Data Analytics

Report on Bus Services

Eashwar Thyagarajan - 6713922

Mansi Goyal - 6709502

Neha Jogula - 6707944

Joseph James - 6713109

Executive Summary

The data provided from the surrey county comprises of 37 bus operators (TILL 2018). A total of 37 bus operators with 270 routes with different bus service numbers are operating in surrey. For our analysis, we have considered a subset of 87 routes with different service numbers. The set goal of the data analytics project is to make a prediction for the travellers(passengers) based on the existing and the updated routes for the county of surrey. The software which are used are RStudio, Power BI, PostgreSQL and PGAdmin. In this report, the data provided will be processed which means data mining, data exploration and data analysis will be done. Initially, server and data connection is done with the use of PostgreSQL in the PgAdmin software. For data analysis, the tables and other data will be viewed and understood using Power BI and SQL. Even for graphical data visualisation, Power BI is used extensively. The Prediction Models like linear regression, decision tree, neural networks and random forest are created to understand the prediction of the latest routes and all this is done in R.

Literature Review

"Short-term bus passenger demand prediction based on time series model and interactive multiple model approach"

The association and periodicity were used to create models. The heteroscedasticity of time series was explored in particularly to increase predictive performance. By using IMM filter method, individual forecasting models were coupled with dynamically forecasted passenger demand for the next interval. The interactive multiple model (IMM) filter algorithm-based models proposed in the research work are used to forecast short-term passenger needs. To integrate distinct aspects of time - series data, the related prediction models ARMA, SARIMA, and ARIMA were established. The error indices MAE, RMSE, MAPE, and vape were utilised to analyse independent and hybrid algorithms. The suggested hybrid model beats the single predictions system in respect of precision and resilience, according to the achievement evaluation.

"A deep learning based architecture for metro passenger flow prediction"

This research tries to evaluate metro passenger flow by using deep learning modeling capabilities and domain expertise in transportation. Deep Passenger Flow (DeepPF), an end-to-end deep learning architecture for forecasting metro inbound/outbound passenger flow is introduced in this paper. The design of the proposed model is exceptionally adaptable and extensible, acknowledging integration and modeling of external environmental elements, temporal dependencies, geographical features, and metro operational qualities in short-term metro passenger flow forecasting. Furthermore, due to the simplicity with which multi-source data may be integrated, the suggested framework attains a high prediction accuracy. Symmetric Mean Absolute Percent Error (SMAPE), Root Mean Square Error (RMSE), Mean Absolute Error (MAE), and Mean Relative Error (MRE) are just the performance methods applied in this study (MRE).

"Predicting short-term bus passenger demand using a pattern hybrid approach"

This study presents an Interactive Multiple Model-based Pattern Hybrid (IMMPH) technique to forsee short-term passenger demand. By assembling knowledge from pattern models using historical data and improving the interaction between them using real-time observations, the technique boosts the effective information content. It can anticipate the priori pattern model combination for the following time interval dynamically. Over the course of a year, the source demand data was accumulated using a Smart Card system along one bus service route. The weekly (weekend/weekday), daily (off-peak/peak), and hourly pattern time series are produced as time relevance pattern time series. Furthermore, in this study, statistic design algorithms are created to incorporate various time-series patterns. Finally, an updated IMM method is used to dynamically collaborate the pattern model estimations to achieve the final result.

"Using a simulated annealing algorithm to solve the transit route network design problem"

This research paper aims to resolve the ideal bus transit route network design issue BTRNDP at the distribution node level by employing a simulated annealing approach. For the BTRNDP, a multi-objective nonlinear mixed integer model is evolved. The proposed solution framework consists of three segments: an initial candidate route set generation procedure that generates all achievable routes while taking into consideration practical bus transit industry guidelines; a network analysis procedure that assigns transit trips, determines service frequencies, and computes performance measures; and a simulated annealing procedure that incorporates these two parts, guides the candidate solution generation process, and finally determines an optimal candidate solution. As part of a pilot research, the three experimental networks were successfully tested for efficiency. Sensitivity studies are executed, as well as associated features and trade-offs reinforcing the BTRNDP.

"Urban bus transit route network design using genetic algorithm"

The application of genetic algorithms (GAs), research and optimizing derived from natural genes & selecting, to the route network design problem is discussed in this research. There are two stages to the design. Initially, a set of possible paths are developed that compete for the best approach. Next, a GA is used to choose the best selection. The GA is addressed by combining a traditional constant string length coding technique with something like a unique variable string length coding strategy introduced in the paper to tackle the problem. The latter considers the size of the solutions route set and uses a GA to identify the several optimal paths out from the candidate route set. This approach is verified on a system of case studies, and thus the findings are provided.

“A simple bus line model for optimisation of service frequency and bus size”

This study investigates the implications of integrating some social expenses in bus operational analytics. Passengers’ work, wait, travelling, and other social expenses are included. Those were monetised and factored in along with producer expenses, effectively establishing passenger intakes as an aspect of bus transportation supply pricing. The investigation is limited to the supply sector. That will be impossible to identify whatever the best degree of supplies will be in a given circumstance because the demand side has been omitted. Nonetheless, using a basic bus line model, it’s really easy to demonstrate that social expense minimization produces a sequence of service features that is fundamentally distinct from most current services, particularly in these areas: provided demand, more buses must be operated, small buses etc.

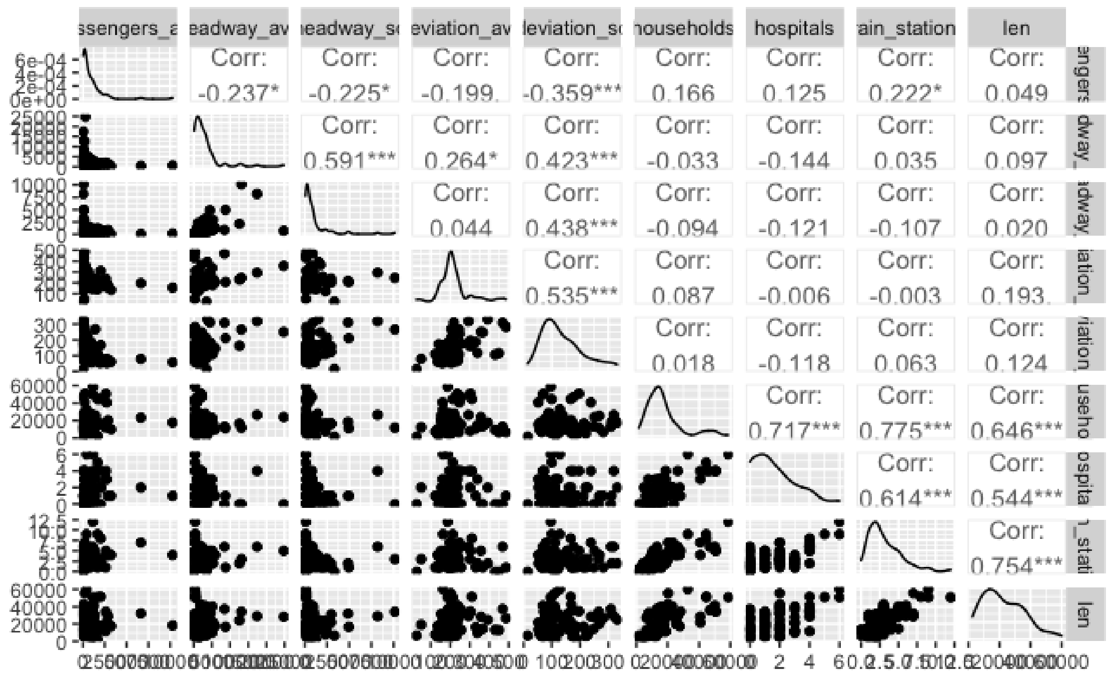
Data Explanation and preparation

In the southeast of England is the county Surrey with an area of 1,663 km2 with a population of 1,189,934 (estimate in 2018). Surrey is a county with a population of around 1.18 million (1,189,934) within an area of 1663 km2. There are almost 29.1K postal codes accordingly in this beautiful county. The Royal Mail- PAF (Postcode Address File) and Ordnance survey’s portal provided the data of general geographical data, Delivery points and addresses required for the analysis. All this information is taken from BusAnalytics.uk. The Database we are using has a population of 1.16M and 1.86M including the buffer. There are 2.41 people per single household on average and up to 480k households in the county according to estimation in 2018. There is a total of 37 bus operators from the county of surrey shown in our bus service database. There are many other bus operators present in the county but due to the non-disclosure agreement, they cannot be mentioned or considered. So, a total of 37 bus operators with 270 routes with different bus service numbers are operating in surrey. For our analysis, we have considered a subset of 87 routes with different service numbers. To summarise, actions will be broken down into the following categories: 1. Machine Learning Algorithms are used for new routes. 2. Time Series Analysis is used for current Routes Machine learning algorithms are done in R programming, but time series analysis was performed in both R and Power BI. SQL queries are used to perform data mining straight from the pgAdmin server. The map of Surrey bus routes is made up of ten separate tables, they are: Routes_aggr, routes _daily, bus_route, bus _stop, train _station, hospital, household, household_domestic_surrey, town, surrey_boundary. During data processing, mostly two sets of data were employed.
Routes_aggr: Routes data for a long period of time
Routes_daily: database on daily routes • There are numerical and character elements in two tables. • The target variable in any forecasting model is the passenger’s variable, which is also the dependent variable. There are too many records in the original tables route_daily and route_aggregate, which must be adjusted. Data cleansing is the process of eliminating any duplicate entries from a database so that only distinct observations are

present in the new derived tables employing SQL queries. A new table for hospital variable was created, and then this table was modified by adding a column (geom 4326) and updating it. In addition, SQL queries revealed the presence of train station in the buffer area. There are 39 Domestic Hospitals in the above setup and each hospital has a varied number of domestic households around them with a 1000m , such as 5444 household are present around The Old Cottage Hospital and 2452 households are present around Guildford Nuffield Hospital. Following data cleansing, it was discovered that the routes aggregate dataset had 244 different rows, whilst the daily table included 57,834 rows. Several tables, such as household domestic surrey & hospital, also are cleansed in the same way. Passengers, weekday, peak, deviation, household, and other factors are among the 14 modelling variables present in the data.

Correlation

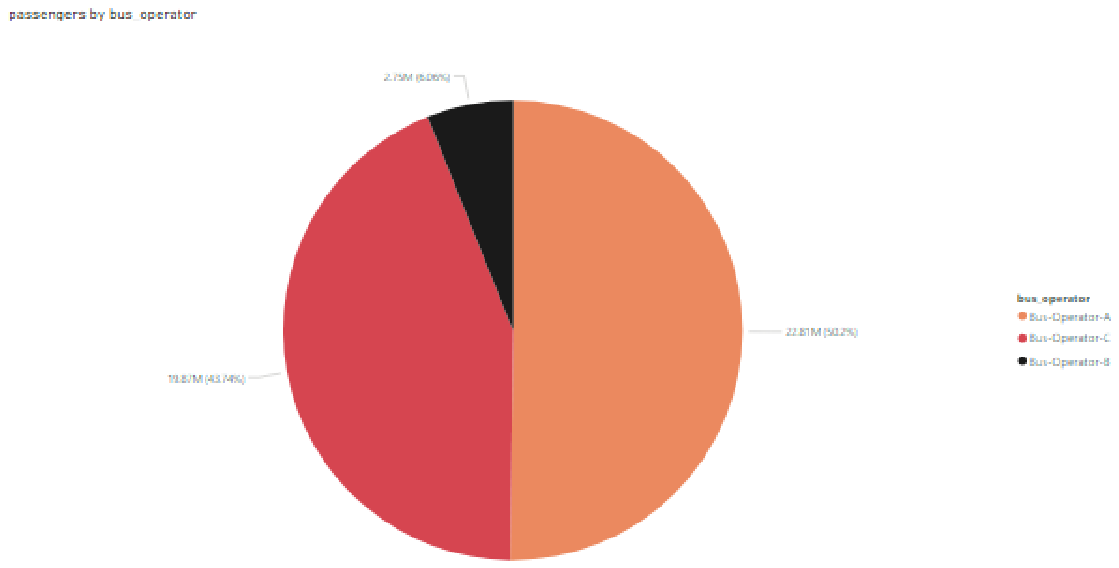
The fields which are used for this project are: bus_route, bus_stop, train_station, hospital, household, household_domestic_surrey, town, surrey_boundary, routes_aggr, routes_daily. The given variables were tested for correlation A correlation matrix has been utilised to examine the relationship amongst numerous variables at once. Overall route information is collected by the bus company, service (route), peak, and weekday. For passengers, headways, and variances, the daily averages were averaged again, and also the standard deviation of regular averages was calculated. There were no differences in the homes, hospitals, railroad stations, or route length. The correlation matrix among variables is used to determine the link among elements as well as the importance of the association. The response variable in the evaluation is passenger, and that all analyses are built around it. The passenger number is derived from the ticket information gathered. ggpairs() is the command used to graphically & statistically illustrate the relationship among variables. Households have a substantial favourable link towards train stations and hospitals, as seen.



(Figure1: correlation)

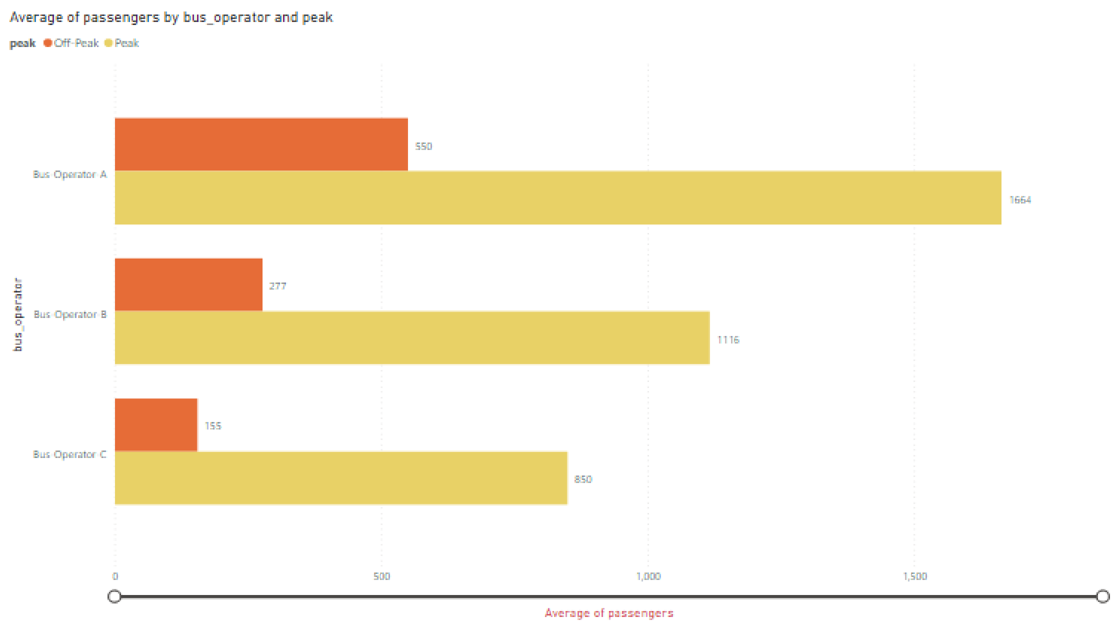
Data visualisation

We have used Power BI for the data visualisation where we plotted some charts and graphs between different fields to visualize and analyse the data for a better prediction of the model. We have created pie charts, bar graphs, stacked graphs, key influencer charts and histograms.



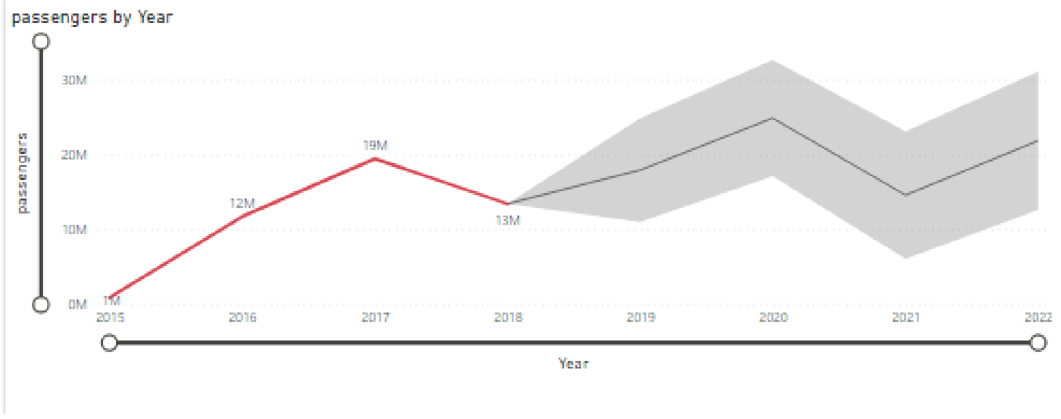
(Figure2: visualisation of passengers by bus operator)

Here a pie chart is created with passengers and bus_operators as data fields as shown in the above figure. From the given pie chart, we can observe that Bus-Operator-A has the highest passengers at 22,809,672.48 followed by Bus-Operator-C at 19,872,869.81 and Bus-Operator-B at 2,751,726.13. Bus-Operator-A accounted for 50.20% of total passengers being the highest whereas Bus Operator-B with lowest percentage of 6.06% of total population.



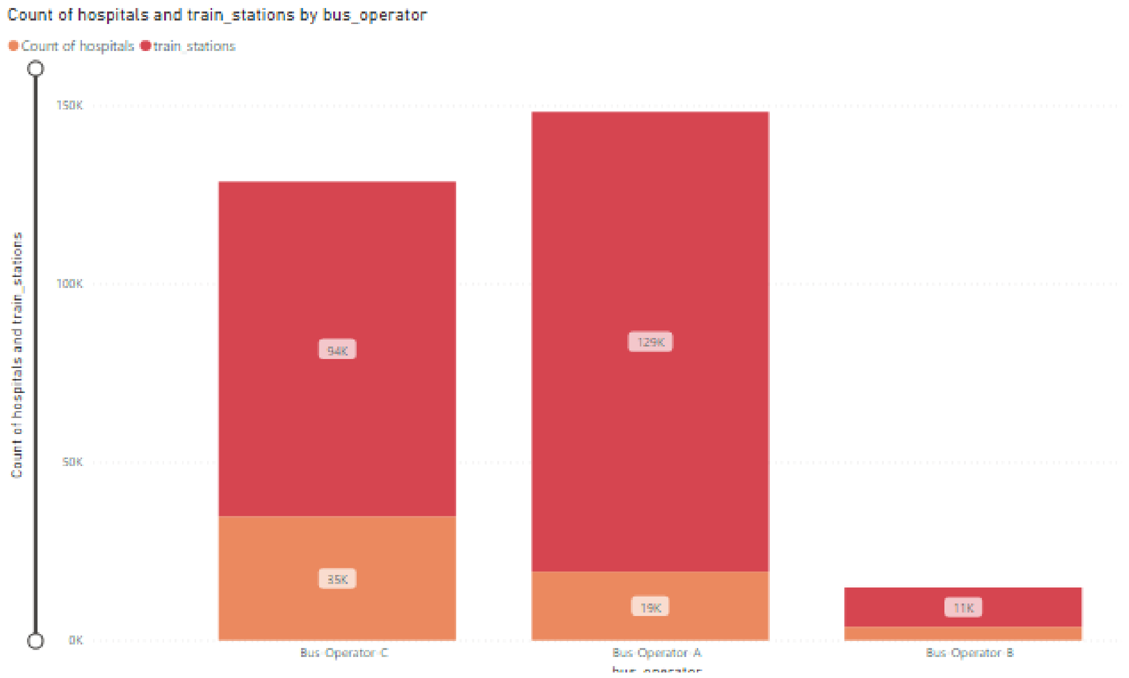
(Figure3: visualisation of Average of passengers by peak and off-peak)

In the above graph we have generated a Cluster bar chart for average of passengers by bus_operator and peak. We can perceive that average of passengers was higher for Peak (1,209.88) than Off-Peak (327.35). In general,peak time (7:00 to 17:00) people travel more as these are the general working hours. Bus-Operator-A in peak made up 36.07% of Average of passengers. Average of passengers for Peak and Off-Peak diverged the most when the bus was operated by Bus-Operator-A, when Peak were 1,113.43 higher than Off-Peak. Bus-operator-B and bus-operator-C have an average very similar to each other as we can see in the graph given.



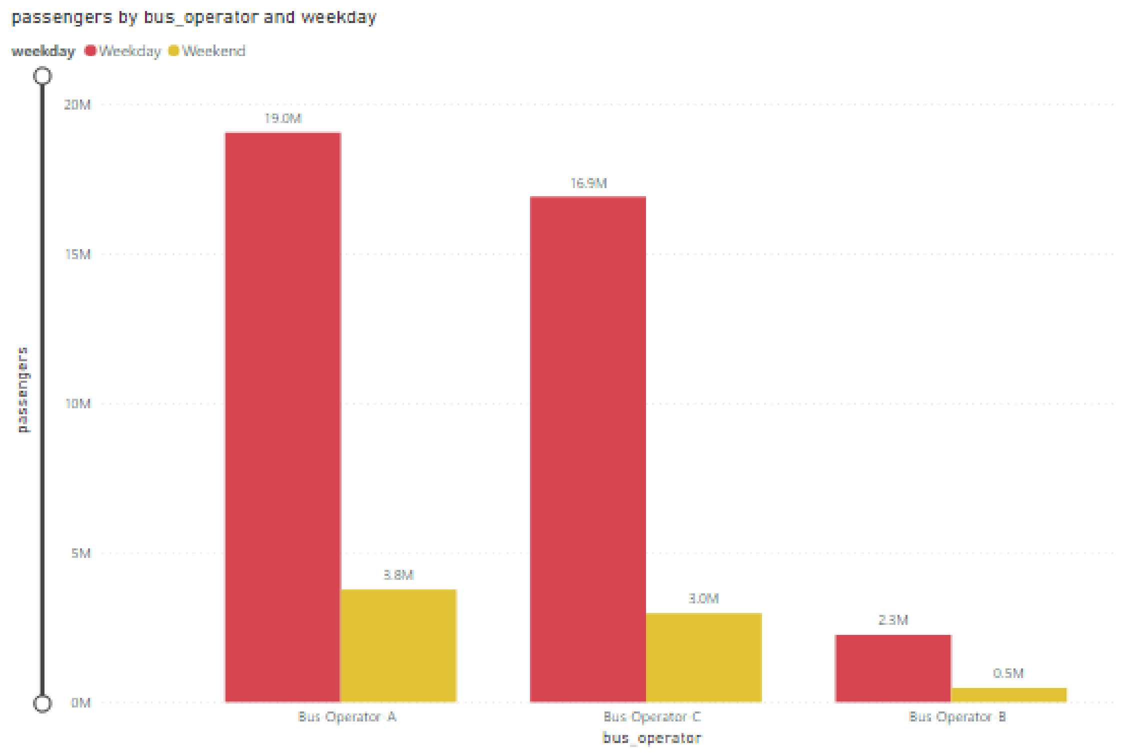
(Figure4:visualisation of passengers by year)

The above graph shows the Number of passengers increased for the last 4 years on record according to our data. We have used a line chart to create this graph between Passengers and year. Passengers number jumped from 843,377.65 to 19,454,432.35 during its steepest incline from 2015 and 2017 and then it has gradually declined to 13,431,697.56 passengers in 2018. In spite of the small decline passengers experienced the longest period of growth (+12,588,319.91) between 2015 and 2018 only. Based on the data provided from 2015 to 2018 the line chart of the passengers is predicted till the year 2022 in the given chart below.



(Figure5: visualisation of count of hospitals and train_stations by bus_operator)

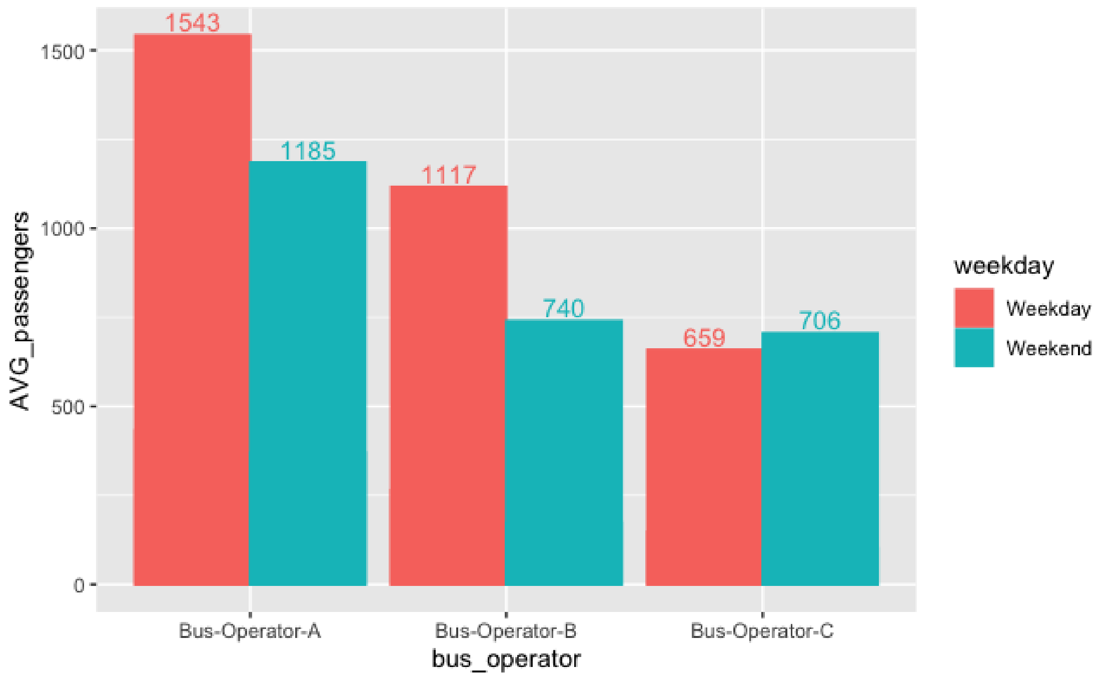
This graph was created using a line and stacked column chart. The graph shows the count of hospitals and train stations by the bus_operator. At 34,755, Bus-Operator-C had the highest Count of hospitals and was 818.47% higher than Bus-Operator-B, which had the lowest Count of hospitals at 3,784. We can see that the Count of hospitals and total train_stations are positively correlated with each other. Here Bus-Operator-C accounted for 60.09% of Count of hospitals. Whereas Bus-Operator-A had 19,295 Count of hospitals but had the highest number of train stations which is 128902. Now coming to Bus-Operator-B, it has the least Count of hospitals as well as least number of train_stations with 11092 train stations only.



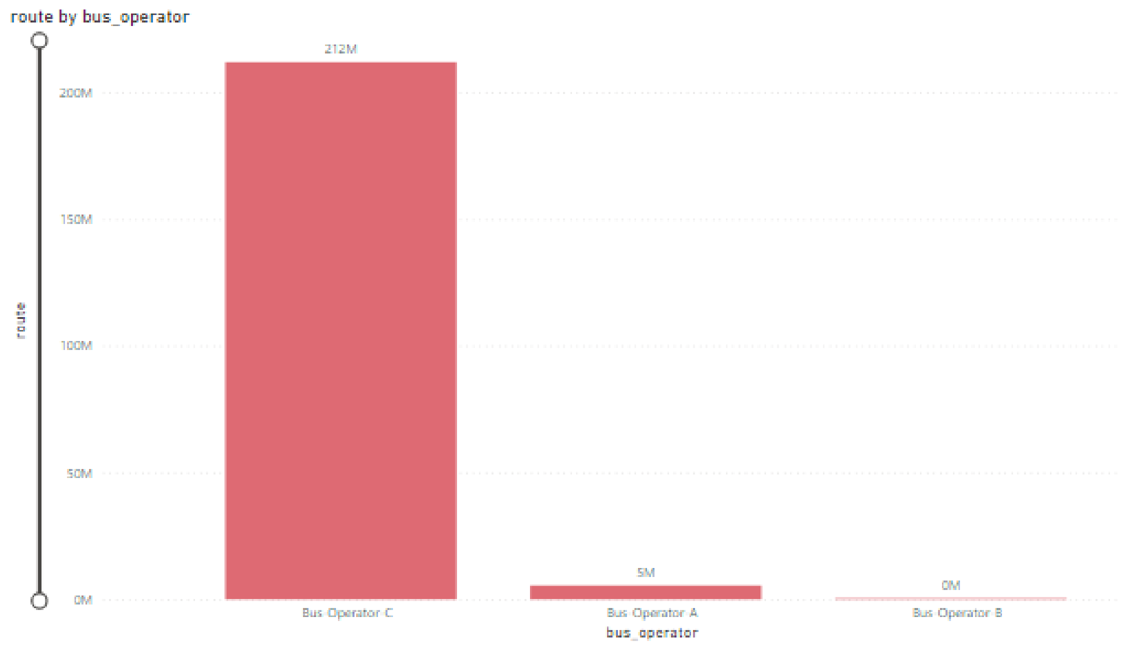
(Figure6: visualisation of passengers by bus_operator and weekday)

The above graph is made by using a clustered column chart in power BI. This graph is for passengers by bus_operator and weekday fields. We can visualize that average passenger was higher for Weekday (12,736,541.66) than Weekend (2,408,214.48) as majority of the people move around more in the weekdays because they need to go to work and complete other daily activities in the weekdays. So in total passengers was higher for Weekday (38,209,624.97) than Weekend (7,224,643.44). We can observe that Bus-Operator-A in weekday made up 41.91% of passengers and passengers for Weekday and Weekend diverged the most with Bus-Operator-A, when Weekday were 15,276,834.02 higher than Weekend. Bus operator-B as usual has the least number of passengers in both weekday and weekend respectively.

Histogram using R:



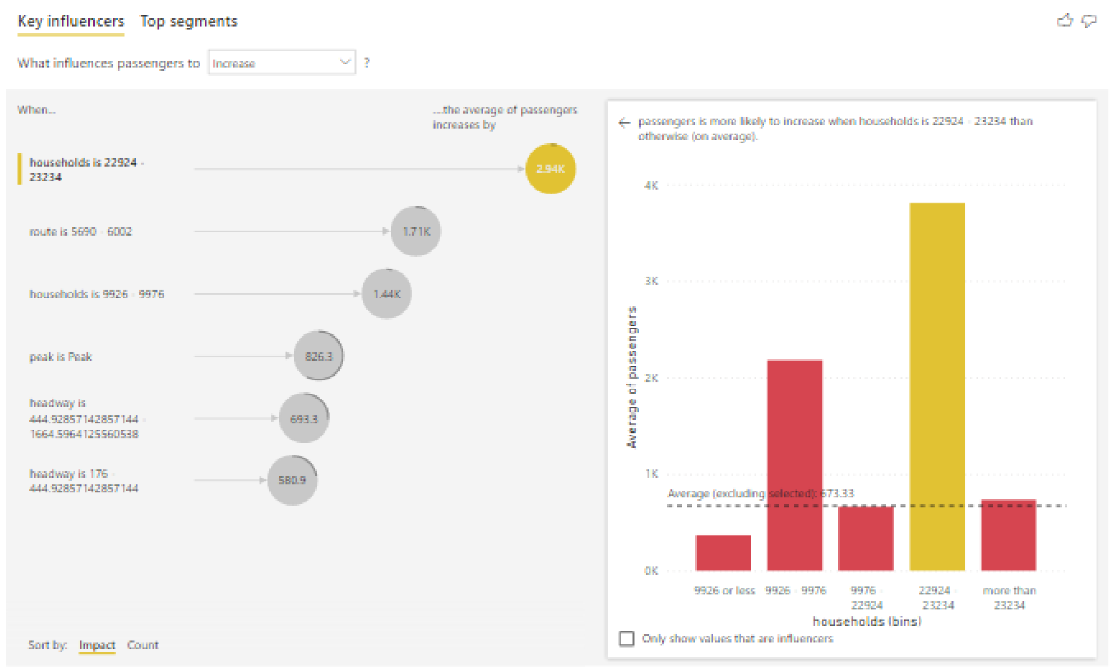
(Figure7: Histogram representation of weekday and weekend using R)



(Figure8: visualisation of route by bus_operator)

The above chart is created between the routes and all the bus operators present in the data. In this graph we can observe that at 211846012, Bus-Operator-C has the highest route and is 81,881.21% higher than Bus-Operator-B, which has the lowest route at 258408. Bus-Operator-C has the highest route at 211846012, followed by Bus-Operator-A at 5472916. Bus-Operator-C has alone accounted for 97.37% of route leaving Bus-Operator-A and Bus-

Operator-B very few routes in total. From here we can understand that the reason Bus operator-C has been standing out in every other visualization of the data.



(Figure9: visualisation of Key Influencers)

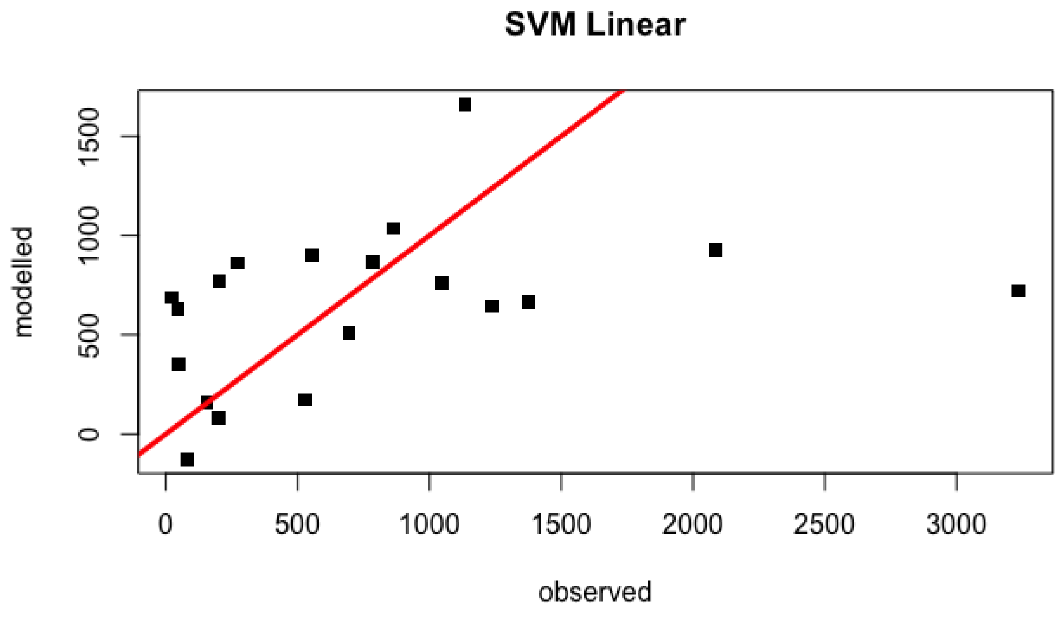
The given graph is showing the key influencers. From the key influencers we can observe that passengers are more likely to increase when household is 22924-23234 than otherwise on average. It is also found that the average of passengers increases across route 5690-6002 with 1.71k and during peak hours with 826.3.

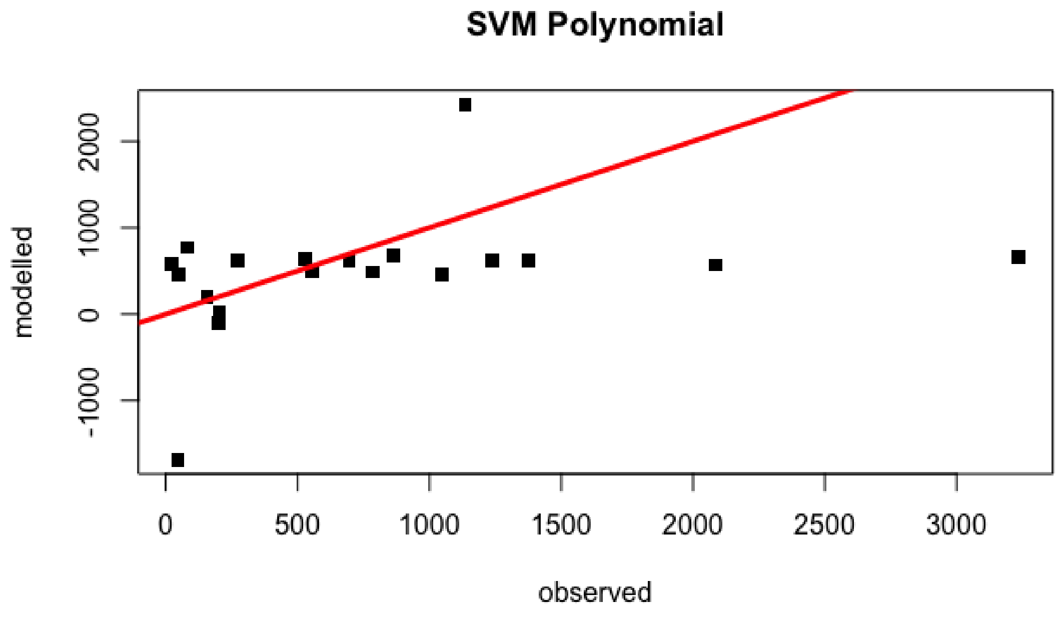
Prediction / Classifications of Models

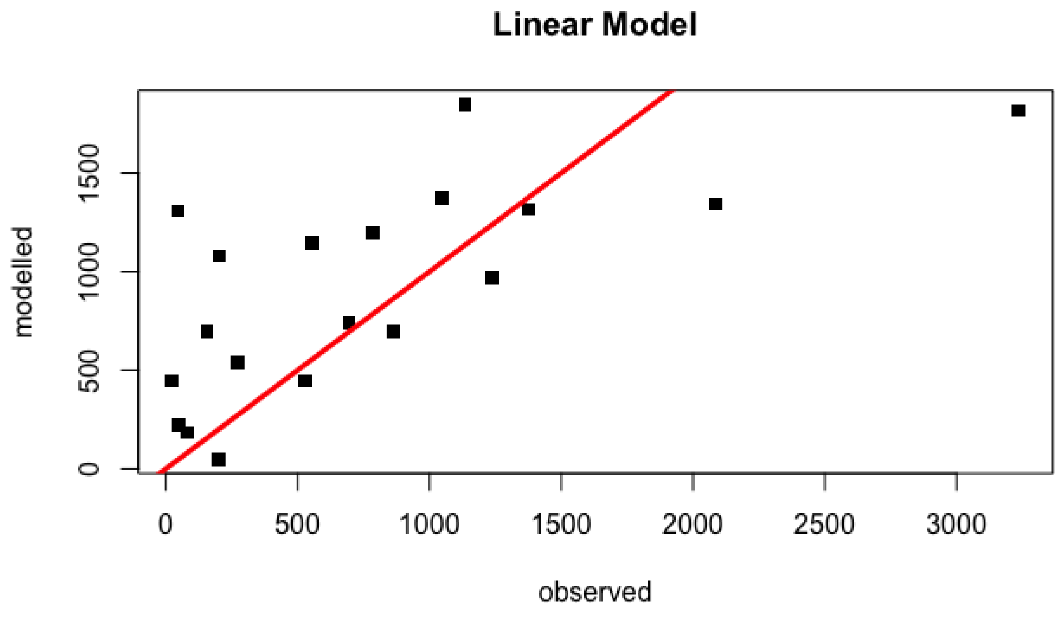
The dataset is split into "peak" and "weekday" segments. The data set was first segmented into train and test data, with the models being tested on various splits such as 70:30, 75:25 and 80:20 on many models. A linear model, linear regression model using the normal equation, Decision Tree, Random Forest, Neural Networks, and SVM (Support Vector Machines) for linear and polynomial performance were used to determine the results. The metrics used to assess each model's performance were Rmse (root mean square error) and mape (mean average percentage error). In a regression study, the root mean square error (RMSE) is a quadratic scoring method that additionally calculates the average size of the error. So basically, the model prediction error is measured as RMSE. In other words, it gives how densely the data is clustered around the best-fit line. It's the square root of the average of squared differences between the predicted and observed values. The model is better if the RMSE is low. A forecast system's accuracy is measured by the mean absolute percentage error (MAPE). It is determined as the average absolute percent error for each time period minus actual values divided by actual values, and it is expressed as a percentage. It is a simple average of absolute percentage errors.

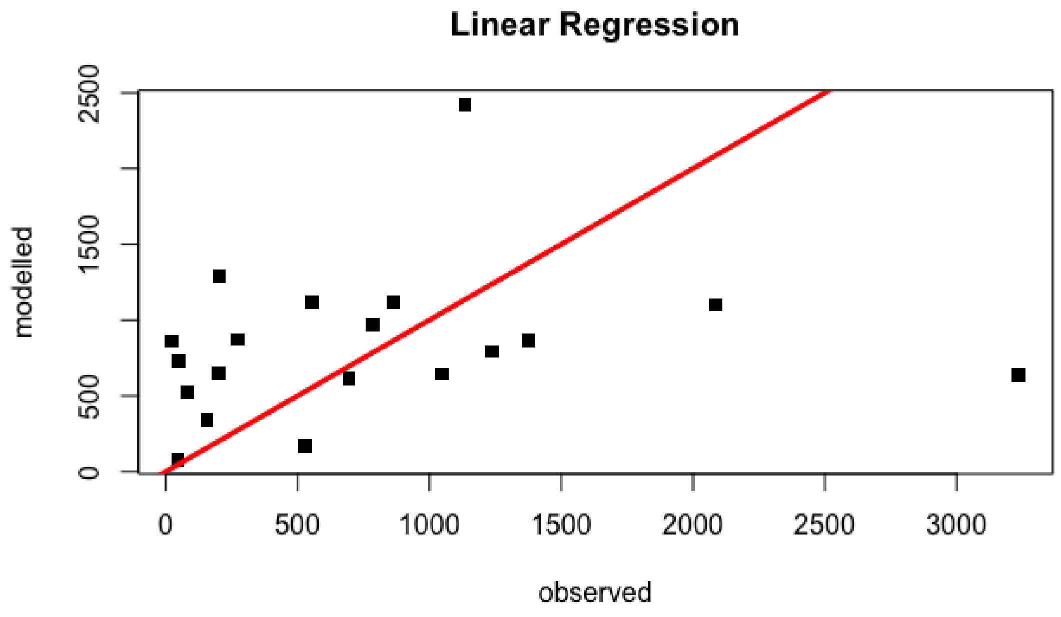
MAPE is not scale-dependent, although RMSE is. So, you can't use RMSE to compare accuracy across time series with various scales. MAPE is frequently used in business since it appears that managers grasp percentages better than squared errors. Because our dataset contains numerous outliers, it is advised that we use the model with the lowest MAPE percent for routes_aggr distinct. Below are the scatter plots showing the Frequency distribution of all the prediction models. All the models were tested on the data sets in order to obtain the lowest RMSE and MAPE for various data distributions which is shown in the below table for comparison purposes.

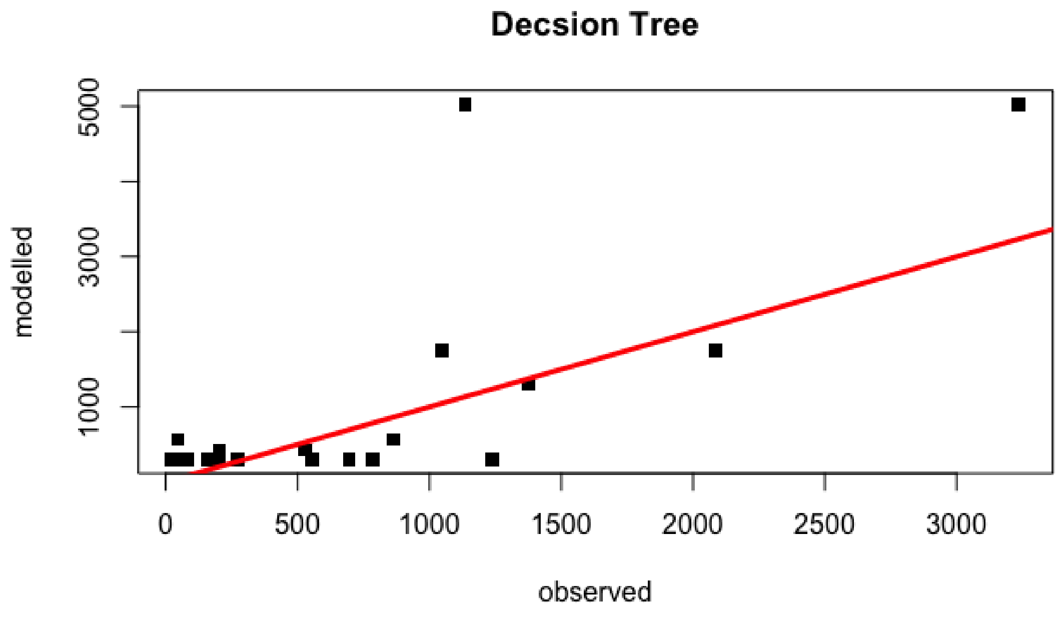
As a consequence, the random forest model was observed to have the least root mean square value with a 0.8 train test splitting, and it was eventually adopted as the prediction model. MAPE value is less than 100 for different training set as well (Neural network). As a result, it was determined that the Random Forest model outperformed competing models, and the number of passengers for new and existing routes was estimated. While the SVM model (both linear and polynomial) was implemented for RMSE and MAPE analyses, it was overlooked because it was better fitted for smaller datasets.

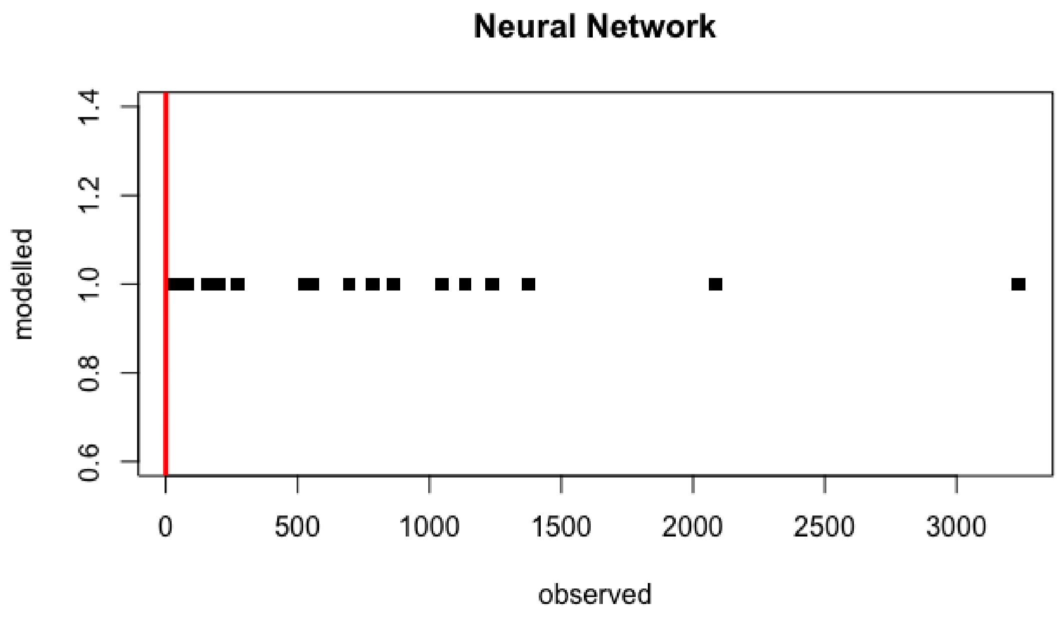

SVM Linear
modelled
observed
0
500
1000
1500
2000
2500
3000

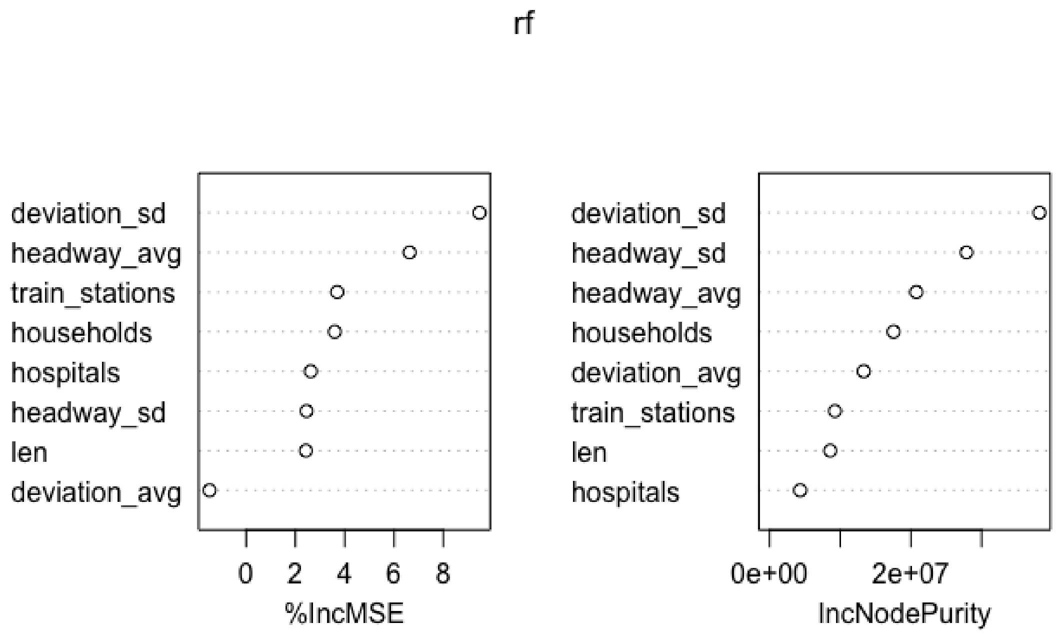

SVM Polynomial
modelled
observed
-1000
0
1000
2000
500
1500
2500
3000


Linear Model
modelled
observed
0
500
1000
1500
2000
2500
3000


Linear Regression
modelled
observed
0
500
1500
2500
1000
2000
3000


Decsion Tree
modelled
observed
0
500
1000
1500
2000
2500
3000
1000
3000
5000


Neural Network
modelled
observed
0
500
1000
1500
2000
2500
3000
0.6
0.8
1.0
1.2
1.4



	Data Trimming					
	70% Models		75% Models		80% Models	
Models	RMSE	MAPE	RMSE	MAPE	RMSE	MAPE
Linear Model	1124.6	289.20%	1206.1	334.00%	596.8	335.40%
Linear Regression	1326	281.60%	1407.6	306.10%	848.2	374.20%
Decision Tree	1018.3	224.10%	1004.1	200%	1045.9	216.80%
Random Forest	1114.1	272.30%	1170.3	299.40%	468.8	286.50%
Neural Networks	1717.5	99.50%	1805.8	99.40%	1105.6	99.30%
SVM(Linear)	1350.5	208.30%	1414.2	240.60%	753.3	312.90%
SVM(Polynomial)	1443.8	307.20%	1532.1	335.80%	923.3	458.80%

(Figure17: Prediction of models)

Summary & Conclusion

References

• Xue, R., Sun, D. (Jian) and Chen, S. (2015). Short-Term Bus Passenger Demand Prediction Based on Time Series Model and Interactive Multiple Model Approach. Discrete Dynamics in Nature and Society, 2015, pp.1–11.

• Yang Liu, Zhiyuan Liu, Ruo Jia, 'DeepPF: A deep learning based architecture for metro passenger flow prediction' (2019).

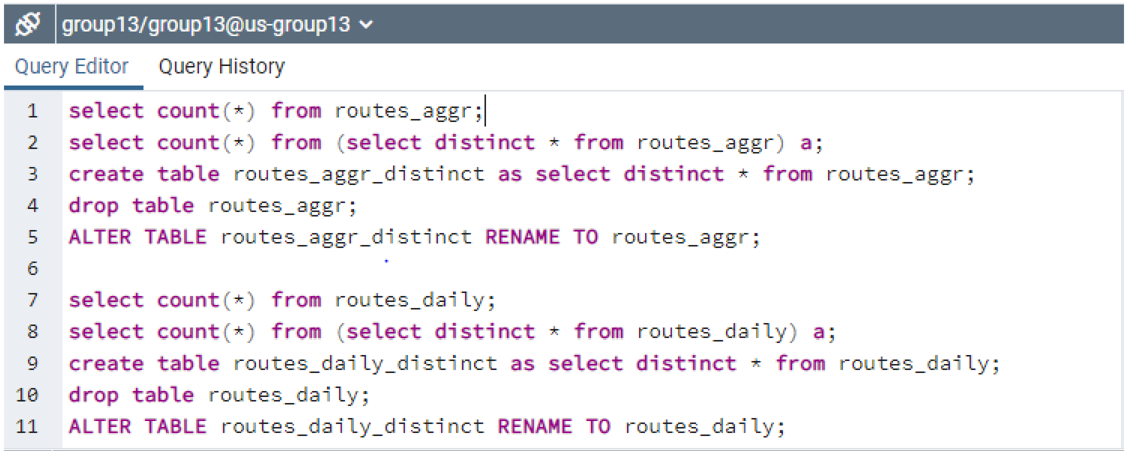
•Ma, Z., Xing, J., Mesbah, M. and Ferreira, L. (2014). Predicting short-term bus passenger demand using a pattern hybrid approach. Transportation Research Part C: Emerging Technologies, 39, pp.148–163.

• Fan, W. and Machemehl, R.B. (2006). Using a Simulated Annealing Algorithm to Solve the Transit Route Network Design Problem. Journal of Transportation Engineering, 132(2), pp.122–132.

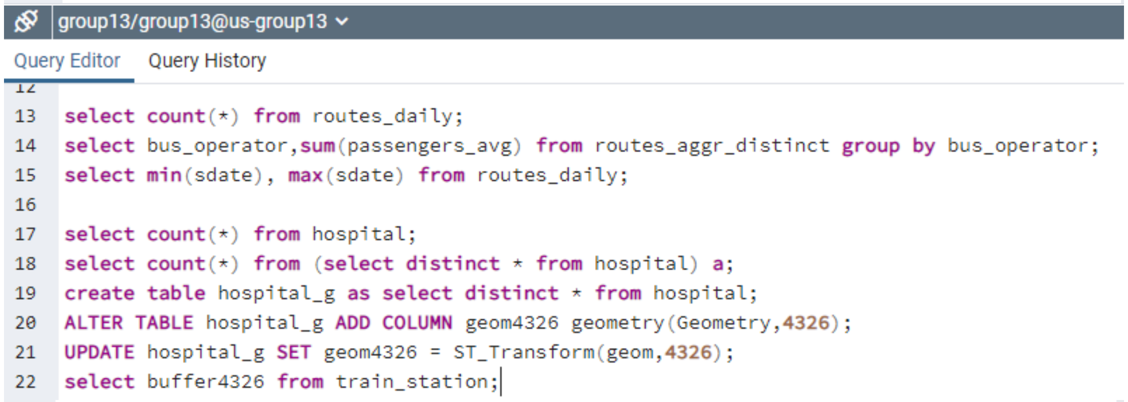
• Pattnaik, S.B., Mohan, S. and Tom, V.M. (1998). Urban Bus Transit Route Network Design Using Genetic Algorithm. Journal of Transportation Engineering, 124(4), pp.368–375.

• Jansson, J.O., 1980. A simple bus line model for optimisation of service frequency and bus size. Journal of Transport Economics and Policy, pp.53-80.

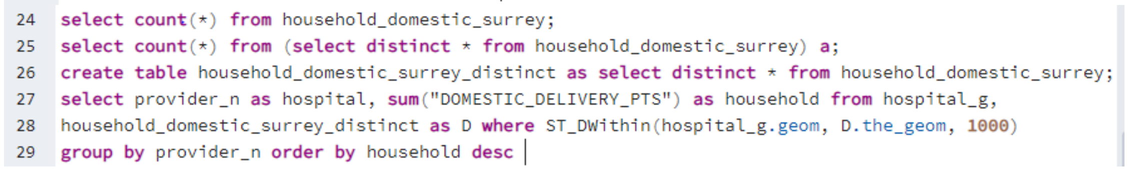
Appendix



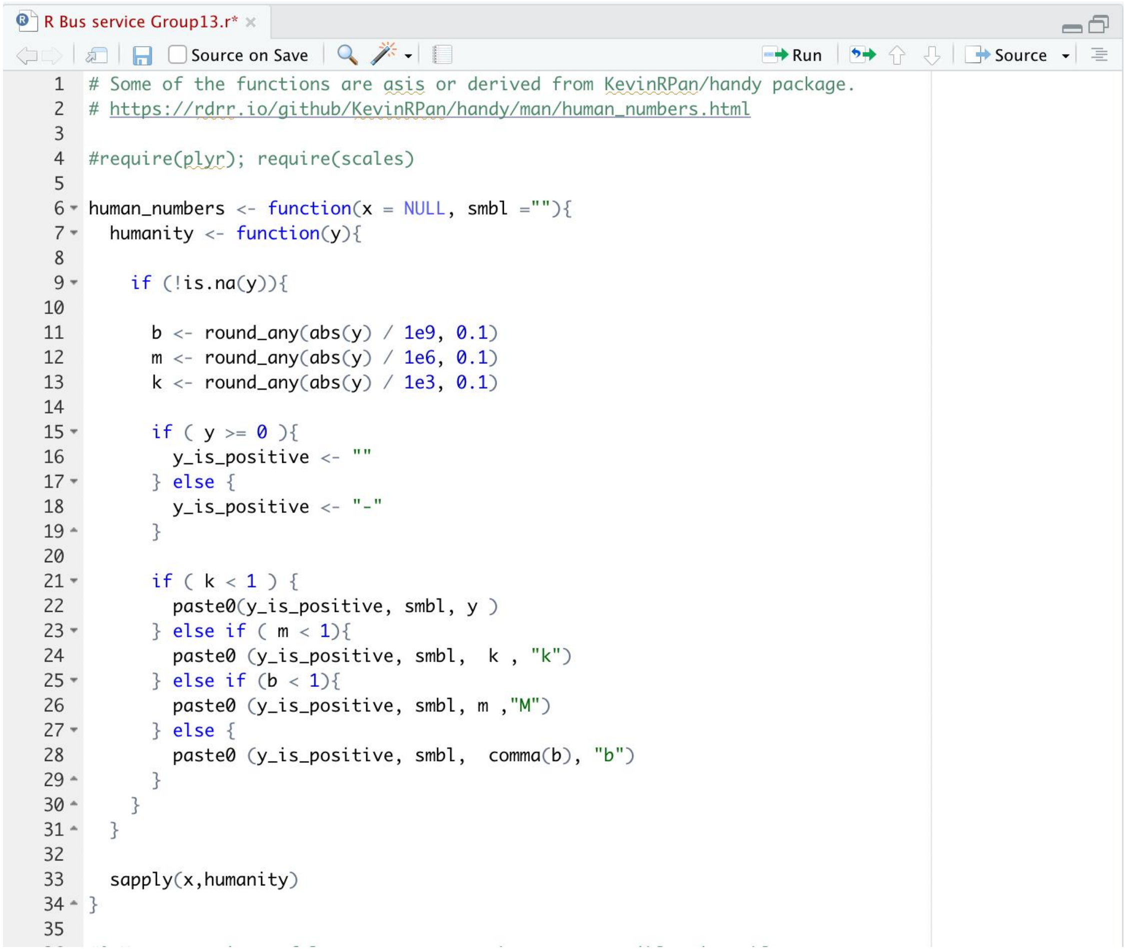
```
select count(*) from routes_aggr;
select count(*) from (select distinct * from routes_aggr) a;
create table routes_aggr_distinct as select distinct * from routes_aggr;
drop table routes_aggr;
ALTER TABLE routes_aggr_distinct RENAME TO routes_aggr;

select count(*) from routes_daily;
select count(*) from (select distinct * from routes_daily) a;
create table routes_daily_distinct as select distinct * from routes_daily;
drop table routes_daily;
ALTER TABLE routes_daily_distinct RENAME TO routes_daily;
```



```
select count(*) from routes_daily;
select bus_operator,sum(passengers_avg) from routes_aggr_distinct group by bus_operator;
select min(sdate), max(sdate) from routes_daily;

select count(*) from hospital;
select count(*) from (select distinct * from hospital) a;
create table hospital_g as select distinct * from hospital;
ALTER TABLE hospital_g ADD COLUMN geom4326 geometry(Geometry,4326);
UPDATE hospital_g SET geom4326 = ST_Transform(geom,4326);
select buffer4326 from train_station;
```

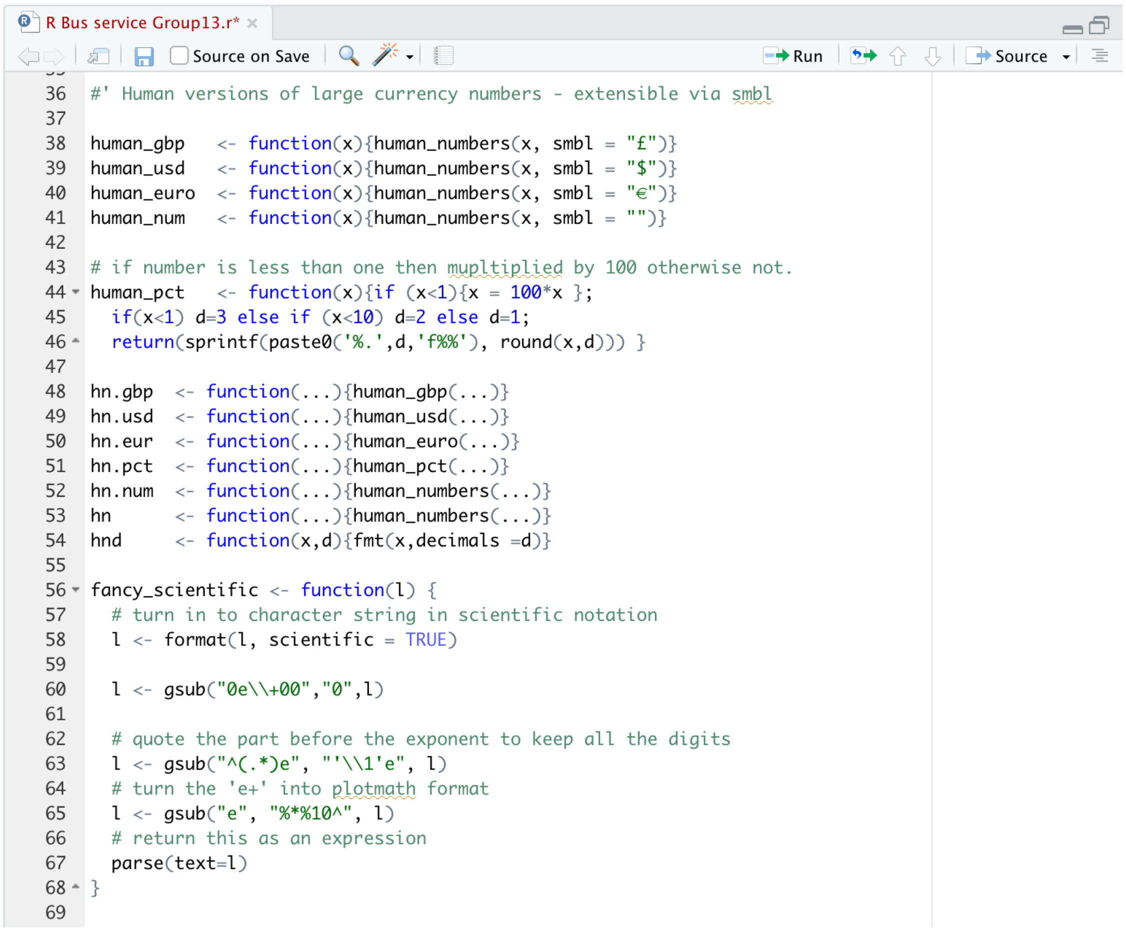


```
select count(*) from household_domestic_surrey;
select count(*) from (select distinct * from household_domestic_surrey) a;
create table household_domestic_surrey_distinct as select distinct * from household_domestic_surrey;
select provider_n as hospital, sum("DOMESTIC_DELIVERY_PTS") as household from hospital_g,
household_domestic_surrey_distinct as D where ST_DWithin(hospital_g.geom, D.the_geom, 1000)
group by provider_n order by household desc
```

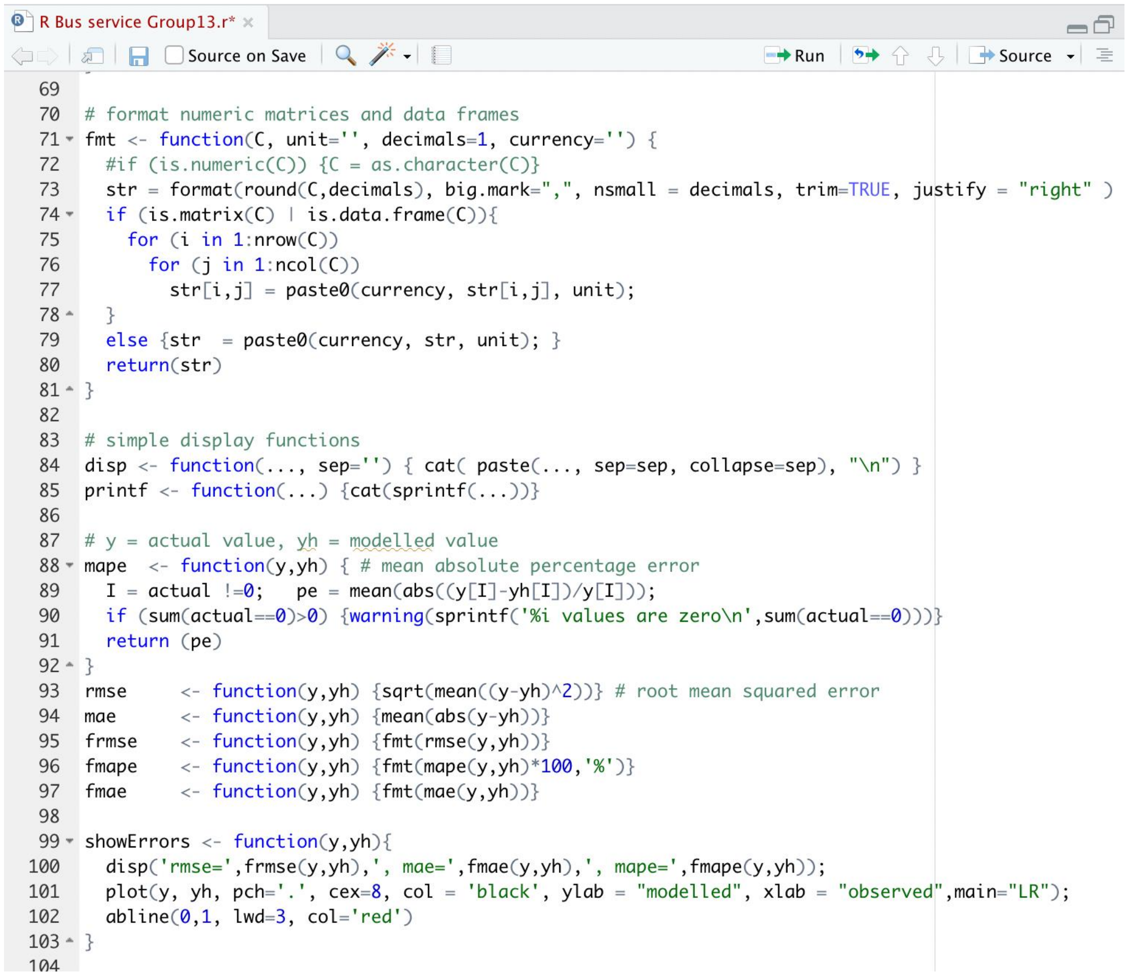


R Bus service Group13.r*

```
# Some of the functions are asis or derived from KevinRPan/handy package.
# https://rdrr.io/github/KevinRPan/handy/man/human_numbers.html

#require(plyr); require(scales)

human_numbers <- function(x = NULL, smbl =""){
  humanity <- function(y){

    if (!is.na(y)){

      b <- round_any(abs(y) / 1e9, 0.1)
      m <- round_any(abs(y) / 1e6, 0.1)
      k <- round_any(abs(y) / 1e3, 0.1)

      if ( y >= 0 ){
        y_is_positive <- ""
      } else {
        y_is_positive <- "-"
      }

      if ( k < 1 ) {
        paste0(y_is_positive, smbl, y )
      } else if ( m < 1){
        paste0 (y_is_positive, smbl,  k , "k")
      } else if (b < 1){
        paste0 (y_is_positive, smbl, m ,"M")
      } else {
        paste0 (y_is_positive, smbl,  comma(b), "b")
      }
    }
  }

  sapply(x,humanity)
}

```



```r
#' Human versions of large currency numbers - extensible via smbl

human_gbp    <- function(x){human_numbers(x, smbl = "£")}
human_usd    <- function(x){human_numbers(x, smbl = "$")}
human_euro   <- function(x){human_numbers(x, smbl = "€")}
human_num    <- function(x){human_numbers(x, smbl = "")}

# if number is less than one then mupltiplied by 100 otherwise not.
human_pct    <- function(x){if (x<1){x = 100*x };
  if(x<1) d=3 else if (x<10) d=2 else d=1;
  return(sprintf(paste0('%.',d,'f%%'), round(x,d))) }

hn.gbp   <- function(...){human_gbp(...)}
hn.usd   <- function(...){human_usd(...)}
hn.eur   <- function(...){human_euro(...)}
hn.pct   <- function(...){human_pct(...)}
hn.num   <- function(...){human_numbers(...)}
hn       <- function(...){human_numbers(...)}
hnd      <- function(x,d){fmt(x,decimals =d)}

fancy_scientific <- function(l) {
  # turn in to character string in scientific notation
  l <- format(l, scientific = TRUE)

  l <- gsub("0e\\+00","0",l)

  # quote the part before the exponent to keep all the digits
  l <- gsub("^(.*)e", "'\\1'e", l)
  # turn the 'e+' into plotmath format
  l <- gsub("e", "%*%10^", l)
  # return this as an expression
  parse(text=l)
}

```



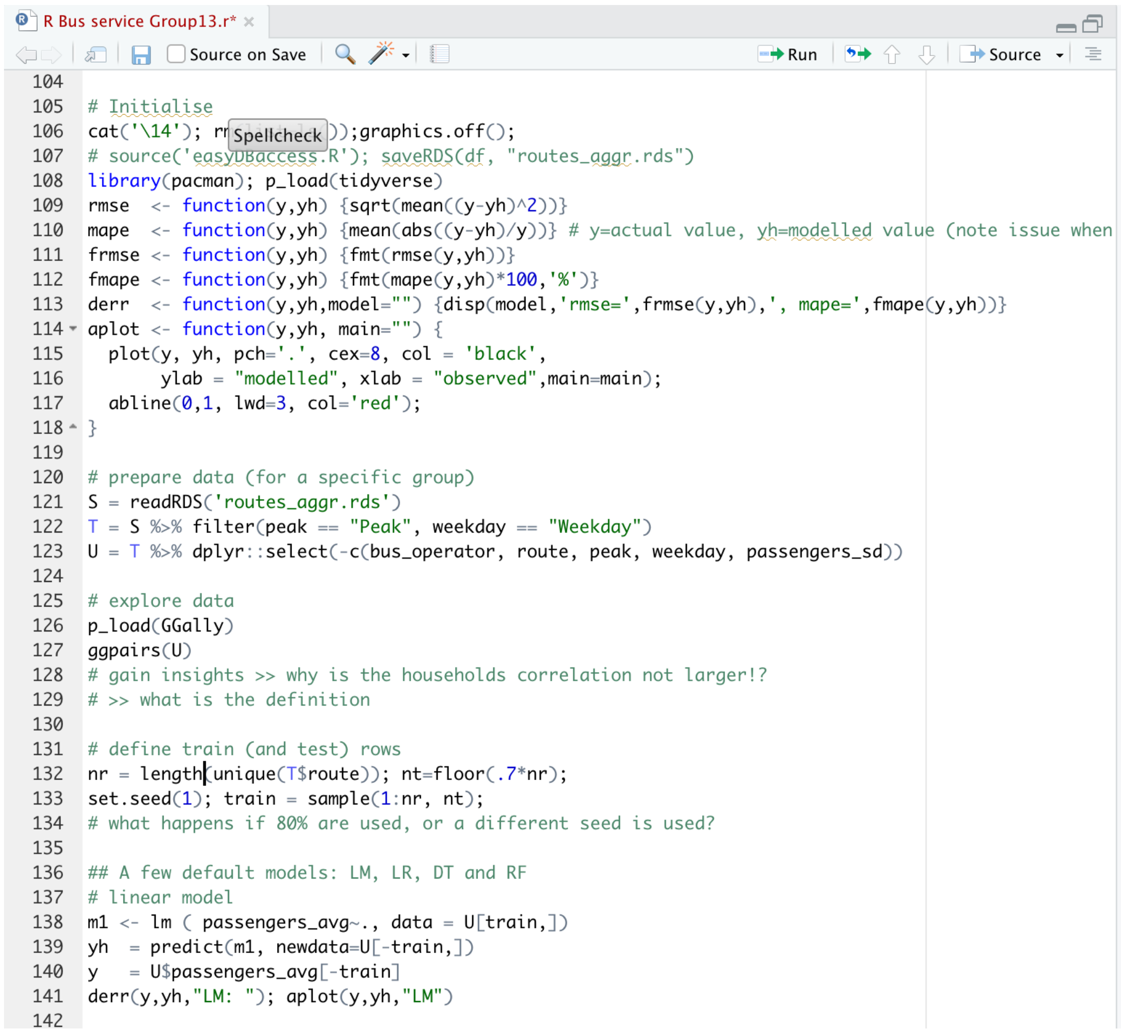
R Bus service Group13.r*

Source on Save

Run

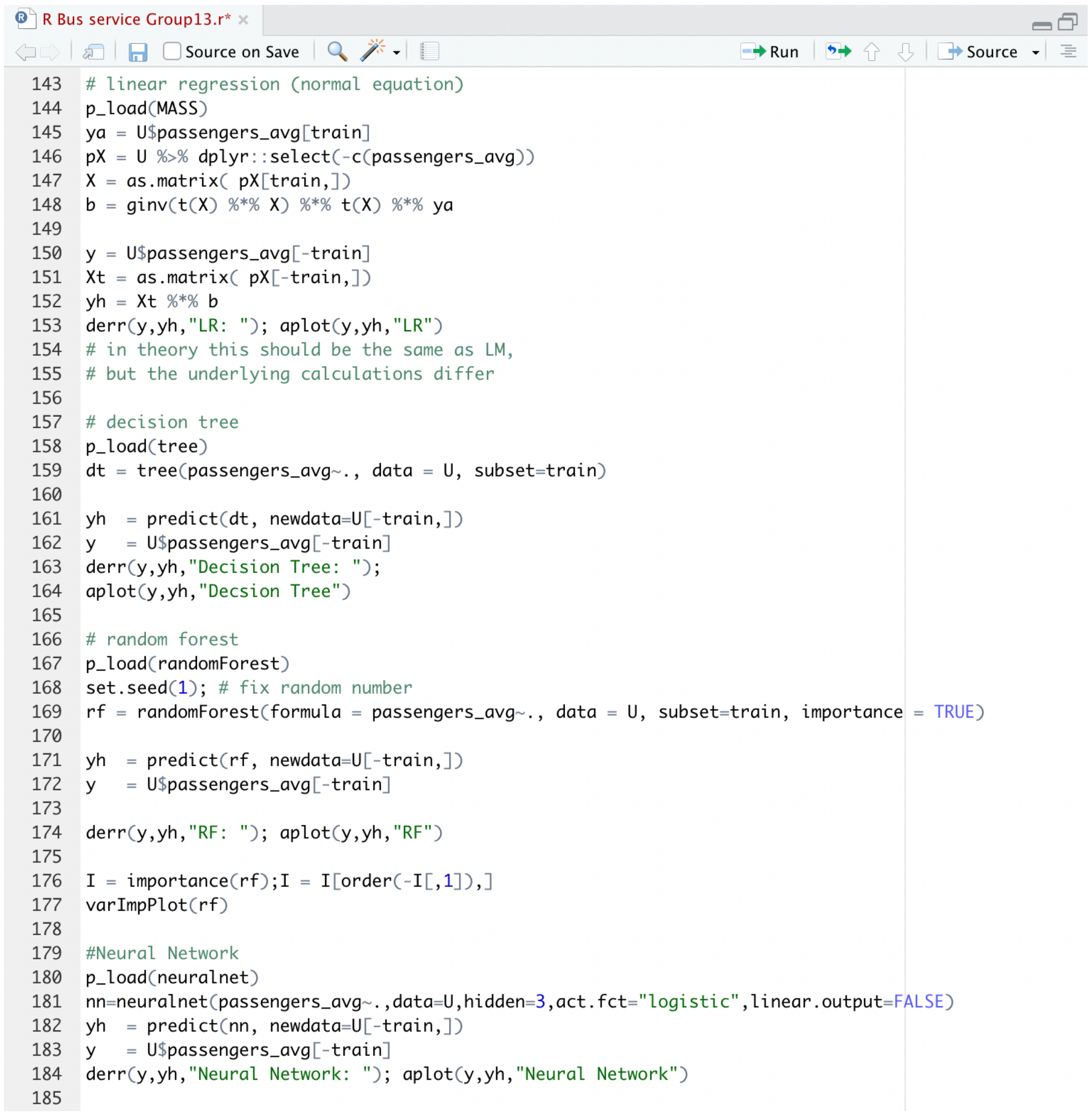
Source

```
# format numeric matrices and data frames
fmt <- function(C, unit='', decimals=1, currency='') {
  #if (is.numeric(C)) {C = as.character(C)}
  str = format(round(C,decimals), big.mark=",", nsmall = decimals, trim=TRUE, justify = "right" )
  if (is.matrix(C) | is.data.frame(C)){
    for (i in 1:nrow(C))
      for (j in 1:ncol(C))
        str[i,j] = paste0(currency, str[i,j], unit);
  }
  else {str  = paste0(currency, str, unit); }
  return(str)
}

# simple display functions
disp <- function(..., sep='') { cat( paste(..., sep=sep, collapse=sep), "\n") }
printf <- function(...) {cat(sprintf(...))}

# y = actual value, yh = modelled value
mape  <- function(y,yh) { # mean absolute percentage error
  I = actual !=0;   pe = mean(abs((y[I]-yh[I])/y[I]));
  if (sum(actual==0)>0) {warning(sprintf('%i values are zero\n',sum(actual==0)))}
  return (pe)
}
rmse     <- function(y,yh) {sqrt(mean((y-yh)^2))} # root mean squared error
mae      <- function(y,yh) {mean(abs(y-yh))}
frmse    <- function(y,yh) {fmt(rmse(y,yh))}
fmape    <- function(y,yh) {fmt(mape(y,yh)*100,'%')}
fmae     <- function(y,yh) {fmt(mae(y,yh))}

showErrors <- function(y,yh){
  disp('rmse=',frmse(y,yh),', mae=',fmae(y,yh),', mape=',fmape(y,yh));
  plot(y, yh, pch='.', cex=8, col = 'black', ylab = "modelled", xlab = "observed",main="LR");
  abline(0,1, lwd=3, col='red')
}
```



```r

# Initialise
cat('\14'); r[illegible]));graphics.off();
# source('easyDBaccess.R'); saveRDS(df, "routes_aggr.rds")
library(pacman); p_load(tidyverse)
rmse  <- function(y,yh) {sqrt(mean((y-yh)^2))}
mape  <- function(y,yh) {mean(abs((y-yh)/y))} # y=actual value, yh=modelled value (note issue when
frmse <- function(y,yh) {fmt(rmse(y,yh))}
fmape <- function(y,yh) {fmt(mape(y,yh)*100,'%')}
derr  <- function(y,yh,model="") {disp(model,'rmse=',frmse(y,yh),', mape=',fmape(y,yh))}
aplot <- function(y,yh, main="") {
  plot(y, yh, pch='.', cex=8, col = 'black',
       ylab = "modelled", xlab = "observed",main=main);
  abline(0,1, lwd=3, col='red');
}

# prepare data (for a specific group)
S = readRDS('routes_aggr.rds')
T = S %>% filter(peak == "Peak", weekday == "Weekday")
U = T %>% dplyr::select(-c(bus_operator, route, peak, weekday, passengers_sd))

# explore data
p_load(GGally)
ggpairs(U)
# gain insights >> why is the households correlation not larger!?
# >> what is the definition

# define train (and test) rows
nr = length(unique(T$route)); nt=floor(.7*nr);
set.seed(1); train = sample(1:nr, nt);
# what happens if 80% are used, or a different seed is used?

## A few default models: LM, LR, DT and RF
# linear model
m1 <- lm ( passengers_avg~., data = U[train,])
yh   = predict(m1, newdata=U[-train,])
y    = U$passengers_avg[-train]
derr(y,yh,"LM: "); aplot(y,yh,"LM")

```

Rcode



```r
# linear regression (normal equation)
p_load(MASS)
ya = U$passengers_avg[train]
pX = U %>% dplyr::select(-c(passengers_avg))
X = as.matrix( pX[train,])
b = ginv(t(X) %*% X) %*% t(X) %*% ya

y = U$passengers_avg[-train]
Xt = as.matrix( pX[-train,])
yh = Xt %*% b
derr(y,yh,"LR: "); aplot(y,yh,"LR")
# in theory this should be the same as LM,
# but the underlying calculations differ

# decision tree
p_load(tree)
dt = tree(passengers_avg~., data = U, subset=train)

yh  = predict(dt, newdata=U[-train,])
y   = U$passengers_avg[-train]
derr(y,yh,"Decision Tree: ");
aplot(y,yh,"Decsion Tree")

# random forest
p_load(randomForest)
set.seed(1); # fix random number
rf = randomForest(formula = passengers_avg~., data = U, subset=train, importance = TRUE)

yh  = predict(rf, newdata=U[-train,])
y   = U$passengers_avg[-train]

derr(y,yh,"RF: "); aplot(y,yh,"RF")

I = importance(rf);I = I[order(-I[,1]),]
varImpPlot(rf)

#Neural Network
p_load(neuralnet)
nn=neuralnet(passengers_avg~.,data=U,hidden=3,act.fct="logistic",linear.output=FALSE)
yh  = predict(nn, newdata=U[-train,])
y   = U$passengers_avg[-train]
derr(y,yh,"Neural Network: "); aplot(y,yh,"Neural Network")
```

Rcode