Tidy Tuesday - IKEA Data Set

Arnab Panja

07/11/2020

Data Analysis

The Tidy Tuesday Week involved the IKEA Data Set. We show you how a typical data analysis of a data set progresses in R.

The data analysis starts with loading the data in an R dataframe. The below code snippet loads the IKEA data set and also shows the below two very important attributes of the data

1. Number of observations
2. Number of variables/features

```
names(ikea_data)
```

```
##  [1] "x1"                "item_id"           "name"
##  [4] "category"          "price"             "old_price"
##  [7] "sellable_online"   "link"              "other_colors"
## [10] "short_description" "designer"          "depth"
## [13] "height"            "width"
```

```
dim(ikea_data)
```

```
## [1] 3694   14
```

As we can see above there are 14 variables in the data set and 3694 observations.

Now let us see the first few records of the data set.

```
head(ikea_data)
```

```
## # A tibble: 6 x 14
##      x1 item_id name  category price old_price sellable_online link
##   <dbl>   <dbl> <chr> <chr>    <dbl> <chr>     <lgl>           <chr>
## 1     0  9.04e7 FREK~ Bar fur~   265 No old p~ TRUE            http~
## 2     1  3.69e5 NORD~ Bar fur~   995 No old p~ FALSE           http~
## 3     2  9.33e6 NORD~ Bar fur~  2095 No old p~ FALSE           http~
## 4     3  8.02e7 STIG  Bar fur~    69 No old p~ TRUE            http~
## 5     4  3.02e7 NORB~ Bar fur~   225 No old p~ TRUE            http~
## 6     5  1.01e7 INGO~ Bar fur~   345 No old p~ TRUE            http~
## # ... with 6 more variables: other_colors <chr>, short_description <chr>,
## #   designer <chr>, depth <dbl>, height <dbl>, width <dbl>
```

We now observe the below variables of the data set. The below variables are focussed as we would try and predict the prices of the furniture based on a few predictors.

```
## # A tibble: 6 x 6
##   category      name                    sellable_online depth height width
##   <chr>         <chr>                   <lgl>           <dbl>  <dbl> <dbl>
## 1 Bar furniture FREKVENS                TRUE               NA     99    51
## 2 Bar furniture NORDVIKEN               FALSE              NA    105    80
## 3 Bar furniture NORDVIKEN / NORDVIKEN FALSE              NA     NA    NA
## 4 Bar furniture STIG                    TRUE               50    100    60
## 5 Bar furniture NORBERG                 TRUE               60     43    74
## 6 Bar furniture INGOLF                  TRUE               45     91    40
```

As we can see the data has lots of NA values. Lets study them first.

```
rbind(ikea_data %>% count(depth, sort = TRUE) %>% filter(is.na(depth)) %>%
  mutate(type = "depth") %>% select(type, count_na = n),
ikea_data %>% count(height, sort = TRUE) %>% filter(is.na(height)) %>%
  mutate(type = "height") %>% select(type, count_na = n),
ikea_data %>% count(width, sort = TRUE) %>% filter(is.na(width)) %>%
  mutate(type = "width") %>% select(type, count_na = n))
```

```
## # A tibble: 3 x 2
##   type   count_na
##   <chr>     <int>
## 1 depth      1463
## 2 height      988
## 3 width       589
```

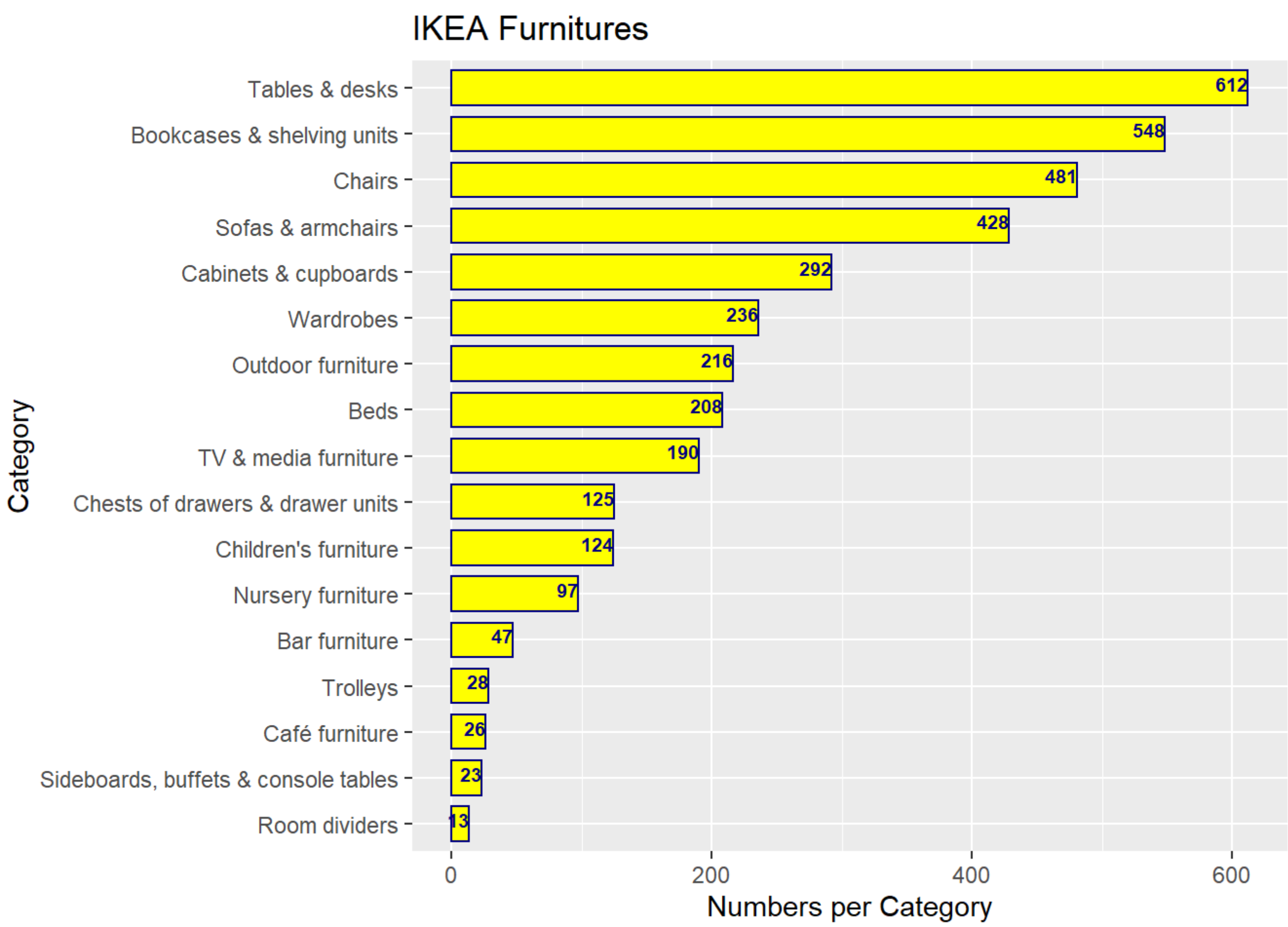
We have identified lots of observations with NA values. How we deal with NA values is a very interesting concern in its own right. Here we assume that not all furnitures will have all three dimensions and in some cases we will have 2 out of 3 dimensions and the third dimension may not be relevant at all. With this understanding (this is where domain knowledge plays a crucial role) lets try and replace the NA values in depth, height and width with zero.

```
ikea_data$height <- ifelse(is.na(ikea_data$height), 0, ikea_data$height)
ikea_data$width <- ifelse(is.na(ikea_data$width), 0, ikea_data$height)
ikea_data$depth <- ifelse(is.na(ikea_data$depth), 0, ikea_data$height)
```

The above piece of code quite succintly replaces all NA values with zeroes in height, depth and width.

So we now have 3694 observations to work with all having complete values in them.

```
ikea_data %>% count(category, sort = TRUE) %>%
  ggplot() +
  geom_col(mapping = aes(x = n,
                         y = reorder(category, n)),
           show.legend = FALSE,
           width = 0.75,
           fill = "yellow",
           color = "navyblue") +
  geom_text(mapping = aes(x = n,
                          y = reorder(category, n),
                          label = n),
            hjust = "top",
            nudge_y = 0.1,
            size = 2.5,
            color = "navyblue",
            fontface = "bold") +
  labs(x = "Numbers per Category",
       y = "Category",
       title = "IKEA Furnitures")
```



We , for the purposes of further study will concentrate on the furtniture category that has most number of observations.

```
max_cat <- ikea_data %>%
  group_by(category) %>%
  summarise(cnt = n(), .groups = "drop_last") %>%
  ungroup() %>%
  filter(cnt == max(cnt))

max_cat
```

```
## # A tibble: 1 x 2
##   category         cnt
##   <chr>          <int>
## 1 Tables & desks   612
```

So there are 612 Tables & desks that are the most frequent in this data set. Now lets create a smaller subset of data with only this furniture and create a regression model to predict the price of this category of the furniture based on the following 4 variables

1. Depth
2. Width
3. Height
4. Whether the Tables & desks is sellable online or not

So lets first create this smaller subset of data as below

```
ikea_data_sub <- ikea_data %>% select(item_id,
                                      category,
                                      sellable_online,
                                      depth,
                                      height,
                                      width,
                                      price) %>%
  filter(category == max_cat$category) %>%
  mutate(sellable_online = case_when(sellable_online == "TRUE" ~ 1,
                                     TRUE ~ 0))

head(ikea_data_sub)
```

```
## # A tibble: 6 x 7
##    item_id category       sellable_online depth height width price
##      <dbl> <chr>                    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1 19011777 Tables & desks               1     0     74    74   199
## 2 70466496 Tables & desks               1     0     28    28   245
## 3 60214159 Tables & desks               1    73     73    73   475
## 4 49011766 Tables & desks               1     0     72    72   179
## 5 59133593 Tables & desks               1     0     74    74   270
## 6 19216694 Tables & desks               1    56     56    56    80
```

What is the distribution of the sellable online marker? Here it is.

```
## # A tibble: 2 x 2
##   sellable_online     n
##             <dbl> <int>
## 1               1   608
## 2               0     4
```

Since most of the observations are sellable online, so we remove this from our prediction analysis. A variable that does not vary will not impact the response. With this argument we further narrow down the predictors by removing sellable online indicator.

```
ikea_data_sub <-  ikea_data_sub %>% select(item_id,
                                           height,
                                           width,
                                           depth,
                                           price)

head(ikea_data_sub)
```

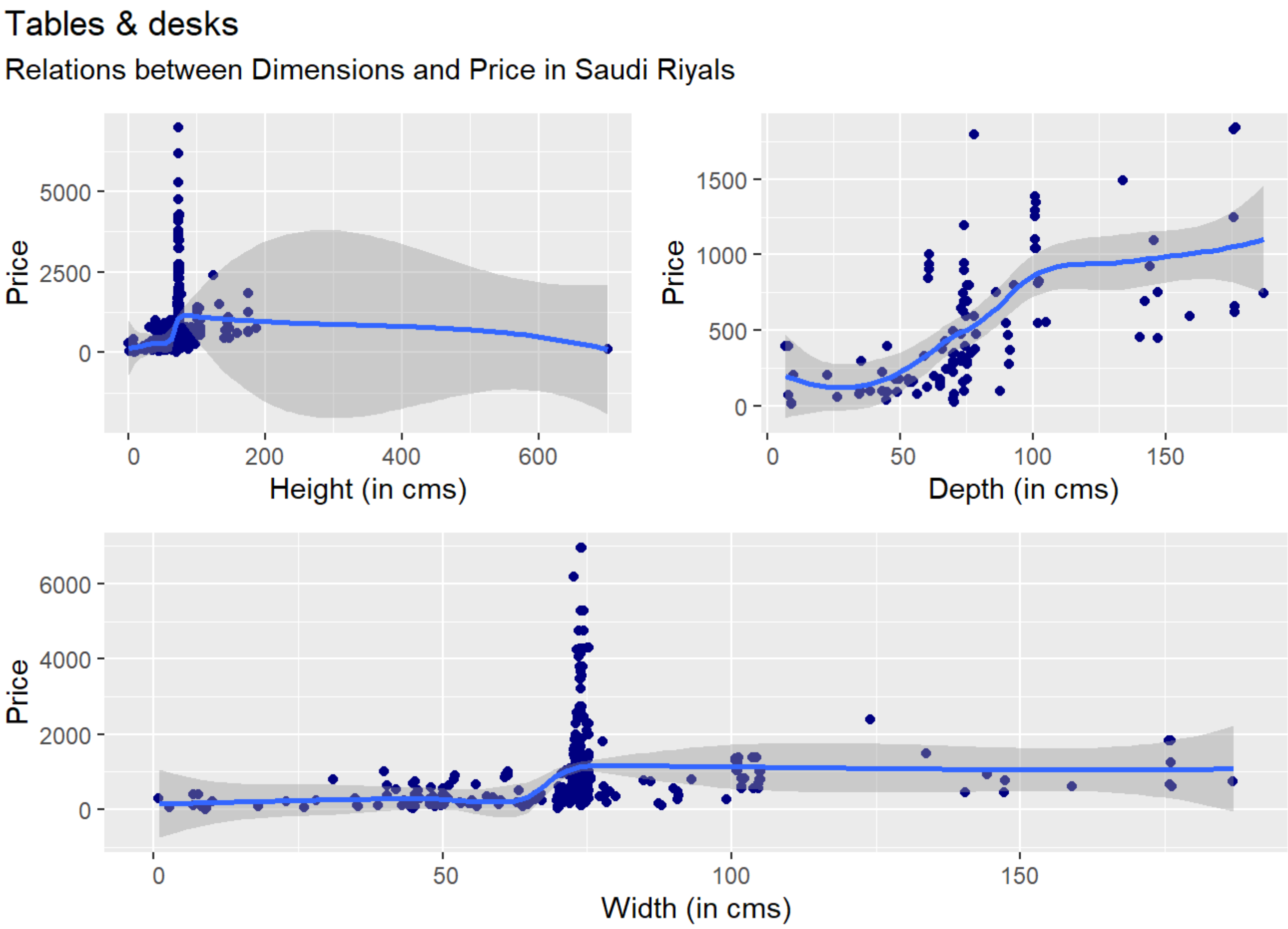
```
## # A tibble: 6 x 5
##    item_id height width depth price
##      <dbl>  <dbl> <dbl> <dbl> <dbl>
## 1 19011777     74    74     0   199
## 2 70466496     28    28     0   245
## 3 60214159     73    73    73   475
## 4 49011766     72    72     0   179
## 5 59133593     74    74     0   270
## 6 19216694     56    56    56    80
```

Let us now plot to see the relationship between the different dimensions and the price of the item.

```
p_height <- ikea_data_sub %>% filter(height != 0) %>%
  ggplot() +
  geom_point(mapping = aes(x = height, y = price),
             color = "navyblue",
             show.legend = FALSE, position = "jitter") +
  geom_smooth(mapping = aes(x = height, y = price),
              show.legend = FALSE,
              method = "loess",
              formula = "y ~ x") +
  scale_x_continuous(labels = function(x) format(x, scientific = FALSE)) +
  labs(x = "Height (in cms)",
       y = "Price")



p_width <- ikea_data_sub %>% filter(width != 0) %>%
  ggplot() +
  geom_point(mapping = aes(x = width, y = price),
             color = "navyblue",
             show.legend = FALSE, position = "jitter") +
  geom_smooth(mapping = aes(x = width, y = price),
              show.legend = FALSE,
              method = "loess",
              formula = "y ~ x") +
  scale_x_continuous(labels = function(x) format(x, scientific = FALSE)) +
  labs(x = "Width (in cms)",
       y = "Price")

p_depth <- ikea_data_sub %>% filter(depth != 0) %>%
  ggplot() +
  geom_point(mapping = aes(x = depth, y = price),
             color = "navyblue",
             show.legend = FALSE, position = "jitter") +
  geom_smooth(mapping = aes(x = depth, y = price),
              show.legend = FALSE,
              method = "loess",
              formula = "y ~ x") +
  scale_x_continuous(labels = function(x) format(x, scientific = FALSE)) +
  labs(x = "Depth (in cms)",
       y = "Price")
```

The patchwork package in R helps in combining more than one plots into a single plot as shown below.

```
(p_height + p_depth)/p_width +
  plot_annotation(title = paste0(max_cat$category),
                  subtitle = "Relations between Dimensions and Price in Saudi Riyals")
```



As we can see the 3 predictors and the repsonse do not really follow a linear relationship. The residuals of a linear model may be too high to reconcile with the actual response value. Also at width of 75 cms, height of 90 cms there is a large variation of price. This itself indicates there must be other predictors that also control the price of the furniture.

A non-linear model i.e. a decision tree or an ensemble of decision trees might be a better model of the response based on the predictors height, width and depth.

In the next section of the document we will build a decision tree model using the R package randomForest to predict the price of Tables & desks using the depth, width and height as the predictors.

Statistical Modelling - Random Forest/Bagging

We now split the data into a 80:20 training and test data set and create the X matrix and the response vector to be fed into the random forest model

```
set.seed(1234)

# convert the tibble into a data frame before modelling
ikea_data_sub <- as.data.frame(ikea_data_sub)

train <- sample(1:nrow(ikea_data_sub), round(0.8 * nrow(ikea_data_sub),
                                              digits = 0))

test <- setdiff(1:nrow(ikea_data_sub), train)

ikea_x <- model.matrix(object = price ~ .,
                       data = ikea_data_sub[train, -1])[, -1]

ikea_y <- as.vector(ikea_data_sub[train, "price"])

n_tree <- 64 # no of trees to be used for modelling
n_mtry <- 3 # no of predictors to be used to determine internal nodes
```

We now create the random forest model with 64 trees and all 3 predictors to decide the splits of the internal nodes in the decision tree.

```
rf_price_model <- randomForest(x = ikea_x,
                               y = ikea_y,
                               ntree = n_tree,
                               mtry = n_mtry,
                               importance = TRUE)
```

Now having created the random forest model with all 3 predictors we will use this model for predicting the price of the Tables & desks on the test data split. A random forest model using all 3 predictors is usually called bagging. So we have effectively created a bagging model to predict the price.

```
set.seed(1234)

ikea_data_test <- model.matrix(object = price ~ ., data = ikea_data_sub[test, -1])[, -1]


rf_price_predict <- predict(object = rf_price_model, newdata = ikea_data_test)
```

The predicted values of the price on the test data set is stored in the rf_price_predict vector. Now let us calculate the RMSE of the model on the test data. The MSE(Mean Squared Error), RMSE (Root Mean Squared Error) or the RMLSE (Root Mean Log Squared Error) are the common parameters to judge a statistical model. Here let us use the RMSE and the RMLSE to judge this model.

```
v_test_rmse <- round(sqrt(mean((round(rf_price_predict, digits = 0) - ikea_data_sub[test, "pr
ice"]) ^ 2)), digits = 2)



v_test_rmlse <- round(mltools::rmsle(round(rf_price_predict, digits = 0),
               ikea_data_sub[test, "price"]), digits = 2)

as.data.frame(rbind(c("param" = "rmse", "value" = v_test_rmse),
                    c("param" = "rmlse", "value" = v_test_rmlse)),
              stringsAsFactors = FALSE)
```

```
##   param   value
## 1  rmse 1085.26
## 2 rmlse    0.99
```

So the random forest model above has a test RMSE of 1085.26 and test test RMLSE of 0.99. The test RMSE is a high value indicating our model has not been great in predicting the price.

Let us see the importance of the predictors in the model.

```
importance(rf_price_model)
```

```
##          %IncMSE IncNodePurity
## height 31.095799     128808778
## width   8.189922      22068644
## depth  12.934303      13314493
```

As can be seen height has a high influence on the price. Width and Depth do not have that significant an impact. The Node Purity figures also indicate that including height gives a high node purity and better decision on the price. Width and Depth do not have that much of an impact on node purity.

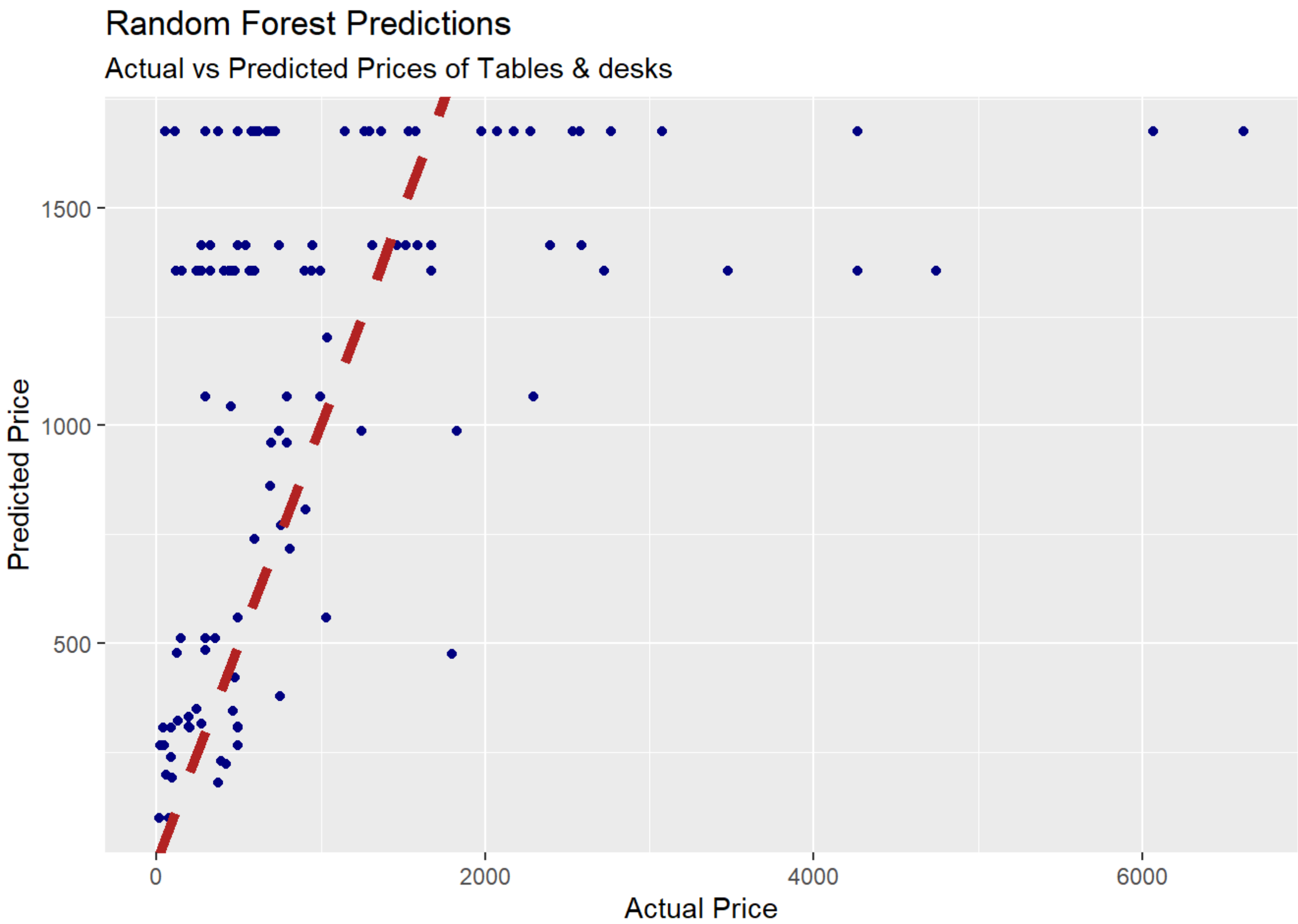
What does the above modelling indicate? There may be some other crucial predictors which we might have missed in the data set and hence in the model. What could be other crucial predictors not included here? the build material, the color, the designer and many other features could be having influence in predicting price of the furniture.

Let us observe the predicted values and actual values side by side for a visualization.

```
pred_actuals <- as.data.frame(cbind("pred_price" = round(rf_price_predict, digits = 0), "act_
price" = ikea_data_sub[test, "price"]), stringsAsFactors = FALSE)

head(pred_actuals)
```

```
##    pred_price act_price
## 3         421       475
## 5        1355       270
## 9         305        89
## 24       1355       119
## 27        305       495
## 32        308       195
```

```
p_pred_actuals <- ggplot(data = pred_actuals) +
  geom_point(mapping = aes(x = act_price, y = pred_price), color = "navyblue", show.legend =
FALSE) +
  geom_abline(mapping = aes(intercept = 0, slope = 1), show.legend = FALSE, color = "firebric
k", linetype = "dashed", size = 2) +
  labs(x = "Actual Price",
       y = "Predicted Price",
       title = "Random Forest Predictions",
       subtitle = paste0("Actual vs Predicted Prices of ", max_cat$category))

p_pred_actuals
```



The above graphic shows a very interesting fact. The red dashed line is the slope = 1 line. The model would have predicted well when most of the points are near to this line. But the random forest model that we developed is able to predict the prices when the actual prices are within the range of 800 - 900 Saudi Riyals. It is is in this range that the predicted and actual values are closer to the red dashed line. For Tables & desks having prices above this range the predicted prices are way off the mark.

So this is a demonstration of a data science project comprising the below tasks when presented with a new data set.

1. Loading the data
2. Analysing the data
3. Visualizing the data
4. Modelling, Predictions & Accuracy Measure
5. Reporting and Communications via R Markdown