Predicting electrical power generation of a combined gas & steam turbine

This dataset was obtained from the UCI repository:
https://archive.ics.uci.edu/ml/datasets/Combined+Cycle+Power+Plant

The goal of this study is to present two machine learning algorithms, the Multivariate Linear Regression and the Random Forest models to predict the electrical energy output (PE) generation per hour, in a combined gas and steam turbine power plant, as response variable.

The predictive variables are composed by the ambient temperature (AT), ambient pressure (AP) and relative humidity (RH), which are environmental variables. The equipment represented by the exhaust vacuum (V) variable measures the pressure related on the turbine work process in the Rankine cycle, creating pressure difference on the last stages of the turbine expansion, avoiding the water formation on the turbine blades that could cause erosion on it.

To evaluate and compare the performance for each machine learning model, the R_squared and the Root Mean Square Error (RMSE) values were calculated, as well the mean relative error (%) between the predicted and the actual values.

Attribute Information

Features consist of hourly average ambient variables:

- Ambient Temperature(AT) range 1.81 to 37.11°C
- Ambient Pressure(AP) range 992.89 to 1033.30 milibar
- Relative Humidity(RH) range 25.56 to 100.16%
- Exhaust Vacuum(V) range 25.36 to 81.56 cm Hg
- Net hourly electrical energy output(PE) range 420.26 to 495.76 MW

Part 1 - Predicting output variable by using multivariate linear regression model

Importing and naming the data

```r
library(readxl)

outpwr <- read_excel('Cycle power plant dataset.xlsx')
```

Analyzing the data

```
str(outpwr)

## Classes 'tbl_df', 'tbl' and 'data.frame':    9568 obs. of  5 variables:
##  $ AT: num  14.96 25.18 5.11 20.86 10.82 ...
##  $ V : num  41.8 63 39.4 57.3 37.5 ...
##  $ AP: num  1024 1020 1012 1010 1009 ...
##  $ RH: num  73.2 59.1 92.1 76.6 96.6 ...
##  $ PE: num  463 444 489 446 474 ...
```

```
summary(outpwr)

##        AT               V               AP               RH        
##  Min.   : 1.81   Min.   :25.36   Min.   : 992.9   Min.   : 25.56  
##  1st Qu.:13.51   1st Qu.:41.74   1st Qu.:1009.1   1st Qu.: 63.33  
##  Median :20.34   Median :52.08   Median :1012.9   Median : 74.97  
##  Mean   :19.65   Mean   :54.31   Mean   :1013.3   Mean   : 73.31  
##  3rd Qu.:25.72   3rd Qu.:66.54   3rd Qu.:1017.3   3rd Qu.: 84.83  
##  Max.   :37.11   Max.   :81.56   Max.   :1033.3   Max.   :100.16  
##        PE       
##  Min.   :420.3  
##  1st Qu.:439.8  
##  Median :451.6  
##  Mean   :454.4  
##  3rd Qu.:468.4  
##  Max.   :495.8  
```

```
cor(outpwr)

##            AT          V          AP          RH         PE
## AT  1.0000000  0.8441067 -0.50754934 -0.54253465 -0.9481285
## V   0.8441067  1.0000000 -0.41350216 -0.31218728 -0.8697803
## AP -0.5075493 -0.4135022  1.00000000  0.09957432  0.5184290
## RH -0.5425347 -0.3121873  0.09957432  1.00000000  0.3897941
## PE -0.9481285 -0.8697803  0.51842903  0.38979410  1.0000000
```

Splitting the data in train and test sets

```
set.seed(1)
n <- nrow(outpwr)
shuffled <- outpwr[sample(n),]
train_indices <- 1:round(0.7 * n)
train <- shuffled[train_indices, ]
test_indices <- (round(0.7 * n) + 1):n
test <- shuffled[test_indices, ]
```

Training the model, predicting the output and summarizing the statistics

```
model.lm <- lm(PE ~., train)
test$pred.lm <- predict(model.lm, test)
head(test)

## # A tibble: 6 x 6
##      AT     V    AP    RH    PE pred.lm
##   <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl>
## 1  29.3  70.0  1010  46.9   439     436
## 2  19.7  56.6  1021  72.1   455     454
## 3  28.3  68.7  1006  69.9   435     434
## 4  25.2  68.5  1013  72.3   441     440
## 5  10.9  40.1  1014  91.4   478     472
## 6  10.5  41.9  1017  92.8   480     473
```

```
rmse <- sqrt((1/nrow(test)) * sum((test$PE - test$pred.lm) ^ 2))
rmse

## [1] 4.588754
```

```
summary(model.lm)

## 
## Call:
## lm(formula = PE ~ ., data = train)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -43.033  -3.157  -0.122   3.203  17.814 
## 
## 
```

```
Coefficients:
##                Estimate Std. Error  t value Pr(>|t|)
## (Intercept) 457.211664  11.722397   39.003  < 2e-16 ***
## AT           -1.978739   0.018459 -107.199  < 2e-16 ***
## V            -0.236140   0.008765  -26.943  < 2e-16 ***
## AP            0.059726   0.011371    5.252 1.55e-07 ***
## RH           -0.159173   0.005019  -31.711  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.545 on 6693 degrees of freedom
## Multiple R-squared:  0.9292, Adjusted R-squared:  0.9292
## F-statistic: 2.197e+04 on 4 and 6693 DF,  p-value: < 2.2e-16
```

Calculating the mean relative error (%) between the predicted and the actual values

```
library(dplyr)

test %>%
  select(PE, pred.lm) %>%
  summarise(error = mean((abs(PE - pred.lm)/PE)*100))

## # A tibble: 1 x 1
##   error
##   <dbl>
## 1 0.806
```

Part 2 - Predicting output variable by using the Random Forest model

Training the model, predicting the output and summarizing the statistics

```
library(party)

model.rf <- cforest(PE ~., data = train, controls = cforest_unbiased(
                    ntree = 500, mtry = 2))

cforestStats(model.rf)

##      RMSE  Rsquared       MAE
## 3.6458564 0.9544987 2.7219670
```

```
varImp(model.rf)

##       Overall
## AT 228.254304
## V   59.914796
## AP   6.830999
## RH   4.642428

test$pred.rf <- predict(model.rf, newdata = test, type = 'response')
head(test)

## # A tibble: 6 x 7
##      AT     V    AP    RH    PE pred.lm pred.rf
##   <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl>   <dbl>
## 1  29.3  70.0  1010  46.9   439     436     437
## 2  19.7  56.6  1021  72.1   455     454     456
## 3  28.3  68.7  1006  69.9   435     434     434
## 4  25.2  68.5  1013  72.3   441     440     438
## 5  10.9  40.1  1014  91.4   478     472     476
## 6  10.5  41.9  1017  92.8   480     473     476
```

Calculating the mean relative error (%) between the predicted and the actual values

```
test %>%
  select(PE, pred.rf) %>%
  summarise(error = mean((abs(PE - pred.rf)/PE)*100))

## # A tibble: 1 x 1
##   error
##   <dbl>
## 1 0.595
```

Conclusion

For the proposal of this study to predict the electrical output power generation, both machine learning models, the Linear Regression and the Random Forest presented efficient results with the mean relative error results being below 1%, and it was revealed the major predictive variables, the temperature and exhaust vacuum in this order of importance.

Despite the slightly better result presented by the Random Forest model, the performance and the time to process and to fit the model were much higher compared to Linear Regression, even working in a not big dataset and with few variables involved.