Regression

Aloksai Choudari, Alekhya Pinnamaneni

2022-10-17

Select a data set

Data set: Power Consumption of Tetouan city

Source: https://archive.ics.uci.edu/ml/datasets/Power+consumption+of+Tetouan+city#

Attributes: DateTime, Temperature, Humidity, Wind Speed, general diffuse flows, diffuse flows, Zone 1 Power Consumption, Zone 2 Power Consumption, Zone 3 Power Consumption

No. of instances (rows): 52,417 rows

Load the data

```
power <- read.csv("power.csv")
```

Clean the data

```
# Remove the DateTime column
power <- power[-c(1)]

# Sum the last 3 columns into one target column 'Total Power Consumption'
power$Total.Power.Consumption <- power$Zone.1.Power.Consumption + power$Zone.2..Power.Consumption + powe
power <- power[-c(6:8)]
```

Divide data into train, test, and validate sets

```
set.seed(1234)
spec <- c(train=.7, test=.2, validate=.1)
i <- sample(cut(1:nrow(power),
                nrow(power)*cumsum(c(0,spec)), labels=names(spec)))
train <- power[i=="train",]
test <- power[i=="test",]
val <- power[i=="validate",]
```

Explore the data statistically and graphically

```r
# Outputs the first 5 rows of the train data
head(train, n=5)
```

```
##   Temperature Humidity Wind.Speed general.diffuse.flows diffuse.flows
## 4       6.121     75.0      0.083                  0.091         0.096
## 5       5.921     75.7      0.081                  0.048         0.085
## 6       5.853     76.9      0.081                  0.059         0.108
## 7       5.641     77.7      0.080                  0.048         0.096
## 8       5.496     78.2      0.085                  0.055         0.093
##   Total.Power.Consumption
## 4                65489.23
## 5                63650.45
## 6                62171.34
## 7                60937.36
## 8                59566.75
```

```r
# Outputs the mean temperature
mean(train$Temperature)
```

```
## [1] 18.83691
```

```r
# Outputs the lowest and highest temperatures
range(train$Temperature)
```

```
## [1]  3.247 39.760
```

```r
# Outputs the mean humidity
mean(train$Humidity)
```

```
## [1] 68.24749
```

```r
# Outputs the lowest and highest humidity
range(train$Humidity)
```

```
## [1] 11.34 94.80
```

```r
# Outputs the median humidity
median(train$Humidity)
```

```
## [1] 69.86
```

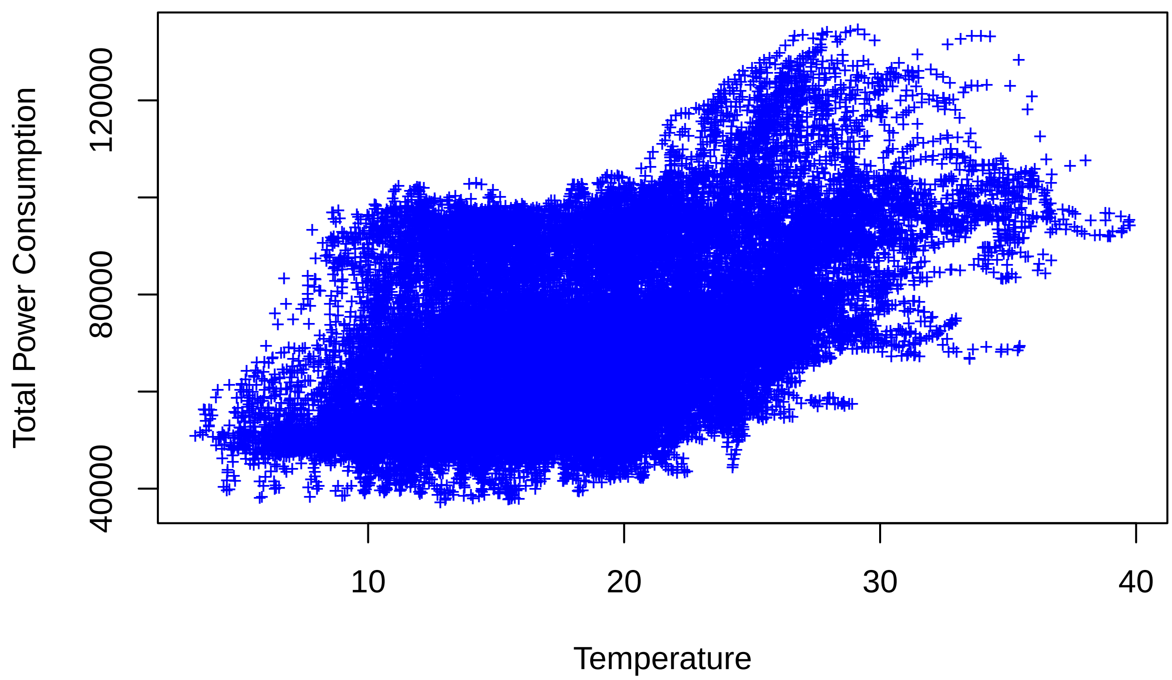
```r
# Outputs the mean wind speed
mean(train$Wind.Speed)
```

```
## [1] 1.966005
```

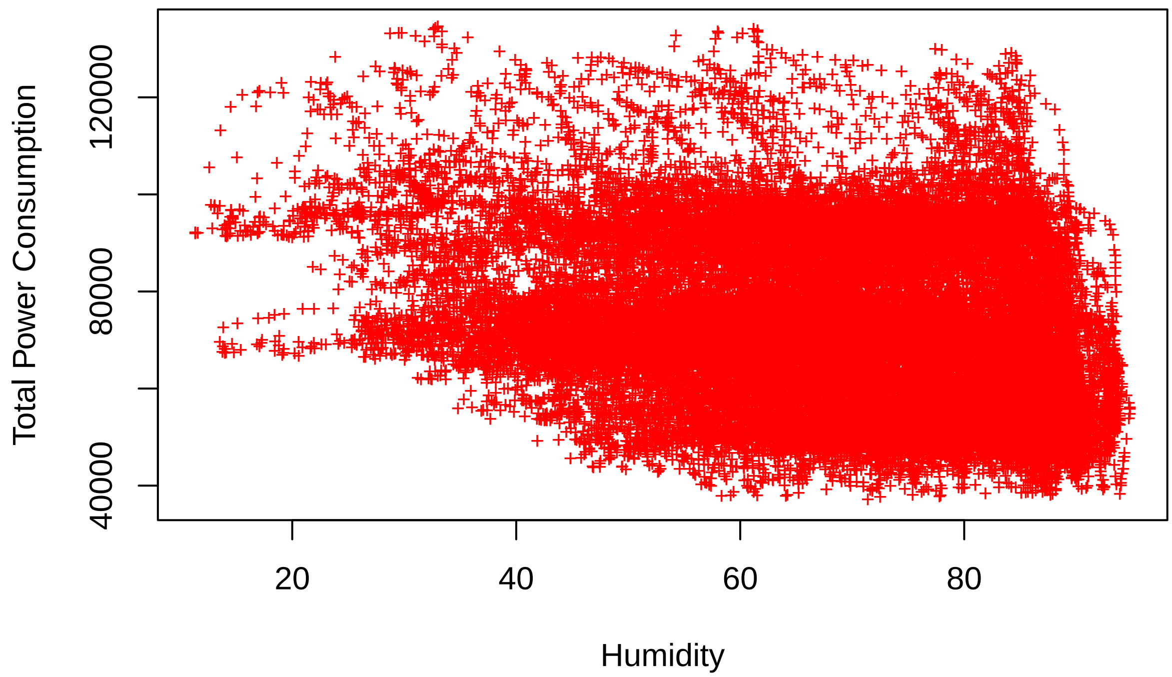
```r
# Outputs statistics for Total Power Consumption
summary(train$Total.Power.Consumption)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   36785   56552   69823   71273   83756  134208
```

```r
# Outputs a scatterplot of temperature vs. total power consumption
plot(train$Temperature, train$Total.Power.Consumption, pch='+', cex=0.75, col="blue", xlab="Temperature
```




```r
# Outputs a scatterplot of humidity vs. total power consumption
plot(train$Humidity, train$Total.Power.Consumption, pch='+', cex=0.75, col="red", xlab="Humidity", ylab=
```



SVM Regression

Linear Kernel

```
library(e1071)
library(MASS)
svm1 <- tune(svm, Total.Power.Consumption~., data=val, kernel="linear", ranges=list(cost=c(0.1, 10, 100)
summary(svm1)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##   100
##
## - best performance: 232918636
##
## - Detailed performance results:
##    cost     error dispersion
## 1   0.1 233027531   17516457
```

```
## 2  10.0 232939426   17550706
## 3 100.0 232918636   17530283
```

```r
pred <- predict(svm1$best.model, newdata=test)
cor_svm1 <- cor(pred, test$Total.Power.Consumption)
print(paste("correlation = ", cor_svm1))
```

```
## [1] "correlation =  0.496819053897192"
```

```r
mse_svm1 <- mean((pred - test$Total.Power.Consumption)^2)
print(paste("mean squared error = ", mse_svm1))
```

```
## [1] "mean squared error =  228268776.37207"
```

Polynomial Kernel

```r
svm2 <- tune(svm, Total.Power.Consumption~., data=val, kernel="polynomial", ranges=list(cost=c(0.1, 1, 1
summary(svm2)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##     1
##
## - best performance: 219067517
##
## - Detailed performance results:
##   cost     error dispersion
## 1  0.1 220902656   15552705
## 2  1.0 219067517   15486063
## 3 10.0 219135048   15292595
```

```r
pred2 <- predict(svm2$best.model, newdata=test)
cor_svm2 <- cor(pred2, test$Total.Power.Consumption)
print(paste("correlation = ", cor_svm2))
```

```
## [1] "correlation =  0.53462279699273"
```

```r
mse_svm2 <- mean((pred2 - test$Total.Power.Consumption)^2)
print(paste("mean squared error = ", mse_svm2))
```

```
## [1] "mean squared error =  214612699.20731"
```

Radial Kernel

```r
library(e1071)
library(MASS)
svm3 <- tune(svm, Total.Power.Consumption~., data=val, kernel="radial", ranges=list(cost=c(1, 10, 100),
summary(svm3)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##     1     1
##
## - best performance: 196304215
##
## - Detailed performance results:
##   cost gamma     error dispersion
## 1    1  0.25 199389584    7214447
## 2   10  0.25 196946280    9121916
## 3  100  0.25 196547068   10238669
## 4    1  1.00 196304215    8724710
## 5   10  1.00 199878247   11626077
## 6  100  1.00 219646763   25837676
## 7    1  2.00 197042011    9576039
## 8   10  2.00 204675226   13608356
## 9  100  2.00 240327212   51481394
```

```r
pred3 <- predict(svm3$best.model, newdata=test)
cor_svm3 <- cor(pred3, test$Total.Power.Consumption)
print(paste("correlation = ", cor_svm3))
```

```
## [1] "correlation =  0.610376220002194"
```

```r
mse_svm3 <- mean((pred3 - test$Total.Power.Consumption)^2)
print(paste("mean squared error = ", mse_svm3))
```

```
## [1] "mean squared error =  188455666.355684"
```

Analysis

The linear kernel had the worst performance out of the three kernels with a correlation of 0.4968 and a mean squared error (mse) of 228,268,776, when using C=100 (cost hyperparameter/slack). When tuning for the most optimal cost hyperparameter, we discovered that 100 generated the highest correlation because higher slack values prevent overfitting and lower variance. The linear SVM kernel was not suitable for this dataset because the dataset does not have a linear relationship between the predictors and the target variable. This explains why the correlation was so low and the error was so high for the linear kernel algorithm. The polynomial kernel had a slightly improved performance with a correlation of 0.5346 and a mse of 214,612,699, when C=10. This is because the polynomial kernel maps the observations to a higher dimensional space, thus improving accuracy. The radial kernel had the best performance with 0.6104 for correlation and 188,455,666 for mse, when C=1 and gamma=2. The radial kernel has another hyperparamter: gamma. Gamma can be can be used to control the bias-variance tradeoff, which results in even higher accuracy.