MA5832_Capstone Project

Friday Yusuf

15/02/2021

Driving Factors of Unemployment rate in Australia

Abstract

The unemployment rate in Australia is a serious problem which can result in poverty, low standard of living, poor mental health, increased pressure on the health, welfare systems and other public services. Therefore, this study is aimed at identifying the determining factors for the unemployment rate in the Australian economy from June 1981 to September 2020 by applying machine learning methods and deep learning methods.

The data was cleaned and pre-processing, data contained five missing values on both variables X6: Number of job vacancies measured in thousands, and X7: Estimated Resident Population measured in thousands

The Data was split into two, 80 percent for training set and remaining 20% for testing or validation for both machine learning and deep learning method. The missing values were imputed using r package called MICE which mean multivariate imputation by chain equations. The missing values can lead a significant amount of bias in the analysis and makes the analysis of the data more problematic. Imputation seems to be the best options to avoid drawbacks involved

From the analysis the main driving factors for unemployment rate in Australia economy are consumer price index (CPI), population growth, and to number of job vacancies available.

One of the objectives of this analysis is to communicate the rate of unemployment and the driven factor to the public.

1.1. The Australian Unemployment Rate overview of the last 21 years ranging from 1999 - 2020 and some driving factor of the unemployment rate.

```
# subsetting the data from 1999 to 2020
emp_rate <-data[72:158,]
summary(emp_rate)
##      X              Y           X1            X2
##  Length:87        Min.   :4.100   Min.   :-7.0000   Min.   :-1.4000
##  Class :character   1st Qu.:5.100   1st Qu.: 0.4000   1st Qu.: 0.3000
##  Mode  :character   Median :5.500   Median : 0.7000   Median : 0.8000
##                     Mean   :5.591   Mean   : 0.6391   Mean   : 0.8529
##                     3rd Qu.:6.100   3rd Qu.: 1.0000   3rd Qu.: 1.5000
##                     Max.   :7.100   Max.   : 3.3000   Max.   : 3.0000
```

```
##
##       X3              X4              X5              X6
##  Min.   :-8.3000   Min.   :-8.1000   Min.   : 67.80   Min.   : 87.2
##  1st Qu.: 0.5000   1st Qu.:-1.0500   1st Qu.: 80.75   1st Qu.:119.7
##  Median : 0.7000   Median : 1.0000   Median : 94.30   Median :154.5
##  Mean   : 0.7299   Mean   : 0.7149   Mean   : 93.89   Mean   :155.1
##  3rd Qu.: 1.0000   3rd Qu.: 2.1500   3rd Qu.:107.15   3rd Qu.:182.1
##  Max.   : 5.9000   Max.   :13.2000   Max.   :116.60   Max.   :232.3
##                                      NA's   :5
##       X7
##  Min.   :187705
##  1st Qu.:199038
##  Median :216467
##  Mean   :217310
##  3rd Qu.:234583
##  Max.   :253643
##  NA's   :5
d <-emp_rate %>%
    group_by(Y) %>%
    arrange(desc(Y))
head(d)
## # A tibble: 6 x 9
## # Groups:   Y [3]
##   X         Y    X1    X2    X3    X4    X5    X6     X7
##   <chr>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>   <dbl>
## 1 Mar-99   7.1   0.8   0.1   1.5   1.1  67.8  87.2 187705.
## 2 Sep-20   7.1   3.3   1.4   5.9   0.7 116.  206.      NA
```

The above statistics description of all the variables for 20 years (1999 - 2020) shows that the average unemployment rate (Y) was of 5.59%. The lowest unemployment rate was 4.1% in March 2008 and the highest unemployment rate was 7.1% in March of 1999 and in September 2020. The increase in unemployment was due to covid19 pandemic which impacted the worldwide economy. The estimated resident population (X7) measured in thousands showed a population growth of 6.5938 million people, from 18.770 million people to 25.3643 million people, a 34.6% increase in population over this period. As population grew over the past 20 years, the job vacancies(X6) also increased at an average of 155.1. The minimum numbers of job vacancies were 87.2 and the maximum were 232.3.

The Consumer Price Index (CPI) is a measure of changes in retail prices of a constant basket of goods and services over time, representative of consumption expenditure by resident families in metropolitan Australian areas (Australian Bureau of Statistics). The maximum CPI of 116.6 was achieved in March 2020 and a minimum of 67.8 in March 1999.

The term of trade index (X4), which is the ratio between the index of export prices and the index of import prices had an average value of 0.7149%, a minimum value of -8 % and a maximum of 13.2 %. This shows that Australia has increased in export prices than the import prices.

These factors are major indications of the economic growth of any nation or state; which have an impact on the unemployment rate.

It is understandable from the AUS_Data data, that the main signs for the unemployment rate are the estimated resident population, the number of job vacancies, and the consumer price index (CPI). The consumer price index (CPI) is a very vital sign for the unemployment rate; since is the factor that shows if there is adequate economy growth to create more jobs opportunities and if the nation or state in the study has or not a positive economic balance.

2.0 DATA

The dataset used in the report is gathered and collected from the Australia Bureau of statistic. This is from the last 40years (1981-2020) and the data contains quarterly values for this duration. The data can also be found in a public domain from Australia Bureau of statistic website.

The dataset set consist of 158 observations, and 9 variables which are;

1. Period: quarterly period the data start from June 1981 which is the second quarter of the year 1981 to September 2020 which is the last quarter of the 2020.
2. Y: unemployment rate measured in percentage
3. X1: Percentage change in Gross domestic product
4. X2: Percentage change in the Government final consumption expenditure
5. X3: Percentage change in final consumption expenditure of all industry sectors
6. X4: Term of trade index (percentage)
7. X5: Consumer Price Index of all groups (CPI)
8. X6: Number of job vacancies measured in thousands
9. X7: Estimated Resident Population measured in thousands

There are some missing values in the data, five missing values on X6: Number of job vacancies measured in thousands, and another five values missing on X7: Estimated Resident Population measured in thousands variable. The data was saved using CSV file format, it was loaded in Rmarkdown using read.table function in rstudio. The first row of the data was removed since it is descriptive names of the variables. The response variable Y and the eight predictors variable were transformed to numeric variables with a transform function to get statistic description.

Data Summary

```
summary(data)
##      X                Y               X1               X2
##  Length:158         Min.   : 4.100   Min.   :-7.0000   Min.   :-4.6000
##  Class :character   1st Qu.: 5.500   1st Qu.: 0.4000   1st Qu.: 0.1000
##  Mode  :character   Median : 6.250   Median : 0.7000   Median : 1.0000
##                     Mean   : 6.855   Mean   : 0.7247   Mean   : 0.8532
##                     3rd Qu.: 8.275   3rd Qu.: 1.1000   3rd Qu.: 1.6000
##                     Max.   :11.100   Max.   : 3.3000   Max.   : 7.5000
##
```

```
##       X3              X4              X5              X6
##  Min.   :-8.300   Min.   :-8.1000   Min.   : 28.40   Min.   : 26.8
##  1st Qu.: 0.400   1st Qu.:-1.1000   1st Qu.: 59.00   1st Qu.: 66.9
##  Median : 0.800   Median : 0.3000   Median : 73.50   Median : 99.7
##  Mean   : 0.762   Mean   : 0.3456   Mean   : 75.08   Mean   :111.4
##  3rd Qu.: 1.200   3rd Qu.: 1.6500   3rd Qu.: 96.80   3rd Qu.:160.8
##  Max.   : 5.900   Max.   :13.2000   Max.   :116.60   Max.   :232.3
##                                     NA's   :5
##       X7
##  Min.   :149233
##  1st Qu.:171698
##  Median :190288
##  Mean   :194851
##  3rd Qu.:218656
##  Max.   :253643
##  NA's   :5
```

The above statistic descriptive table from June 1981 till June 2020 shows that the mean unemployment rate was 6.25%. The Consumer Price Index of all groups (CPI) grew steadily as the population increased. Over the period of forty-one years, more than 10 million people have been added to the population. With an increase in the labour market, the number of jobs vacancies also increased from 28.4 thousand to 1.16 million.

Imputing missing data

Missing data are vital parts of most actual datasets. To provide an effective and accurate analytical report of data, the datasets need to be treated using imputation and cleaning methods. The missing values were imputed using a package called MICE which mean multivariate imputation by chain equations. A function was used to calculate the percentage of missing data on each variable by using sum of the missing value divide by the length, and multiplied by 100. The MICE function which uses a predictive mean matching was then used to impute the values.

```
#Imputing  missing value

p <-function(x){sum(is.na(x))/length(x)*100}
apply(data, 2, p)
```

```
##       X       Y      X1      X2      X3      X4      X5      X6
## 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 0.000000 3.164557
##       X7
## 3.164557

imputed <- mice(data, m = 1, seed = 1234)

data_m <-complete(imputed, 1)
```

3.0 Analysis and Investigation of Machine learning (ML) method

Random Forest is a common a commonly used supervised machine learning algorithm that is simple and diverse. It is based on random sample of a training data that consist of multiple single trees (Rodriguez, 2015), which makes it a strong modelling method and much more robust than single decision tree.

In this method, two libraries were used; randomForest and caret. A seed was set in order to reproduce the same result whenever the model is run with the same seed number. The dataset was partitioned, 80% of the data set was allocated for training and 20% for test. The model was applied by loading the library 'randomForest' with the function. A numeric variable, Y was the respond variable. It had 7 numeric predictor variables and uses 4 mtry.

```
##
## Call:
##  randomForest(formula = Y ~ X1 + X2 + X3 + X4 + X5 + X6 + X7,      data = training, mtry = 4,
importance = TRUE)
##               Type of random forest: regression
##                     Number of trees: 500
## No. of variables tried at each split: 4
##
##          Mean of squared residuals: 0.220792
##                    % Var explained: 93.74
```

Only four variables were used to used and 93.74 percent variance is explained furthermore we see the importance variables

```
# plot
importance(rfm_au)
```

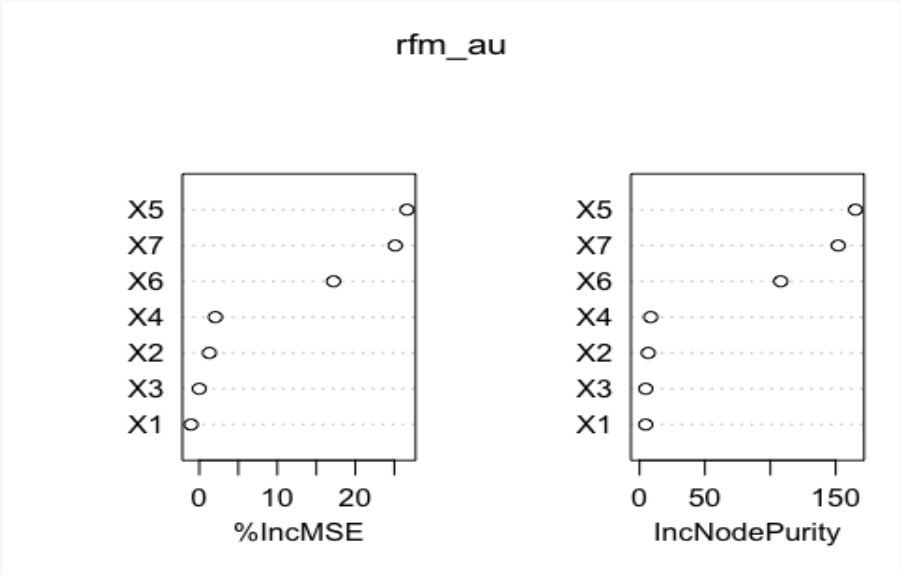
```
##        %IncMSE IncNodePurity
## X1 -1.056575415     4.712464
## X2  1.292365269     6.662729
## X3 -0.003781022     4.778179
## X4  2.063328745     8.525811
## X5 26.622533006   165.086106
## X6 17.208199310   107.952126
## X7 25.119865194   151.895757
```

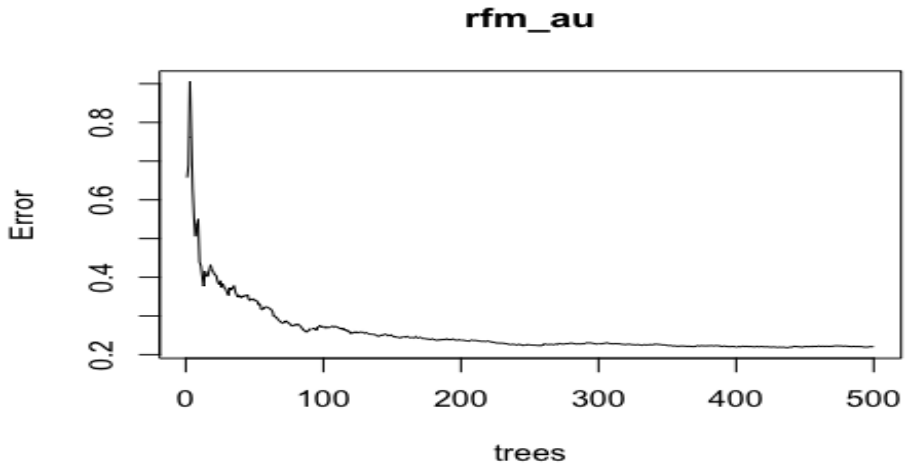
```
varImpPlot(rfm_au)
```

The model shows that variables X5: Consumer Price Index of all groups (CPI) ,X7: Estimated Resident Population measured in thousands, X6: Number of job vacancies measured in thousands are the most important variables to determine the employment rate

Error Rate Plot

```
plot(rfm_au)
```





The above plot shows the error rate in the random forest model. There was a high error rate of 0.9 with 0 number of trees. As the number of trees increased from 0 to 100, the error rate gradually decreases from 0.9 to 0.25. There was no significant increase in the error rate even as the number of trees increased from 100 to 400.

Prediction and Confusion Matrix

We use prediction function to predict the model

```
## confusion Matrix training Data

rfm_pred <- predict(rfm_au, training, type = "class")
```

```
tab <-table( training$Y, rfm_pred)
accuracy <- 1 - sum(diag(tab))/sum(tab)

summary(rfm_pred)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   4.526   5.479   6.521   6.944   8.258  10.800

accuracy

## [1] 0.968
```

The prediction shows that the average unemployment rate is 6.521% with an accuracy of 96.8%

Prediction of the Australia Unemployment rate from period of 2018 to 2020

```
# subset data from first quarter 2018 - 2020
newdata <- data_m[148:158,]
print(newdata)

##         X   Y   X1  X2   X3   X4    X5    X6      X7
## 148 Mar-18 5.5  0.9 1.3  0.7  3.3 112.6 212.8 248986.3
## 149 Jun-18 5.4  0.9 0.0  0.6 -1.2 113.0 223.8 249826.9
## 150 Sep-18 5.2  0.4 1.3  0.5  1.0 113.5 228.6 250917.5
## 151 Dec-18 5.0  0.2 1.7  0.6  2.4 114.1 230.4 251701.8
## 152 Mar-19 5.0  0.5 0.9  0.5  3.3 114.1 232.3 252887.6
## 153 Jun-19 5.2  0.6 2.8  1.0  1.5 114.8 228.0 253643.1
## 154 Sep-19 5.2  0.6 0.8  0.2  0.6 115.4 224.2 253643.1
## 155 Dec-19 5.2  0.4 1.2  0.6 -4.4 116.2 227.0 251701.8
## 156 Mar-20 5.2 -0.3 1.9 -0.4  0.4 116.6 227.3 252887.6
## 157 Jun-20 7.0 -7.0 3.0 -8.3  0.9 114.4 129.2 225222.0
## 158 Sep-20 7.1  3.3 1.4  5.9  0.7 116.2 206.1 221724.7

## predicting from march 2018  to june 2019
sub_pred <- predict(rfm_au, newdata, type = "class")

summary(sub_pred)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   5.076   5.221   5.278   5.468   5.392   6.478
```

For this period of March 2018 to September 2020 the average unemployment rate was predicted to be 5.468%.

4. 0. Analysis and Investigation neural network (NN) method

Lately, deep learning is considered as the most powerful part of machine learning method, which is used for finding out the hidden knowledge within a very large dataset to make predictions more accurate(Xu, 2014). Neural networks are known for classification analysis, for instance, they are used

in handwritten digits classification, but the question is will it be productive if it is used for regression problems?

The neural network was visualised using function neuralnet. The neural network model uses numerical variables for both response and predictor.
Step 1: Processing the dataset and set seed to default
The data was converted in matrix with the as.matrix function. The data was split into training and testing. For the training, the predictor variables X1, X2, X3, X4, X5, X6, X7 were kept and the respond variable Y as training target was stored.
The colMean function was used to calculate the mean, also apply function was used to find the standard deviation of the training set. Scale function was used to normalized the training set. The mean and the standard deviation of the training set was applied to scale the testing set.

Step 2 : creating the Neural Network model

The sequential model was defined, two dense layers were added, each layer with 64 neurons, the output layer was defined with only one layer dense, '**relu**' as used as the activation function for the hidden layers. "relu" means rectified linear unit

Step 3: complie

- mean_square_error (mse) was used since the response variable is numeric, which means is a regression problem.
- mean_absolute_error (mae) as a loss function was used
  Adam was used, adam is the best optimizers. when training the neural network in less time and more efficiently. (Paszke, 2016)

Step 4: fit the model

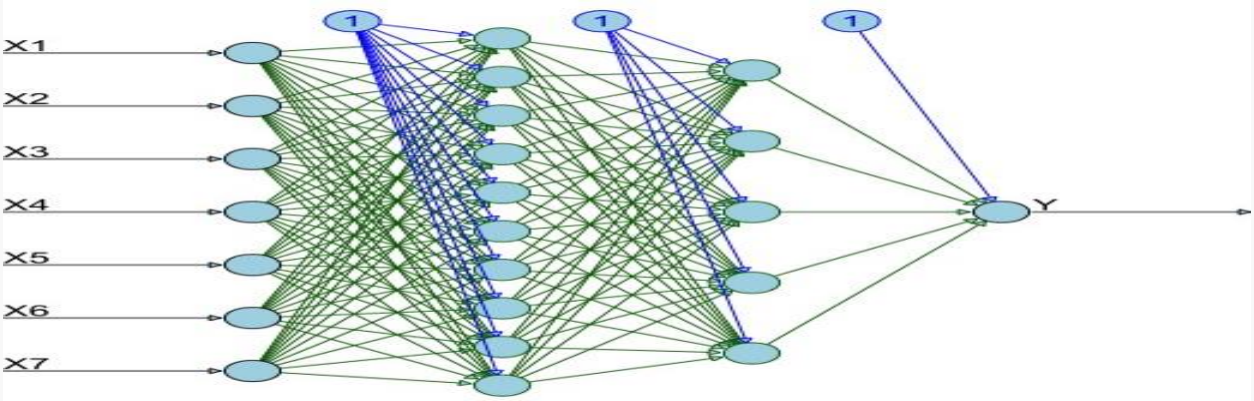
The dplyr function was used to connect the model to fit model. The training data and the target training respond variable, epoch is set to 100, batch size set to 1, and validation split is set to 0.2 were inputted into the fit model. Validation split was used for cross validation, by keeping 20% of the of the training dataset to calculate the out of sample error.

The plot below shows the neural network with 10 neurons in the first hidden layer, 5 neurons in the second hidden layer, 7 input predictor variables with one neuron each, and the outer layer have one node for the response variable Y.

```
# Neural Network Visualisation
n <- neuralnet(Y~ X1 + X2 + X3 + X4 + X5 + X6 +X7, data = data1, hidden = c(10,5) ,linear.output = F,
               lifesign = "full",rep = 1)
```

```
## hidden: 10, 5    thresh: 0.01    rep: 1/1    steps:     23  error: 2958.83406   time: 0.03 secs
```

```
plot(n, col.hidden = "darkgreen", col.hidden.synapse = "darkgreen", show.weights = FALSE,
    fill = "lightblue", information = FALSE)
```



Interpretation of the visualisation:
We have two hidden layer there are 512 parameters in the first layer, the input layer have 7 neurons and first layers have 64 neurons there is a constant neurons connected to the first neuron the model compute the parameter in this format (7 X 64) + 64 = 512. Similarly, we have 64 neuron in the second hidden layer connected to the first neuron and to a constant (64X64) + 64 = 4160 and the output layer have one neuron which is connected to the first neuron which is (1X64) + 1 = 65. The model has total of 4,737 parameters

```
## converting to matrix
data1 <- as.matrix(data1)
dimnames(data1) <- NULL
```

Neural Network algorithm using keras package in Rstudio

```
## partioning data1
set.seed(1234)

ind <- sample(2, nrow(data1), replace = TRUE, prob = c(0.8, 0.2))

nn_training <- data1[ind==1, 2:8]
train_target <-data1[ind==1, 1]

nn_testing <-data1[ind==2, 2:8]
test_target <-data1[ind==2, 1]
```

Scaling the data

```
## Normalising
m <-colMeans(nn_training)
s <- apply(nn_training, 2, sd)
nn_training <-scale(nn_training, center = m, scale = s)
nn_testing <-scale(nn_testing, center = m, scale = s)
```

Crerating the model

```r
# creating the model
set.seed(1234)
model <- keras_model_sequential()
model %>%
 layer_dense(units = 64, activation = "relu", input_shape = c(7)) %>%
 layer_dense(units = 64, activation = "relu") %>%
           layer_dense(units = 1)

print(model)
```

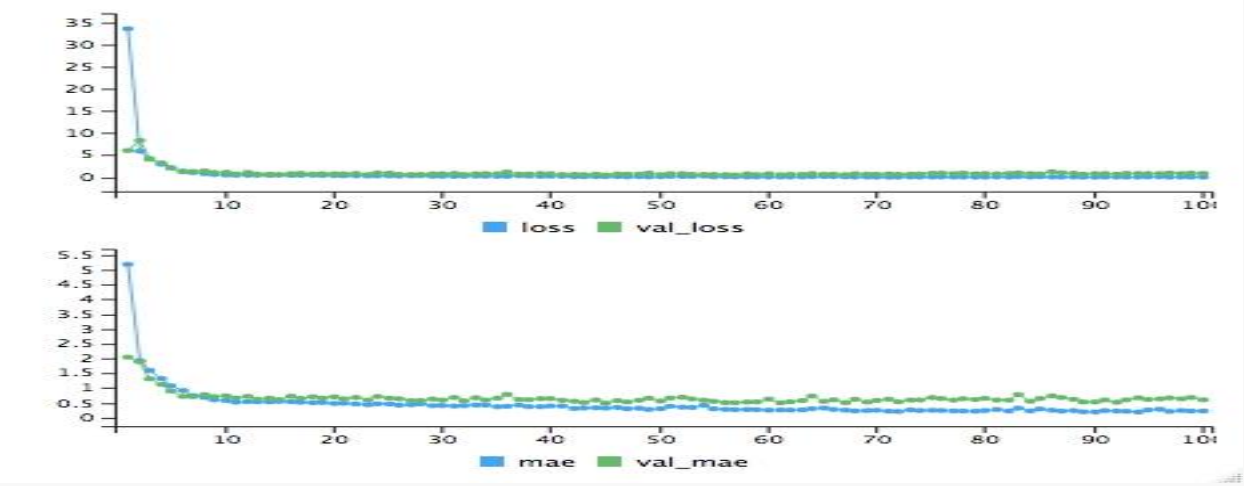
```
## Model
## Model: "sequential"
##
_________________________________________________________________________________
## Layer (type)                 Output Shape              Param #
##
=================================================================================
## dense_2 (Dense)              (None, 64)                512
##
_________________________________________________________________________________
## dense_1 (Dense)              (None, 64)                4160
##
_________________________________________________________________________________
## dense (Dense)                (None, 1)                 65
##
=================================================================================
## Total params: 4,737
## Trainable params: 4,737
## Non-trainable params: 0
##
_________________________________________________________________________________
```

Interpretation of the summary:

We have two hidden layer there are 512 parameters in the first layer, the input layer have 7 neurons and first layers have 64 neurons there is a constant neurons connected to the first neuron the model compute the parameter in this format (7 X 64) + 64 = 512. Similarly, we have 64 neuron in the second hidden layer connected to the first neuron and to a constant (64X64) + 64 = 4160 and the output layer have one neuron which is connected to the first neuron which is (1X64) + 1 = 65. The model has total of 4,737 parameters

```r
set.seed(1234)
# compile
model %>% compile(optimizer = "adam",loss = "mse", metrics = "mae")=
# Fit model
nn_model <- model %>%
 fit(nn_training, train_target, epochs = 100, batch_size = 1, validation_data = list(nn_testing, test_t
```

arget))
print(nn_model)



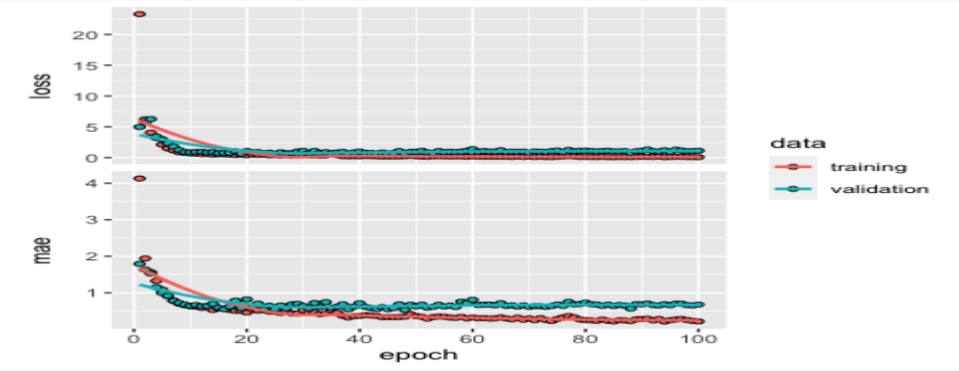
The first graph above shows the plot of mean square error or validation and means absolute error (mse). The green line represents the validation lost or means square error while the blue line represents the mean square error based on the data used to develop the model.
In the second plot of mean_absolute_error (mae) graph, the green line represents the mean_absolute_error (mae) for the model while the blue colour represents the mean_absolute_error (mae) base on the data that is been used in developing the model. The graph shows that the error rate is very minimal.

print(nn_model)

```
##
## Final epoch (plot to see history):
##     loss: 0.07621
##      mae: 0.2149
## val_loss: 1.121
##  val_mae: 0.676
```

plot(nn_model)

```
## `geom_smooth()` using formula 'y ~ x'
```



The graph above show the loss and mae plot of training and validation (testing) data. The loss plot indicate that model has similar performance on both training and validation dataset (target test). The epoch became almost constant from 20 epoch.
The mean absolute error (mae) plot shows that the error rate was very minima.

```
# Evaluation

score <-model %>% evaluate(nn_testing, test_target)
print(score)
```

```
##     loss      mae
## 1.1214371 0.6759767
```

Prediction and confusion matrix

```
pred_nn <- model %>% predict(nn_testing)

tab <-table(test_target, pred_nn)

accuracy2 <-1 - sum(diag(tab))/sum(tab)
accuracy2
```

```
## [1] 0.9666667
```

The prediction has an accuracy of 96.7%

5.0 Comparison and Contrasting ML and NN models

Random forest can have a different interpretation of a decision tree but with improved performance . Neural Networks will need larger data to be effective.
Random forest is quite easy to interpret, neural is complex in terms of interpretation of the model. The cross-validation shows that the random forest was slightly better than the neural network because random forest had an accuracy of 96.8 % while neutral network (NN) had an accuracy of 96 .7%. The neural network also had a longer computation time in training the model compare to the random forest model. Neutral network is mostly used for large datasets and because the training data in this case was small, there are tendencies of bias in the model.

6.0 Conclusions/findings

This analysis compared the performance of the broadly used neural network (NN), and random forest (RF) algorithm to evaluate the unemployment rate in Australia. Based on the performance m etric used in this analysis, the random forest method performed marginally better than the neural network. Both of this model performed very well on the training and validation data. However, from both results, it is concluded that both models can be viable for the analysis.

Reference

Gad, I., Hosahalli, D., Manjunatha, B. R., & Ghoneim, O. A. (2020). A robust deep learning model for missing value imputation in big NCDC dataset. *Iran Journal of Computer Science*, 1-18.

Paszke, A., Chaurasia, A., Kim, S., & Culurciello, E. (2016). Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint arXiv:1606.02147*.

Rodriguez-Galiano, V., Sanchez-Castillo, M., Chica-Olmo, M., & Chica-Rivas, M. J. O. G. R. (2015). Machine learning predictive models for mineral prospectivity: An evaluation of neural networks, random forest, regression trees and support vector machines. *Ore Geology Reviews*, *71*, 804-818.

Xu, Y., Du, J., Dai, L. R., & Lee, C. H. (2014). A regression approach to speech enhancement based on deep neural networks. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, *23*(1), 7-19.

Appendix:

```
data <-read.table("/Users/macbookpro/Downloads/AUS_Data.csv", sep = ",", na.strings = "?",
header = TRUE)

## Tranforming data in numeric values
data <-transform(data, Y = as.numeric(Y), X1 = as.numeric(X1), X2 = as.numeric(X2),
            X3 = as.numeric(X3), X4 = as.numeric(X4), X5 = as.numeric(X5),
            X6 = as.numeric(X6), X7 = as.numeric(X7))
data <-data[-1,]

# Data partition
set.seed(1234)
index <- sample(2, nrow(data_m), replace = TRUE, prob = c(0.8, 0.2))

training <- data_m[index ==1,]
testing <- data_m[index==2,]

rfm_au <- randomForest(Y ~ X1 + X2 + X3 + X4 + X5 + X6 +X7,
                  data = training, mtry = 4, importance = TRUE)
print(rfm_au)
```