Final Project: Report 2

Eric Wu

2023-05-04

Introduction

Background

The purpose of this project is to take a deep look into the use of single-layer neural network. In the part one of the project, we have examine the lab on single-layer neural network on Hitter's data. The author is trying to compare and contrast model performances of three different models based on the mean absolute error of each model. The result is significant, despite the fact that LASSO outperform the multiple linear regression and single-layer neural network, we still get to see how single-layer neural network produces a loss function graph and its basic structure and parameters. For the second part of the project, our goal was to fit a similar single-layer neural network to another data set.

Dataset

We decided to use the "House Rent Prediction" (HRP) dataset found on Kaggle, a data science community known for its enriched various data sources. The HRP dataset consists of 4700+ observations and 11 attributes that allow us to make a prediction on the number of rents given a variety of attributes.

Here's the list of variables used and explained by the contributor of the dataset:

- BHK: Number of Bedrooms, Hall, Kitchen
- Rent: Rent of the Houses/Apartments/Flats
- Size: Size of the Houses/Apartments/Flats in Square Feet
- Floor: Houses/Apartments/Flats situated on which Floor and Total Number of Floors (Example: Ground out of 2, 3 out of 5, etc.)
- Area Type: Size of the Houses/Apartments/Flats calculated on either Super Area or Carpet Area or Build Area.
- Area Locality (Area.Locality): Locality of the Houses/Apartments/Flats
- City: City where the Houses/Apartments/Flats are Located
- Furnishing Status: Furnishing Status of the Houses/Apartments/Flats, either it is Furnished or Semi-Furnished or Unfurnished
- Tenant Preferred: Type of Tenant Preferred by the Owner or Agent
- Bathroom: Number of Bathrooms
- Point of Contact (Posted.On): Whom should you contact for more information regarding the Houses/Apartments/Flats

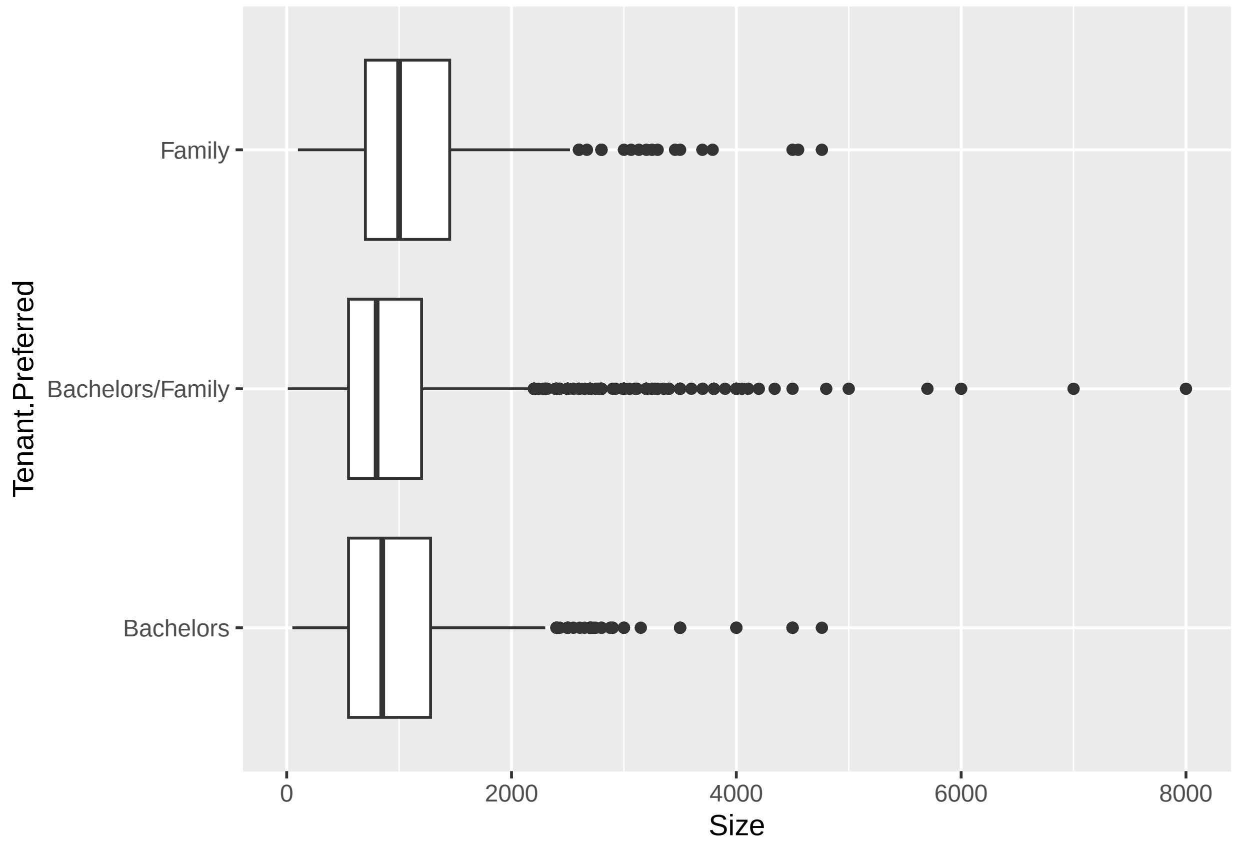
Exploratory Data Analysis

Data Visualization

```r
hrp %>%
  group_by(Tenant.Preferred) %>%
  count()
```

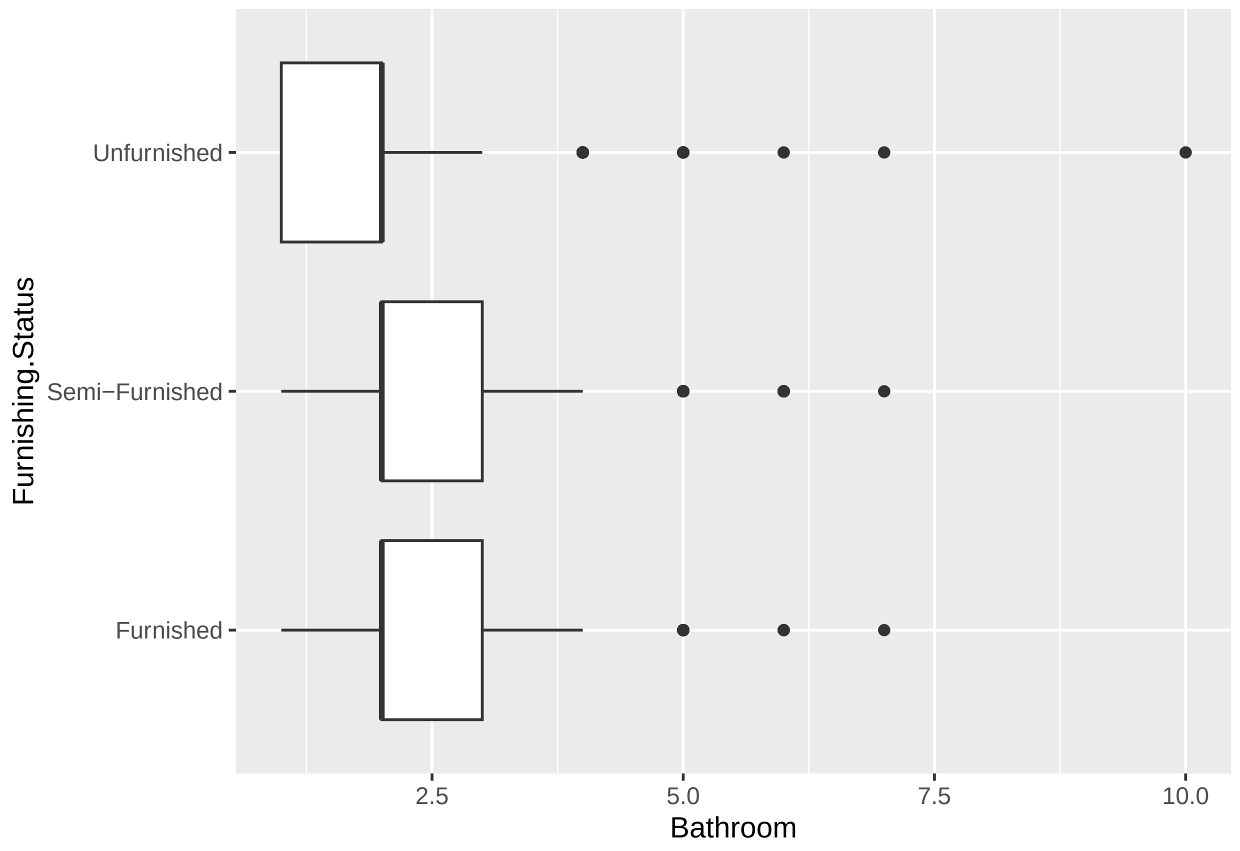
```
## # A tibble: 3 x 2
## # Groups:   Tenant.Preferred [3]
##   Tenant.Preferred     n
##   <chr>            <int>
## 1 Bachelors          830
## 2 Bachelors/Family  3444
## 3 Family             472
```

```r
hrp %>%
  ggplot(mapping=aes(x=Size, y=Tenant.Preferred)) +
  geom_boxplot()
```



It seems as though houses that prefer families as tenants may be slightly larger on average than those that prefer bachelors, although the difference is not significant. Houses with no preference, however, seem to be slightly smaller than both on average but there are many more upper outliers. These results may be skewed however, as there are significantly more houses with no preference than those with a preference either way.

```r
hrp %>%
  ggplot(mapping=aes(x=Bathroom, y=Furnishing.Status)) +
  geom_boxplot()
```



Clearly Unfurnished houses tend to have less bathrooms than Semi-Furnished and Furnished houses, while Semi-Furnished and Furnished houses seem to generally have the same amount of bathrooms as each other.

Hierarchical clustering

Methodology

Our plan is to compare and contrast a multiple regression model, a KNN regression model, and a single-layer neural network in predicting the rent based using other predictors in the dataset. All models will involve some sort of variable selection, whether it's using other functions or included in model building, this is very beneficial when we are doing model selections. The primary focus is to compare the RMSE produced by each model.

Multiple Linear Regression

```r
base_model <- lm(Rent ~ . - Area.Locality - Floor - Posted.On, data=train)

linear_model <- step(base_model, direction="backward")
```

```
## Start:  AIC=84620.79
## Rent ~ (Posted.On + BHK + Size + Floor + Area.Type + Area.Locality +
##     City + Furnishing.Status + Tenant.Preferred + Bathroom +
##     Point.of.Contact) - Area.Locality - Floor - Posted.On
##
##                     Df  Sum of Sq        RSS   AIC
## - Area.Type          2 8.4516e+09 1.8071e+13 84619
## - BHK                1 5.4526e+09 1.8068e+13 84620
## - Furnishing.Status  2 1.8056e+10 1.8080e+13 84621
## <none>                            1.8062e+13 84621
## - Tenant.Preferred   2 2.1561e+10 1.8084e+13 84621
## - Point.of.Contact   2 3.9625e+10 1.8102e+13 84625
## - Bathroom           1 9.2255e+10 1.8154e+13 84638
## - Size               1 8.5590e+11 1.8918e+13 84795
## - City               5 1.2257e+12 1.9288e+13 84860
##
## Step:  AIC=84618.57
## Rent ~ BHK + Size + City + Furnishing.Status + Tenant.Preferred +
##     Bathroom + Point.of.Contact
##
##                     Df  Sum of Sq        RSS   AIC
## - BHK                1 6.6024e+09 1.8077e+13 84618
## - Furnishing.Status  2 1.8471e+10 1.8089e+13 84618
## <none>                            1.8071e+13 84619
## - Tenant.Preferred   2 1.9721e+10 1.8090e+13 84619
## - Point.of.Contact   2 6.3143e+10 1.8134e+13 84628
## - Bathroom           1 9.1734e+10 1.8162e+13 84636
## - Size               1 8.4887e+11 1.8920e+13 84791
## - City               5 1.2770e+12 1.9348e+13 84868
##
## Step:  AIC=84617.96
## Rent ~ Size + City + Furnishing.Status + Tenant.Preferred + Bathroom +
##     Point.of.Contact
##
##                     Df  Sum of Sq        RSS   AIC
## - Furnishing.Status  2 1.8159e+10 1.8095e+13 84618
## - Tenant.Preferred   2 1.8828e+10 1.8096e+13 84618
## <none>                            1.8077e+13 84618
## - Point.of.Contact   2 6.3087e+10 1.8140e+13 84627
## - Bathroom           1 1.8758e+11 1.8265e+13 84655
## - Size               1 9.6237e+11 1.9040e+13 84813
## - City               5 1.2704e+12 1.9348e+13 84866
##
## Step:  AIC=84617.77
## Rent ~ Size + City + Tenant.Preferred + Bathroom + Point.of.Contact
##
##                    Df  Sum of Sq        RSS   AIC
## - Tenant.Preferred  2 1.8475e+10 1.8114e+13 84618
## <none>                           1.8095e+13 84618
## - Point.of.Contact  2 6.5553e+10 1.8161e+13 84627
## - Bathroom          1 1.8746e+11 1.8283e+13 84655
## - Size              1 9.8759e+11 1.9083e+13 84817
## - City              5 1.3177e+12 1.9413e+13 84875
##
```

```
## Step:  AIC=84617.64
## Rent ~ Size + City + Bathroom + Point.of.Contact
##
##                     Df  Sum of Sq        RSS   AIC
## <none>                            1.8114e+13 84618
## - Point.of.Contact  2 6.4514e+10 1.8178e+13 84627
## - Bathroom          1 1.8254e+11 1.8296e+13 84654
## - Size              1 9.8726e+11 1.9101e+13 84817
## - City              5 1.3129e+12 1.9427e+13 84873
```

```
linpred <- predict(linear_model, newdata=test)

test_linpred <-
  test %>%
  cbind(linpred) %>%
  mutate(Resid.Square = (Rent-linpred)^2)

sqrt(sum(test_linpred$Resid.Square)/nrow(test_linpred))
```

```
## [1] 42803.14
```

RMSE with this seed (123) for multiple linear regression is 42803.14.

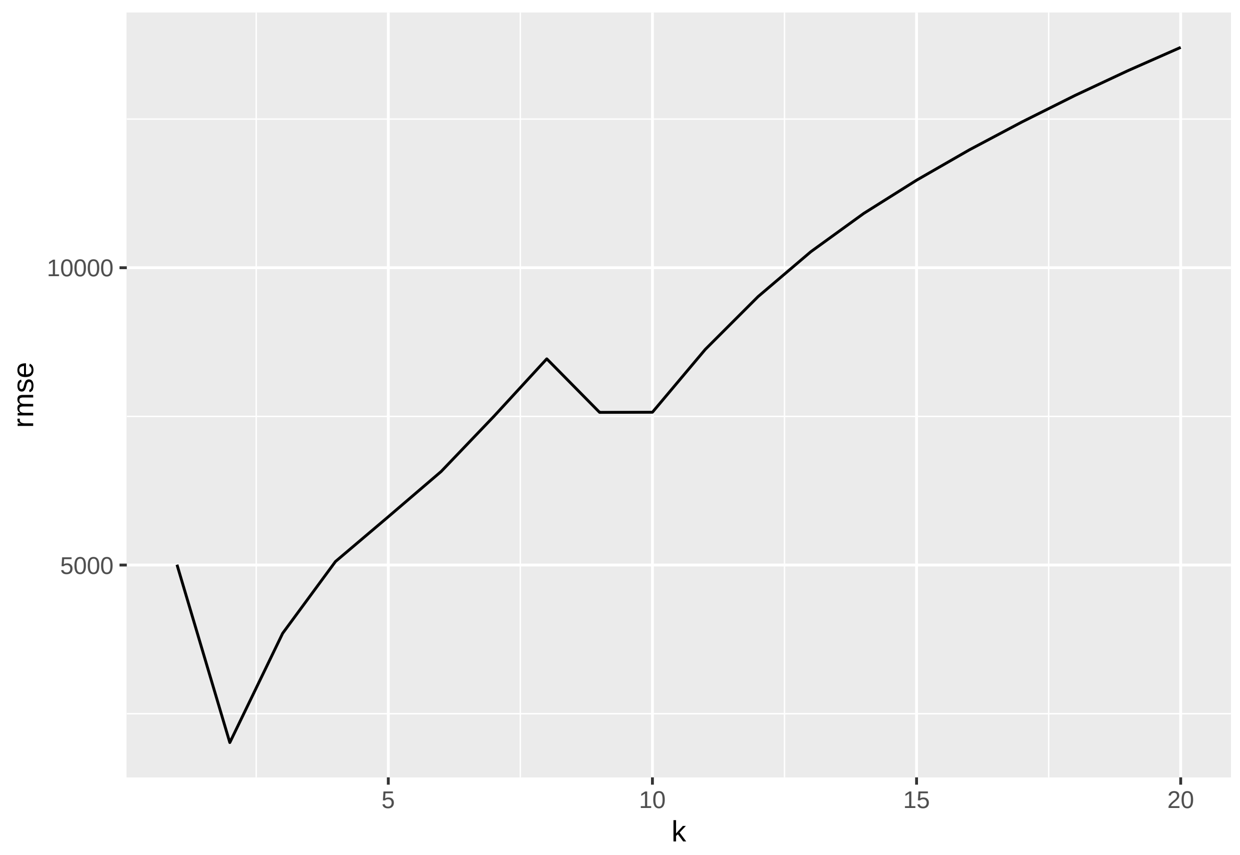
KNN Regression

```
xvars <- names(select_if(train_scale, is.numeric))
maxK <- 20
mse_vec <- rep(NA, maxK)
rmse_vec <- rep(NA, maxK)

for(i in 1:maxK){
  knn_res <- knn.reg(train = train_scale[ , xvars, drop = FALSE],
                     test = test_scale[ , xvars, drop = FALSE],
                     y = train_scale$Rent,
                     k = i)

  mse_vec[i] <- mean((test_scale$Rent - knn_res$pred)^2)

  rmse_vec[i] <- sqrt(mse_vec[i])
}
choice_k <- data.frame(k = 1:maxK, rmse = rmse_vec)
ggplot(data = choice_k, mapping = aes(x = k, y = rmse)) +
   geom_line()
```



```r
# Based on the graph the best K value seems to be the lowest point on the graph.
```

```r
# Based on the front of this vector the best K value is 2.
head(choice_k)
```

```
##   k     rmse
## 1 1 5005.750
## 2 2 2014.333
## 3 3 3851.691
## 4 4 5059.494
## 5 5 5812.080
## 6 6 6572.819
```

Single-Layer Neural Network

```r
modnn <- keras_model_sequential() %>%
     layer_dense(units = 2351, activation = "relu",
        input_shape = ncol(x)) %>%
     layer_dropout(rate = 0.4) %>%
     layer_dense(units = 1)

modnn %>% compile(loss = "mse",
    optimizer = optimizer_rmsprop(),
    metrics = list("mse")
   )

#history <- modnn %>% fit(
#     x[train_ind, ], y[train_ind], epochs = 600, batch_size = 32, #
#    validation_data = list(x[-train_ind, ], y[-train_ind])
#   )

#npred <- predict(modnn, x[-train_ind, ])
#nn_mse <- mean((y[-train_ind] - npred)^2)
#sqrt(nn_mse)

## 43418.85
```

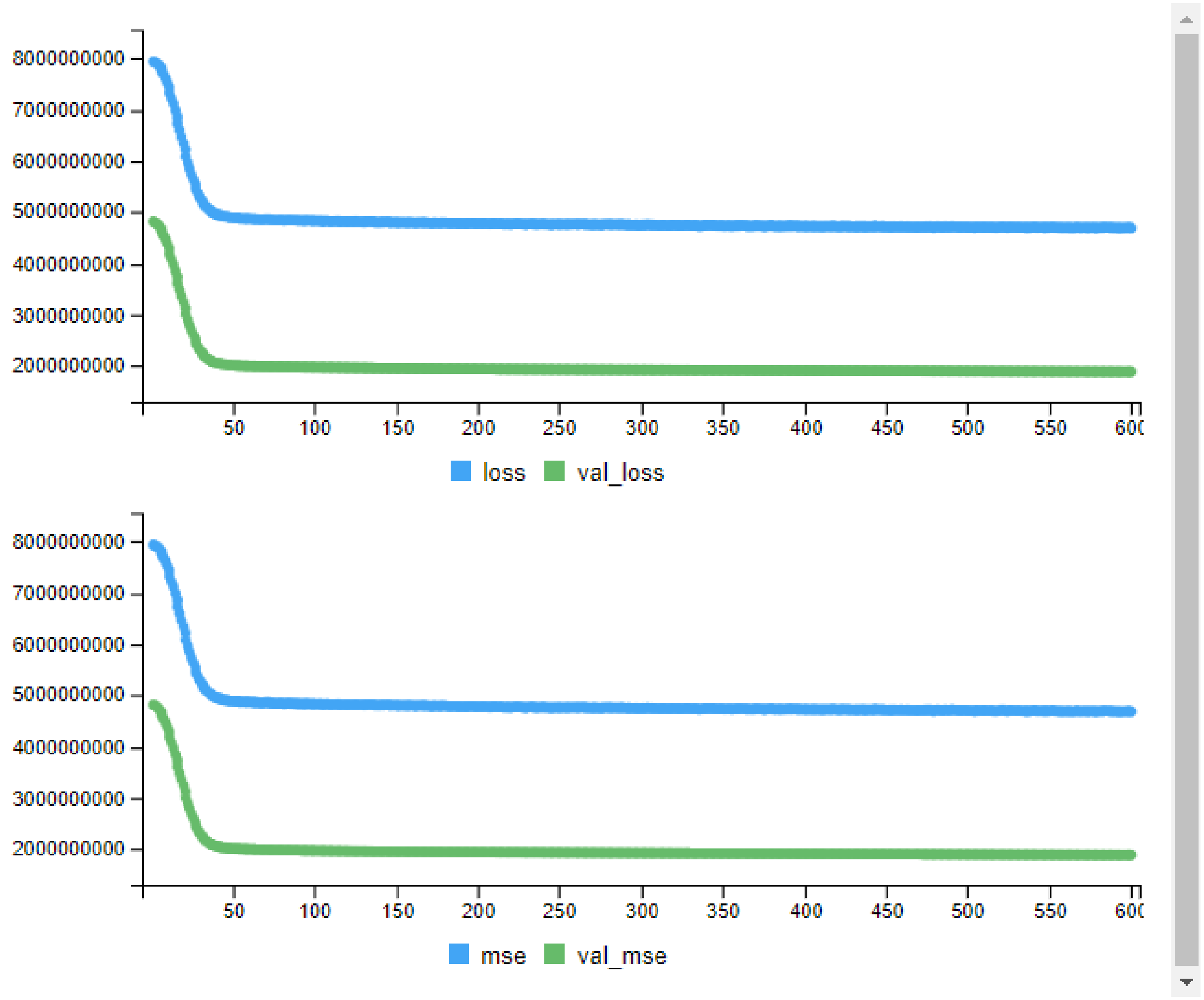


Figure 1: Image 1: Training Results from single-layer Neural Network

The structure of the single-layer neural network is very similar to the lab, except for the 2351 hidden units, and we are measuring the Mean Square Error(MSE) instead of the Mean Absolute Error(MAE). The RMSE resulting from the training end up being 43418.85. And base on the loss function graph, the MSE of the validation(testing) set is significantly smaller than the training set. Which indicates better performance in the testing set.

```r
#modnn2 <- keras_model_sequential() %>%
#     layer_dense(units = 2351, activation = "relu",
#        input_shape = ncol(x)) %>%
#     layer_dropout(rate = 0.4) %>%
#     layer_dense(units = 1)

#modnn2 %>% compile(loss = "mse",
#    optimizer = optimizer_rmsprop(),
#    metrics = list("mse")
#   )

#history2 <- modnn2 %>% fit(
#     x[train_ind_alter, ], y[train_ind_alter], epochs = 600, batch_size = 32, #
#    validation_data = list(x[-train_ind_alter, ], y[-train_ind_alter])
#  )

#npred2 <- predict(modnn2, x[-train_ind_alter, ])
#nn_mse2 <- mean((y[-train_ind_alter] - npred2)^2)
#sqrt(nn_mse2)

# 37620.59
```

Image 2 demonstrates the effect of random seed on the neural network, as shown, the MSE is greatly reduced and the RMSE decreases from 43418.85 to 37620.59.

Discussion

Conclusion

Code Appendix

```r
# Load essential libraries
library(tidyverse)
library(ggplot2)
library(keras)
library(ISLR2)
library(FNN)

# Load the dataset
hrp <- read.csv("House_Rent_Dataset.csv")

# Create indicator variables
hrp_update <- hrp %>%
  mutate(
    Area.indicator = case_when(
      Area.Type == "Built Area" ~ 1,
      Area.Type == "Carpet Area" ~ 2,
      Area.Type == "Super Area" ~ 3
    ),
    Furnishing.indicator = case_when(
      Furnishing.Status == "Furnished" ~ 1,
```

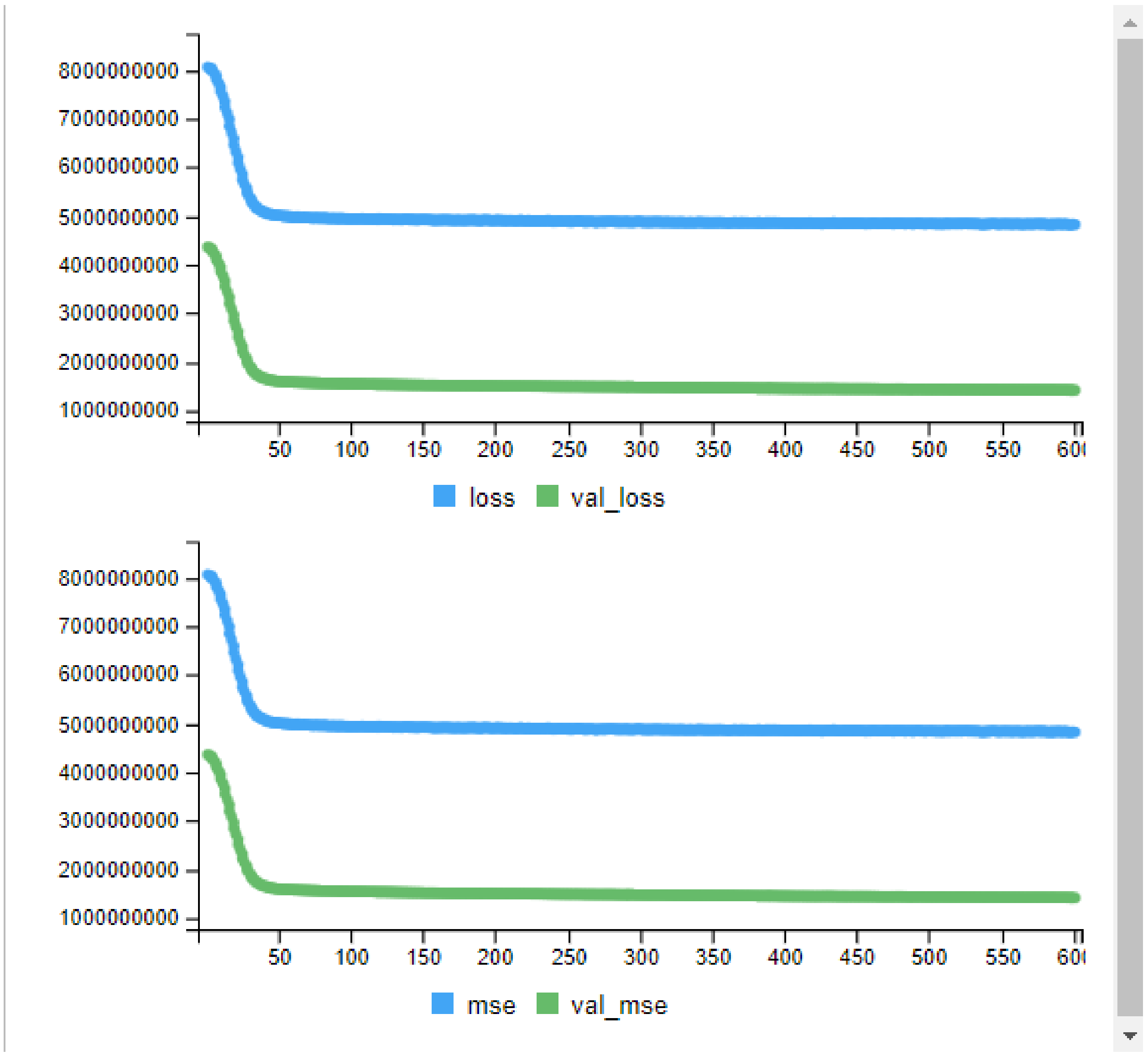


Figure 2: Image 2: Training Results from a single-layer Neural Network with alternative seed

```
      Furnishing.Status == "Semi-Furnished" ~ 2,
      Furnishing.Status == "Unfurnished" ~ 3
    ),
    Contact.indicator = case_when(
      Point.of.Contact == "Contact Agent" ~ 1,
      Point.of.Contact == "Contact Builder" ~ 2,
      Point.of.Contact == "Contact Owner" ~ 3
    )
  )

# Set up all parameters
seed <- 123
seed_alter <- 12345

# Scale the data
xvars <- c("BHK", "Size", "Area.indicator", "Furnishing.indicator", "Bathroom", "Contact.indicator")
hrp_update[ , xvars] <- scale(hrp_update[ , xvars], center = TRUE, scale = TRUE)

# Set up the necessary matrix for neural network
x <- scale(model.matrix(Rent ~ BHK + Size + Area.indicator + Furnishing.indicator + Bathroom + Contact.
y <- hrp_update$Rent

# Set up the training/testing split
set.seed(seed)
train_ind <- sample(1:nrow(hrp_update), floor(0.8 * nrow(hrp_update)))
set.seed(NULL)

set.seed(seed_alter)
train_ind_alter <- sample(1:nrow(hrp_update), floor(0.8 * nrow(hrp_update)))
set.seed(NULL)

# Not scaled data
train <- hrp[train_ind, ]
test <- hrp[-train_ind, ]

# Scaled data
train_scale <- hrp_update[train_ind, ]
test_scale <- hrp_update[-train_ind, ]

# Scaled data with alternative seed
train_alter <- hrp_update[train_ind_alter, ]
test_alter <- hrp_update[-train_ind_alter, ]

hrp %>%
  group_by(Tenant.Preferred) %>%
  count()

hrp %>%
  ggplot(mapping=aes(x=Size, y=Tenant.Preferred)) +
  geom_boxplot()

hrp %>%
  ggplot(mapping=aes(x=Bathroom, y=Furnishing.Status)) +
```

```r
  geom_boxplot()

base_model <- lm(Rent ~ . - Area.Locality - Floor - Posted.On, data=train)

linear_model <- step(base_model, direction="backward")

linpred <- predict(linear_model, newdata=test)

test_linpred <-
  test %>%
  cbind(linpred) %>%
  mutate(Resid.Square = (Rent-linpred)^2)

sqrt(sum(test_linpred$Resid.Square)/nrow(test_linpred))

xvars <- names(select_if(train_scale, is.numeric))
maxK <- 20
mse_vec <- rep(NA, maxK)
rmse_vec <- rep(NA, maxK)

for(i in 1:maxK){
  knn_res <- knn.reg(train = train_scale[ , xvars, drop = FALSE],
                   test = test_scale[ , xvars, drop = FALSE],
                   y = train_scale$Rent,
                   k = i)

  mse_vec[i] <- mean((test_scale$Rent - knn_res$pred)^2)

  rmse_vec[i] <- sqrt(mse_vec[i])
}
choice_k <- data.frame(k = 1:maxK, rmse = rmse_vec)
ggplot(data = choice_k, mapping = aes(x = k, y = rmse)) +
   geom_line()
# Based on the graph the best K value seems to be the lowest point on the graph.

# Based on the front of this vector the best K value is 2.
head(choice_k)

modnn <- keras_model_sequential() %>%
     layer_dense(units = 2351, activation = "relu",
        input_shape = ncol(x)) %>%
     layer_dropout(rate = 0.4) %>%
     layer_dense(units = 1)

modnn %>% compile(loss = "mse",
    optimizer = optimizer_rmsprop(),
    metrics = list("mse")
   )

#history <- modnn %>% fit(
#     x[train_ind, ], y[train_ind], epochs = 600, batch_size = 32, #
#    validation_data = list(x[-train_ind, ], y[-train_ind])
#  )
```

```
#npred <- predict(modnn, x[-train_ind, ])
#nn_mse <- mean((y[-train_ind] - npred)^2)
#sqrt(nn_mse)

## 43418.85

#modnn2 <- keras_model_sequential() %>%
#     layer_dense(units = 2351, activation = "relu",
#        input_shape = ncol(x)) %>%
#     layer_dropout(rate = 0.4) %>%
#     layer_dense(units = 1)

#modnn2 %>% compile(loss = "mse",
#    optimizer = optimizer_rmsprop(),
#    metrics = list("mse")
#   )

#history2 <- modnn2 %>% fit(
#     x[train_ind_alter, ], y[train_ind_alter], epochs = 600, batch_size = 32, #
#    validation_data = list(x[-train_ind_alter, ], y[-train_ind_alter])
#  )

#npred2 <- predict(modnn2, x[-train_ind_alter, ])
#nn_mse2 <- mean((y[-train_ind_alter] - npred2)^2)
#sqrt(nn_mse2)

# 37620.59
```