Regression analysis

Shrinath Rajeshirke

02/12/2021

#importing libraries

```
library(rlang)
library(tidymodels)

## Registered S3 method overwritten by 'tune':
##   method                   from
##   required_pkgs.model_spec parsnip

## -- Attaching packages -------------------------------------- tidymodels
0.1.4 --

## v broom        0.7.10     v recipes      0.1.17
## v dials        0.0.10     v rsample      0.1.1
## v dplyr        1.0.7      v tibble       3.1.6
## v ggplot2      3.3.5      v tidyr        1.1.4
## v infer        1.0.0      v tune         0.1.6
## v modeldata    0.1.1      v workflows    0.2.4
## v parsnip      0.1.7      v workflowsets 0.1.0
## v purrr        0.3.4      v yardstick    0.0.9

## -- Conflicts -----------------------------------------
tidymodels_conflicts() --
## x purrr::%@%()         masks rlang::%@%()
## x purrr::as_function() masks rlang::as_function()
## x purrr::discard()     masks scales::discard()
## x dplyr::filter()      masks stats::filter()
## x purrr::flatten()     masks rlang::flatten()
## x purrr::flatten_chr() masks rlang::flatten_chr()
## x purrr::flatten_dbl() masks rlang::flatten_dbl()
## x purrr::flatten_int() masks rlang::flatten_int()
## x purrr::flatten_lgl() masks rlang::flatten_lgl()
## x purrr::flatten_raw() masks rlang::flatten_raw()
## x purrr::invoke()      masks rlang::invoke()
## x dplyr::lag()         masks stats::lag()
## x purrr::list_along()  masks rlang::list_along()
## x purrr::modify()      masks rlang::modify()
## x purrr::prepend()     masks rlang::prepend()
## x purrr::splice()      masks rlang::splice()
## x recipes::step()      masks stats::step()
## * Dig deeper into tidy modeling with R at https://www.tmwr.org

library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse
1.3.1 --

## v readr   2.1.0     v forcats 0.5.1
## v stringr 1.4.0

## -- Conflicts ------------------------------------------
tidyverse_conflicts() --
## x purrr::%@%()         masks rlang::%@%()
## x purrr::as_function() masks rlang::as_function()
## x readr::col_factor()  masks scales::col_factor()
## x purrr::discard()     masks scales::discard()
## x dplyr::filter()      masks stats::filter()
## x stringr::fixed()     masks recipes::fixed()
## x purrr::flatten()     masks rlang::flatten()
## x purrr::flatten_chr() masks rlang::flatten_chr()
## x purrr::flatten_dbl() masks rlang::flatten_dbl()
## x purrr::flatten_int() masks rlang::flatten_int()
## x purrr::flatten_lgl() masks rlang::flatten_lgl()
## x purrr::flatten_raw() masks rlang::flatten_raw()
## x purrr::invoke()      masks rlang::invoke()
## x dplyr::lag()         masks stats::lag()
## x purrr::list_along()  masks rlang::list_along()
## x purrr::modify()      masks rlang::modify()
## x purrr::prepend()     masks rlang::prepend()
## x readr::spec()        masks yardstick::spec()
## x purrr::splice()      masks rlang::splice()

library(stringr)
library(glmnet)

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

## Loaded glmnet 4.1-3

The `seoul_bike_sharing_converted_normalized.csv` will be our main dataset
which has following variables:

The response variable:

*   `RENTED BIKE COUNT`- Count of bikes rented at each hour

Weather predictor variables:
```

```
*   `TEMPERATURE` - Temperature in Celsius
*   `HUMIDITY` - Unit is `%`
*   `WIND_SPEED` - Unit is `m/s`
*   `VISIBILITY` - Multiplied by 10m
*   `DEW_POINT_TEMPERATURE` - The temperature to which the air would have to
cool down in order to reach saturation, unit is Celsius
*   `SOLAR_RADIATION` - MJ/m2
*   `RAINFALL` - mm
*   `SNOWFALL` - cm

Date/time predictor variables:

*   `DATE` - Year-month-day
*   `HOUR`- Hour of he day
*   `FUNCTIONAL DAY` - NoFunc(Non Functional Hours), Fun(Functional hours)
*   `HOLIDAY` - Holiday/No holiday
*   `SEASONS` - Winter, Spring, Summer, Autumn
```

#importing dataset

```
setwd("C:/Users/ASUS/Desktop/Weather Bike project")
bike_sharing_df <- read_csv("seoul_bike_sharing_converted_normalized.csv")

## Rows: 8465 Columns: 41

## -- Column specification -----------------------------------------------------
------
## Delimiter: ","
## chr  (2): DATE, FUNCTIONING_DAY
## dbl (39): RENTED_BIKE_COUNT, TEMPERATURE, HUMIDITY, WIND_SPEED,
VISIBILITY, ...

##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this
message.

spec(bike_sharing_df)

## cols(
##   DATE = col_character(),
##   RENTED_BIKE_COUNT = col_double(),
##   TEMPERATURE = col_double(),
##   HUMIDITY = col_double(),
##   WIND_SPEED = col_double(),
##   VISIBILITY = col_double(),
##   DEW_POINT_TEMPERATURE = col_double(),
##   SOLAR_RADIATION = col_double(),
##   RAINFALL = col_double(),
##   SNOWFALL = col_double(),
##   FUNCTIONING_DAY = col_character(),
```

```
##   `0` = col_double(),
##   `1` = col_double(),
##   `2` = col_double(),
##   `3` = col_double(),
##   `4` = col_double(),
##   `5` = col_double(),
##   `6` = col_double(),
##   `7` = col_double(),
##   `8` = col_double(),
##   `9` = col_double(),
##   `10` = col_double(),
##   `11` = col_double(),
##   `12` = col_double(),
##   `13` = col_double(),
##   `14` = col_double(),
##   `15` = col_double(),
##   `16` = col_double(),
##   `17` = col_double(),
##   `18` = col_double(),
##   `19` = col_double(),
##   `20` = col_double(),
##   `21` = col_double(),
##   `22` = col_double(),
##   `23` = col_double(),
##   AUTUMN = col_double(),
##   SPRING = col_double(),
##   SUMMER = col_double(),
##   WINTER = col_double(),
##   HOLIDAY = col_double(),
##   NO_HOLIDAY = col_double()
## )
```

We won't be using the DATE column, because 'as is', it basically acts like an data entry index. (However, given more time, we could use the DATE colum to create a 'day of week' or 'isWeekend' column, which we might expect has an affect on preferred bike rental times.) We also do not need the FUNCTIONAL DAY column because it only has one distinct value remaining (YES) after missing value processing.

```
bike_sharing_df <- bike_sharing_df %>%
                   select(-DATE, -FUNCTIONING_DAY)
```

Split training and testing data
First, we need to split the full dataset into training and testing datasets.

The training dataset will be used for fitting regression models, and the testing dataset will be used to evaluate the trained models.

```
set.seed(1234)
split <- initial_split(bike_sharing_df,prop = 3/4)
```

```
train_data <- training(split)
test_data <-testing(split)
```

Build a linear regression model using weather variables only
weather conditions may affect people's bike renting decisions.Thus, can we predict a city's bike-sharing demand based on its local weather information? Let's try to build a regression model to do that.

Building a linear regression model called lm_model_weather using the following variables:

TEMPERATURE - Temperature in Celsius
HUMIDITY - Unit is %
WIND_SPEED - Unit is m/s
VISIBILITY - Multiplied by 10m
DEW_POINT_TEMPERATURE - The temperature to which the air would have to cool down in order to reach saturation, unit is Celsius
SOLAR_RADIATION - MJ/m2
RAINFALL - mm
SNOWFALL - cm

```
lm <- linear_reg()
lm_model_weather <- lm(RENTED_BIKE_COUNT ~ TEMPERATURE + HUMIDITY +
WIND_SPEED + VISIBILITY + DEW_POINT_TEMPERATURE +
                         SOLAR_RADIATION + RAINFALL + SNOWFALL, train_data)
summary(lm_model_weather)
```

```
##
## Call:
## lm(formula = RENTED_BIKE_COUNT ~ TEMPERATURE + HUMIDITY + WIND_SPEED +
##     VISIBILITY + DEW_POINT_TEMPERATURE + SOLAR_RADIATION + RAINFALL +
##     SNOWFALL, data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1348.46  -294.03   -57.28   208.59  2329.78
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)               156.71      58.07   2.699  0.00698 **
## TEMPERATURE              2399.74     261.66   9.171  < 2e-16 ***
## HUMIDITY                 -918.38     126.79  -7.243  4.9e-13 ***
## WIND_SPEED                404.47      48.16   8.399  < 2e-16 ***
## VISIBILITY                 12.56      24.86   0.505  0.61351
## DEW_POINT_TEMPERATURE    -316.92     278.83  -1.137  0.25575
## SOLAR_RADIATION          -444.85      34.69 -12.824  < 2e-16 ***
## RAINFALL                -1764.01     182.65  -9.658  < 2e-16 ***
## SNOWFALL                  317.78     131.58   2.415  0.01576 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 487.3 on 6339 degrees of freedom
## Multiple R-squared:  0.4303, Adjusted R-squared:  0.4296
## F-statistic: 598.5 on 8 and 6339 DF,  p-value: < 2.2e-16
```

Building a linear regression model using all variables

```
lm <- linear_reg()
lm_model_all <- lm(RENTED_BIKE_COUNT ~ ., train_data)
summary(lm_model_all)
```

```
##
## Call:
## lm(formula = RENTED_BIKE_COUNT ~ ., data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1401.45  -218.96    -7.31   199.53  1780.67
##
## Coefficients: (3 not defined because of singularities)
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)               316.008     52.341   6.037 1.65e-09 ***
## TEMPERATURE               782.658    212.129   3.690 0.000227 ***
## HUMIDITY                 -886.730     99.492  -8.913  < 2e-16 ***
## WIND_SPEED                 31.913     40.275   0.792 0.428169
## VISIBILITY                 21.872     20.262   1.079 0.280439
## DEW_POINT_TEMPERATURE     598.387    221.369   2.703 0.006888 **
## SOLAR_RADIATION           276.882     41.466   6.677 2.64e-11 ***
## RAINFALL                -2064.638    143.276 -14.410  < 2e-16 ***
## SNOWFALL                  260.973    103.498   2.522 0.011709 *
## `0`                      -133.107     33.323  -3.994 6.56e-05 ***
## `1`                      -220.655     32.838  -6.719 1.98e-11 ***
## `2`                      -341.020     32.910 -10.362  < 2e-16 ***
## `3`                      -423.680     33.498 -12.648  < 2e-16 ***
## `4`                      -490.101     33.297 -14.719  < 2e-16 ***
## `5`                      -466.528     32.826 -14.212  < 2e-16 ***
## `6`                      -307.927     32.990  -9.334  < 2e-16 ***
## `7`                         2.949     33.207   0.089 0.929246
## `8`                       347.169     32.967  10.531  < 2e-16 ***
## `9`                      -103.808     33.853  -3.066 0.002175 **
## `10`                     -341.327     35.106  -9.723  < 2e-16 ***
## `11`                     -351.192     36.879  -9.523  < 2e-16 ***
## `12`                     -312.150     37.820  -8.253  < 2e-16 ***
## `13`                     -295.163     38.411  -7.684 1.77e-14 ***
## `14`                     -296.250     37.268  -7.949 2.21e-15 ***
## `15`                     -213.542     36.764  -5.808 6.61e-09 ***
## `16`                      -80.680     35.369  -2.281 0.022575 *
## `17`                      201.739     34.547   5.839 5.50e-09 ***
## `18`                      690.995     33.487  20.634  < 2e-16 ***
## `19`                      419.180     33.099  12.664  < 2e-16 ***
```

```
## `20`                       328.187     32.827   9.997  < 2e-16 ***
## `21`                       342.772     32.918  10.413  < 2e-16 ***
## `22`                       238.833     32.723   7.299 3.26e-13 ***
## `23`                            NA         NA      NA       NA
## AUTUMN                     358.999     20.290  17.694  < 2e-16 ***
## SPRING                     191.365     19.362   9.884  < 2e-16 ***
## SUMMER                     198.142     29.187   6.789 1.24e-11 ***
## WINTER                          NA         NA      NA       NA
## HOLIDAY                   -124.424     22.948  -5.422 6.11e-08 ***
## NO_HOLIDAY                      NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 377.9 on 6312 degrees of freedom
## Multiple R-squared:  0.6589, Adjusted R-squared:  0.657
## F-statistic: 348.4 on 35 and 6312 DF,  p-value: < 2.2e-16
```

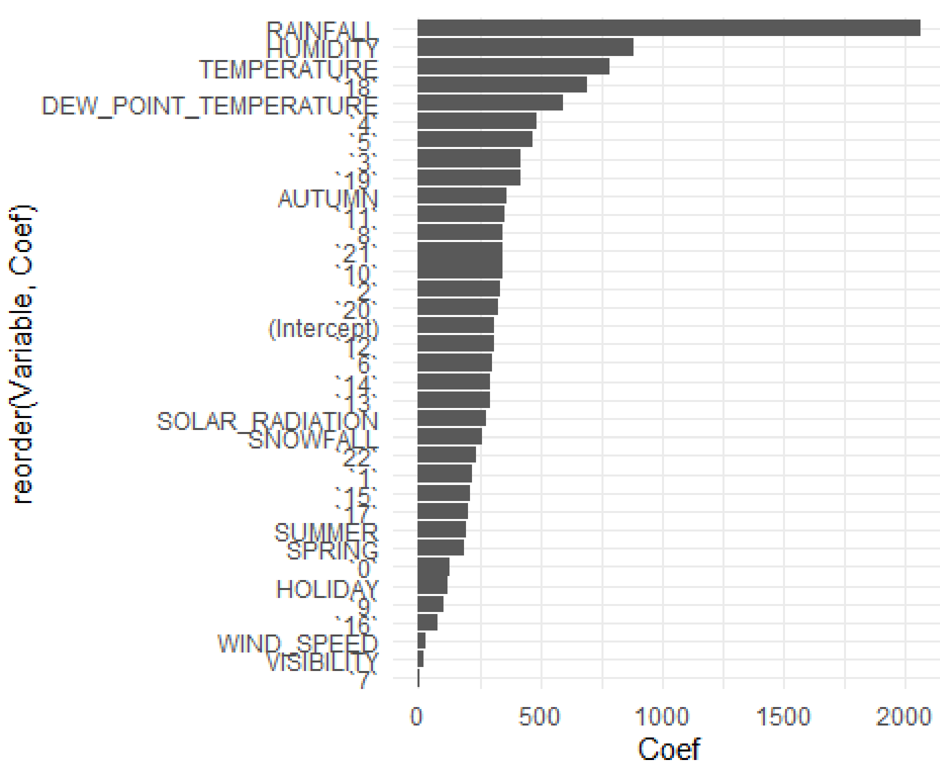
predictions on the testing dataset using both lm_model_weather and lm_model_all models

```
pred_weather <- predict(lm_model_weather,test_data)
pred_all <- predict(lm_model_all,test_data)

## Warning in predict.lm(lm_model_all, test_data): prediction from a rank-
deficient
## fit may be misleading

pred_df <-
data.frame(truth=test_data$RENTED_BIKE_COUNT,weather=pred_weather,all=pred_al
l)

rsq_weather <- rsq(pred_df,truth = truth,estimate=weather);rsq_weather

## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rsq     standard       0.439

rsq_all <- rsq(pred_df,truth = truth,estimate=all);rsq_all

## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rsq     standard       0.669

rmse_weather <- rmse(pred_df,truth=truth,estimate=weather);rmse_weather

## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard        475.
```

```r
rmse_all <- rmse(pred_df,truth=truth,estimate=all);rmse_all
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard        364.
```

```r
abs_cof_df <- stack(abs(lm_model_all$coefficients))
names(abs_cof_df) <- c("Coef", "Variable")
abs_cof_df <- abs_cof_df %>%
                select(Variable, Coef)
coefs_sorted <- arrange(abs_cof_df, -Coef)
coefs_sorted <- na.omit(coefs_sorted)

ggplot(data=coefs_sorted, aes(x= reorder(Variable,Coef),Coef)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  theme_minimal()
```

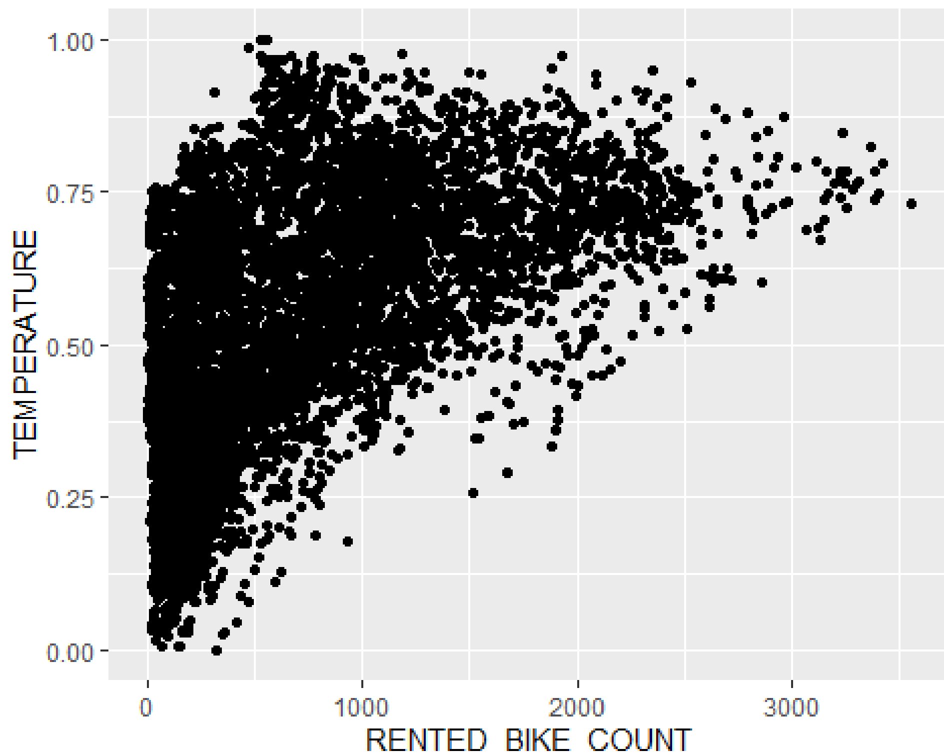



```
Define a linear regression model specification.

lm_spec <- linear_reg() %>%
  set_engine("lm") %>%
  set_mode("regression")

Split the data into training and testing datasets.
```

```
set.seed(1234)
data_split <- initial_split(bike_sharing_df, prop = 4/5)
train_data <- training(data_split)
test_data <- testing(data_split)
```

```
scatter plot of RENTED_BIKE_COUNT vs TEMPERATURE
```

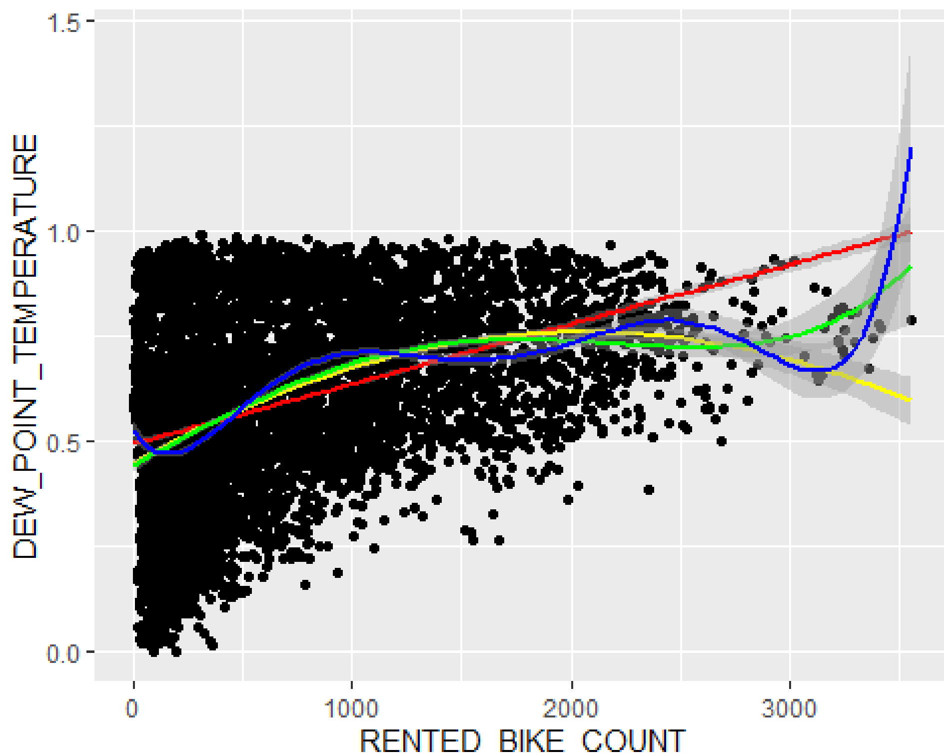
```
ggplot(data = train_data, aes(RENTED_BIKE_COUNT, TEMPERATURE)) +
    geom_point()
```




```
correlation between RENTED_BIKE_COUNT and TEMPERATURE does not look like
linear
```

```
plotting higher order polynomial fits
```

```
ggplot(data=train_data, aes(RENTED_BIKE_COUNT, DEW_POINT_TEMPERATURE)) +
    geom_point() +
    geom_smooth(method = "lm", formula = y ~ x, color="red") +
    geom_smooth(method = "lm", formula = y ~ poly(x, 2), color="yellow") +
    geom_smooth(method = "lm", formula = y ~ poly(x, 4), color="green") +
    geom_smooth(method = "lm", formula = y ~ poly(x, 6), color="blue")
```



From above plot, we can see that curve with degree 6 fits good.

Fitting of regreession model with popynomial terms

```
lm_poly <- lm(RENTED_BIKE_COUNT ~ .+ poly(TEMPERATURE,6) + poly(HUMIDITY,4) +
poly(RAINFALL,4),train_data)
summary(lm_poly)
```

```
##
## Call:
## lm(formula = RENTED_BIKE_COUNT ~ . + poly(TEMPERATURE, 6) + poly(HUMIDITY,
##     4) + poly(RAINFALL, 4), data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1417.15  -207.69    -5.18   188.89  1484.60
##
## Coefficients: (6 not defined because of singularities)
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -13.91      53.22  -0.261 0.793843
## TEMPERATURE               3365.40     252.96  13.304  < 2e-16 ***
## HUMIDITY                   354.62     122.09   2.904 0.003691 **
## WIND_SPEED                 -31.96      35.00  -0.913 0.361232
## VISIBILITY                 -73.55      18.39  -4.000 6.39e-05 ***
## DEW_POINT_TEMPERATURE    -2365.93     274.72  -8.612  < 2e-16 ***
## SOLAR_RADIATION            216.19      36.95   5.851 5.10e-09 ***
## RAINFALL                 -1740.24     141.59 -12.291  < 2e-16 ***
```

```
## SNOWFALL                  267.65      90.83   2.947 0.003221 **
## `0`                      -121.70      28.78  -4.228 2.39e-05 ***
## `1`                      -233.54      28.58  -8.170 3.64e-16 ***
## `2`                      -341.79      28.64 -11.935  < 2e-16 ***
## `3`                      -401.77      29.05 -13.830  < 2e-16 ***
## `4`                      -474.47      28.85 -16.444  < 2e-16 ***
## `5`                      -448.43      28.51 -15.731  < 2e-16 ***
## `6`                      -288.60      28.70 -10.057  < 2e-16 ***
## `7`                        20.23      28.80   0.703 0.482320
## `8`                       347.71      28.83  12.062  < 2e-16 ***
## `9`                       -82.36      29.38  -2.803 0.005078 **
## `10`                     -301.76      30.56  -9.874  < 2e-16 ***
## `11`                     -285.55      32.27  -8.850  < 2e-16 ***
## `12`                     -199.68      33.10  -6.032 1.70e-09 ***
## `13`                     -190.17      33.68  -5.646 1.71e-08 ***
## `14`                     -168.10      32.58  -5.160 2.54e-07 ***
## `15`                      -89.10      32.10  -2.775 0.005529 **
## `16`                       43.24      30.92   1.398 0.162033
## `17`                      314.45      29.97  10.491  < 2e-16 ***
## `18`                      762.01      29.25  26.053  < 2e-16 ***
## `19`                      469.12      28.65  16.374  < 2e-16 ***
## `20`                      353.67      28.50  12.411  < 2e-16 ***
## `21`                      356.48      28.63  12.453  < 2e-16 ***
## `22`                      243.42      28.53   8.531  < 2e-16 ***
## `23`                          NA         NA      NA       NA
## AUTUMN                    376.30      20.71  18.170  < 2e-16 ***
## SPRING                    232.01      20.19  11.491  < 2e-16 ***
## SUMMER                    262.29      26.21  10.009  < 2e-16 ***
## WINTER                        NA         NA      NA       NA
## HOLIDAY                  -107.01      20.02  -5.345 9.32e-08 ***
## NO_HOLIDAY                    NA         NA      NA       NA
## poly(TEMPERATURE, 6)1         NA         NA      NA       NA
## poly(TEMPERATURE, 6)2  -1729.48     506.38  -3.415 0.000641 ***
## poly(TEMPERATURE, 6)3 -10071.86     356.11 -28.283  < 2e-16 ***
## poly(TEMPERATURE, 6)4  -5281.72     383.68 -13.766  < 2e-16 ***
## poly(TEMPERATURE, 6)5    353.07     347.71   1.015 0.309935
## poly(TEMPERATURE, 6)6   1822.54     354.94   5.135 2.90e-07 ***
## poly(HUMIDITY, 4)1            NA         NA      NA       NA
## poly(HUMIDITY, 4)2     -8808.27     454.83 -19.366  < 2e-16 ***
## poly(HUMIDITY, 4)3     -3115.68     399.88  -7.792 7.62e-15 ***
## poly(HUMIDITY, 4)4      1152.09     441.68   2.608 0.009116 **
## poly(RAINFALL, 4)1            NA         NA      NA       NA
## poly(RAINFALL, 4)2      3050.42     366.77   8.317  < 2e-16 ***
## poly(RAINFALL, 4)3     -1903.41     355.98  -5.347 9.24e-08 ***
## poly(RAINFALL, 4)4      2611.87     348.94   7.485 8.06e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 338 on 6725 degrees of freedom
```

```
## Multiple R-squared:  0.727,  Adjusted R-squared:  0.7251
## F-statistic: 389.2 on 46 and 6725 DF,  p-value: < 2.2e-16
```

```
test_results <- predict(lm_poly,test_data)
```

```
## Warning in predict.lm(lm_poly, test_data): prediction from a rank-
deficient fit
## may be misleading
```

```
#minor improvement we could do here is to convert all negative prediction
results to zero, because we can not have negative rented bike counts
test_results[test_results<0] <- 0
df <- data.frame(truth=test_data$RENTED_BIKE_COUNT,estimate=test_results)

rmse_poly <- rmse(df,truth =truth,estimate=estimate);rmse_poly
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard        317.
```

```
rsq_poly <- rsq(df,truth,estimate);rsq_poly
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rsq     standard       0.753
```

Fitting of regression model with interaction and polynomial terms

```
lm_poly2 <- lm(RENTED_BIKE_COUNT ~ .+ RAINFALL*HUMIDITY +poly(TEMPERATURE,6)
+ poly(HUMIDITY,4) + poly(DEW_POINT_TEMPERATURE,4) ,train_data)
summary(lm_poly2)
```

```
##
## Call:
## lm(formula = RENTED_BIKE_COUNT ~ . + RAINFALL * HUMIDITY +
poly(TEMPERATURE,
##     6) + poly(HUMIDITY, 4) + poly(DEW_POINT_TEMPERATURE, 4),
##     data = train_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1560.10  -203.91     5.57   184.69  1467.53
##
## Coefficients: (6 not defined because of singularities)
##                                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)                          178.70      52.34   3.414 0.000643 ***
## TEMPERATURE                         2160.10     255.57   8.452  < 2e-16 ***
## HUMIDITY                            -286.80     122.47  -2.342 0.019223 *
## WIND_SPEED                            13.76      34.66   0.397 0.691278
## VISIBILITY                           -30.78      18.23  -1.689 0.091299 .
```

```
## DEW_POINT_TEMPERATURE              -1007.46     278.05  -3.623 0.000293 ***
## SOLAR_RADIATION                      -24.74      38.49  -0.643 0.520293    
## RAINFALL                          -40976.73    5378.30  -7.619 2.91e-14 ***
## SNOWFALL                             -66.83      91.86  -0.728 0.466927    
## `0`                                 -118.91      28.24  -4.211 2.58e-05 ***
## `1`                                 -230.91      28.04  -8.236  < 2e-16 ***
## `2`                                 -339.86      28.10 -12.097  < 2e-16 ***
## `3`                                 -407.57      28.50 -14.301  < 2e-16 ***
## `4`                                 -478.16      28.31 -16.893  < 2e-16 ***
## `5`                                 -452.69      27.97 -16.187  < 2e-16 ***
## `6`                                 -297.99      28.16 -10.580  < 2e-16 ***
## `7`                                   19.96      28.25   0.707 0.479834    
## `8`                                  367.22      28.29  12.980  < 2e-16 ***
## `9`                                  -37.30      28.96  -1.288 0.197720    
## `10`                                -228.46      30.28  -7.544 5.16e-14 ***
## `11`                                -195.74      32.08  -6.101 1.11e-09 ***
## `12`                                -102.01      32.96  -3.095 0.001979 ** 
## `13`                                 -94.12      33.43  -2.815 0.004886 ** 
## `14`                                 -88.87      32.28  -2.753 0.005924 ** 
## `15`                                 -29.04      31.68  -0.917 0.359399    
## `16`                                  78.23      30.40   2.573 0.010089 *  
## `17`                                 320.14      29.41  10.886  < 2e-16 ***
## `18`                                 759.43      28.70  26.464  < 2e-16 ***
## `19`                                 464.00      28.11  16.507  < 2e-16 ***
## `20`                                 351.31      27.95  12.569  < 2e-16 ***
## `21`                                 352.89      28.07  12.570  < 2e-16 ***
## `22`                                 240.73      27.99   8.601  < 2e-16 ***
## `23`                                     NA         NA      NA       NA    
## AUTUMN                               348.34      20.40  17.080  < 2e-16 ***
## SPRING                               204.65      19.89  10.288  < 2e-16 ***
## SUMMER                               305.45      25.85  11.817  < 2e-16 ***
## WINTER                                   NA         NA      NA       NA    
## HOLIDAY                             -108.30      19.67  -5.504 3.84e-08 ***
## NO_HOLIDAY                               NA         NA      NA       NA    
## poly(TEMPERATURE, 6)1                    NA         NA      NA       NA    
## poly(TEMPERATURE, 6)2               7461.89     734.43  10.160  < 2e-16 ***
## poly(TEMPERATURE, 6)3             -10554.08     473.81 -22.275  < 2e-16 ***
## poly(TEMPERATURE, 6)4              -7065.41     448.97 -15.737  < 2e-16 ***
## poly(TEMPERATURE, 6)5                159.15     375.47   0.424 0.671684    
## poly(TEMPERATURE, 6)6               1529.21     349.04   4.381 1.20e-05 ***
## poly(HUMIDITY, 4)1                       NA         NA      NA       NA    
## poly(HUMIDITY, 4)2                 -5337.17     506.55 -10.536  < 2e-16 ***
## poly(HUMIDITY, 4)3                 -3689.63     385.42  -9.573  < 2e-16 ***
## poly(HUMIDITY, 4)4                 -2141.01     454.49  -4.711 2.52e-06 ***
## poly(DEW_POINT_TEMPERATURE, 4)1          NA         NA      NA       NA    
## poly(DEW_POINT_TEMPERATURE, 4)2 -13110.24     752.82 -17.415  < 2e-16 ***
## poly(DEW_POINT_TEMPERATURE, 4)3  -2380.16     475.49  -5.006 5.71e-07 ***
## poly(DEW_POINT_TEMPERATURE, 4)4    915.95     428.53   2.137 0.032601 *  
## HUMIDITY:RAINFALL                  40345.40    5455.12   7.396 1.58e-13 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 331.5 on 6724 degrees of freedom
## Multiple R-squared:  0.7373, Adjusted R-squared:  0.7354
## F-statistic: 401.5 on 47 and 6724 DF,  p-value: < 2.2e-16
```

```
test_results2 <- predict(lm_poly2,test_data)
```

```
## Warning in predict.lm(lm_poly2, test_data): prediction from a rank-
deficient fit
## may be misleading
```

```
#minor improvement we could do here is to convert all negative prediction
results to zero, because we can not have negative rented bike counts
test_results2[test_results2<0] <- 0
df1 <- data.frame(truth=test_data$RENTED_BIKE_COUNT,estimate=test_results2)
```

```
rmse_poly2 <- rmse(df1,truth =truth,estimate=estimate);rmse_poly2
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard        314.
```

```
rsq_poly2 <- rsq(df1,truth,estimate);rsq_poly2
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rsq     standard       0.757
```