BUS445

2024-11-07

Summary

Did some investigation for which variables are important. Removed NA missing values (4 total). Renamed some NULLS values to better explain why there were NULL. Did some feature engineering to look at continents, holiday, and seasonal results. I chose to remove the reservation_status as the data is mainly after a cancellation/non cancellation has occurred.

Used random forests, decision tree, and logistic regression primarily. Random forests found that the top 5 important variables are: deposit_type NonRefundable,country: Portugal, lead_time, total_of_special_requests, and previous_cancellations. Decision trees found that the top 5 important variables are: deposit_type Non Refundable, country Portugal, previous_cancellation, market_segment: Online TA and lead_time. Quite similar due to random forests being the average of a bunch of decision trees.

Logistic regression was quite different and found company204, assigned_room_typeI, agent252, company321, and required_car_parking_spaces important. Probably due to logistic regression being linear function compared to the other methods.

Did 2 Neural networks since can't do one on all of the variables. One on the important variables from the random forest and decision tree. One on the important variables from logistic regression. The nn from the variables from the tree methods (RF and DT) performed pretty well. Important variables are deposit_type Non Refund, previous_cancellations, market_segment Offline TA/TO, market_segment Direct, and country PRT.

The nn on the variables from logistic regression performed pretty poorly. (Probably due to how it could only run with one hidden layer.) Important variables found are agent7, agent492, required_car_parking_spaces,company48,agent28.

In terms of accuracy on the test set, Logistic Regression (84%) > Decision Tree (81%) > Neural Network Tree Variables (79%) > Random Forest (75%) > Neural Network Logistic Regression Variables (71%).

Still need to do plots on the important variables. Not sure if should do cross validation as it takes a while to run vs train test sets. Did a bit of train test sets at the end. Seems to perform better.

Exploration of Missing Values

```
set.seed(123)
data=read.csv("hotel_bookings.csv")
originalData=data
#Checking for missing values (NA). Observed 4 missing values in the children column.
data[rowSums(is.na(data))>0,]
```

```
##             hotel is_canceled lead_time arrival_date_year arrival_date_month
## 40601 City Hotel           1         2              2015             August
## 40668 City Hotel           1         1              2015             August
## 40680 City Hotel           1         1              2015             August
## 41161 City Hotel           1         8              2015             August
##       arrival_date_week_number arrival_date_day_of_month
## 40601                       32                         3
## 40668                       32                         5
## 40680                       32                         5
## 41161                       33                        13
##       stays_in_weekend_nights stays_in_week_nights adults children babies meal
## 40601                       1                    0      2       NA      0   BB
## 40668                       0                    2      2       NA      0   BB
## 40680                       0                    2      3       NA      0   BB
## 41161                       2                    5      2       NA      0   BB
##       country market_segment distribution_channel is_repeated_guest
## 40601     PRT      Undefined            Undefined                 0
## 40668     PRT         Direct            Undefined                 0
## 40680     PRT      Undefined            Undefined                 0
## 41161     PRT      Online TA            Undefined                 0
##       previous_cancellations previous_bookings_not_canceled reserved_room_type
## 40601                      0                              0                  B
## 40668                      0                              0                  B
## 40680                      0                              0                  B
## 41161                      0                              0                  B
##       assigned_room_type booking_changes deposit_type agent company
## 40601                  B               0   No Deposit  NULL    NULL
## 40668                  B               0   No Deposit    14    NULL
## 40680                  B               0   No Deposit  NULL    NULL
## 41161                  B               0   No Deposit     9    NULL
##       days_in_waiting_list   customer_type  adr required_car_parking_spaces
## 40601                    0 Transient-Party 12.0                           0
## 40668                    0 Transient-Party 12.0                           0
## 40680                    0 Transient-Party 18.0                           0
## 41161                    0 Transient-Party 76.5                           0
##       total_of_special_requests reservation_status reservation_status_date
## 40601                         1           Canceled              2015-08-01
## 40668                         1           Canceled              2015-08-04
## 40680                         2           Canceled              2015-08-04
## 41161                         1           Canceled              2015-08-09
```

```
#Removing these 4 instances as there is a lot of observations
data=na.omit(data)
```

Contingency table of all the columns

```
#lapply(data,table) Commented out as it's too big of a print.
```

It’s observed that there are NULL values in the data. The columns with NULL values are company, agent, and country.

```
colSums(data=="NULL")
```

```
##                          hotel                    is_canceled 
##                              0                              0 
##                      lead_time              arrival_date_year 
##                              0                              0 
##             arrival_date_month       arrival_date_week_number 
##                              0                              0 
##      arrival_date_day_of_month        stays_in_weekend_nights 
##                              0                              0 
##           stays_in_week_nights                         adults 
##                              0                              0 
##                       children                         babies 
##                              0                              0 
##                           meal                        country 
##                              0                            488 
##                 market_segment           distribution_channel 
##                              0                              0 
##              is_repeated_guest         previous_cancellations 
##                              0                              0 
## previous_bookings_not_canceled             reserved_room_type 
##                              0                              0 
##             assigned_room_type                booking_changes 
##                              0                              0 
##                   deposit_type                          agent 
##                              0                          16338 
##                        company           days_in_waiting_list 
##                         112589                              0 
##                  customer_type                            adr 
##                              0                              0 
##    required_car_parking_spaces      total_of_special_requests 
##                              0                              0 
##             reservation_status        reservation_status_date 
##                              0                              0
```

The contigency table for the company feature.

```
#table(data$company) Commented out as it's too big of a print.
```

It is observed that the most common element is the NULL value with 112589 observations which is much more than 50% of the data. This is most likely due to a majority of the hotel bookings not be associated with a company booking. As a result, this implys that the NULL values are important so they will be renamed to “No Company”

```
data=data%>%mutate(company=ifelse(company=="NULL","No Company",company))
```

The agent feature has 16338 NULL values. As the agent number is related to the distribution channel of the booking, we will investigate the distribution channel.

```
#table(data$agent) Commented out as it's too big of a print.
```

```
agentNullData=data%>% filter(agent=="NULL")
#table(agentNullData$agent,agentNullData$distribution_channel) Commented out as it's too big of
a print.
```

Of the 16338 NULL values in the agent field, 13168 (5543+7625) of them belong to the corporate and direct distribution channels which have no agents as they directly contact the hotel for the booking. We will fill these with “No Travel Agency” as they don’t use any travel agency. There is 3167 NULL values with TA/TO distribution channels. We will fill in these with “TA/TO No Agent Number” as they have travel agents but have no agent id. The remaining 3 NULL values will be removed as they are only 3 of them.

```
data=data%>%mutate(agent=ifelse(distribution_channel %in% c("Corporate","Direct") & agent=='NUL
L','No Travel Agency',agent))
data=data%>%mutate(agent=ifelse(distribution_channel=="TA/TO" & agent=="NULL","TA/TO No Agent Nu
mber",agent))
data=data%>%filter(agent!="NULL")
```

Looking at the Contingency table of the country column we see that there is 488 NULL values.

```
#table(data$country) Commented out as it's too big of a print.
```

```
countryNulldata=data%>% filter(country=="NULL")
x=table(countryNulldata$country,countryNulldata$agent)
#x["NULL",] Commented out as it's too big of a print.
```

It is observed that majority of the observations with NULL for countries also had no agents which are now “No Travel Agency” and “TA/TO No Agent Number”. We will fill these with countries with “Unknown”. For all the other NULL countries, we will remove them as there is a small amount of them.

```
data=data%>%mutate(country=ifelse(agent %in% c("No Travel Agency","TA/TO No Agent Number") & cou
ntry=='NULL','Unknown',country))
data=data%>%filter(data$country!="NULL")
```

```
#lapply(data,table)
```

It is observed that there is 1168 undefined columns in the meal feature. As the other options are BB (Bed and Breakfast), FB(Full Board), HB(Half Board), and SC (Self Catering) it is observed that there is no option for no meal services. As a result, we will fill these undefined values with “Other”

```
data=data%>%mutate(meal=ifelse(meal=='Undefined','Other',meal))
table(data$meal)
```

```
## 
##    BB    FB    HB Other    SC 
## 92164   798 14450  1168 10649
```

```
head(data)
```

```
##          hotel is_canceled lead_time arrival_date_year arrival_date_month
## 1 Resort Hotel           0       342              2015               July
## 2 Resort Hotel           0       737              2015               July
## 3 Resort Hotel           0         7              2015               July
## 4 Resort Hotel           0        13              2015               July
## 5 Resort Hotel           0        14              2015               July
## 6 Resort Hotel           0        14              2015               July
##   arrival_date_week_number arrival_date_day_of_month stays_in_weekend_nights
## 1                       27                         1                       0
## 2                       27                         1                       0
## 3                       27                         1                       0
## 4                       27                         1                       0
## 5                       27                         1                       0
## 6                       27                         1                       0
##   stays_in_week_nights adults children babies meal country market_segment
## 1                    0      2        0      0   BB     PRT         Direct
## 2                    0      2        0      0   BB     PRT         Direct
## 3                    1      1        0      0   BB     GBR         Direct
## 4                    1      1        0      0   BB     GBR      Corporate
## 5                    2      2        0      0   BB     GBR      Online TA
## 6                    2      2        0      0   BB     GBR      Online TA
##   distribution_channel is_repeated_guest previous_cancellations
## 1               Direct                 0                      0
## 2               Direct                 0                      0
## 3               Direct                 0                      0
## 4            Corporate                 0                      0
## 5                TA/TO                 0                      0
## 6                TA/TO                 0                      0
##   previous_bookings_not_canceled reserved_room_type assigned_room_type
## 1                              0                  C                  C
## 2                              0                  C                  C
## 3                              0                  A                  C
## 4                              0                  A                  A
## 5                              0                  A                  A
## 6                              0                  A                  A
##   booking_changes deposit_type            agent    company days_in_waiting_list
## 1               3   No Deposit No Travel Agency No Company                    0
## 2               4   No Deposit No Travel Agency No Company                    0
## 3               0   No Deposit No Travel Agency No Company                    0
## 4               0   No Deposit              304 No Company                    0
## 5               0   No Deposit              240 No Company                    0
## 6               0   No Deposit              240 No Company                    0
##   customer_type adr required_car_parking_spaces total_of_special_requests
## 1     Transient   0                           0                         0
## 2     Transient   0                           0                         0
## 3     Transient  75                           0                         0
## 4     Transient  75                           0                         0
## 5     Transient  98                           0                         1
## 6     Transient  98                           0                         1
##   reservation_status reservation_status_date
## 1          Check-Out              2015-07-01
## 2          Check-Out              2015-07-01
```

```
## 3          Check-Out               2015-07-02
## 4          Check-Out               2015-07-02
## 5          Check-Out               2015-07-03
## 6          Check-Out               2015-07-03
```

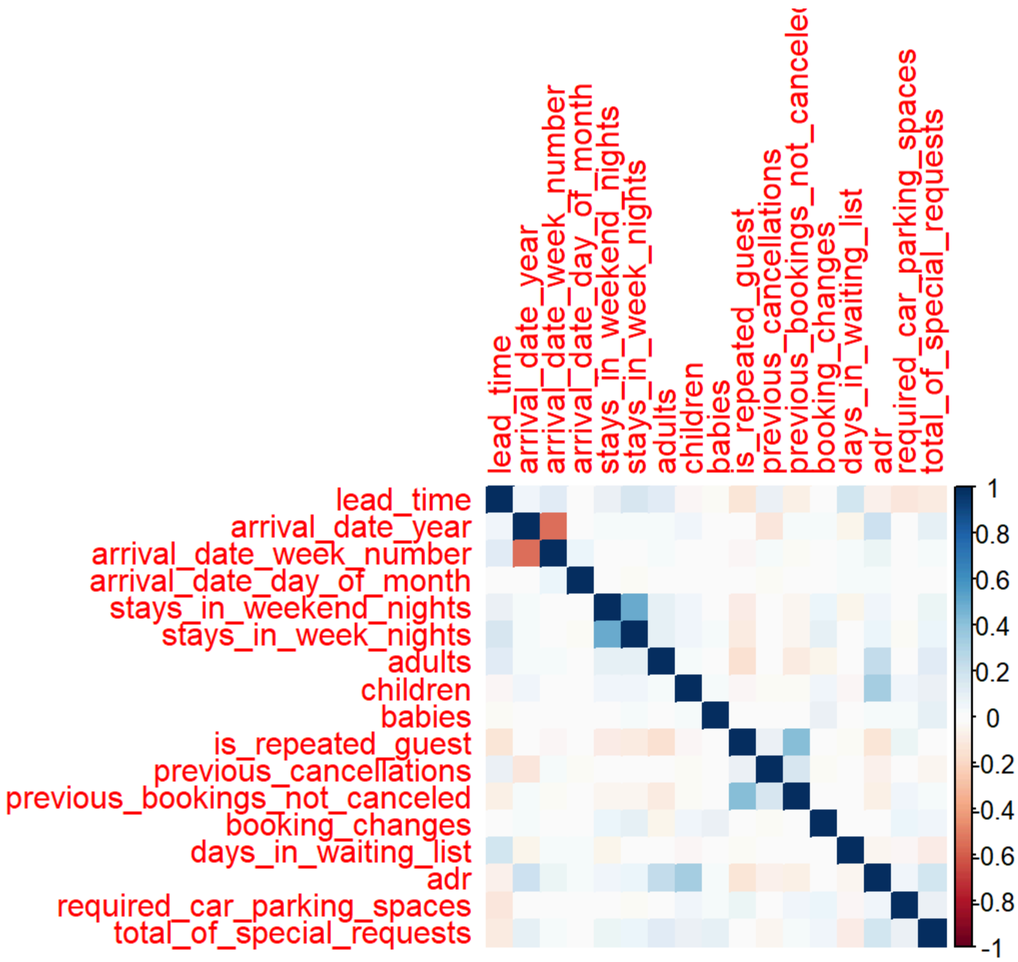
```
write.csv(data,"data.csv",row.names = FALSE) # Writing out for easier factor conversion
```

```
data=read.csv("data.csv",stringsAsFactors = TRUE)
data$is_canceled=as.factor(data$is_canceled)
file.remove("data.csv")
```

```
## [1] TRUE
```

Correlation Exploration

```
library(corrplot)
numericData=data[sapply(data,is.numeric)]
corr=cor(numericData)
corrplot(corr,method="color")
```



As their isn't any highly correlated columns, no columns will be removed.

Creating New Features

Binning the lead time into quartiles

```
q1LeadTime=quantile(data$lead_time,0.25)
q2LeadTime=quantile(data$lead_time,0.50)
q3LeadTime=quantile(data$lead_time,0.75)
data$lead_timeCategories=cut(data$lead_time,breaks=c(-Inf,q1LeadTime,q2LeadTime,q3LeadTime,Inf),
labels=c("Very Low Lead Time", "Below Average Lead Time", "Above Average Lead Time", "High Lead
Time"))
```

Making a continent Column

```
data$Continent=countrycode(data$country, origin = "iso3c", destination = "continent")
```

```
## Warning: Some values were not matched unambiguously: ATF, CN, TMP, UMI, Unknown
```

```
southAmerica=c("ARG", "BRA", "CHL", "PER", "COL", "VEN", "SUR", "ECU", "GUY", "PRY", "BOL", "GU
Y")

#Manually fixing continent values that the country code couldn't define

#South America is linked together as Americas with North America
data$Continent=ifelse(data$country %in% southAmerica & data$Continent == "Americas","South Ameri
ca",data$Continent)
data$Continent=ifelse(data$country == "ATF", "None",data$Continent)  #French South Territories i
sn't associated with a continent
data$Continent=ifelse(data$country == "CN","Asia",data$Continent) #China
data$Continent=ifelse(data$country == "TMP","Asia",data$Continent) #East Timor, part of ASIA
data$Continent=ifelse(data$country == "UMI","None",data$Continent) #United States Minor Outlying
Islands isn't associated with a continent
data$Continent=ifelse(data$country == "Unknown","Unknown",data$Continent)
```

Making a holiday seasons column (Summer, Chirstmas, New years)

```
data$ArrivalHolidaySeason=cut(data$arrival_date_week_number,breaks=c(-Inf,1,20,26,47,51,Inf),lab
els=c("New Year","Regular","Summer","Regular","Chirstmas","New Year"))
```

Making a seasonal column

```
data=data%>%mutate(ArrivalSeason=case_when(
    arrival_date_month %in% c("December", "January", "February") ~ "Winter",
    arrival_date_month %in% c("March", "April", "May") ~ "Spring",
    arrival_date_month %in% c("June", "July", "August") ~ "Summer",
    arrival_date_month %in% c("September", "October", "November") ~ "Fall")
  )
data$ArrivalSeason=as.factor(data$ArrivalSeason)
```

```
originalData=data#Before removing columns stored original with features engineered for later us
e.

data=subset(data,select=-reservation_status) #Dropping variables that are observed after a hotel
booking is finalized (Canceled, No Show, etc)
data=subset(data,select=-reservation_status_date)

data=subset(data,select=-arrival_date_week_number)#Dropping arrival week number as I used it to
create the Seasonal columns
```

Splitting the data for ML

```
data$is_canceled=as.factor(data$is_canceled)
data$Continent=as.factor(data$Continent)
partition=createDataPartition(data$is_canceled,p=0.75,list=FALSE)
data_train=data[partition,]
data_test=data[-partition,]
```

Random Forest

Used cross validation to tune for parameters for RF.

```
#Commented out as it takes a while to run. The final values used for the model were mtry = 6, sp
litrule = gini and min.node.size = 10.
#gridRF=expand.grid(mtry=round(sqrt(ncol(data_train))),splitrule="gini",min.node.size= c(1, 5, 1
0, 20, 50))
#control=trainControl(method="cv",number=5,verboseIter=TRUE)
#model=train(is_canceled~.,data=data_train,method="ranger",tuneGrid=gridRF,trControl=control,imp
ortance = "impurity",num.trees=1000)
#varImp(model)
```

```
gridRF=expand.grid(mtry=6,splitrule="gini",min.node.size=10)
control=trainControl(method="cv",number=5)
rfmodel=train(is_canceled~.,data=data_train,method="ranger",tuneGrid=gridRF,trControl=control,im
portance = "impurity",num.trees=1000)
```

Accuracy

```
rf_preds=predict(rfmodel,newdata=data_test)
mean(rf_preds==data_test$is_canceled)
```

```
## [1] 0.7492871
```

Confusion Matrix

```
table(rf_preds,data_test$is_canceled)
```

```
##
## rf_preds     0     1
##        0 18748  7461
##        1    12  3586
```

Variable Importance

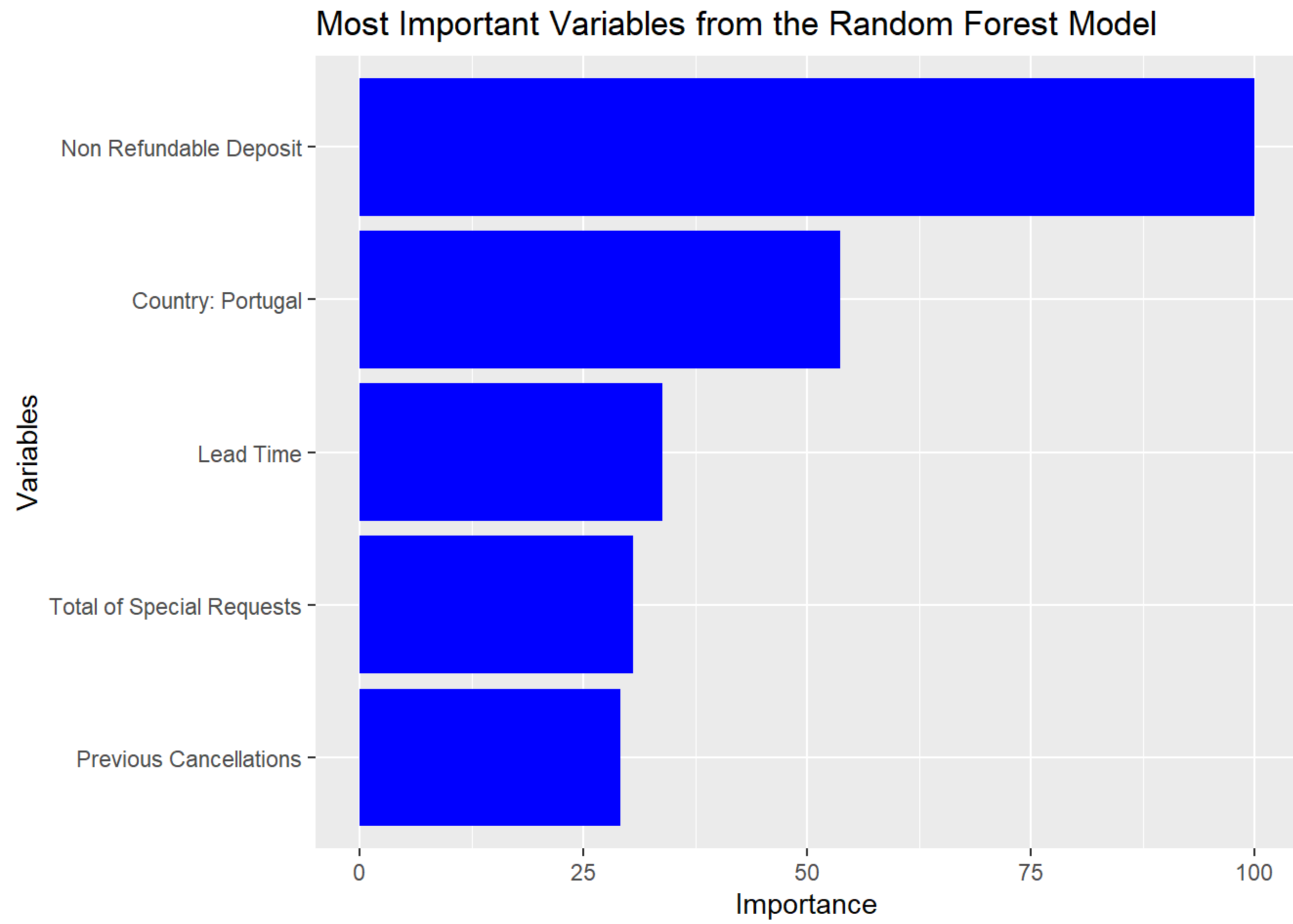
```
rfImportance=varImp(rfmodel)
Top5RfImportance=rfImportance$importance%>%as.data.frame()%>%rownames_to_column("Feature") %>% a
rrange(desc(Overall))%>%head(5)
Top5RfImportance
```

```
##                      Feature   Overall
## 1    deposit_typeNon Refund 100.00000
## 2                 countryPRT  53.65040
## 3                  lead_time  33.83009
## 4 total_of_special_requests  30.54662
## 5    previous_cancellations  29.08125
```

```
#Found Deposit type:Non refundable, country:Portugal, lead_time, total of special requests, and
previous_cancellations and lead_time important
```

Variable Importance Plot

```
ggplot(data=Top5RfImportance,mapping=aes(x=Overall,y= reorder(Feature, Overall)))+geom_bar(stat
="identity",fill="blue")+scale_y_discrete(labels=c("Previous Cancellations","Total of Special Re
quests","Lead Time","Country: Portugal","Non Refundable Deposit"))+xlab("Importance")+ylab("Vari
ables")+ggtitle("Most Important Variables from the Random Forest Model")
```



Decision Tree

As with Random Forests, used cross validation to tune the parameters.

```
#Commented out as it takes a while to tune for the parameters. The final value used for the mode
l was cp = 0.01
#tune_gridTree=expand.grid(cp=seq(0.01,0.1, by=0.01))
#train_controlTree=trainControl(method="cv",number=5,verboseIter=TRUE)
#TreeModel=train(is_canceled~.,data=data_train,method="rpart",trControl=train_controlTree, tuneG
rid = tune_gridTree)
```

```
tune_gridTree=expand.grid(cp=0.01)
train_controlTree=trainControl(method="cv",number=5)
TreeModel=train(is_canceled~.,data=data_train,method="rpart",trControl=train_controlTree, tuneGr
id = tune_gridTree)
Treepreds=predict(TreeModel,newdata=data_test)
```

Accuracy

```
mean(Treepreds==data_test$is_canceled)
```

```
## [1] 0.8131647
```

Confusion Matrix

```
table(Treepreds,data_test$is_canceled)
```

```
##
## Treepreds     0     1
##         0 16210  3019
##         1  2550  8028
```

```
TreeImportance=varImp(TreeModel)
```

Important Variables

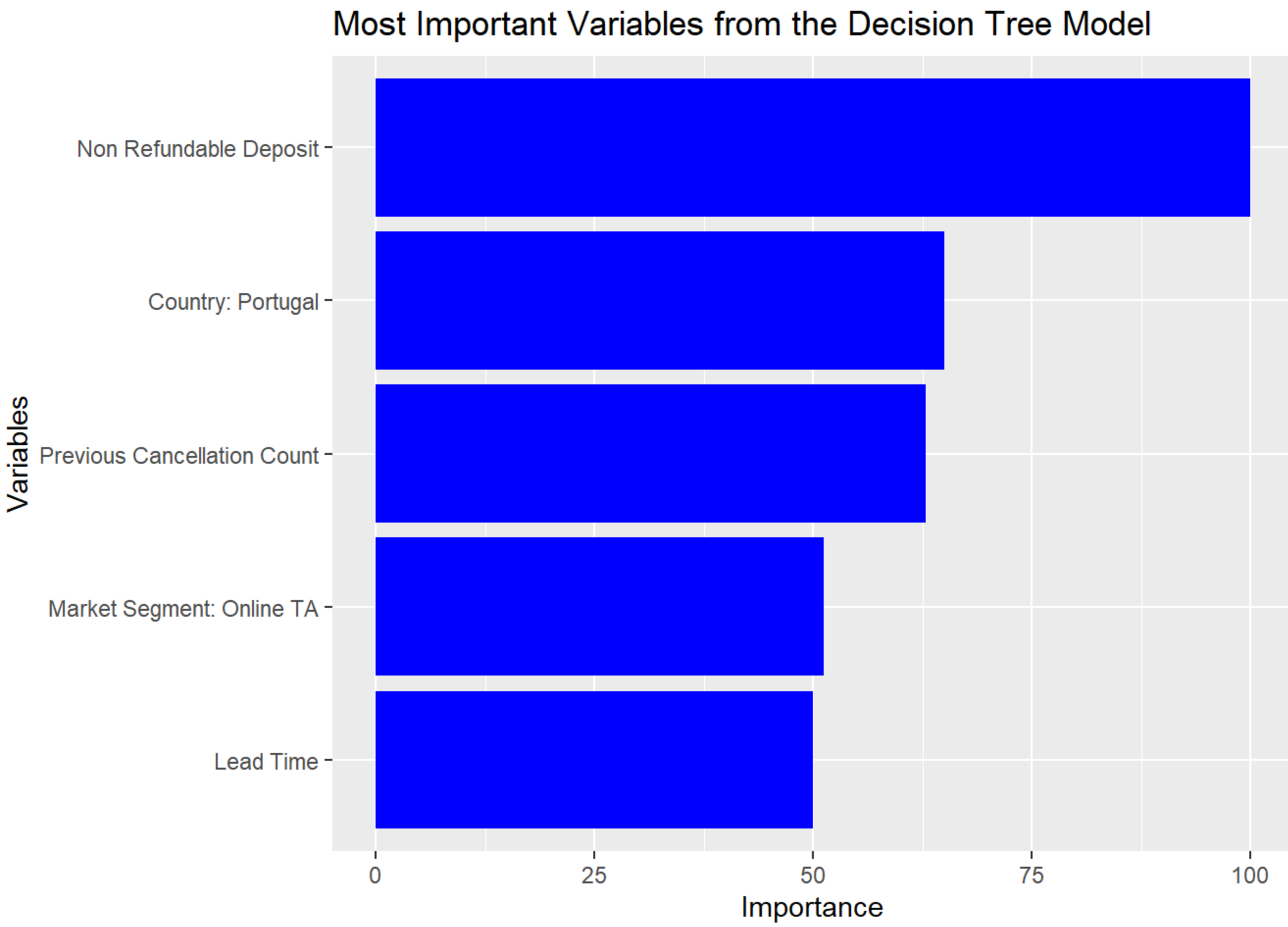
```
Top5TreeImportance=TreeImportance$importance%>%as.data.frame()%>%rownames_to_column("Feature") %
>% arrange(desc(Overall))%>%head(5)
Top5TreeImportance
```

```
##                    Feature   Overall
## 1  deposit_typeNon Refund 100.00000
## 2              countryPRT  65.03180
## 3  previous_cancellations  62.85566
## 4 market_segmentOnline TA  51.15788
## 5               lead_time  49.92142
```

```
#Found Deposity type:Non refundable, country:Portugal, previous_cancellations, Market Segment:On
line TA, and lead_time important
```

Important Variables plot

```
ggplot(data=Top5TreeImportance,mapping=aes(x=Overall,y= reorder(Feature, Overall)))+geom_bar(sta
t="identity",fill="blue")+scale_y_discrete(labels=c("Lead Time","Market Segment: Online TA","Pre
vious Cancellation Count","Country: Portugal","Non Refundable Deposit"))+xlab("Importance")+ylab
("Variables")+ggtitle("Most Important Variables from the Decision Tree Model")
```



Logistic Regression

Was gonna use glm but it wouldn't run. So used multinom instead. Takes a while to run.

```
train_control=trainControl(method="cv",number=5)
LogitModel=train(is_canceled~.,data=data_train,method="multinom",trControl=train_control)
```

```
## # weights:  947 (946 variable)
## initial  value 49586.363003
## iter  10 value 35130.282435
## iter  20 value 32863.802475
## iter  30 value 31530.438436
## iter  40 value 29120.515376
## iter  50 value 27399.954803
## iter  60 value 26457.063805
## iter  70 value 25329.907761
## iter  80 value 24954.890807
## iter  90 value 24922.557808
## iter 100 value 24887.106597
## final  value 24887.106597
## stopped after 100 iterations
## # weights:  947 (946 variable)
## initial  value 49586.363003
## iter  10 value 35131.616109
## iter  20 value 32868.943902
## iter  30 value 31550.142755
## iter  40 value 29257.492742
## iter  50 value 27647.161521
## iter  60 value 26504.867007
## iter  70 value 25376.870077
## iter  80 value 25050.185836
## iter  90 value 25018.079964
## iter 100 value 24996.557525
## final  value 24996.557525
## stopped after 100 iterations
## # weights:  947 (946 variable)
## initial  value 49586.363003
## iter  10 value 35130.283767
## iter  20 value 32863.807579
## iter  30 value 31530.458143
## iter  40 value 29120.592000
## iter  50 value 27400.012965
## iter  60 value 26457.097132
## iter  70 value 25330.033852
## iter  80 value 24955.062251
## iter  90 value 24922.508283
## iter 100 value 24887.263721
## final  value 24887.263721
## stopped after 100 iterations
## # weights:  947 (946 variable)
## initial  value 49585.669856
## iter  10 value 33636.708238
## iter  20 value 31154.954915
## iter  30 value 30151.357492
## iter  40 value 28615.198547
## iter  50 value 26808.677740
## iter  60 value 26038.458354
## iter  70 value 25617.954283
## iter  80 value 25129.716465
```

```
## iter  90 value 24750.587557
## iter 100 value 24703.343942
## final  value 24703.343942
## stopped after 100 iterations
## # weights:  947 (946 variable)
## initial  value 49585.669856
## iter  10 value 33638.550142
## iter  20 value 31161.399562
## iter  30 value 30172.130823
## iter  40 value 28666.277212
## iter  50 value 27112.568121
## iter  60 value 26145.962930
## iter  70 value 25250.287108
## iter  80 value 25019.173422
## iter  90 value 24892.270172
## iter 100 value 24851.812200
## final  value 24851.812200
## stopped after 100 iterations
## # weights:  947 (946 variable)
## initial  value 49585.669856
## iter  10 value 33636.710110
## iter  20 value 31154.961730
## iter  30 value 30151.380721
## iter  40 value 28615.257688
## iter  50 value 26808.793815
## iter  60 value 26047.986347
## iter  70 value 25615.693912
## iter  80 value 25212.830809
## iter  90 value 24794.296059
## iter 100 value 24752.206068
## final  value 24752.206068
## stopped after 100 iterations
## # weights:  947 (946 variable)
## initial  value 49585.669856
## iter  10 value 34590.210632
## iter  20 value 32981.716428
## iter  30 value 32270.426818
## iter  40 value 30088.719669
## iter  50 value 28141.243414
## iter  60 value 27250.106544
## iter  70 value 25518.887691
## iter  80 value 25005.997912
## iter  90 value 24880.792430
## iter 100 value 24872.834126
## final  value 24872.834126
## stopped after 100 iterations
## # weights:  947 (946 variable)
## initial  value 49585.669856
## iter  10 value 34590.684472
## iter  20 value 32984.158575
## iter  30 value 32278.675305
## iter  40 value 30119.890093
```

```
## iter  50 value 28207.396786
## iter  60 value 27323.901460
## iter  70 value 25604.747913
## iter  80 value 25119.442918
## iter  90 value 24941.637885
## final  value 24940.770390
## converged
## # weights:  947 (946 variable)
## initial  value 49585.669856
## iter  10 value 34590.211105
## iter  20 value 32981.718850
## iter  30 value 32270.435001
## iter  40 value 30088.751934
## iter  50 value 28141.313103
## iter  60 value 27250.146714
## iter  70 value 25518.997685
## iter  80 value 25006.214417
## iter  90 value 24884.182214
## iter 100 value 24874.875718
## final  value 24874.875718
## stopped after 100 iterations
## # weights:  947 (946 variable)
## initial  value 49586.363003
## iter  10 value 33138.146033
## iter  20 value 30825.989087
## iter  30 value 29943.777784
## iter  40 value 28412.774451
## iter  50 value 27140.328427
## iter  60 value 26217.086123
## iter  70 value 25130.198695
## iter  80 value 24784.788052
## iter  90 value 24682.904002
## iter 100 value 24662.891933
## final  value 24662.891933
## stopped after 100 iterations
## # weights:  947 (946 variable)
## initial  value 49586.363003
## iter  10 value 33139.970792
## iter  20 value 30832.075871
## iter  30 value 29961.995101
## iter  40 value 28461.787162
## iter  50 value 27233.663495
## iter  60 value 26419.606143
## iter  70 value 25451.916770
## iter  80 value 24952.069108
## iter  90 value 24759.716875
## iter 100 value 24711.555745
## final  value 24711.555745
## stopped after 100 iterations
## # weights:  947 (946 variable)
## initial  value 49586.363003
## iter  10 value 33138.147822
```

```
## iter  20 value 30825.994753
## iter  30 value 29943.793773
## iter  40 value 28412.819375
## iter  50 value 27140.419642
## iter  60 value 26217.200665
## iter  70 value 25130.344284
## iter  80 value 24784.962877
## iter  90 value 24683.104299
## iter 100 value 24656.987505
## final  value 24656.987505
## stopped after 100 iterations
## # weights:  947 (946 variable)
## initial  value 49586.363003
## iter  10 value 34954.567017
## iter  20 value 32489.723584
## iter  30 value 31373.384046
## iter  40 value 29053.889967
## iter  50 value 27300.105836
## iter  60 value 26413.097206
## iter  70 value 25318.445725
## iter  80 value 24982.023299
## iter  90 value 24772.611531
## iter 100 value 24739.628827
## final  value 24739.628827
## stopped after 100 iterations
## # weights:  947 (946 variable)
## initial  value 49586.363003
## iter  10 value 34956.232471
## iter  20 value 32495.572822
## iter  30 value 31390.942122
## iter  40 value 29393.663034
## iter  50 value 27502.475240
## iter  60 value 26498.658652
## iter  70 value 25405.915796
## iter  80 value 25086.032423
## iter  90 value 24919.211107
## iter 100 value 24886.953251
## final  value 24886.953251
## stopped after 100 iterations
## # weights:  947 (946 variable)
## initial  value 49586.363003
## iter  10 value 34954.568681
## iter  20 value 32489.729416
## iter  30 value 31373.402137
## iter  40 value 29053.945801
## iter  50 value 27300.204452
## iter  60 value 26413.225819
## iter  70 value 25318.594343
## iter  80 value 24982.193450
## iter  90 value 24772.818413
## iter 100 value 24739.785320
## final  value 24739.785320
```

```
## stopped after 100 iterations
## # weights:  947 (946 variable)
## initial  value 61982.607180
## iter  10 value 42306.598067
## iter  20 value 40200.162677
## iter  30 value 39495.687122
## iter  40 value 37263.275748
## iter  50 value 34531.822209
## iter  60 value 33469.930740
## iter  70 value 32297.303685
## iter  80 value 31356.902877
## iter  90 value 31058.200651
## iter 100 value 31055.281845
## final  value 31055.281845
## stopped after 100 iterations
```

```
Logitpreds=predict(LogitModel,newdata=data_test)
```

Confusion Matrix

```
table(Logitpreds,data_test$is_canceled)
```

```
##
## Logitpreds     0     1
##          0 16891  2876
##          1  1869  8171
```

Accuracy

```
mean(Logitpreds==data_test$is_canceled)
```

```
## [1] 0.8408092
```

Important variables

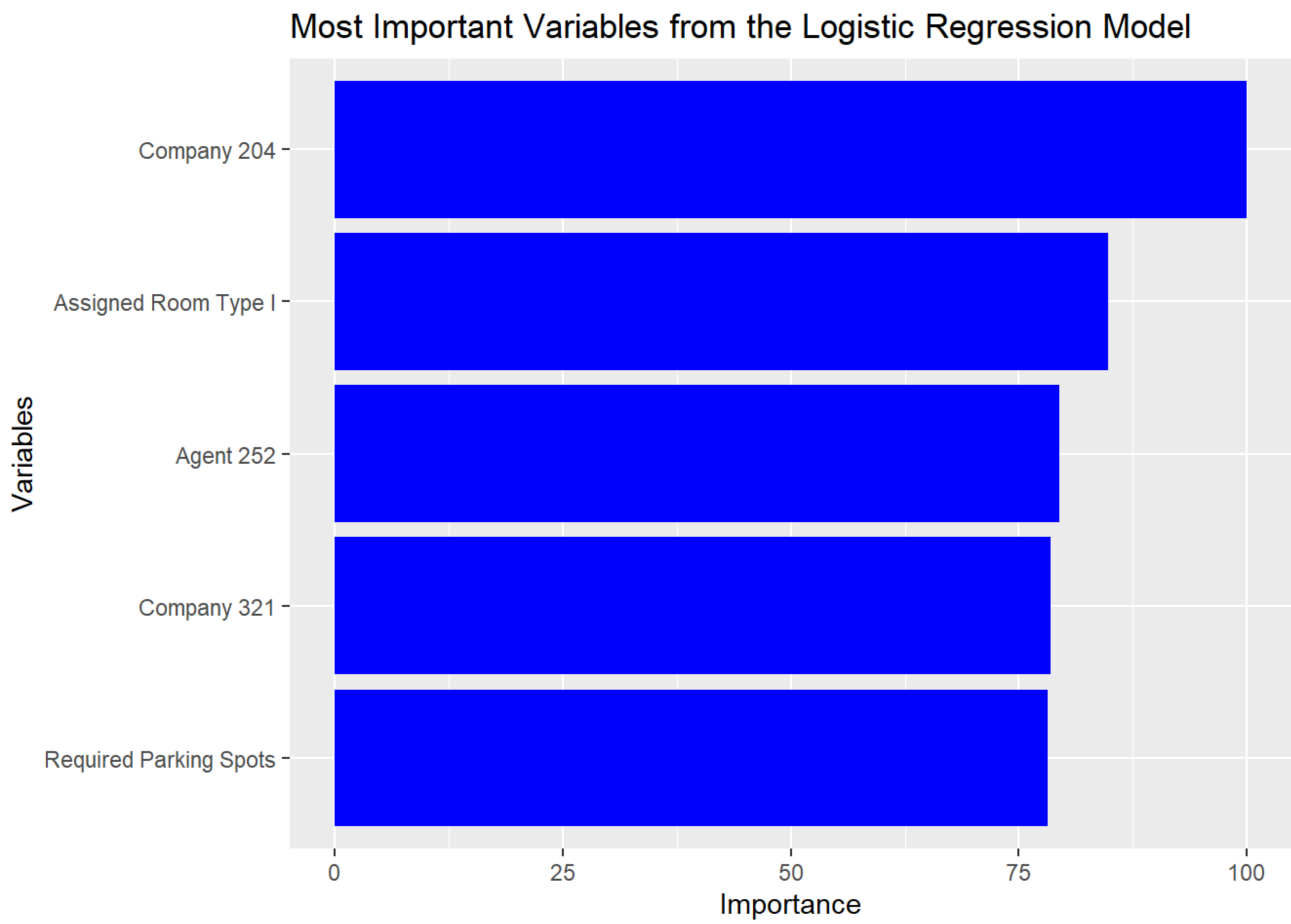
```
lgImportance=varImp(LogitModel)
Top5LgImportance=lgImportance$importance%>%as.data.frame()%>%rownames_to_column("Feature") %>% a
rrange(desc(Overall))%>%head(5)
Top5LgImportance
```

```
##      Feature   Overall
## 1   agent252 100.00000
## 2 company321  84.83656
## 3   agent341  79.43836
## 4   agent276  78.52560
## 5 company204  78.15446
```

```
#Found company204, assigned_room_typeI, agent252, company321, and required_car_parking_spaces im
portant
```

Important Variables Plot

```
ggplot(data=Top5LgImportance,mapping=aes(x=Overall,y= reorder(Feature, Overall)))+geom_bar(stat
="identity",fill="blue")+scale_y_discrete(labels=c("Required Parking Spots","Company 321","Agent
252","Assigned Room Type I","Company 204"))+xlab("Importance")+ylab("Variables")+ggtitle("Most I
mportant Variables from the Logistic Regression Model")
```



Very different result from random forests and decision tree. Probably due to random forests struggling with linear relationships.

Neural Net on variables from tree methods (RF and DT)

Can’t use all the variables so used the ones that random forest and decision tree found important. Used CV to tune for parameters.

```
#size = 5 and decay = 0.1. Commneted out as it takes too long to compute
#nn_trainControl=trainControl(method="cv",number=5,verboseIter = TRUE)
#nn_tuneGrid=expand.grid(size=c(1,2,3,5,10), decay = c(0.1, 0.2, 0.3))
#nnModel=train(is_canceled~lead_time+deposit_type+country+market_segment+previous_cancellations,
data=data_train,method="nnet",trControl=nn_trainControl,tuneGrid=nn_tuneGrid)

#only using the variables found as important in the previous sections as it gets too computation
ally complex
nn_trainControl=trainControl(method="cv",number=5)
nn_tuneGrid=expand.grid(size=5, decay = 0.1)
nnModel=train(is_canceled~lead_time+deposit_type+country+market_segment+previous_cancellations,d
ata=data_train,method="nnet",trControl=nn_trainControl,tuneGrid=nn_tuneGrid)
```

```
## # weights:  946
## initial  value 49453.927508
## iter  10 value 43863.460340
## iter  20 value 40382.798635
## iter  30 value 37505.261873
## iter  40 value 35236.559998
## iter  50 value 33608.339828
## iter  60 value 32861.702378
## iter  70 value 31819.216939
## iter  80 value 30934.205429
## iter  90 value 30428.354909
## iter 100 value 30033.687437
## final  value 30033.687437
## stopped after 100 iterations
## # weights:  946
## initial  value 48910.634061
## iter  10 value 44545.493973
## iter  20 value 37716.796804
## iter  30 value 34941.270849
## iter  40 value 31208.559049
## iter  50 value 30022.255571
## iter  60 value 29642.618032
## iter  70 value 29311.277651
## iter  80 value 29154.734835
## iter  90 value 29041.250737
## iter 100 value 28947.894981
## final  value 28947.894981
## stopped after 100 iterations
## # weights:  946
## initial  value 56286.614438
## iter  10 value 42205.296805
## iter  20 value 37985.206672
## iter  30 value 35697.429350
## iter  40 value 34460.406845
## iter  50 value 33065.625971
## iter  60 value 31455.782480
## iter  70 value 30957.721985
## iter  80 value 30520.923455
## iter  90 value 29855.698416
## iter 100 value 29549.725967
## final  value 29549.725967
## stopped after 100 iterations
## # weights:  946
## initial  value 48871.443807
## iter  10 value 43746.106173
## iter  20 value 42865.606983
## iter  30 value 39205.976855
## iter  40 value 34849.020670
## iter  50 value 33161.249913
## iter  60 value 32042.388785
## iter  70 value 31086.448700
## iter  80 value 30452.783919
```

```
## iter  90 value 30223.174408
## iter 100 value 29702.194831
## final  value 29702.194831
## stopped after 100 iterations
## # weights:  946
## initial  value 51121.708489
## iter  10 value 43479.016413
## iter  20 value 37609.681272
## iter  30 value 32645.483271
## iter  40 value 30269.527269
## iter  50 value 29687.528085
## iter  60 value 29305.195501
## iter  70 value 29090.062403
## iter  80 value 28963.168127
## iter  90 value 28826.093169
## iter 100 value 28749.071089
## final  value 28749.071089
## stopped after 100 iterations
## # weights:  946
## initial  value 71011.062020
## iter  10 value 54448.240675
## iter  20 value 51143.565351
## iter  30 value 42756.411149
## iter  40 value 40960.243892
## iter  50 value 38870.242087
## iter  60 value 38206.771824
## iter  70 value 37596.286203
## iter  80 value 37232.105544
## iter  90 value 36882.554511
## iter 100 value 36608.874204
## final  value 36608.874204
## stopped after 100 iterations
```

```
nnPreds=predict(nnModel,newdata=data_test)
```

Confusion Matrix

```
table(nnPreds,data_test$is_canceled)
```

```
##
## nnPreds     0     1
##       0 17361  4803
##       1  1399  6244
```

Accuracy

```
mean(nnPreds==data_test$is_canceled)
```

```
## [1] 0.7919281
```

Important variables

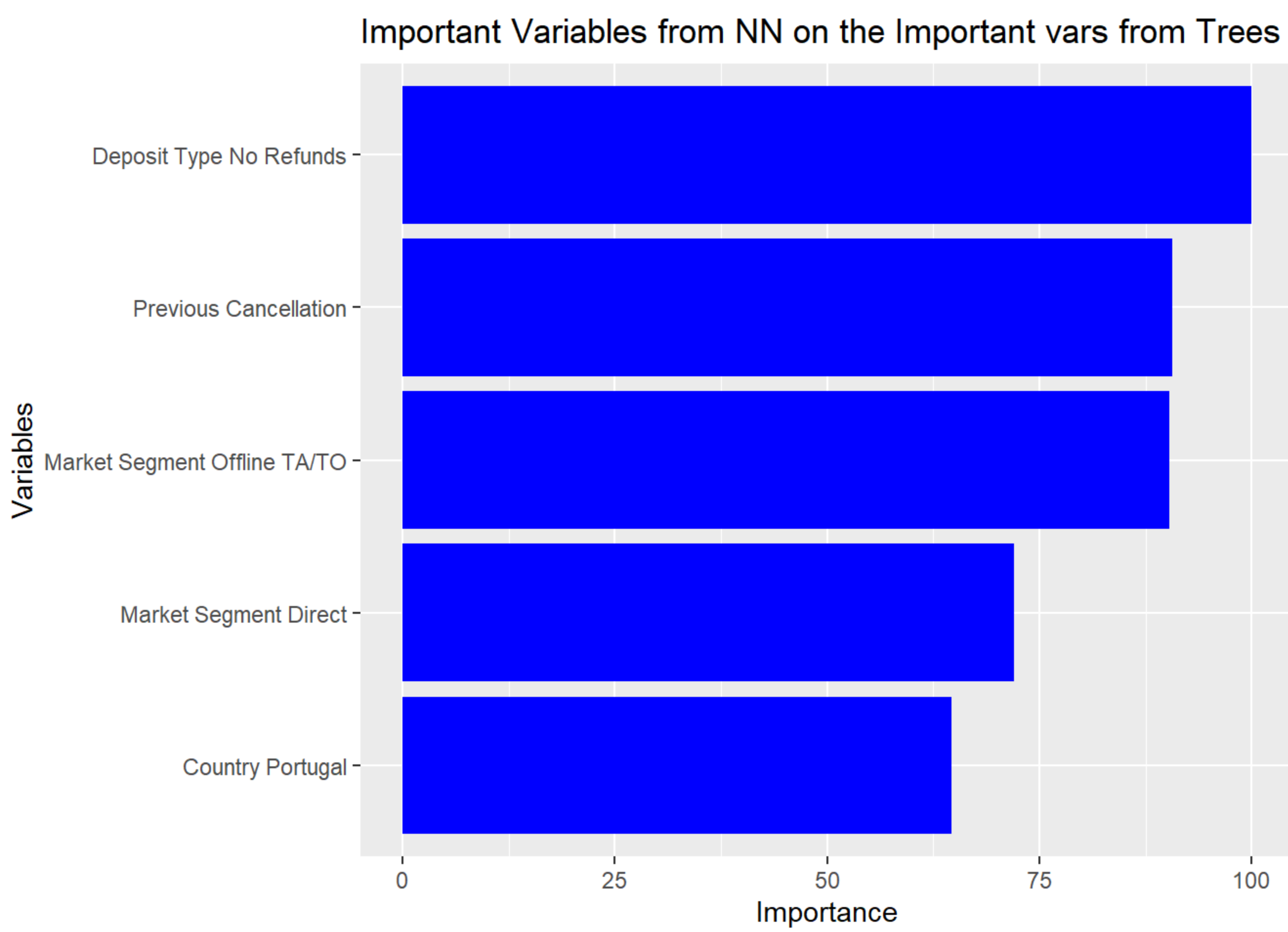
```
nnImportance=varImp(nnModel)
Top5nnImportance=nnImportance$importance%>%as.data.frame()%>%rownames_to_column("Feature") %>% a
rrange(desc(Overall))%>%head(5)
Top5nnImportance
```

```
##                    Feature   Overall
## 1               countryESP 100.00000
## 2               countryARE  90.65934
## 3               countryAGO  90.30891
## 4 market_segmentCorporate  72.05516
## 5               countryBRA  64.59718
```

```
#Found deposit_typeNon Refund, previous_cancellations, market_segmentOffline TA/TO, market_segme
ntDirect, countryPRT important
```

Important Variables Plot

```
ggplot(data=Top5nnImportance,mapping=aes(x=Overall,y= reorder(Feature, Overall)))+geom_bar(stat
="identity",fill="blue")+scale_y_discrete(labels=c("Country Portugal","Market Segment Direct","M
arket Segment Offline TA/TO","Previous Cancellation","Deposit Type No Refunds"))+xlab("Importanc
e")+ylab("Variables")+ggtitle("Important Variables from NN on the Important vars from Trees")
```



Neural Net on Important variables from Logistic Regression

```
#The final values used for the model were size = 1 and decay = 0.1. Commented out as it took too long to run
#NN_trainControl=trainControl(method="cv",number=5,verboseIter = TRUE)
#NN_tuneGrid=expand.grid(size=c(1,2), decay = c(0.1, 0.2, 0.3))
#NNModel=train(is_canceled~lead_time+deposit_type+country+market_segment+previous_cancellations,
data=data_train,method="nnet",trControl=NN_trainControl,tuneGrid=NN_tuneGrid, trace = FALSE)
#NNModel
```

```
NN_trainControl=trainControl(method="cv",number=5)
NN_tuneGrid=expand.grid(size=1, decay = 0.1)
NNModel=train(is_canceled~company+agent+assigned_room_type+required_car_parking_spaces,data=data_train,method="nnet",trControl=NN_trainControl,tuneGrid=NN_tuneGrid, trace = FALSE)
NNPreds=predict(NNModel,newdata=data_test)
```

Confusion Matrix

```
table(NNPreds,data_test$is_canceled)
```

```
##
## NNPreds     0     1
##       0 16425  6199
##       1  2335  4848
```

Accuracy

```
mean(NNPreds==data_test$is_canceled)
```

```
## [1] 0.7136914
```

Important variables

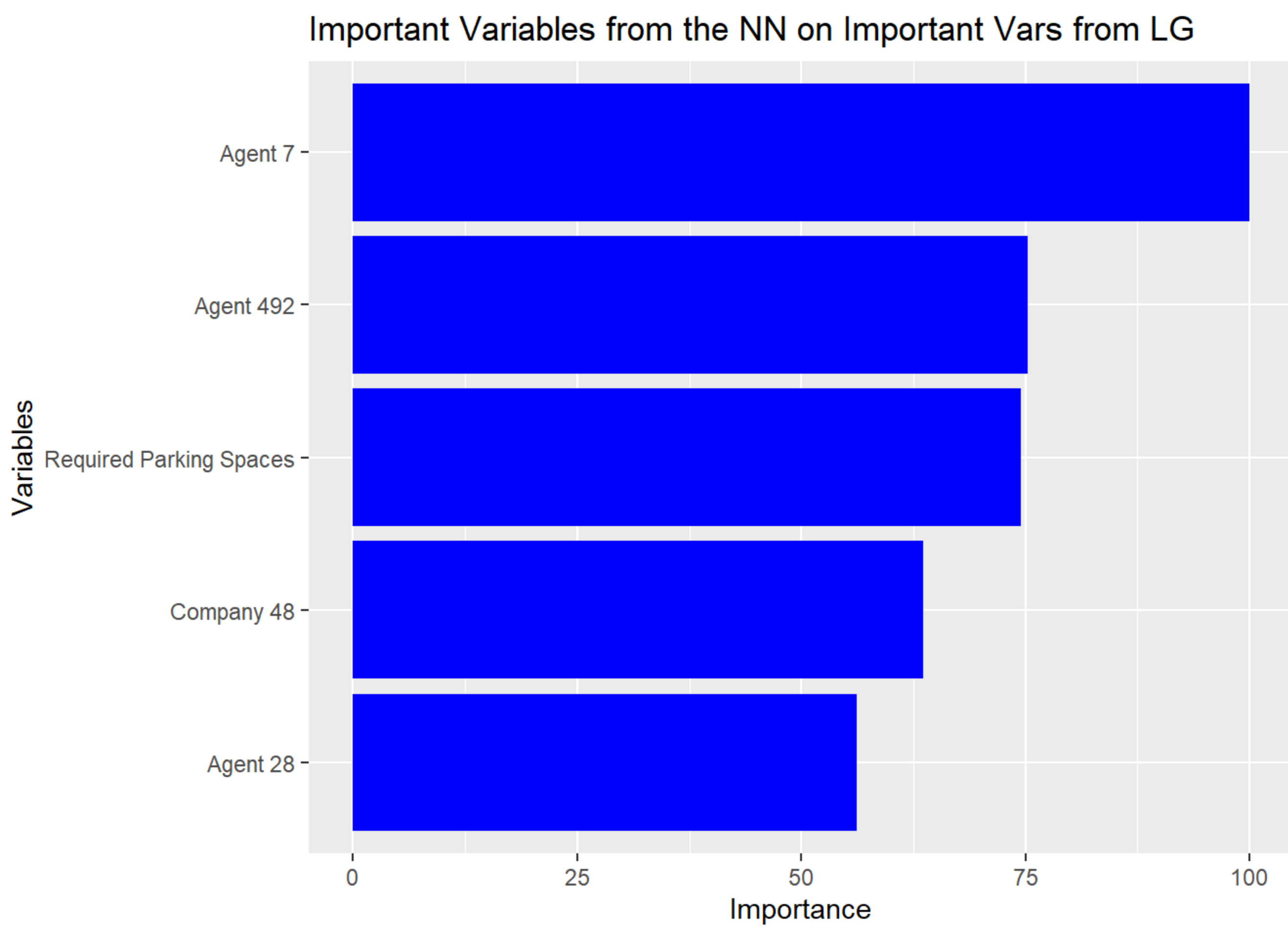
```
NNImportance=varImp(NNModel)
Top5NNImportance=NNImportance$importance%>%as.data.frame()%>%rownames_to_column("Feature") %>% arrange(desc(Overall))%>%head(5)
Top5NNImportance
```

```
##                       Feature   Overall
## 1         assigned_room_typeK 100.00000
## 2 required_car_parking_spaces  75.22501
## 3                    agent236  74.50452
## 4                    agent220  63.59099
## 5                    agent464  56.17624
```

```
#Found agent7, agent492, required_car_parking_spaces,company48,agent28 to be important
```

Important Variables Plot

```
ggplot(data=Top5NNImportance,mapping=aes(x=Overall,y= reorder(Feature, Overall)))+geom_bar(stat
="identity",fill="blue")+scale_y_discrete(labels=c("Agent 28","Company 48","Required Parking Spa
ces","Agent 492","Agent 7"))+xlab("Importance")+ylab("Variables")+ggtitle("Important Variables f
rom the NN on Important Vars from LG")
```



Train test split Only. No cross validation. Computationally faster. Better Results too. Will explore more tmr.

```
rg=ranger(is_canceled~.,data=data_train,num.trees=1000,importance="impurity")
```

```
rgPreds=predict(rg,data=data_test)
mean(rgPreds$predictions==data_test$is_canceled)
```

```
## [1] 0.8965679
```

```
importance(rg)
```

```
##                          hotel                      lead_time
##                      393.04851                     3841.73627
##              arrival_date_year             arrival_date_month
##                      945.25155                     1074.25613
##      arrival_date_day_of_month        stays_in_weekend_nights
##                     1795.40185                      748.70827
##           stays_in_week_nights                         adults
##                     1168.39678                      434.71805
##                       children                         babies
##                      229.69418                       32.08636
##                           meal                        country
##                      451.36078                     4278.89217
##                 market_segment           distribution_channel
##                     1750.30362                      395.76485
##              is_repeated_guest         previous_cancellations
##                       73.87236                     1232.20314
## previous_bookings_not_canceled             reserved_room_type
##                      143.92530                      557.75932
##             assigned_room_type                booking_changes
##                      923.09535                      830.48841
##                   deposit_type                          agent
##                     4565.07869                     2234.72725
##                        company           days_in_waiting_list
##                      171.00354                       75.43999
##                  customer_type                            adr
##                     1008.58413                     2552.53808
##    required_car_parking_spaces      total_of_special_requests
##                      895.47107                     2326.98966
##            lead_timeCategories                      Continent
##                     1488.44875                      401.37790
##           ArrivalHolidaySeason                  ArrivalSeason
##                      386.83233                      562.61466
```

```
table(rgPreds$predictions,data_test$is_canceled)
```

```
##
##         0     1
##   0 17595  1918
##   1  1165  9129
```

```
tree=rpart(is_canceled~.,data=data_train, method = "class")
```

```
treePreds=predict(tree,newdata=data_test,type="class")
mean(treePreds==data_test$is_canceled)
```

```
## [1] 0.8121582
```

```
table(treePreds,data_test$is_canceled)
```

```
##
## treePreds     0     1
##         0 17364  4203
##         1  1396  6844
```

Logistic Regression

```
lg=multinom(is_canceled~.,data=data_train,method="Binomial")
```

```
## # weights:  947 (946 variable)
## initial  value 61982.607180
## iter  10 value 42306.597569
## iter  20 value 40200.160295
## iter  30 value 39495.681300
## iter  40 value 37263.253669
## iter  50 value 34531.768919
## iter  60 value 33469.863585
## iter  70 value 32297.193898
## iter  80 value 31356.737584
## iter  90 value 31057.993696
## final  value 31055.180283
## converged
```

```
lgPreds=predict(lg,newdata=data_test)
mean(lgPreds==data_test$is_canceled)
```

```
## [1] 0.8408092
```

```
table(lgPreds,data_test$is_canceled)
```

```
##
## lgPreds     0     1
##       0 16891  2876
##       1  1869  8171
```