

Model Ideal

Load Libraries

```
library(glmnet) # create model
library(caret) # ml library
library(tidyverse) # common libraries
```

Load Data

```
df<-read.csv("rent-amount-clear-preprocessing.csv")
```

```
attach(df)
```

Split Data

```
set.seed(2018) # random state

training.ids<-createDataPartition(rent.amount,p=0.7,list = F)

train_data<-df[training.ids,] # select train data index
test_data<-df[-training.ids,] # select test data index
```

```
X_train<- train_data %>% select(-rent.amount)
X_train<-as.matrix(X_train)# transform to matrix
Y_train<-train_data$rent.amount
```

```
X_test<- test_data %>% select(-rent.amount)
X_test<-as.matrix(X_test) # transform to matrix
Y_test<-test_data$rent.amount
```

Model Creation

```
lm_ridge<-glmnet(x=X_train,y=Y_train,alpha = 0.1,lambda = 0.001)
```

We assign the same parameters of the best model, which we had previously estimated.

```
predict_model<-function(data){

  y_pred<-predict(lm_ridge,newx =data)
  y_pred<-as.vector(y_pred)
}
```

```
pred_train<-predict_model(X_train)
pred_test<-predict_model(X_test)
```

```
train_data<- train_data %>% mutate(predictions=pred_train)

test_data<- test_data %>% mutate(predictions=pred_test)
```

MSE

Measures the average error between the predicted value and the original.

Train

```
train_data %>% summarise(mse=RMSE(predictions,rent.amount))
```

	mse <dbl>
	0.1915151
1 row	

Test

```
test_data %>% summarise(mse=RMSE(predictions,rent.amount))
```

	mse <dbl>
	0.2209271

1 row

R²

It measures the degree of fit of the model. The higher the value, the better the model fit.

```
train_data %>% summarise(mse=R2(predictions,rent.amount))
```

	mse <dbl>
	0.9392654

1 row

```
test_data %>% summarise(mse=R2(predictions,rent.amount))
```

	mse <dbl>
	0.917652

1 row

Both metrics are very similar. Therefore, it indicates that our model is not overtrained. This means that the model is only good for the training data but it is unable to generalize with new data that it has never seen.

Exponential transformation

```
test_data<-test_data %>%  
  mutate(rent.amount=exp(rent.amount)) %>%  
  mutate(predictions=exp(predictions))
```

Let us remember that the rent.amount variable is in logarithmic scale and we perform the exponential transformation because it is its opposite operation.

Variable coefficients

```
coef(lm_ridge)
```

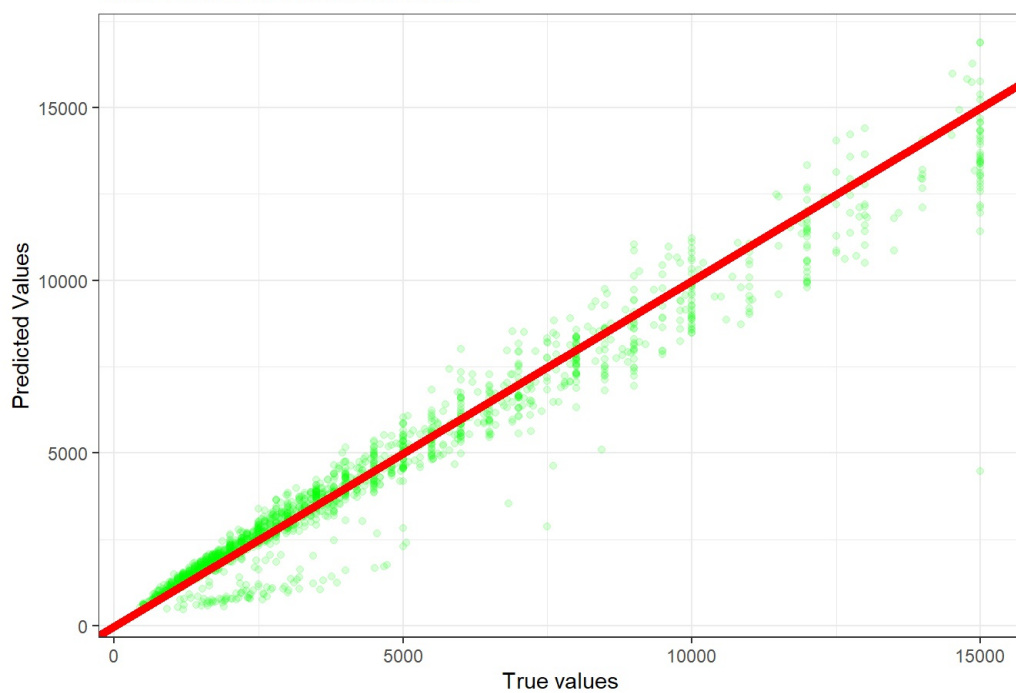
```
## 10 x 1 sparse Matrix of class "dgCMatrix"  
##              s0  
## (Intercept)  4.6666887400  
## city        0.1171973949  
## animal      -0.0003875935  
## furniture    0.0928719962  
## area        0.0502690908  
## fire.insurance 0.8194256100  
## rooms       -0.0041296755  
## bathrooms    0.0293551434  
## floor        0.0483440792  
## parking.spaces 0.0118199662
```

We previously knew that the health insurance charge variable. It had a great relationship with respect to the rental price, which I assign the highest coefficient.

To the other complementary variables that add value, I minimize the weight of the coefficient as we had previously explained.

```
test_data %>%  
  ggplot(aes(x=rent.amount,y=predictions))+  
  geom_point(color="green",alpha=0.15) +  
  geom_abline(intercept = 0,slope = 1,color="red",lwd=2) +  
  theme_bw() +  
  labs(title = "True Values vs Predicted Values",  
       x="True values",y="Predicted Values")
```

True Values vs Predicted Values



```
test_data %>% select(rent.amount,predictions) %>% head(50)
```

7
10
11
15
17
18
19
23
27
29

1-10 of 50 rows | 1-1 of 3 columns

Previous **1** 2 Next

In most predictions, it gives predictions very close to the original value.

Save Model

```
saveRDS(lm_ridge,"lm_ridge_rent.RDS")
```