Fleet Optimization

Final project submission for Certificate course in Engineering Excellence course from INSOFE

10/8/2014

Contents

Executive Summary..3
Introduction...3
Fuel & Carbon Offset Savings Calculations...4
Crash Cost..5
Problem Definition:...8
Data driven visualization..10
Processing Techniques Used:...16
Model Outputs:..18
Future enhancements..19
R code...19

Executive Summary

Fleet Optimization is a complex problem. This paper discusses accurate prediction as a first step to optimization goal. There are multiple factors and stakeholders affecting the optimization process for a large fleet. Some of the common factors affecting fleet efficiencies are Driver, Vehicle, Customer, Route, Fuel and Payload. For large fleet the problems are more complex and could result in potentially large savings to the operational costs, which typically increase at the rate of 1-2% year on year.

Introduction

Advantages of implementing optimized solution are

- Improved customer Satisfaction via improved or on-time service
- Improved operational efficiency

  Predictive/forward planning

  Monitoring and flagging infractions

  Adoption of technology and training
- Increased Safety

  Driver & Field Staff safety

  Enforced fleet security

  Major crash investigation and vehicle recalls
- Compliance

Toll administration

Fleet and Driver Licensing and Violation reporting

Crew safety & Haz-Mat Compliance and Certification

Fuel Surcharge Calculations and report mile for IFTA miles per states and paying surcharges as applicable per state

- Environment

Reducing Carbon Footprint - Carbon Offset

Fuel & Carbon Offset Savings Calculations

Fuel and Carbon Saving calculator shown below shows the savings that can be achieved for a 500 vehicle fleet.

Calculation parameters:

Total Vehicles In Fleet = 500

Est. Miles per Day per Vehicle = 120

Average Miles per Gallon per Vehicle = 7.8

Days Worked per Week = 6

Estimated Idling Hours per Day per Vehicle = 3.0

Fuel Cost = $3.09

Potential Savings as per Calculator

$1,156,896 per year by decreasing excess idling

$1,112,400 per year through route optimization

$593,280 per year by improving driving behavior

Potential savings of $2,862,576 per year and reduce our carbon emissions by 8,026.88metric tons. For a fleet size of few thousands vehicles the savings really add up.

Crash Cost

Beyond the basic optimization problem predictive analysis in other areas are also important to fleet cost reductions. The ability to predict and curb rash driving behavior and infractions could also lead to efficiencies for a large fleet. Another factor that drives up the fleet costs are accidents According to FMSCA cost calculator the average cost of accidents for a Straight truck in 2008 were as indicated:

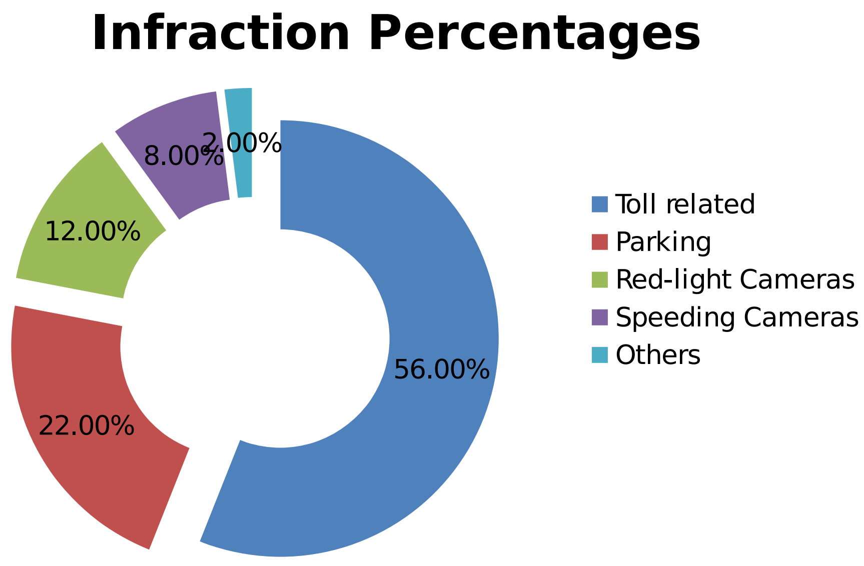
Average Cost of Crash for Straight Truck – No Trailer

No injuries Crash - $ 97,811

Crash where at least one person was injured - $247,343

Fatal Crash - $6,314,659

The chart below shoes the infractions distribution for the type of infractions occurred in one metro area:



Driving behavior, fleet composition, fuel type and pricing, payload for customer service agreement, Customer Services agreements and routes all factors affect the efficiency of a fleet. Table below shows how all of these factors are interrelated and their influence on the efficiency of the operations.

Summary of the effects of factors influencing vehicle fuel economy.

Level	Factor	Effect
Strategic	Vehicle class	38%
	Vehicle model	800% all cars; 355% cars excluding fully electric; 227% cars excluding fully electric and hybrids; 100% all pickups
	Vehicle configuration	18% cars, 28% pickups
	Out-of-tune engine	4-40%
	Tires with 25% higher rolling resistance	3-5%
	Tires underinflated by 5 psi	1.5%
	Improper engine oil	1-2%
Tactical	Route selection: road type	variable
	Route selection: grade profile	15-20%
	Route selection: congestion	20-40%
	Carrying extra 100 pounds	≤2%
Operational	Idling	variable
	Driving at very high speeds	30%
	Not using cruise control	7% (while at highway speeds)
	Using air conditioner	5-25%
	Aggressive driving	20-30%

There are a multitude of optimization problems that can be addressed in the fleet optimization area. Some of them are listed under:

- Minimize accidents
- Minimize Vehicles break down
- Minimize fuel cost
- Optimize routes based on traffic and risk- e.g. avoiding areas with schools during school start and let off time, avoiding peak rush hours, avoiding construction zones, avoiding areas with special events

- Optimize routes based on Cost – e.g. garbage collections on weekend for downtown districts in the city Vs cost of Overtime / Double time pay for employees on the route
- Optimize routes based on Customer demands and schedules
- Optimize routes based on route characteristics – Tolls, Ferries, Overpass, Underpass and Bridges with payload restrictions
- Minimum Staffing levels to meet customer SLAs
- Min Staffing during an emergency to maintain Operations

Maximum Route Circuity

Problem Definition:

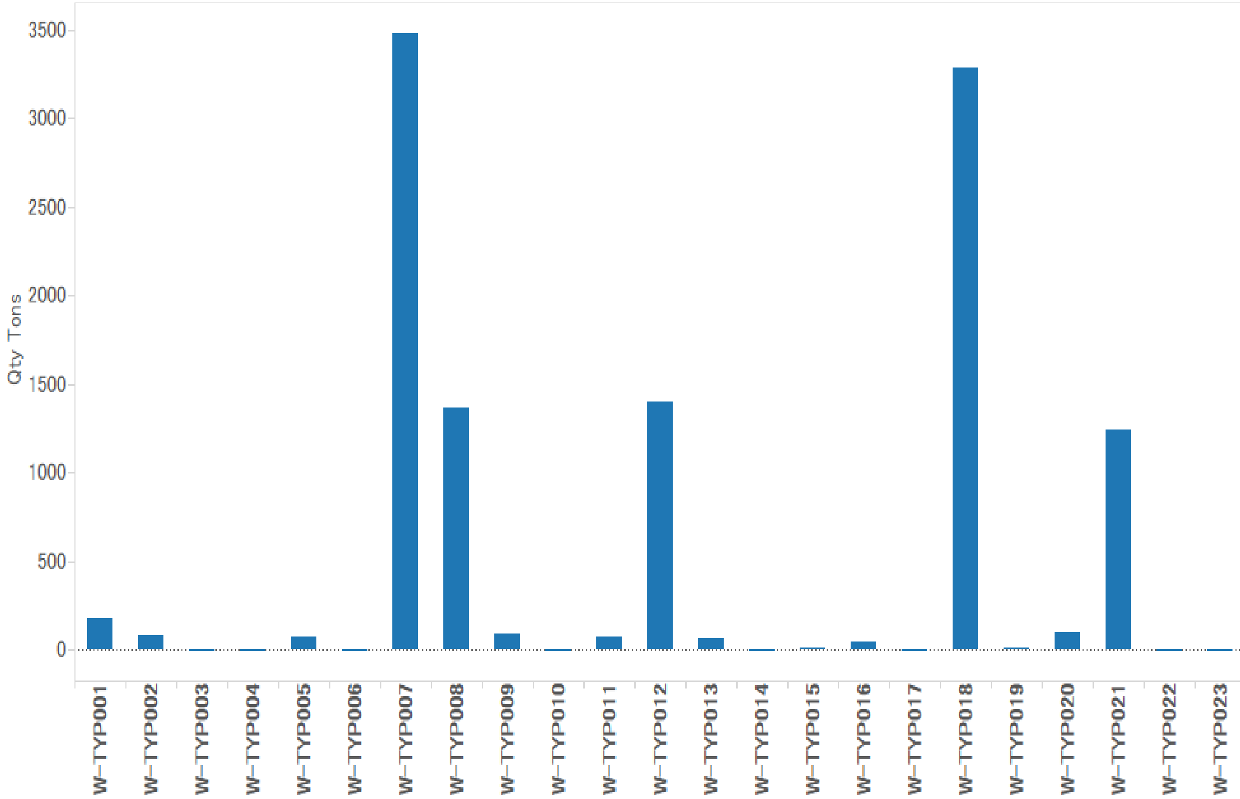
Junk Busters Inc wanted to optimize the number of customers serviced in a given day by its fleet and as a first step to optimization, the time take to service was accurately needed to be predicted using machine learning techniques. The GPS data for the vehicles was available. This GPS vehicle data was used to calculate the start and stop times for the vehicle from each location. The summary statistics of the data is as given.

Summary Statistics:

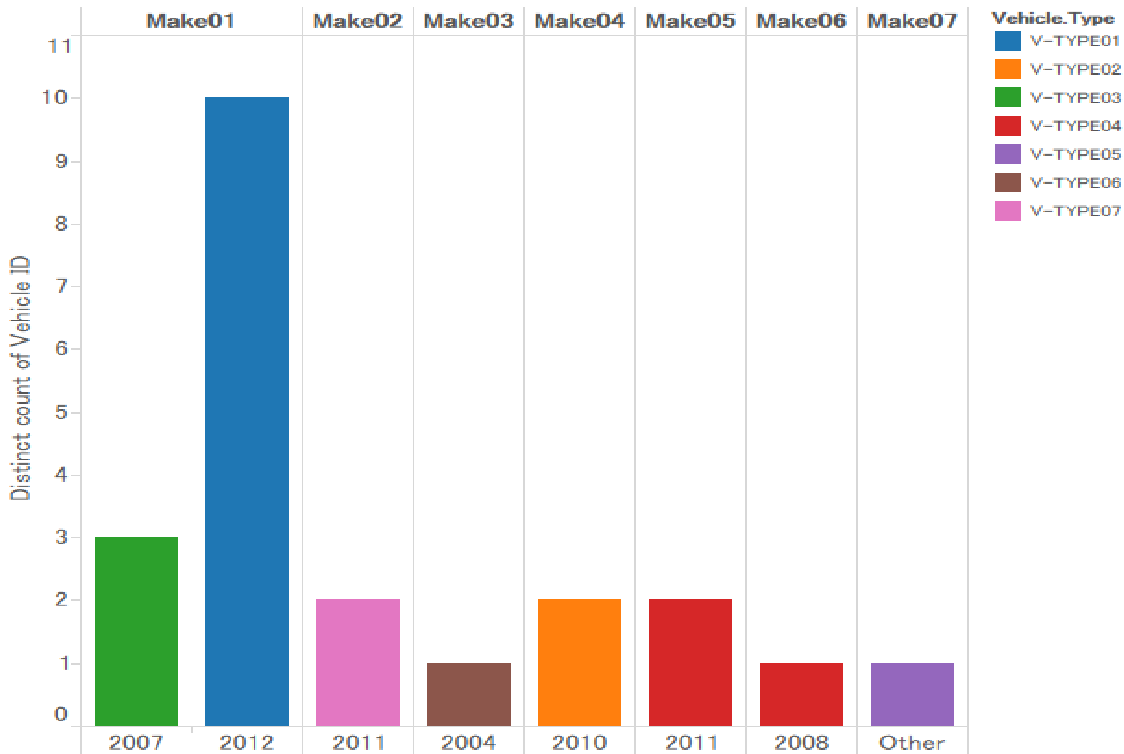
Attributes	**Attribute type**	**Summary Statistics**	**Comments**
Customer	Categorical	109 Customers	4 customer account for 40% of the trips
Driver	Categorical	32 Drivers	DriverID 16 used for only one trip
Vehicle Id	Categorical	22 Vehicle IDs	(11-85)Min Max range for trips per Vehicle by Vehicle ID
Vehicle Type	Categorical	7 Vehicle types	VTYPE-01 accounts for 48.9% of the trips
Make	Categorical	7 Makes	MAKE01 accounts for 65.7% of all trips
Model	Categorical	9 categories in model year	Model years 2011 and 2012 accounts for 51.9% of the trips
Waste Type	Categorical	23 waste categories	5 waste categories account for approximatey 90% of the trips
Waste Qty	Numeric	Min = -0.44Tons ,Max =29.35 Tons	Mean10.24 - Median 10.85
Kms	Numeric	Min =18KM ,Max =254KM	Mean79.49 - Median 53.74
Date	Date	Aug to Dec 2013	Data spans over 5 months
FLStartTime	Time	Recorded start time from Facilities	
FLReachTime	Time	Recorded reach time to Facilities	
CustLeftTime	Time	Recorded start time from Customer	
CLReachTime	Time	Recorded reach time to Customer	
Class Variables – Derived			
Time to Customer	Time		FLStartTime - CLReachTime
Time at Customer	Time		CLReachTime - CustLeftTime
Time to FL	Time		CustLeftTime - FLReachTime

Data driven visualization

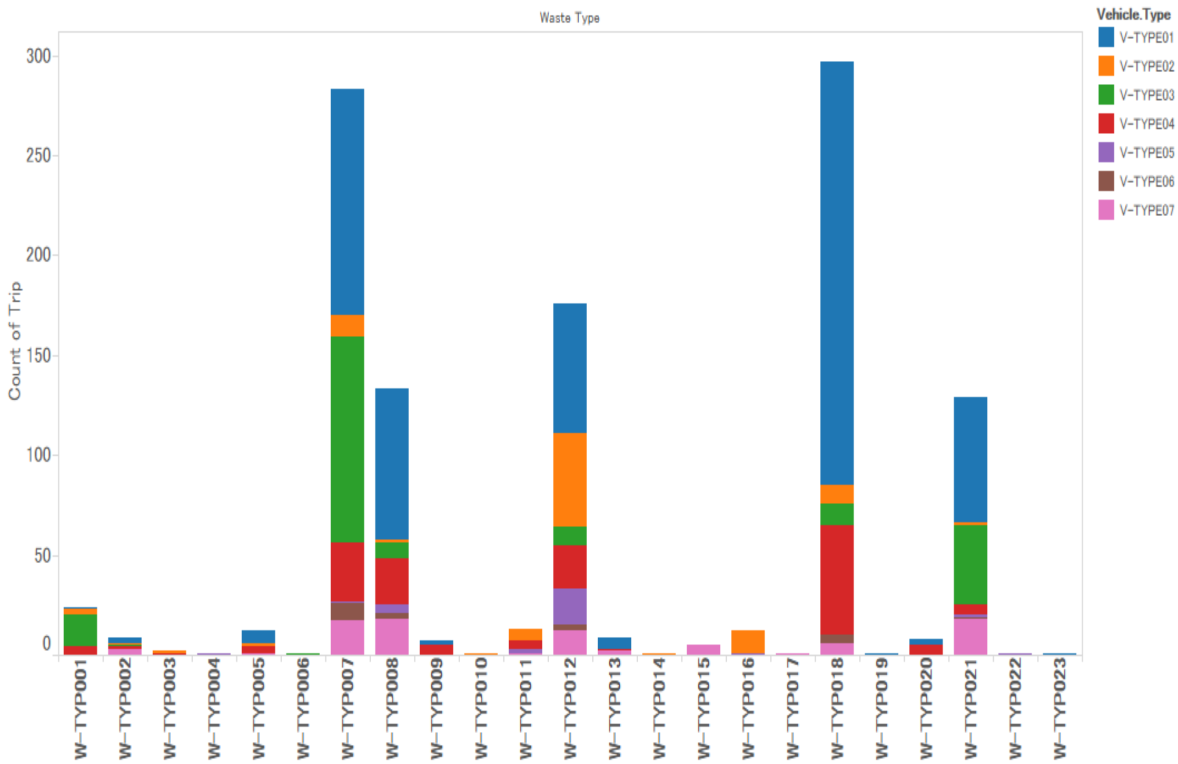
Waste Type Distribution shows Waste types W-TYP007, W-TYP008, W-TYP0012, W-TYP0018 & W-TYP0021 account for 90% of all waste.



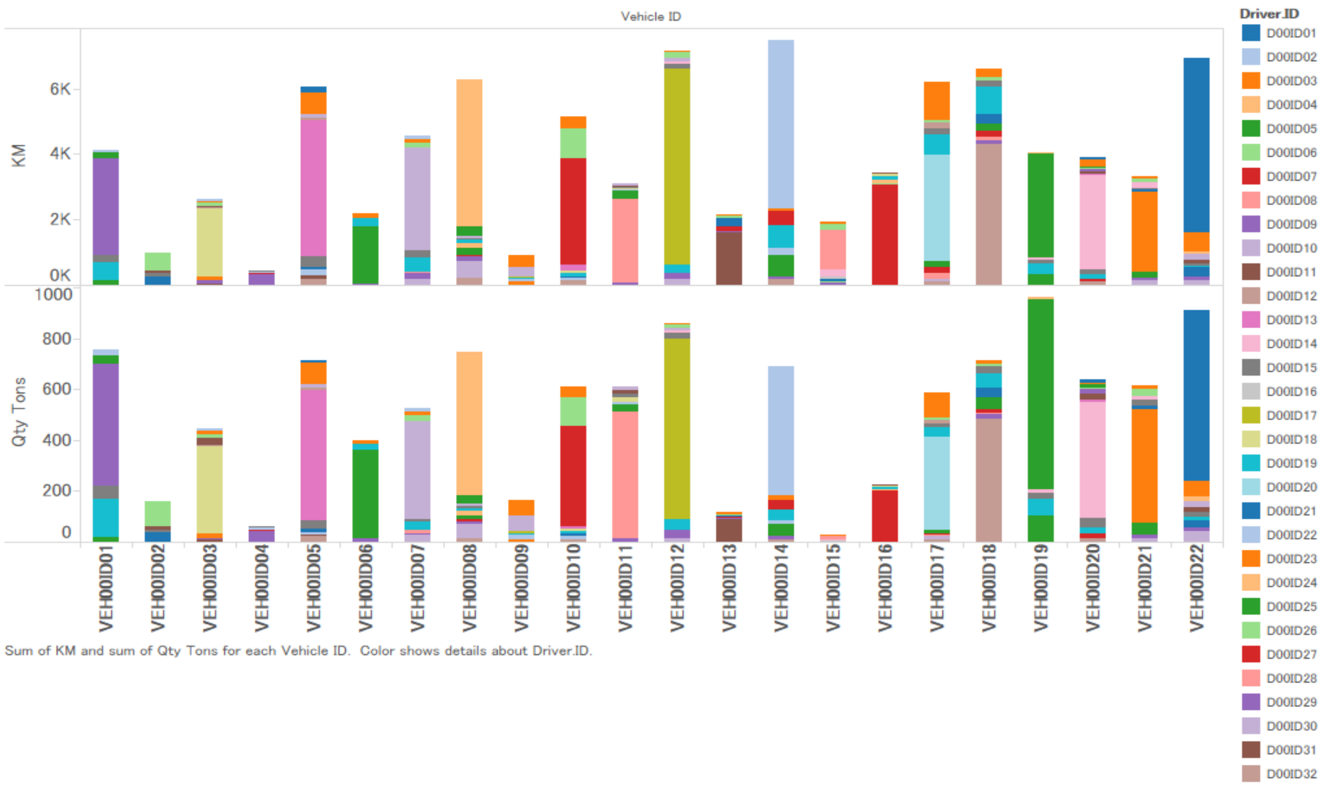
Vehicle Details by make model and type show Older/Unknown Vehicles Models may need to be replaced at some point.



WasteType by Vehicle Type Used shows no dependency of waste type while replacing vehicles.

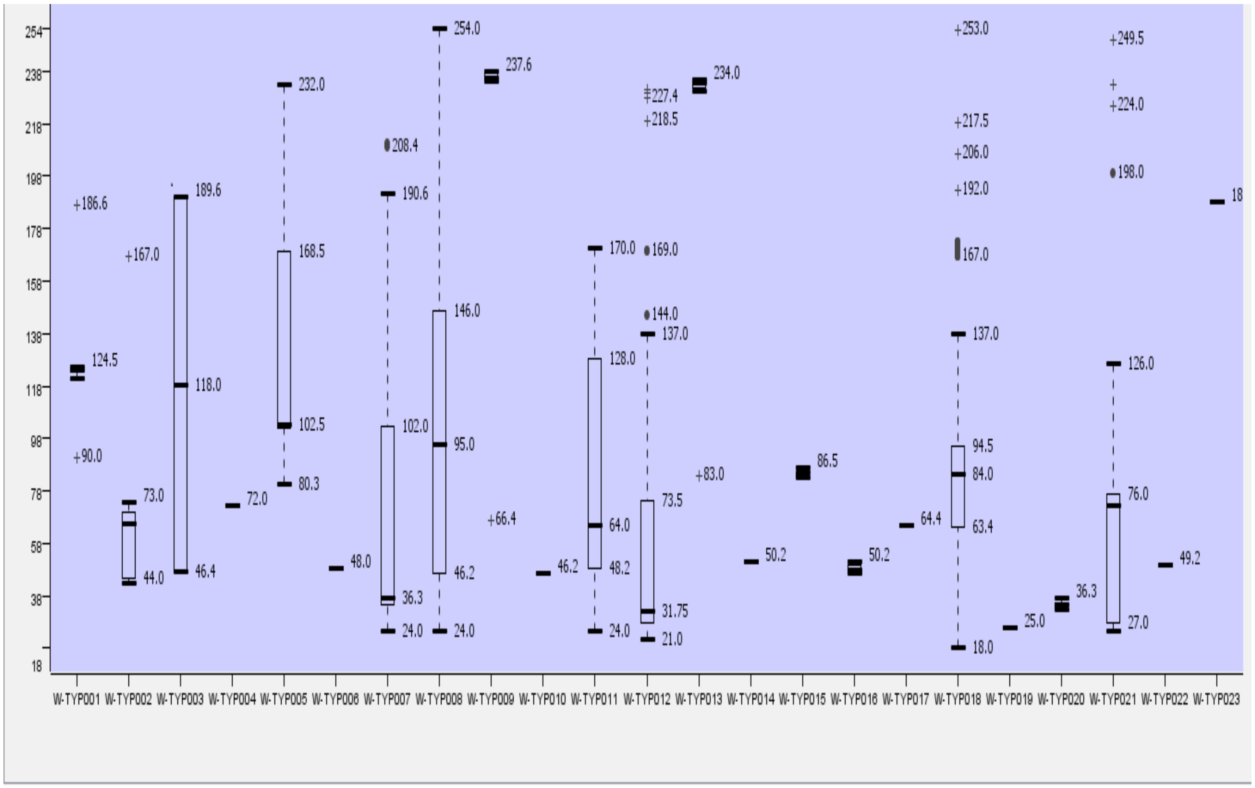


Vehicle Driver Pattern graph indicates there is one primary Driver for each Vehicle

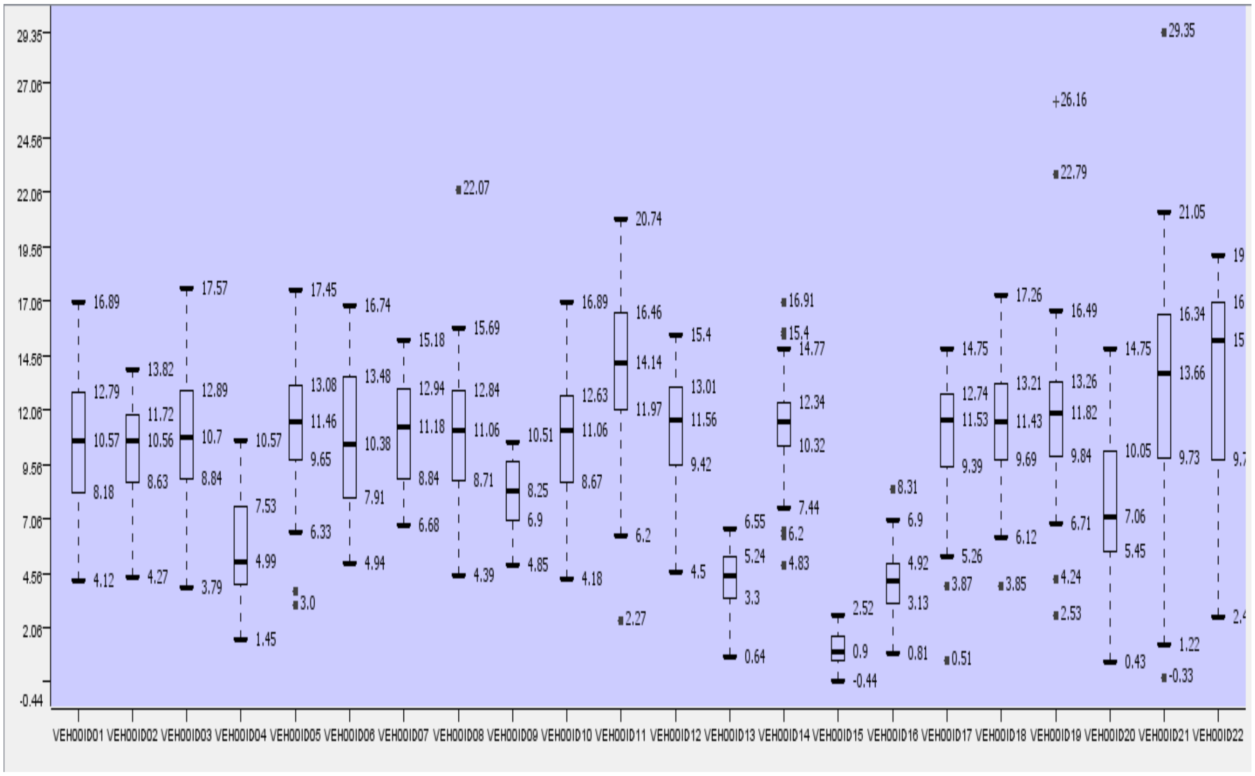


Sum of KM and sum of Qty Tons for each Vehicle ID. Color shows details about Driver.ID.

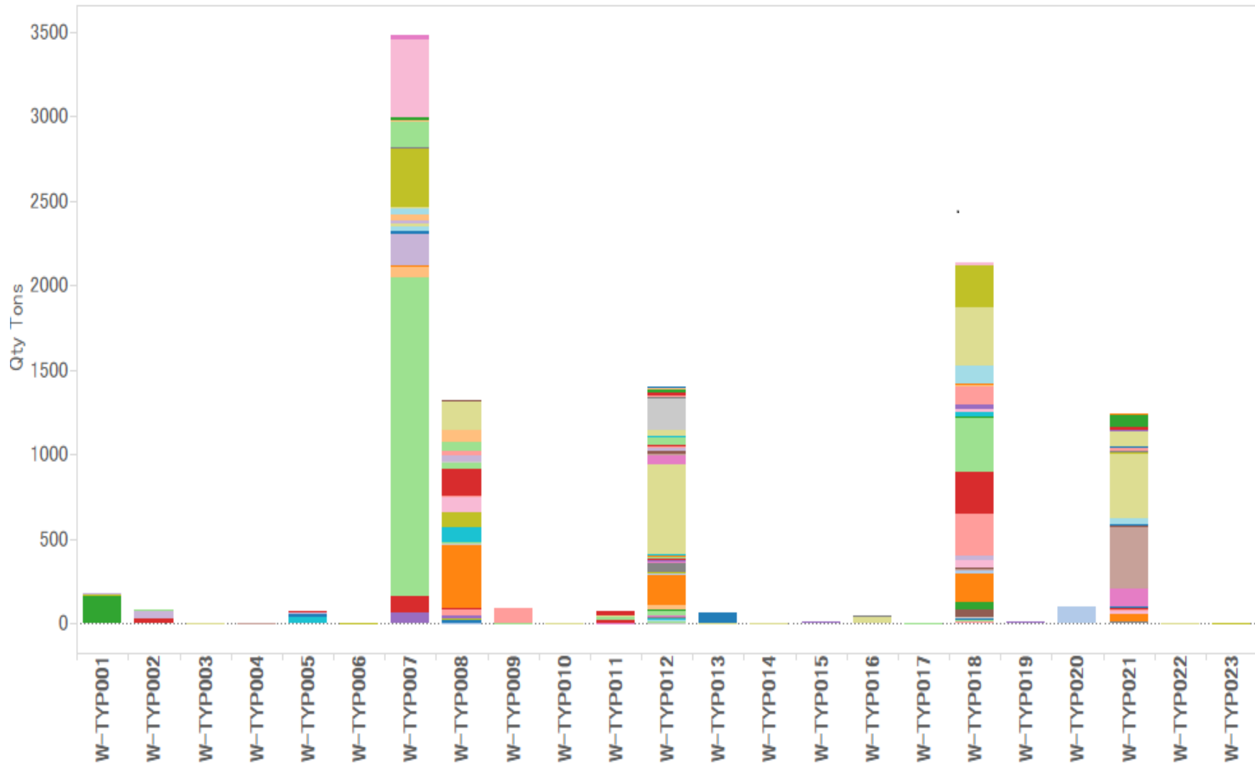
Waste Type By Distance shows waste types that may be potential targets for future expansions:



VehicleID by Qty in Tons shows that few vehicles are used to carry lighter loads exclusively.



Waste Type Distribution by customers shows huge reliance on one Customer Waste Type combination



Processing Techniques Used:

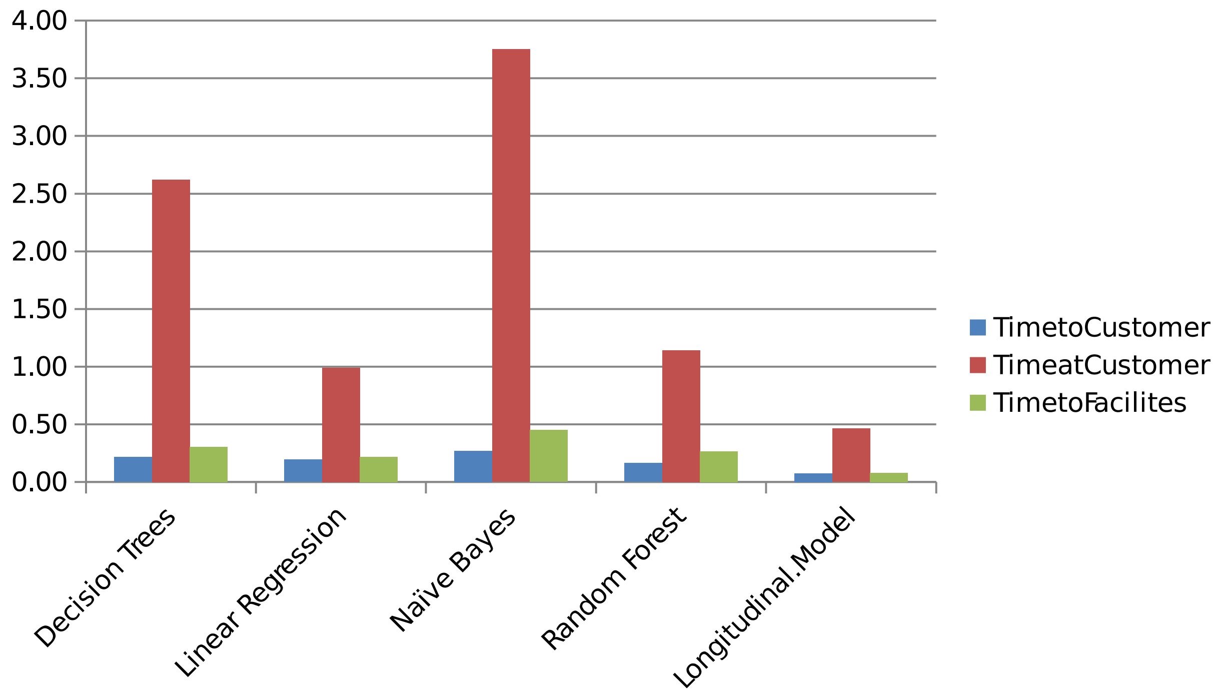
To predict time, various machine learning algorithms were used. The algorithms used were Linear Regression, Naïve Bayes, Decision Trees and Random forest. Cross sectional models as well as longitudinal models were developed using these techniques and the algorithms that gave good output for all the three time prediction was Linear Regression

The goodness of fit for these models was determined using Mean square Error. The longitudinal Model with Linear Regression has the least error and the error metric and graph are shown below.

Error Metric:

Class Variables	Decision Trees	Linear Regression	Naïve Bayes	Random Forest	Longitudin al.Model
TimetoCusto mer	0.22	0.19	0.27	0.17	0.07
TimeatCusto mer	2.62	0.99	3.75	1.14	0.46
TimetoFaciliti es	0.31	0.22	0.45	0.27	0.08

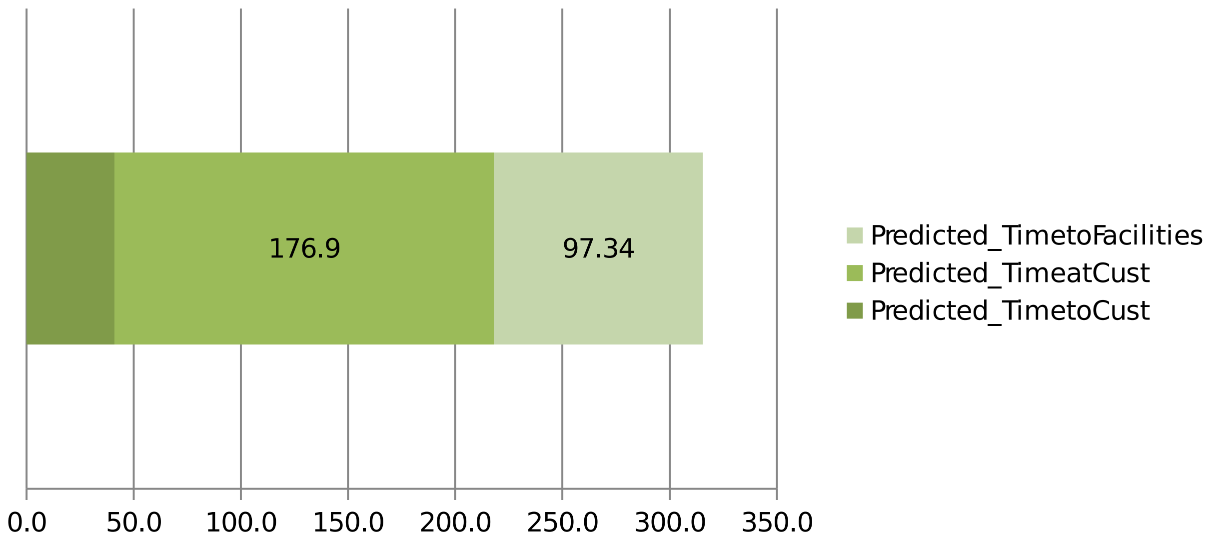
Error metric plotted:



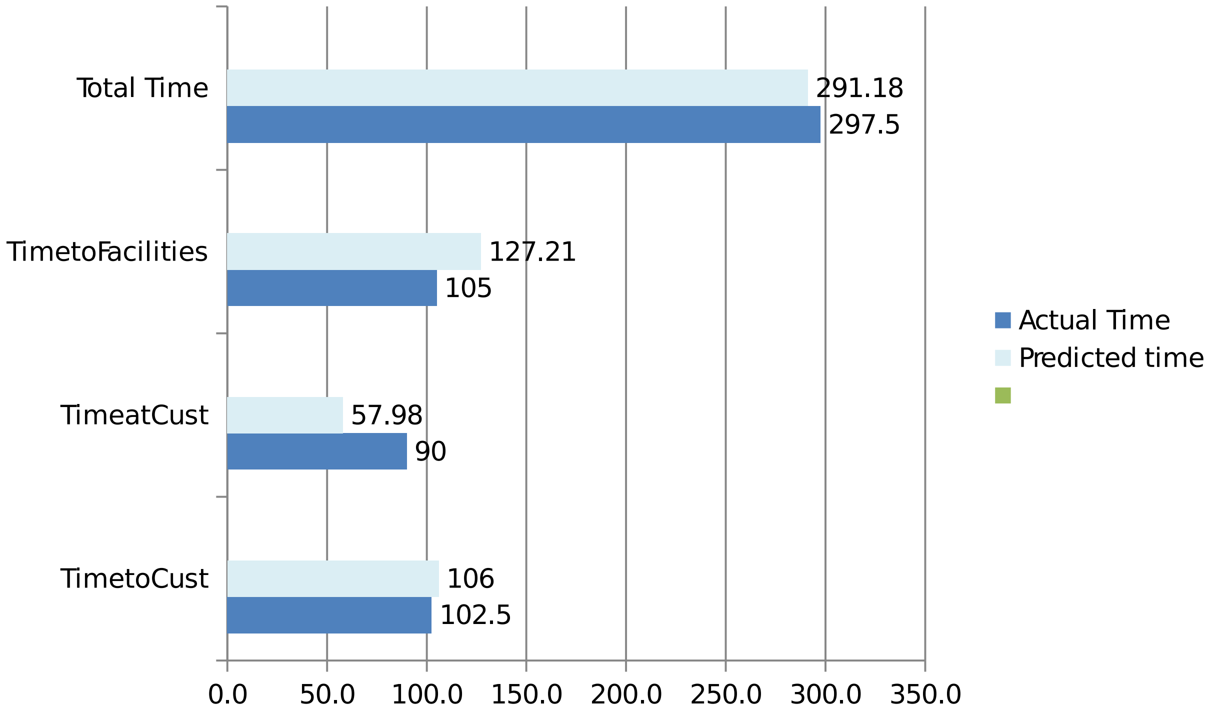
Model Outputs:

Prediction Based on input selection

WasteType	QtyTons	KM
W-TYP018	10	81



Actual Vs Predicted time based on Customer selected:



Future enhancements

Beyond accurate prediction the goal of this projects future enhancement is to develop a shiny interface an use genetic algorithm to optimize the routing for maximum customers serviced(trips) in a given day.

R code

```
rm(list=ls(all=TRUE))

setwd("C:/Users/ Desktop/Fleet Management")

library(lubridate)

fleetdata<-read.table("FleetManagement.csv", header = TRUE, sep = ",")

##Converting from 12HR AM:PM format to 24 HR HH:MM format

fleetdata$FLStartTime<-substr(strptime(fleetdata$FLStartTime, "%I:%M %p"),11,19)
```

```
fleetdata$CLReachTime<-substr(strptime(fleetdata$CLReachTime, "%I:%M
%p"),11,19)

fleetdata$CustLeftTime<-substr(strptime(fleetdata$CustLeftTime, "%I:%M
%p"),11,19)

fleetdata$FLReachTime<-substr(strptime(fleetdata$FLReachTime, "%I:%M
%p"),11,19)

fleetdata$MM<-formatC(fleetdata$MM, width = 2)

fleetdata$DD<-formatC(fleetdata$DD, width = 2)

fleetdata$Date <- as.character(paste(fleetdata$YY, fleetdata$MM,

                                     fleetdata$DD, sep = "-"))

fleetdata<-fleetdata[,-(2:4)]

fleetdata$Date.FLStartTime <- as.character(paste(fleetdata$Date,

                                                 fleetdata$FLStartTime

                                                 , sep = " "))

fleetdata$Date.FLReachTime <- as.character(paste(fleetdata$Date,

                                                 fleetdata$FLReachTime

                                                 , sep = " "))

fleetdata$Date.CustLeftTime <- as.character(paste(fleetdata$Date,

                                                  fleetdata$CustLeftTime

                                                  , sep = " "))

fleetdata$Date.CLReachTime<- as.character(paste(fleetdata$Date,

                                                fleetdata$CLReachTime

                                                , sep = " "))

head(fleetdata$Date.FLReachTime)

fleetdata<-fleetdata[,-(8:11)]

fleetdata$TimetoCust<-
difftime(fleetdata$Date.CLReachTime,fleetdata$Date.FLStartTime,units="hours")
```

```
fleetdata$TimeatCust<-
difftime(fleetdata$Date.CustLeftTime,fleetdata$Date.CLReachTime,units="hours")

fleetdata$TimetoFL<-
difftime(fleetdata$Date.FLReachTime,fleetdata$Date.CustLeftTime,units="hours")

fleetdata$TimetoCust<-as.numeric(fleetdata$TimetoCust)

fleetdata$TimeatCust<-as.numeric(fleetdata$TimeatCust)

fleetdata$TimetoFL<-as.numeric(fleetdata$TimetoFL)

summary(fleetdata)

write.csv(fleetdata, "preprocessfleet.csv")


fleet<-fleetdata[,c(1:12,17:19)]

write.csv(fleet,"fleet.csv")

rm(fleetdata)

summary(fleet)


# Plotting histograms in r

hist(fleet$QtyTons)

hist(fleet$KM)

hist(fleet$TimetoCust)

hist(fleet$TimeatCust)

hist(fleet$TimetoFL)


# Plotting simple barchart in R

customer <- table(fleet$Customer.ID)

barplot(customer, main="count",

        xlab="Customer$ID")
```

```
driver<- table(fleet$Driver.ID)

driver

barplot(driver, main="count",

        xlab="Driver$ID")

# Boxplot of WasteType by QtyTons

boxplot(QtyTons~WasteType,data=fleet, main="Waste type by Qty in Tons",

        xlab="QtyTons", ylab="WasteType")

####Using table command to evaluate multiple variables

library(reshape2)

driver.waste <- data.frame(table(fleet$Driver.ID,fleet$WasteType))

driver.waste<- acast(driver.waste, Var1~Var2,mean)

barplot(driver.waste, main="count",

        xlab="Driver counts by Waste Type")

customer.waste<-data.frame(table(fleet$Customer.ID,fleet$WasteType))

customer.waste<-acast(customer.waste,Var1~Var2,mean)

barplot(customer.waste, main="count",

        xlab="Customer counts by Waste Type", col=c("400"))

summary(customer.waste)

driver.model <- data.frame(table(fleet$Driver.ID,fleet$Model))

driver.model<- acast(driver.model, Var1~Var2,mean)


barplot(driver.model, main="count",

        xlab="Driver counts by Model")

driver.make <- data.frame(table(fleet$Driver.ID,fleet$Make))

driver.make<- acast(driver.make, Var1~Var2,mean)

barplot(driver.make, main="count",
```

```
        xlab="Driver counts by make")

driver.vehicle<- data.frame(table(fleet$Driver.ID,fleet$VehicleID))

driver.vehicle<- acast(driver.vehicle, Var1~Var2,mean)

bp<-barplot(driver.vehicle, main="Driver Trip Count by Vehicle ID")

vehicle.make<- data.frame(table(fleet$VehicleID,fleet$Make))

vehicle.make<- acast(vehicle.make, Var1~Var2,mean)

###replace with Group bar plot

barplot(vehicle.make, main="count",

        xlab="Vehicle counts by Make")

###Binning numeric variable in R ###

library(car)

###Only use this code for plotting

fleet$QtyTons1<-cut(fleet$QtyTons,breaks=10,labels=c(1,2,3,4,5,6,7,8,9,10))

fleet$QtyTons1<-as.numeric(fleet$QtyTons1)

hist(fleet$QtyTons1)

fleet$KM1<-cut(fleet$KM,breaks=11,labels=c(1,2,3,4,5,6,7,8,9,10,11))

fleet$KM1<-as.numeric(fleet$KM1)

hist(fleet$KM1)


###Binning variables using Recode in Deducer

####fleet11[c("WasteType")] <- recode.variables(fleet11[c("WasteType")] ,

####                    "'W-TYP007' -> 'W-TYP007';

####                     'W-TYP008' -> 'W-TYP008';

####                     'W-TYP012' -> 'W-TYP012';

####                     'W-TYP018' -> 'W-TYP018';

####                     'W-TYP021' -> 'W-TYP021';
```

```
####                          else -> 'Others';")

#### fleet11[c("Customer.ID.Bin")] <- recode.variables(fleet11[c("Customer.ID")] ,

####"'C00ID106' -> 'C00ID106';

####'C00ID009' -> 'C00ID009';

####'C00ID039' -> 'C00ID039';

####'C00ID019' -> 'C00ID019';

####'C00ID047' -> 'C00ID047';

####'C00ID084' -> 'C00ID084';

####'C00ID058' -> 'C00ID058';

####'C00ID048' -> 'C00ID048';

####'C00ID053' -> 'C00ID053';

####'C00ID015' -> 'C00ID015';

####'C00ID066' -> 'C00ID066';

####'C00ID049' -> 'C00ID049';

####else -> 'Others';")

######---Plotting done in DEducer-----------###

########### Aggregate command ----------------------------

library(VIM)

library(stats)

rm(counts,customer,customer.waste,driver,driver.waste)

FLtoCLt<-fleet

library(som)

cor(normalize(fleet[,c(10,11,13:15)],byrow=TRUE), use="complete.obs")

#####Sorting command#####

library(plyr)

FLtoCLt <- ddply(FLtoCLt, c("KM","WasteType","QtyTons"), summarise,
```

```
        N    = length(TimetoCust),
        TimetoCust = mean(TimetoCust),
        TimeatCust = mean(TimeatCust),
        TimetoFL = mean(TimetoFL))
####Aggregate command###
library(plyr)
FltoCLtCount<- ddply(FLtoCLt,.(FLtoCLt$KM, FLtoCLt$WasteType), .fun = NULL)
###################################################
##
#we use set.seed to ensure that you get same results
#every time you run the experiment. Seperating the data into test and train data
rows<-seq(1:1127)
set.seed(7)
trainRows=sample(rows,1000)
testRows<-rows[-(trainRows)]
train = fleet[trainRows,]
test=fleet[testRows,]
summary(train)
summary(test)
head(train)
summary(train)
attach(train)
#### Build a linear model using R, to predict times T1,T2 & T3
##### Predicting TimetoCustT1 using lm
mylmT1<-lm(TimetoCust ~
KM+WasteType+QtyTons+VehicleID+Driver.ID+Vehicle.Type+Make+Model,
```

```
          data = train)
summary(mylmT1)
plot(mylmT1)
testlm<-test[-c(67,76,127),]
predlmTimetoCust <- predict(mylmT1,
                          newdata = testlm,
                          type = ("response"))
test<-cbind(testlm,predlmTimetoCust)
### Plotted using excel plot(test$pred~test$x)
library(hydroGOF)
library(arules)
ErrorlmTimetoCust<-mse(sim=test$predlmTimetoCust, obs=test$TimetoCust)
ErrorlmTimetoCust
####Predicting Time@ CustomerT2
mylmT2<-lm(TimeatCust ~ Customer.ID+KM+WasteType+QtyTons+VehicleID,
          data = train)
summary(mylmT2)
plot(mylmT2)
testlm<-test[-c(30,60,21,31,87,123),]
predlmTimeatCust <- predict(mylmT2,
                          newdata = testlm,
                          type = ("response"))
test<-cbind(testlm,predlmTimeatCust)
### Plotted using excel plot(test$pred~test$x)
ErrorlmTimeatCust<-mse(sim=test$predlmTimeatCust, obs=test$TimeatCust)
ErrorlmTimeatCust
```

```
## Predicting timeT3 TimetoFL

mylmT3<-lm(TimetoFL ~
KM+WasteType+QtyTons+VehicleID+Driver.ID+Vehicle.Type+Make+Model,

          data = train)

summary(mylmT3)

plot(mylmT3)

predlmTimetoFL <- predict(mylmT3,

                         newdata = test,

                         type = ("response"))

test<-cbind(test,predlmTimetoFL)

### Plotted using excel plot(test$pred~test$x)

ErrorlmTimetoFL<-mse(sim=test$predlmTimetoFL, obs=test$TimetoFL)

ErrorlmTimetoFL

write.csv(test,"testlm.csv")

### Build model 2 with Naive Bayes for all 3 time types###

library(e1071)

trainnb<-train

trainnb$TimetoCust<-
cut(trainnb$TimetoCust,breaks=c(0,0.25,0.5,0.75,1.0,1.25,1.5,1.75,2.0,2.25,2.5,2.7
5,3.0,3.25,3.5,3.75,4.0,4.25,4.5,4.75,5.0),


label=c(0.25,0.5,0.75,1.0,1.25,1.5,1.75,2.0,2.25,2.5,2.75,3.0,3.25,3.5,3.75,4.0,4.25,
4.5,4.75,5.0))

trainnb$TimeatCust<-
cut(trainnb$TimeatCust,breaks=c(0,0.25,0.5,0.75,1.0,1.25,1.5,1.75,2.0,2.25,2.5,2.7
5,3.0,3.25,3.5,3.75,4.0,4.25,4.5,4.75,5.0),


label=c(0.25,0.5,0.75,1.0,1.25,1.5,1.75,2.0,2.25,2.5,2.75,3.0,3.25,3.5,3.75,4.0,4.25,
4.5,4.75,5.0))
```

```
trainnb$TimetoFL<-
cut(trainnb$TimetoFL,breaks=c(0,0.25,0.5,0.75,1.0,1.25,1.5,1.75,2.0,2.25,2.5,2.75,
3.0,3.25,3.5,3.75,4.0,4.25,4.5,4.75,5.0),

label=c(0.25,0.5,0.75,1.0,1.25,1.5,1.75,2.0,2.25,2.5,2.75,3.0,3.25,3.5,3.75,4.0,4.25,
4.5,4.75,5.0))

modelTimetoCust <- naiveBayes(trainnb$TimetoCust ~ ., data = trainnb, laplace =
3)

modelTimetoCust$apriori

modelTimeatCust <- naiveBayes(trainnb$TimetoCust ~ ., data = trainnb, laplace =
3)

modelTimeatCust$apriori

modelTimetoFL <- naiveBayes(trainnb$TimetoCust ~ ., data = trainnb, laplace = 3)

modelTimetoFL$apriori

testnb<-test[,-(16:18)]

prednBTimetoCust <- predict(modelTimetoCust, testnb)

prednBTimeatCust <- predict(modelTimeatCust, testnb)

prednBTimetoFL <- predict(modelTimetoFL, testnb)


testnb<-cbind(testnb,prednBTimetoCust,prednBTimeatCust,prednBTimetoFL)

testnb$prednBTimetoCust<-as.numeric(as.character(testnb$prednBTimetoCust))

testnb$prednBTimeatCust<-as.numeric(as.character(testnb$prednBTimeatCust))

testnb$prednBTimetoFL<-as.numeric(as.character(testnb$prednBTimetoFL))

ErrornbTimetoCust<-mse(sim=testnb$prednBTimetoCust, obs=test$TimetoCust)

ErrornbTimetoCust

ErrornbTimeatCust<-mse(sim=testnb$prednBTimeatCust, obs=test$TimeatCust)

ErrornbTimeatCust

ErrornbTimetoFL<-mse(sim=testnb$prednBTimetoFL, obs=test$TimetoFL)
```

```
ErrornbTimetoFL

write.csv(testnb,"testnb.csv")

rm(trainnb,testnb)

### Build all three time with Random Forest

library(randomForest)

trainrf<-train[,c(2:6,8:11,13:15)]

trainrf$Customer.ID<-cut(trainrf$KM,breaks=11,labels=c(1,2,3,4,5,6,7,8,9,10,11))

ModelrfTimetoCust <- randomForest(TimetoCust ~
KM+WasteType+QtyTons+VehicleID+Driver.ID+Vehicle.Type+Make+Model,
data=trainrf)

print(ModelrfTimetoCust) # view results

importance(ModelrfTimetoCust) # importance of each predictor

testrf<-test[,c(2:6,8:11,13:15)]

rfTimetoCust<-predict(ModelrfTimetoCust,newdata=testrf,

                    type="class")

ModelrfTimeatCust <- randomForest(TimeatCust ~
KM+WasteType+QtyTons+VehicleID,   data=trainrf)

print(ModelrfTimeatCust) # view results

importance(ModelrfTimeatCust) # importance of each predictor

rfTimeatCust<-predict(ModelrfTimeatCust,newdata=testrf,

                    type="class")


ModelrfTimetoFL <- randomForest(TimetoFL ~
KM+WasteType+QtyTons+VehicleID+Driver.ID+Vehicle.Type+Make+Model,
data=trainrf)

print(ModelrfTimetoFL) # view results

importance(ModelrfTimetoFL) # importance of each predictor

rfTimetoFL<-predict(ModelrfTimetoFL,newdata=testrf,
```

```
                    type="class")

testrf<-cbind(testrf,rfTimetoCust,rfTimeatCust,rfTimetoFL)

ErrorrfTimetoCust<-mse(sim=testrf$rfTimetoCust, obs=testrf$TimetoCust)

ErrorrfTimetoCust

ErrorrfTimeatCust<-mse(sim=testrf$rfTimeatCust, obs=testrf$TimeatCust)

ErrorrfTimeatCust

ErrorrfTimetoFL<-mse(sim=testrf$rfTimetoFL, obs=testrf$TimetoFL)

ErrorrfTimetoFL

write.csv(testrf,"testrf.csv")

rm(trainrf)

### Build model 4 with Decision Trees

library(party)

library(C50)

trainc50<-train[,-c(1,2,7,12)]

trainc50$TimetoCust<cut(trainc50$TimetoCust,breaks=c(0,0.25,0.5,0.75,1.0,1.25,1
.5,1.75,2.0,2.25,2.5,2.75,3.0,3.25,3.5,3.75,4.0,4.25,4.5,4.75,5.0),


label=c(0.25,0.5,0.75,1.0,1.25,1.5,1.75,2.0,2.25,2.5,2.75,3.0,3.25,3.5,3.75,4.0,4.25,
4.5,4.75,5.0))

trainc50$TimeatCust<-
cut(trainc50$TimeatCust,breaks=c(0,0.25,0.5,0.75,1.0,1.25,1.5,1.75,2.0,2.25,2.5,2.
75,3.0,3.25,3.5,3.75,4.0,4.25,4.5,4.75,5.0),


label=c(0.25,0.5,0.75,1.0,1.25,1.5,1.75,2.0,2.25,2.5,2.75,3.0,3.25,3.5,3.75,4.0,4.25,
4.5,4.75,5.0))



trainc50$TimetoFL<-
cut(trainc50$TimetoFL,breaks=c(0,0.25,0.5,0.75,1.0,1.25,1.5,1.75,2.0,2.25,2.5,2.75
,3.0,3.25,3.5,3.75,4.0,4.25,4.5,4.75,5.0),
```

```
label=c(0.25,0.5,0.75,1.0,1.25,1.5,1.75,2.0,2.25,2.5,2.75,3.0,3.25,3.5,3.75,4.0,4.25,
4.5,4.75,5.0))

dtC50TimetoCust= C5.0(trainc50$TimetoCust ~ .,

           data = trainc50[,-c(10:13)],

           rules=TRUE)

summary(dtC50TimetoCust)

C5imp(dtC50TimetoCust, pct=TRUE)

testc50<-test[,c(3:6,8:11,13:15)]

c50TimetoCust=predict(dtC50TimetoCust,

           newdata=testc50[,(1:8)],

           type="class")

dtC50TimeatCust= C5.0(trainc50$TimeatCust ~ .,

           data = trainc50[,-c(9,11,12,13)],

           rules=TRUE)

summary(dtC50TimeatCust)

C5imp(dtC50TimeatCust, pct=TRUE)

c50TimeatCust=predict(dtC50TimeatCust,

           newdata=testc50[,(1:8)],

           type="class")

dtC50TimetoFL= C5.0(trainc50$TimetoFL ~ .,

          data = trainc50[,-c(9:10,12,13)],

          rules=TRUE)

summary(dtC50TimetoFL)

C5imp(dtC50TimetoFL, pct=TRUE)

c50TimetoFL=predict(dtC50TimetoFL,
```

```
            newdata=testc50[,(1:8)],

            type="class")

testc50<-cbind(testc50,c50TimetoCust,c50TimeatCust,c50TimetoFL)

testc50$c50TimetoCust<-as.numeric(as.character(testc50$c50TimetoCust))

testc50$c50TimeatCust<-as.numeric(as.character(testc50$c50TimeatCust))

testc50$c50TimetoFL<-as.numeric(as.character(testc50$c50TimetoFL))

Errorc50TimetoCust<-mse(sim=testc50$c50TimetoCust, obs=testc50$TimetoCust)

Errorc50TimetoCust

Errorc50TimeatCust<-mse(sim=testc50$c50TimeatCust, obs=test$TimeatCust)

Errorc50TimeatCust

Errorc50TimetoFL<-mse(sim=testc50$c50TimetoFL, obs=test$TimetoFL)

Errorc50TimetoFL

write.csv(testc50,"testc50.csv")

rm(trainc50,testc50)

### Creating Error Matrix for all models

Error<-c (Errorc50TimetoCust, Errorc50TimeatCust, Errorc50TimetoFL,
ErrorlmTimetoCust, ErrorlmTimeatCust, ErrorlmTimetoFL, ErrornbTimetoCust,
ErrornbTimeatCust, ErrornbTimetoFL,        ErrorrfTimetoCust, ErrorrfTimeatCust,
ErrorrfTimetoFL)

Error<-matrix(unlist(Error),nrow=3,ncol=4,byrow=FALSE)

colnames(Error) <- c("C50", "lm", "nb","rf")

Time<-c("TimetoCust","TimeatCust","TimetoFL")

Error<-cbind(Time,Error)

write.csv(Error,"Error.csv")

###############Removing unused object

rm(Errorc50TimetoCust, Errorc50TimeatCust, Errorc50TimetoFL, ErrorlmTimetoCust,
ErrorlmTimeatCust, ErrorlmTimetoFL, ErrornbTimetoCust, ErrornbTimeatCust,
ErrornbTimetoFL,    ErrorrfTimetoCust, ErrorrfTimeatCust, ErrorrfTimetoFL)
```

```
rm(Time)

rm(dtC50TimeatCust, dtC50TimetoCust, dtC50TimetoFL)

rm(modelTimetoCust,modelTimeatCust,modelTimetoFL)

###rm(mylmT1,mylmT2,mylmT3)

rm(rfTimetoCust,rfTimeatCust,rfTimetoFL)

rm(c50TimetoCust,c50TimeatCust,c50TimetoFL)

###rm(predlmTimetoCust,predlmTimeatCust,predlmTimetoFL)

rm(prednBTimetoCust,prednBTimeatCust,prednBTimetoFL)

rm(train,testlm,testrf)

#### Plotting T1+T2+T3###

write.csv(test,"plotdata.csv")

plotdata<-read.table("plotdata.csv", header = TRUE, sep = ",")

plotdata<-plotdata[,-c(1,2,4:13,17:18)]

plotdata<-plotdata[,c(1,2,5,3,6,4,7)]

colnames(plotdata)<-c("Customer.ID","TtoCustAct","TtoCustPred","TatCustAct",

                    "TatCustPred","TtoFLAct","TtoFLPred")

plotdata<-aggregate(x=plotdata,list(plotdata$Customer.ID), mean)

plotdata<-plotdata[,-2]

write.csv(plotdata,"plotdatafinal.csv")

plot1<-plotdata[,1:3]

plot1<-reshape(plot1,idvar="Group.1",varying=(2:3),

             v.names = "TimetoCust",direction="long")

plot1$time<-as.factor(plot1$time)

library(ggplot2)

plot2<-plotdata[,c(1,4,5)]

plot2<-reshape(plot2,idvar="Group.1",varying=(2:3),
```

```
          v.names = "TimeatCust",direction="long")

plot2$time<-as.factor(plot2$time)

plot3<-plotdata[,c(1,6,7)]

plot3<-reshape(plot3,idvar="Group.1",varying=(2:3),

          v.names = "TimetoFL",direction="long")

plot3$time<-as.factor(plot3$time)

library(ggplot2)

qplot(factor(Group.1),data=plot1,geom="bar", fill=time , weight=TimetoCust,position="dodge",

    main = "Customer Prediction", xlab="TimetoCust",ylab="Time")

qplot(factor(Group.1),data=plot2,geom="bar", fill=time , weight=TimeatCust,position="dodge",

    main = "Customer Prediction", xlab="TimeatCust",ylab="Time")

qplot(factor(Group.1),data=plot3,geom="bar", fill=time , weight=TimetoFL,position="dodge",

    main = "Customer Prediction", xlab="TimetoFLCust",ylab="Time")

library(reshape2)

plotdata1tmp<-plotdata[,c(1,2,4,6)]

colnames(plotdata1tmp)<-paste(c("Customer","TtoCust","TatCust","TtoFL"))

plotdata1tmp <- melt(plotdata1tmp, id.var="Customer")

colnames(plotdata1tmp)<-paste(c("Customer","variable","Actual Time"))

plotdata2tmp<-plotdata[,c(1,3,5,7)]

colnames(plotdata2tmp)<-paste(c("Customer","TtoCust","TatCust","TtoFL"))

plotdata2tmp <- melt(plotdata2tmp, id.var="Customer")

colnames(plotdata2tmp)<-paste(c("Customer","variable","Predicted Time"))


plotdatafinal<-cbind(plotdata1tmp,plotdata2tmp)
```

```
plotdatafinal<-plotdatafinal[,c(1:3,6)]

plotdatafinal <- melt(plotdatafinal, id.var=c("Customer","variable"))

plotdatafinal<-plotdatafinal[,c(1,3,2,4)]

colnames(plotdatafinal)<-paste(c("Customer","Time","variable","value"))

write.csv(plotdatafinal,"plotdatafinalpres.csv")

ggplot(plotdatafinal, aes(x = Customer, y = value, fill = variable)) +

  geom_bar(position = "stack", stat = "identity") +

  facet_wrap( ~ Time)

rm(plot1,plot2,plot3,plotdata,plotdata1tmp,plotdata2tmp,plotdatafinal)

rm(FLtoCLt,FltoCLtCount)

#### Building Longitudinal Model for Predicting time #####

fleetdata<-read.table("preprocessfleet.csv", header = TRUE, sep = ",")

fleetdata<-fleetdata[,c(3:13,18:20)]

rfT1<-predict(ModelrfTimetoCust,newdata=fleetdata[,c(2:5,7:10,12:14)],

              type="class")

rfT2<-predict(ModelrfTimeatCust,newdata=fleetdata[,c(2:5,7:10,12:14)],

              type="class")

rfT3<-predict(ModelrfTimetoFL,newdata=fleetdata[,c(2:5,7:10,12:14)],

              type="class")

fleetdata<-cbind(fleetdata,rfT1,rfT2,rfT3)

fleetdata<-fleetdata[with(fleetdata, order(fleetdata$Customer.ID)), ]

fleetdata<-fleetdata[with(fleetdata,order(fleetdata$Date)),]

### Build LM starting at this point


### Predicting Time to Customer from RF

fleetdata$Avg_TimetoCust<-0
```

```
fleetdata$StdDev_TimetoCust<-0

fleetdata$Pred_Time<-0

fleetdata$Error<-0

### 1. First LM for time to Cust

head(fleetdata)

for(i in 1:nrow(fleetdata))

{

  if(i==1){

    fleetdata$Avg_TimetoCust[i]<-NA

    fleetdata$StdDev_TimetoCust[i]<-NA

    fleetdata$Pred_Time[i]<-fleetdata$rfT1[i]

    fleetdata$Error[i]<-abs(fleetdata$Pred_Time[i]-fleetdata$TimetoCust[i])

  }

  if(i==2)

  {

    fleetdata$Avg_TimetoCust[i]<-fleetdata$TimetoCust[i-1]

    fleetdata$StdDev_TimetoCust[i]<-NA

    fleetdata$Pred_Time[i]<-fleetdata$rfT1[i]*0.25+fleetdata$TimetoCust[i-1]*0.75

    fleetdata$Error[i]<-(abs(fleetdata$Pred_Time[i]-
fleetdata$TimetoCust[i])*100/fleetdata$TimetoCust[i])

  }

  if(i==3)

  {

    fleetdata$Avg_TimetoCust[i]<-mean(fleetdata$TimetoCust[1:i-1])

    fleetdata$StdDev_TimetoCust[i]<-sd(fleetdata$TimetoCust[1:i-1])
```

```
    fleetdata$Pred_Time[i]<-fleetdata$rfT1[i]*0.25+(mean(fleetdata$TimetoCust[1:i-
1])*0.75)

    fleetdata$Error[i]<-(abs(fleetdata$Pred_Time[i]-
fleetdata$TimetoCust[i])*100/fleetdata$TimetoCust[i])

  }

  if(i>3)

  {

    fleetdata$Avg_TimetoCust[i]<-mean(fleetdata$TimetoCust[1:i-1])

    fleetdata$StdDev_TimetoCust[i]<-sd(fleetdata$TimetoCust[1:i-1])

    lm<-lm(TimetoCust~ rfT1 + Avg_TimetoCust + StdDev_TimetoCust
,data=fleetdata[1:i-1,])

    fleetdata$Pred_Time[i]<-predict(lm,fleetdata[i,c(15,18:19)])

    fleetdata$Error[i]<-(abs(fleetdata$Pred_Time[i]-
fleetdata$TimetoCust[i])*100/fleetdata$TimetoCust[i])

  }

}

LMErrorT1<-mse(sim=fleetdata$Pred_Time, obs=fleetdata$TimetoCust)

LMErrorT1

fleetdata$Pred_TimetoCust<-fleetdata$Pred_Time


### Predicting time T2 TimeatCust

fleetdata$Avg_TimeatCust<-fleetdata$Avg_TimetoCust

fleetdata$StdDev_TimeatCust<-fleetdata$StdDev_TimetoCust

fleetdata$Pred_Time<-0

fleetdata$Error<-0


fleetdata<-fleetdata[,-c(18,19)]
```

```
fleetdata<-fleetdata[,c(1:17,20,21,22,18,19)]

head(fleetdata)

for(i in 1:nrow(fleetdata))

{

  if(i==1){

    fleetdata$Avg_TimeatCust[i]<-NA

    fleetdata$StdDev_TimeatCust[i]<-NA

    fleetdata$Pred_Time[i]<-fleetdata$rfT2[i]

    fleetdata$Error[i]<-abs(fleetdata$Pred_Time[i]-fleetdata$TimeatCust[i])

  }

  if(i==2)

  {

    fleetdata$Avg_TimeatCust[i]<-fleetdata$TimeatCust[i-1]

    fleetdata$StdDev_TimeatCust[i]<-NA

    fleetdata$Pred_Time[i]<-fleetdata$rfT2[i]*0.25+fleetdata$TimeatCust[i-1]*0.75

    fleetdata$Error[i]<-(abs(fleetdata$Pred_Time[i]-
fleetdata$TimeatCust[i])*100/fleetdata$TimeatCust[i])

  }

  if(i==3)

  {

    fleetdata$Avg_TimeatCust[i]<-mean(fleetdata$TimeatCust[1:i-1])

    fleetdata$StdDev_TimeatCust[i]<-sd(fleetdata$TimeatCust[1:i-1])

    fleetdata$Pred_Time[i]<-fleetdata$rfT2[i]*0.25+(mean(fleetdata$TimeatCust[1:i-
1])*0.75)

    fleetdata$Error[i]<-(abs(fleetdata$Pred_Time[i]-
fleetdata$TimeatCust[i])*100/fleetdata$TimeatCust[i])

  }
```

```
  if(i>3)
  {
    fleetdata$Avg_TimeatCust[i]<-mean(fleetdata$TimeatCust[1:i-1])
    fleetdata$StdDev_TimeatCust[i]<-sd(fleetdata$TimeatCust[1:i-1])
    lm<-lm(TimeatCust~ rfT2 + Avg_TimeatCust + StdDev_TimeatCust
,data=fleetdata[1:i-1,])
    fleetdata$Pred_Time[i]<-predict(lm,fleetdata[i,c(16,19:20)])
    fleetdata$Error[i]<-(abs(fleetdata$Pred_Time[i]-
fleetdata$TimeatCust[i])*100/fleetdata$TimeatCust[i])
  }
}
LMErrorT2<-mse(sim=fleetdata$Pred_Time, obs=fleetdata$TimeatCust)
LMErrorT2
fleetdata$Pred_TimeatCust<-fleetdata$Pred_Time
fleetdata<-fleetdata[,c(1:18,23,19:22)]
fleetdata$Avg_TimetoFL<-fleetdata$Avg_TimeatCust
fleetdata$StdDev_TimetoFL<-fleetdata$StdDev_TimeatCust
fleetdata$Pred_Time<-0
fleetdata$Error<-0
fleetdata<-fleetdata[,c(1:19,22:25)]
fleetdata<-fleetdata[,c(1:19,22,23,20,21)]
head(fleetdata)
for(i in 1:nrow(fleetdata))
{
  if(i==1){
    fleetdata$Avg_TimetoFL[i]<-NA
```

```
    fleetdata$StdDev_TimetoFL[i]<-NA

    fleetdata$Pred_Time[i]<-fleetdata$rfT3[i]

    fleetdata$Error[i]<-abs(fleetdata$Pred_Time[i]-fleetdata$TimetoFL[i])

  }

  if(i==2)

  {

    fleetdata$Avg_TimetoFL[i]<-fleetdata$TimetoFL[i-1]

    fleetdata$StdDev_TimetoFL[i]<-NA

    fleetdata$Pred_Time[i]<-fleetdata$rfT3[i]*0.25+fleetdata$TimetoFL[i-1]*0.75

    fleetdata$Error[i]<-(abs(fleetdata$Pred_Time[i]-
fleetdata$TimetoFL[i])*100/fleetdata$TimetoFL[i])

  }

  if(i==3)

  {

    fleetdata$Avg_TimetoFL[i]<-mean(fleetdata$TimetoFL[1:i-1])

    fleetdata$StdDev_TimetoFL[i]<-sd(fleetdata$TimetoFL[1:i-1])

    fleetdata$Pred_Time[i]<-fleetdata$rfT3[i]*0.25+(mean(fleetdata$TimetoFL[1:i-
1])*0.75)

    fleetdata$Error[i]<-(abs(fleetdata$Pred_Time[i]-
fleetdata$TimetoFL[i])*100/fleetdata$TimetoFL[i])

  }

  if(i>3)

  {

    fleetdata$Avg_TimetoFL[i]<-mean(fleetdata$TimetoFL[1:i-1])

    fleetdata$StdDev_TimetoFL[i]<-sd(fleetdata$TimetoFL[1:i-1])

    lm<-lm(TimetoFL ~ rfT3 + Avg_TimetoFL + StdDev_TimetoFL ,data=fleetdata[1:i-
1,])
```

```
    fleetdata$Pred_Time[i]<-predict(lm,fleetdata[i,c(17,20:21)])

    fleetdata$Error[i]<-(abs(fleetdata$Pred_Time[i]-
fleetdata$TimetoFL[i])*100/fleetdata$TimetoFL[i])

  }

}

LMErrorT3<-mse(sim=fleetdata$Pred_Time, obs=fleetdata$TimetoFL)

LMErrorT3

fleetdata$Pred_TimetoFL<-fleetdata$Pred_Time

fleetdata<-fleetdata[,-c(20:23)]

fleetdata<-fleetdata[,-c(15:17)]

write.csv(fleetdata,"fleetdatalm.csv")

Longitudinal.Model<-c(LMErrorT1,LMErrorT2,LMErrorT3)

Error<-cbind(Error,Longitudinal.Model)

write.csv(Error,"Error.csv")
```