores ***real_cost*** es igual a 0 lo que, a partir de su fórmula, se tenga una fracción dividiendo por cero. En consecuencia a esto se tomaron dos alternativas: Se calculó el **MAPE** omitiendo en su cálculo aquella fracción con resultado infinito lo reportó un valor de **0.777**; también se realizó el cálculo de la **RMSE**, con resultado **0.828**, dado que este no se ve afectado por el valor cero del ***real_cost***. Así también se tiene que el modelo presenta un $\mathbf{R^2}$ de 0,828.

A continuación se presenta el gráfico de dispersión entre el costo real del proyecto y su valor predicho, se puede apreciar que existe un cierto grado de acumulación de los datos en la línea de 45º lo que expresa algún grado de consistencia entre los datos reales y los valores predichos.



Finalmente se ajustó nuevamente la regresión utilizando *10-fold cross validation* y, con esto se tuvo que el **RMSE** es de **4244** con un $\mathbf{R^2}$ de **0,439**. Se realizó con el **RMSE** a causa de que realizando el **MAPE** se obtiene, nuevamente, que este presenta un valor igual a infinito.

Se observa una peor *performance* en la regresión usando 10-fold cross validation (menor $\mathbf{R^2}$). Creemos que esta peor performance se debe al hecho de que la variable ***stations*** (anteriormente mencionada), al estar colineada con las variables, pueda estar impactando la performance del modelo, ocasionando un peor rendimiento.

Pregunta 2:

A partir de los datos transit_cost.xlsx se estimará θ, que representa el costo real medio de los proyectos de tránsito, utilizando boostrap. Realice los siguientes pasos:

(a) Utilice la semilla set.seed(3770) y genere B = 1000 muestras bootstrap de a partir de los datos. Las muestras deben ser del mismo tamaño *n* que la muestra original.

Para la resolución de esta pregunta se procedió a realizar un bootstrap estratificado respecto a la variable *stations* dado que se busca presentar datos entre los de entrenamiento y testeo lo más similares posible en su media y desviación. Los comandos realizados fueron los siguientes:

```
set.seed(3770)
cost_boots = rsample::bootstraps(transit_cost, times = 1000, strata =
stations)
```

(b) Para cada una de las muestras bootstrap, calcule la media de costo del proyecto, denótelo por:

$$\hat{\theta}_i^*, i = 1, ..., B$$

Por medio de un loop del uno al mil se analizó cada una de las muestras bootstrap y se calculó su media, estos valores fueron guardados en un vector. Los comandos utilizados fueron los siguientes:

```
mean_cost = c()

for(x in 1:1000){
  val = (cost_boots$splits[[x]] %>% analysis())$real_cost
  mean_cost[x] =  val %>% mean()
}
```

(c) Calcule el promedio y la desviación estándar de las B estimaciones, y denótelo por: $\widehat{\theta}_{boot}$ y $\widehat{se}_{boot}$ .Compare con la estimación de la media y su error estándar obtenido en la muestra original.

A continuación se presenta la tabla la cual muestra los valores de la media y el error estándar junto con los de la muestra original para su comparación:

	Media	**se**
Estimación (Bootstrap)	4351.608	265.0101
Muestra original	4357.344	6216.262

Los comandos utilizados para la construcción de la tabla fueron los siguientes:

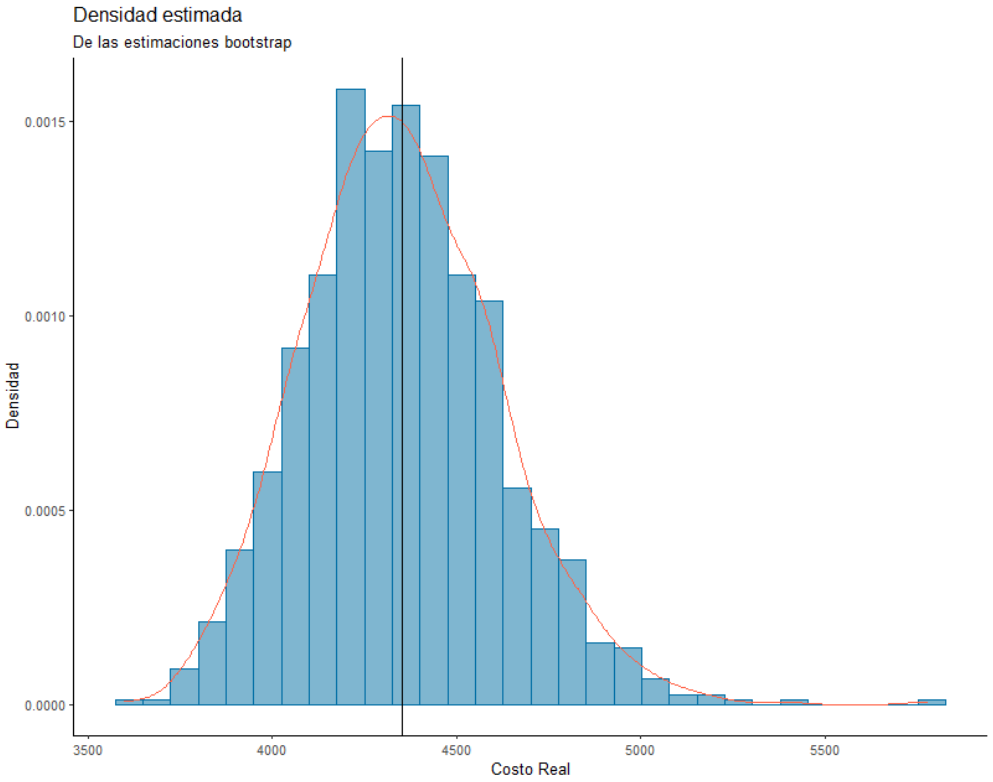
```
transit_cost$real_cost %>%
  mean()
transit_cost$real_cost %>%
  sd()

theta_boot = mean(mean_cost)
theta_boot

se_boot = sd(mean_cost)
se_boot
```

(d) Construya un histograma con las estimaciones bootstrap $\widehat{\theta}_i^*$ y superponga la densidad estimada utilizando la función *density( )*. Esta sería la distribución estimada ($\widehat{F}$) del estimador $\widehat{\theta}$.

Se construyó el siguiente histograma:



El comando para realizar el gráfico fue el siguiente:

```
mean_cost_df = data.frame(mean_cost)

ggplot(data = mean_cost_df, mapping = aes(x = mean_cost)) +
  geom_histogram(aes(y =..density..), fill = '#006EA1', color =
'#006EA5', alpha = 0.5, bins = 30) +
  geom_density(color = 'tomato') +
  geom_vline(xintercept = theta_boot) +
  theme_classic() +
  labs(x = 'Costo Real',
       y = 'Densidad',
       title = 'Densidad estimada',
       subtitle = 'De las estimaciones bootstrap')
```

(d) Calcule el intervalo de 95 % confianza Bootstrap, el cual se obtiene a partir de:

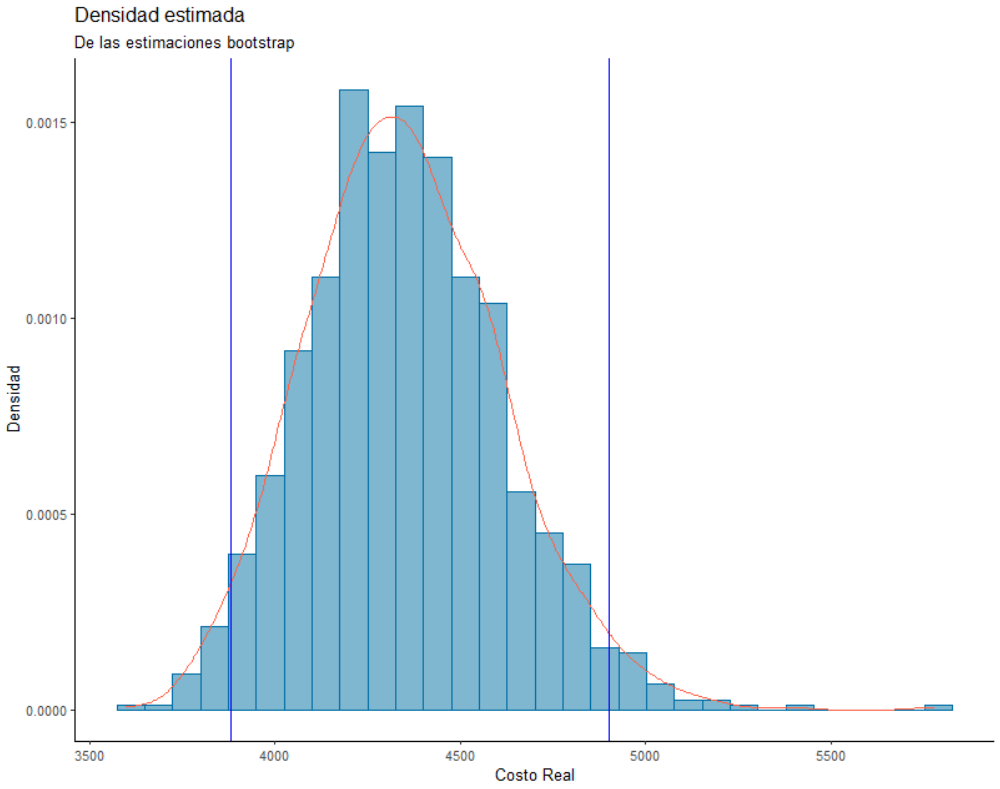
$$[\hat{F}^{-1}(\alpha/2), \hat{F}^{-1}(1-\alpha/2)]$$

Donde, $\widehat{F}^{-1}(\alpha)$ es el percentil α de la distribución estimada del estimador $\widehat{\theta}$.

En la tabla a continuación se presenta el intervalo de 95%:

Límite Inferior	3881.785
Límite Superior	4900.774

Para complementar se añade, respecto al gráfico, tanto el límite inferior como superior para su visualización:



Los comandos utilizados para la construcción de la tabla como el gráfico fueron los siguientes:

```
# Calculamos el límite inferior:
inferior_l = mean_cost_df$mean_cost %>%  quantile(.05/2)
inferior_l
# Calculamos el límite superior:
superior_l = mean_cost_df$mean_cost %>%  quantile(1-(.05/2))
superior_l
```

```
# Graficamos nuevamente el histograma pero con los límites
ggplot(data = mean_cost_df, mapping = aes(x = mean_cost)) +
  geom_histogram(aes(y =..density..), fill = '#006EA1', color =
'#006EA5', alpha = 0.5, bins = 30) +
  geom_density(color = 'tomato') +
  geom_vline(xintercept = inferior_l, color = 'blue') +
  geom_vline(xintercept = superior_l, color = 'blue') +
  theme_classic() +
  labs(x = 'Costo Real',
       y = 'Densidad',
       title = 'Densidad estimada',
       subtitle = 'De las estimaciones bootstrap')
```

(e) Calcule el intervalo de 95 % de confianza para θ utilizando la función t.test( ). ¿La construcción del intervalo t.test( ) asume algún supuesto sobre la distribución de $\widehat{\theta}$? Compare lo obtenido con el intervalo bootstrap. ¿Cuál intervalo le parece mejor para estimar θ? Justifique.

A continuación se presenta la tabla con los intervalos de 95% de confianza utilizando la función t.test( ):

Límite Inferior	3789.017
Límite Superior	4925.671

Con respecto a la construcción de este intervalo se tiene que t.test asume que la distribución de $\widehat{\theta}$ es normal. Es por esto que, con lo que respecta al mejor intervalo para estimar θ, bootstrap realiza un mejor trabajo respecto a la muestra trabajada dado que éste no asume la distribución sino que al realizar diversos muestreos de los datos permite estudiar la distribución real que esta presenta.

Por lo tanto, dado lo anterior, t-test es bastante robusto en el caso de que exista normalidad en la distribución o esté muy cerca de este; pero, como se muestra en este caso, no se presenta una evidencia clara de normalidad en los datos y, por consiguiente, bootstrap presenta mejores intervalos de confianza para la estimación de θ.

Los comandos utilizados para la construcción de la tabla como el gráfico fueron los siguientes:

```
t.test_val = (transit_cost$real_cost %>% t.test())$conf.int
## al realizar t.test se está asumiendo distribución normal
t.test_val
```

Pregunta 3:

Para los bancos es de gran importancia contar con un modelo predictivo para otorgar o no un crédito a un nuevo cliente. A partir de datos históricos de clientes, se registra la variable loan_status, que toma el valor 1 si el cliente incumplió con el pago, y 0 sino. A partir de las variables predictoras log_annual_income que corresponde al logaritmo natural de los ingresos anuales y home_ownership que tiene 4 niveles (Mortgage, Other, Own, Rent), se ajustó un modelo de regresión logística obteniéndose el siguiente modelo:

$$ln\left(\frac{\pi}{1-\pi}\right) = 0.565 + 0.084 \cdot \texttt{home\_ownershipOTHER} - 0.008 \cdot \texttt{home\_ownershipOWN} + 0.007 \cdot \texttt{home\_ownershipRENT} - 0.042 \cdot \texttt{log\_annual\_income}$$

(a) Calcule la probabilidad de caer en incumplimiento para un nuevo cliente que tiene casa propia y su salario anual en escala logarítmica es de 11 USD.

si *home_ownershipOWN* = 1 y *log_annual_income* = 11, entonces:

$$\ln\left(\frac{\pi}{1-\pi}\right) = 0,565 - 0,008 \cdot 1 - 0,042 \cdot 11$$
$$\ln\left(\frac{\pi}{1-\pi}\right) = 0,095$$
$$\left(\frac{\pi}{1-\pi}\right) = e^{0,095}$$
$$\pi = 1,099 \cdot (1-\pi)$$
$$\pi = 52,37\%$$

(b) ¿Cuánto debería ser el salario anual de un nuevo cliente que arrienda su propiedad, para que su probabilidad de caer en incumplimiento sea de 15 %?

Si π = 15% y salario anual = x, entonces:

$$\ln\left(\frac{15\%}{1-15\%}\right) = 0,565 + 0,007 \cdot 1 - 0,042 \cdot x$$
$$\ln(0,176) = 0,572 - 0,042 \cdot x$$
$$\frac{-1,73 - 0,572}{-0,042} = x$$
$$54,80 = x$$

Por lo que el salario anual, en escala logarítmica, debería ser de USD $54,80 para que la probabilidad de caer en incumplimiento sea de 15%.

(c) Se construye un clasificador binario utilizando el punto de corte 0.15. Con los datos con que se ajustó el modelo se calcula la matriz de confusión, obteniéndose:

	real	
pred	0	1
0	21864	2594
1	915	198

Calcule la precisión del modelo, la sensitividad y la especificidad.

$$Sensitividad = \frac{TP}{TP + FN}$$
$$Sensitividad = \frac{21864}{21864 + 915}$$
$$Sensitividad = 95,98\%$$

$$Especificidad = \frac{TN}{PF + TN}$$
$$Especificidad = \frac{2594}{2594 + 198}$$
$$Especificidad = 92,90\%$$

$$Precision = \frac{TP + TN}{total}$$
$$Precision = \frac{21864 + 198}{21864 + 2594 + 915 + 198}$$
$$Precision = 86,27\%$$

(d) Calcule la razón de chances de caer en incumplimiento para una persona que arrienda una propiedad versus una persona que es dueña de su propiedad. Interprete el resultado.

Razón de chances de persona que arrienda una propiedad:

$$RC_{arriendo} = e^{-0,007}$$
$$RC_{arriendo} = 0,9930$$

Razón de chances de una persona que es dueña de su propiedad:

$$RC_{duenaprop} = e^{-0,008}$$
$$RC_{duenaprop} = 0,9920$$

Podemos observar que ambas razones de chances son menores a 1. Esto significa que tanto la variable *home_ownershipRENT*, como la variable *home_ownershipOWN*, reducen la probabilidad de caer en incumplimiento. Por otra parte, podemos observar que $RC_{duenaprop}$ *es menor que* $RC_{arriendo}$, lo que indica que la variable *home_ownershipOWN* disminuye marginalmente más, la probabilidad de caer en incumplimiento que la variable *home_ownershipRENT*.
De igual forma al realizar el cálculo de la razón de chances obtiene que:

$$OR(home\_ownershipRENT, home\_ownershipOWN) = e^{-0.007-0,008}$$
$$OR(home\_ownershipRENT, home\_ownershipOWN) = 0.985$$

Valor que el aumento de estas variables indica que reduce las posibilidades de caer en incumplimiento.

Anexo 1:

Se entrega a continuación el código con las importaciones de librerías utilizadas en todo el proyecto junto a la importación de la base de datos:

```
## Librerías:
if (!require('corrr')) install.packages('corrr'); library(corrr)
if (!require('polycor')) install.packages('polycor'); library(polycor)
if(!require('ggpubr')) install.packages('ggpubr'); library(ggpubr)
if(!require('car')) install.packages('car'); library(car)
if (!require('tidyverse')) install.packages('tidyverse'); library('tidyverse')
#Incluye dyplr y ggplot
if (!require('tree')) install.packages('tree'); library('tree')       #Librería
de clases para hacer el árbol de decisión
if (!require('moments')) install.packages('moments'); library('moments')
if (!require('ggpubr')) install.packages('ggpubr'); library('ggpubr') #Para
juntar gráficos
if (!require('randomForest')) install.packages('randomForest');
library('randomForest') #Para hacer Bagging
if (!require('tidyverse')) install.packages('tidyverse'); library('tidyverse')
if (!require('corrplot')) install.packages('corrplot'); library('corrplot')
#Para gráfico de correlación
install.packages('caret')
if (!require('vcd')) install.packages('vcd'); library('vcd') #Para graficar
matriz de confusión en mosaico

library(ggplot2)
library(gamlss)
library(skimr)
library(dplyr)
library(readr)
library(reshape2)
library(gridExtra)
library(gt)
library(corrr)
library(readxl)
library(vcd)
library(ISLR)
library(caret)
library(scales)
library(DescTools)
library(ResourceSelection)
library(recipes)
library(ipred)
library(moments)

## Carga de BBDD:
transit_cost_full = read_excel("transit_cost.xlsx")
```

```
## Limpieza de BBDD:
transit_cost = transit_cost_full %>%
  dplyr::select(country, city, line, start_year, end_year, rr, length,
                tunnel_per, tunnel, stations, cost, currency,
                year, ppp_rate, real_cost, cost_km_millions) %>%
  dplyr::mutate(rr = dplyr::recode(rr, `0` = 'No', `1` = 'Yes')) %>%
  dplyr::mutate(across(where(is.character), as_factor)) %>%
  dplyr::mutate_if(is.numeric, list( ~ ifelse(is.na(.), 0, .)))

transit_cost$tunnel_per = transit_cost$tunnel / transit_cost$length *100

dplyr::glimpse(transit_cost)
```

Anexo 2:

Se entrega a continuación el código utilizado para el desarrollo de la pregunta 1:

```
##### Pregunta 1: ####
#### 1.a.) ####

###variables categoricas

## real_cost vs. country
ggplot(data = transit_cost, mapping = aes(country, real_cost)) +
  geom_point(color = '#006EA0', alpha = 0.25) +
  theme_classic() +
  labs(x = 'País',
       y = 'Costo real en millones de dólares',
       title = 'Relación País y costo real')

### real_cost v/s city
transit_cost %>%
  ggplot(aes(real_cost, city, color = "default")) +
  geom_point(alpha = 0.4) +
  scale_color_brewer(palette = "Set2") +
  labs(title = "Default status by income and balance")+
  theme_classic()+
  theme(axis.text.y = element_text(size = 6))            # y-axis text size

## real_cost vs. rr:
transit_cost %>%
  dplyr::count(rr) %>%
  dplyr::mutate(prop = n/sum(n))

ggplot(data = transit_cost, mapping = aes(x = rr, y = real_cost, fill = rr)) +
  geom_boxplot() +
  theme_light()+
  labs(x = 'Es un ferrocarril', y = 'Costo real en millones de dólares',
       title = 'Relación costo real y si es ferrocarril')

## real_cost vs. currency
plot8.1 = ggplot(data = transit_cost, mapping = aes(currency, real_cost)) +
  geom_point(color = '#006EA0', alpha = 0.25) +
  theme_light()+
  labs(x = 'Tipo de cambio monetario',
       y = 'Costo real en millones de dólares',
       title = 'Relación entre tipo de cambio monetario y costo real')
plot8.1

### Variables continuas ###
```

```
## real_cost vs. start_year
plot1 = ggplot(data = transit_cost, mapping = aes(start_year, real_cost)) +
  geom_point(color = '#006EA0', alpha = 0.25) +
  theme_classic() +
  stat_smooth(method = 'lm', se = FALSE, col = 'tomato', formula = y ~ x) +
  labs(x = 'Año iniciado',
       y = 'Costo real en millones de dólares',
       title = 'Relación año inicio y costo real')
plot1

transit_cost %>%
  dplyr::select(start_year, real_cost) %>%
  cor()
## correlación del 0.12

## real_cost vs. end_year
plot2 = ggplot(data = transit_cost, mapping = aes(end_year, real_cost)) +
  geom_point(color = '#006EA0', alpha = 0.25) +
  theme_classic() +
  stat_smooth(method = 'lm', se = FALSE, col = 'tomato', formula = y ~ x) +
    labs(x = 'Año termino',
       y = 'Costo real en millones de dólares',
       title = 'Relación año termino y costo real')
plot2

transit_cost %>%
  dplyr::select(end_year, real_cost) %>%
  cor()
## correlación del 0.21

## real_cost vs. length
plot3 = ggplot(data = transit_cost, mapping = aes(length, real_cost)) +
  geom_point(color = '#006EA0', alpha = 0.25) +
  theme_classic() +
  stat_smooth(method = 'lm', se = FALSE, col = 'tomato', formula = y ~ x) +
  labs(x = 'Largo de la linea (en km.)',
       y = 'Costo real en millones de dólares',
       title = 'Relación largo de la linea y costo real')
plot3

transit_cost %>%
  dplyr::select(length, real_cost) %>%
  cor()
## correlación del 0.73

## real_cost vs. tunnel_per
plot4 = ggplot(data = transit_cost, mapping = aes(tunnel_per, real_cost)) +
  stat_smooth(method = 'lm', se = FALSE, col = 'tomato', formula = y ~ x) +
  geom_point(color = '#006EA0', alpha = 0.25) +
  theme_classic() +
```

```
  labs(x = 'Porcentaje de la linea completada',
       y = 'Costo real en millones de dólares',
       title = 'Relación porcentaje de la linea completada y costo real')
plot4

transit_cost %>%
  dplyr::select(tunnel_per, real_cost) %>%
  cor()

## real_cost vs. tunnel
plot5 = ggplot(data = transit_cost, mapping = aes(tunnel, real_cost)) +
  geom_point(color = '#006EA0', alpha = 0.25) +
  stat_smooth(method = 'lm', se = FALSE, col = 'tomato', formula = y ~ x) +
  theme_classic() +
  stat_smooth(method = 'lm', se = FALSE, col = 'tomato', formula = y ~ x) +
  labs(x = 'Kilometros de linea completada',
       y = 'Costo real en millones de dólares',
       title = 'Relación entre Kilometros de linea completada y costo real')
plot5

transit_cost %>%
  dplyr::select(tunnel, real_cost) %>%
  cor()
## correlación del 0.69

## real_cost vs. stations
plot6 = ggplot(data = transit_cost, mapping = aes(stations, real_cost)) +
  geom_point(color = '#006EA0', alpha = 0.25) +
  theme_classic() +
  stat_smooth(method = 'lm', se = FALSE, col = 'tomato', formula = y ~ x) +
  labs(x = 'Número de estaciones donde los pasajeros pueden embarcar/salir',
       y = 'Costo real en millones de dólares',
       title = 'Relación entre número de estaciones donde los pasajeros pueden
embarcar/salir y costo real')
plot6

transit_cost %>%
  dplyr::select(stations, real_cost) %>%
  cor()
## correlación de 0.52

## real_cost vs. cost
plot7 = ggplot(data = transit_cost, mapping = aes(cost, real_cost)) +
  geom_point() +
  geom_point(color = '#006EA0', alpha = 0.25) +
  theme_classic()+
  stat_smooth(method = 'lm', se = FALSE, col = 'tomato', formula = y ~ x) +
  labs(x = 'Costo en millones de moneda local',
       y = 'Costo real en millones de dólares',
       title = 'Relación entre costo en millones de moneda local y costo real')
```

```
plot7

transit_cost %>%
  dplyr::select(cost, real_cost) %>%
  cor()
## correlación del 0.014

## real_cost vs. year
plot9 = ggplot(data = transit_cost, mapping = aes(year, real_cost)) +
  geom_point(color = '#006EA0', alpha = 0.25) +
  theme_classic()+
  stat_smooth(method = 'lm', se = FALSE, col = 'tomato', formula = y ~ x) +
  labs(x = 'Año medio de construcción',
       y = 'Costo real en millones de dólares',
       title = 'Relación entre año medio de construcción y costo real')
plot9

transit_cost %>%
  dplyr::select(year, real_cost) %>%
  cor()
## Correlación del 0.12

## real_cost vs. ppp_rate --> Hay 2 filas nulas

plot10 = ggplot(data = transit_cost, mapping = aes(ppp_rate, real_cost)) +
  geom_point(color = '#006EA0', alpha = 0.25) +
  theme_classic()+
  stat_smooth(method = 'lm', se = FALSE, col = 'tomato', formula = y ~ x) +
  labs(x = 'Paridad del poder adquisitivo',
       y = 'Costo real en millones de dólares',
       title = 'Relación entre paridad del poder adquisitivo y costo real')
plot10

transit_cost %>%
  dplyr::select(ppp_rate, real_cost) %>%
  cor()

## real_cost vs. cost_km_millions --> Hay 2 filas nulas
plot11 = ggplot(data = transit_cost, mapping = aes(cost_km_millions, real_cost))
+
  geom_point(color = '#006EA0', alpha = 0.25) +
  theme_classic()+
  stat_smooth(method = 'lm', se = FALSE, col = 'tomato', formula = y ~ x) +
  labs(x = 'Costo/km en millones de USD',
       y = 'Costo real en millones de dólares',
       title = 'Relación entre Costo/km en millones de USD y costo real')
plot11

transit_cost %>%
  dplyr::select(cost_km_millions, real_cost) %>%
```

```
  cor()


## Gráficos:
grid.arrange(plot1, plot2, plot3, plot4, nrow = 2, ncol = 2)
grid.arrange(plot5, plot6, plot7, plot9, nrow = 2, ncol = 2)
grid.arrange(plot10, plot11, nrow = 1, ncol = 2)

## Correlación policorica??
polycor::polychor(transit_cost$real_cost, transit_cost$country)
polycor::polychor(transit_cost$real_cost, transit_cost$city)
polycor::polychor(transit_cost$real_cost, transit_cost$rr)
polycor::polychor(transit_cost$real_cost, transit_cost$line)
polycor::polychor(transit_cost$real_cost, transit_cost$currency)


#### 1.b.) ####

transit_cost2 <- transit_cost[, c(4, 5, 7, 8, 9, 10, 11, 13, 14, 15, 16)]

x <- transit_cost2 %>%
  correlate() %>%
  focus(real_cost)
x
x %>%
  mutate(term = factor(term, levels = term[order(real_cost)])) %>%  # Order by
correlation strength
  ggplot(aes(x = term, y = real_cost)) +
  theme_light()+
  geom_bar(stat = "identity", fill = "#66d9ff") +
  ylab("Correlación con real_cost") +
  xlab("Variables") +
  ggtitle("Correlación de variables con real_cost")+
  geom_hline(yintercept = 0.3, col='tomato')

#### 1.c.) ####
set.seed(3707)

cost_split <- rsample::initial_split(transit_cost, prop = 0.75, strata =
stations)

## Los set de datos:
# Entrenamiento
data_train  = rsample::training(cost_split)

# Prueba
data_test  = rsample::testing(cost_split)

## Distribution of real_cost en:
# Data Train
```

```
data_train %>%
  dplyr::summarize(min  = min(real_cost),
            max = max(real_cost),
            mean  = mean(real_cost),
            sd  = sd(real_cost))

# Data Test
data_test %>%
  dplyr::summarize(min  = min(real_cost),
            max = max(real_cost),
            mean  = mean(real_cost),
            sd  = sd(real_cost))

## Se define el modelo:
lm_model  =  linear_reg() %>%
  set_engine("lm") %>%
  set_mode("regression")

## Se realiza la receta:
lm_recipe = recipe(real_cost ~ stations + tunnel + length, data = data_train)
%>%
  step_normalize(all_numeric(), -all_outcomes())

summary(lm_recipe)

## Se define el Workflow
lm_workflow =  workflow() %>%
  add_model(lm_model) %>%
  add_recipe(lm_recipe)


## Se realiza el fit del workflow
train_fit =  lm_workflow %>%
  fit(data = data_train)

## Ajuste del modelo de regresión lineal múltiple y sus coeficientes estimados:
tidy(train_fit)

#### Reporte del MAPE, R2
predictions = predict(train_fit, new_data = data_test)

test_results  =  data_test %>%
  bind_cols(predictions) %>%
  bind_cols(predict(train_fit, new_data = data_test, type = 'conf_int'))
head(test_results)

## RMSE y RSQ del testeo
test_results %>%
  rmse(estimate=.pred, truth= real_cost) ## RMSE de 2686.
```

```
test_results %>% mape(truth = real_cost, estimate=.pred) ## MAPE INF

mape2 = abs(test_results$real_cost - test_results$.pred)/test_results$real_cost
for(x in 1:length(mape2)){
  if(is.infinite(mape2[x])){
    mape2[x] = 0
  }
}
mean(mape2) ## MAPE omitiendo los infinitos es de 0.7772809


test_results %>%
  rsq(estimate = .pred, truth = real_cost) ## RSQ de 0.828.

## Gráfico de dispersión entre el costo real del proyecto y su valor predicho
test_results = test_results %>%
  ggplot(mapping = aes(x = .pred, y = real_cost)) +
  theme_light() +
  geom_point(color = '#006EA0', alpha = 0.40) +
  geom_abline(intercept = 0, slope = 1, color = 'tomato') +
  labs(title = 'Resultados de la Regresión Lineal - Test Set',
       x = 'Costos Reales predecidos',
       y = 'Costos Reales del proyecto')
test_results


## Ajustado nuevamente con 10-fold cross super validation 5000:
## Empezamos con la seed
set.seed(3707)

## Hacemos los folds
data_folds = vfold_cv(data_train, v=10, strata = stations)

doParallel::registerDoParallel()
ctrl_pred = control_resamples(save_pred = TRUE)
fit_folds = fit_resamples(lm_workflow, resamples = data_folds, control =
ctrl_pred)

head(fit_folds)

fit_folds %>%
  collect_predictions()

fit_folds %>%
  collect_metrics()

fit_folds %>% collect_predictions() %>%
  mape(estimate = .pred, truth = real_cost)
```

Anexo 3:

Se entrega a continuación el código utilizado para el desarrollo de la pregunta 2:

```
##### Pregunta 2: ####
#### 2.a) ####
set.seed(3770)

cost_boots = rsample::bootstraps(transit_cost, times = 1000, strata = stations)

#### 2.b.) ####
mean_cost = c()


for(x in 1:1000){
  val = (cost_boots$splits[[x]] %>% analysis())$real_cost
  mean_cost[x] =  val %>% mean()
}

#### 2.c #####
transit_cost$real_cost %>%
  mean()
transit_cost$real_cost %>%
  sd()

theta_boot = mean(mean_cost)
theta_boot

se_boot = sd(mean_cost)
se_boot

#### 2.d #####


mean_cost_df = data.frame(mean_cost)

ggplot(data = mean_cost_df, mapping = aes(x = mean_cost)) +
  geom_histogram(aes(y =..density..), fill = '#006EA1', color = '#006EA5', alpha
= 0.5, bins = 30) +
  geom_density(color = 'tomato') +
  geom_vline(xintercept = theta_boot) +
  theme_classic() +
  labs(x = 'Costo Real',
       y = 'Densidad',
       title = 'Densidad estimada',
       subtitle = 'De las estimaciones bootstrap')

# Calculamos el limite inferior:
inferior_l = mean_cost_df$mean_cost %>%  quantile(.05/2)
```

```
inferior_l
# Calculamos el límite superior:
superior_l = mean_cost_df$mean_cost %>%  quantile(1-(.05/2))
superior_l
# Graficamos nuevamente el histograma pero con los límites
ggplot(data = mean_cost_df, mapping = aes(x = mean_cost)) +
  geom_histogram(aes(y =..density..), fill = '#006EA1', color = '#006EA5', alpha
= 0.5, bins = 30) +
  geom_density(color = 'tomato') +
  geom_vline(xintercept = inferior_l, color = 'blue') +
  geom_vline(xintercept = superior_l, color = 'blue') +
  theme_classic() +
  labs(x = 'Costo Real',
       y = 'Densidad',
       title = 'Densidad estimada',
       subtitle = 'De las estimaciones bootstrap')

#### 2.e #####
t.test_val = (transit_cost$real_cost %>% t.test())$conf.int ## al realizar
t.test se esta asumiendo distribución normal
t.test_val
# Graficamos nuevamente el histograma con los limites y les añadimos los
intervalos del t.test
ggplot(data = mean_cost_df, mapping = aes(x = mean_cost)) +
  geom_histogram(aes(y =..density..), fill = '#006EA1', color = '#006EA5', alpha
= 0.5, bins = 30) +
  geom_density(color = 'tomato') +
  geom_vline(xintercept = t.test_val[1], color = 'red') +
  geom_vline(xintercept = t.test_val[2], color = 'red') +
  geom_vline(xintercept = inferior_l, color = 'blue') +
  geom_vline(xintercept = superior_l, color = 'blue') +
  theme_classic() +
  labs(x = 'Costo Real',
       y = 'Densidad',
       title = 'Densidad estimada',
       subtitle = 'De las estimaciones bootstrap')
```