*Where’s my shipment?*

Predicting Arrival Time of International Container Ships

03/27/2017

1. Introduction:

One of the current challenges in supply chain management is how to deliver a product correctly: the right quantity delivered to the right place at the right time, products where and when customers wanted them to be. In addition to this is the rising cost of operations due to rising fuel and energy costs, increasing labor rates, and also rising commodity prices [[1]](http://blog.rbwlogistics.com/the-5-biggest-supply-chain-challenges/).

In order to stay as efficient, effective and competitive in the market, shipping companies must develop strategies that will implement business improvements that give both short and long term results. Alongside innovations in technologies such as cloud computing, internet of things (IOT) and Big Data, shipping companies are now collecting more data than ever. Now, they are facing a challenge on how to make use of this data, how and where to store it, and how to protect it (data security issues).

One aspect of utilizing this large amount of data is to make it visible not only to transportation providers or government agencies but also to shippers and customers. These stakeholders want so see real time data updates on what is going on in their shipment network--for both inbound and outbound movement. Data also becomes an important tool in optimization of inventory and reduction of costs by building a network that minimizes costs and reduces transit time. This will pave the way to creating a balanced and efficient supply chain [[2](http://www.re-transfreight.com/blog/top-supply-chain-issues-trends-for-2017)].

2. Data:

The data used herein is mainly from ships--or any vessel--that sends Automatic Identification System (AIS) [[4]](https://en.wikipedia.org/wiki/Automatic_identification_system) data thru Global Positioning System GPS and/or SATELLITE feed. This data is used to avoid collision among ships and is mandatory for cargo ships of 300 Gross Ton or more and also for all every passenger ships regardless of size. AIS data includes vessel global id, name, destination, position (latitude and longitude), heading and speed and being sent every 10 to 15 minutes. AIS data also supplements marine radar which is still the primary method of avoiding collisions for water transportations. Vessels fitted with AIS transmitters can be tracked by AIS base station in the coastlines and when the vessel it out of range, the data is then a number of satellites fitted with AIS receivers will then continue to receive transmissions from the vessel.

My company receives around 90% of all AIS data sent by all shipping lines internationally which includes from other carriers as well. This will be the sole source of the data that will be used in this project.

3. Data definition:

Below is the description of the data and a sample value.

**COLUMN NAME Sample Value**

AIS\_RAW\_DATA\_KEY NOT NULL VS20140303000000000012000002

DATA\_SOURCE SAT, VT

AISO\_SEQ 1..n

IMO\_MMSI 9638977-#

IMO 9638977

MMSI 563361000

VESSEL\_NAME MAERSK LABREA

CALL\_SIGN 9HA3446

WIDTH 231

LENGTH 69

DIM\_C 25

DIM\_D 15

STATUS NULL

VESSEL\_TYPE cargo\_ships,

TIME\_SEEN\_UTC 03-MAR-17 09.46.39.000000000 AM

LAT 12.665241666666667

LON -9.388526666666667

DESTINATION LOS ANGELES

ETA 11-MAR-17 07.00.00.000000000 AM

COURSE 108.2

HEADING 149

SPEED 23.6

DRAUGHT 7.6

NAV\_STATE 5

RATE\_OF\_TURN NULL

REC\_UPD\_DT 23-MAY-17 02.56.15.290474000 AM

VESSEL\_GID V000004102

DATA\_PROVIDER VT

BATCH\_NUM NULL

EXTERNAL\_USE T

From the sample data below (Figure 1), you may notice that the speed of this vessel slows down as it nears the port. It then reported a full stop (speed = 0) at the destination port, QINGDAO on 14-OCT-14 at 09.18.15AM (arrival time) and ready to unload its cargoes. This will be marked as the origin port QINGDAO for the next voyage.

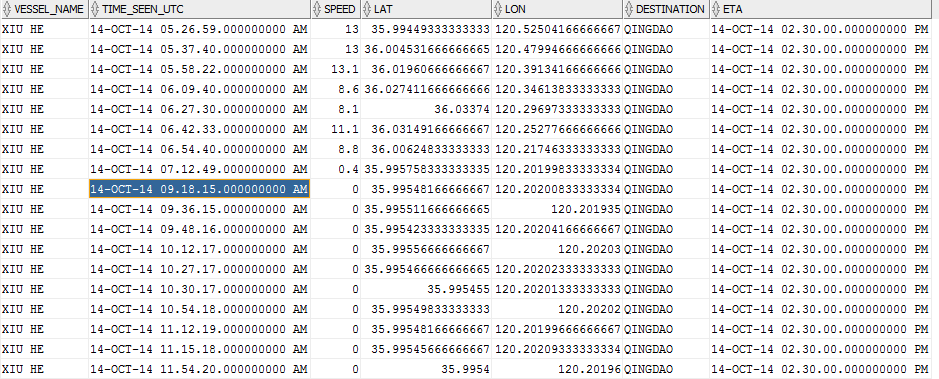


Figure 1: Vessel arrived at port QINGDAO on 14-OCT-2014 at 09.18.15AM and early by +5hrs (14-OCT-2014 02.30.00 PM) based on Estimated Time of Arrival (ETA)

The vessel then stayed at this port for around 12 hours (14-OCT-2014 at 09.21.43 PM) to unload/load containers until it leaves for the next destination to port NINGBO on 14-OCT-2014 09.43.34PM. This departure is indicated by a speed of > 0.1 (or speed = 3.8), a new destination port NINGBO and a new ETA datetime 16-OCT-2014 02.00.00PM.

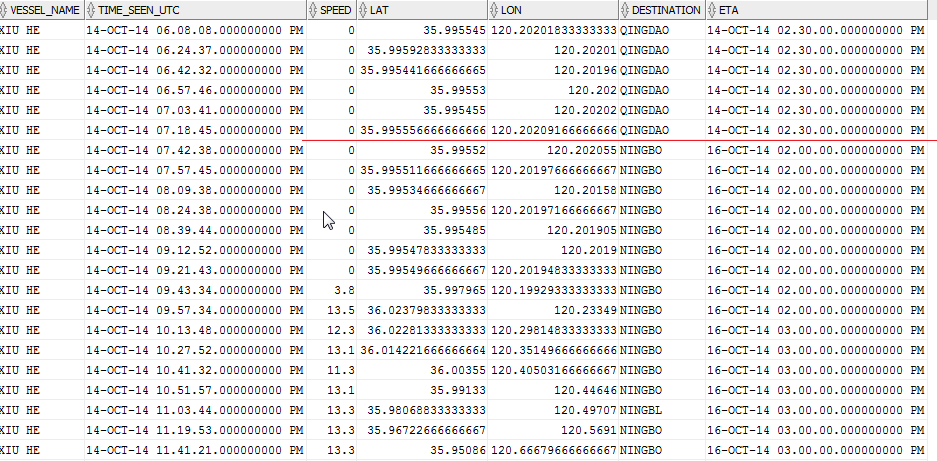


Figure 2: Vessel stayed at port QINGDAO from 14-OCT-2014 09.18.15AM until it departs for destination NINGBO on 14-OCT-2014 09.43.34PM

Also notice that when vessel changes the destination to port NINGBO and after few minutes, it also reported a change on the ETA from 3PM to 16-OCT-2014 03.00.00PM. This change will be ignored since we will record the ETA at the time the vessel departed. Thus, the estimated travel time of from QINGDAO to NINGBO is 1day 16hrs and 16 minutes.

However, the actual arrival datetime based on numerous reported speed=0 at port NINGBO is 16-OCT-2014 08.56.48AM which is early by 5.5 hours compared to the estimated time of arrival (ETA) reported by vessel thru AIS data.

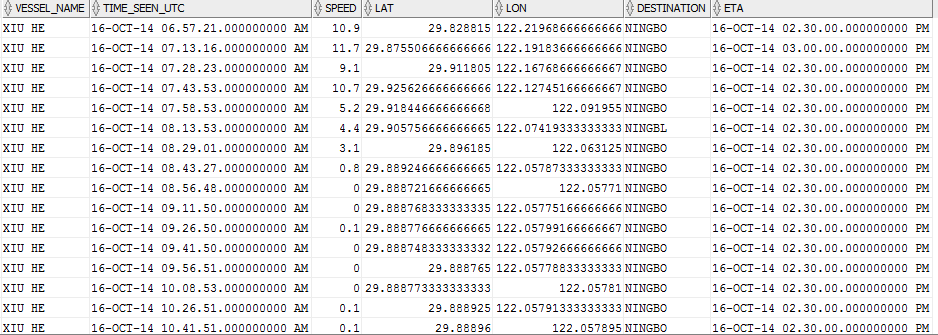


Figure 3: Vessel arrives at port NINGBO on 16-OCT-2014 08.56.48AM and around +5.5 hours ahead of ETA

The difference between the arrival datetime and the departure datetime will be called the actual travel time and will be compared to the estimated travel time based on the ETA. The difference will be analyzed for accuracy in terms of mean absolute percentage error (MAPE) and mean square error (MSE). While the predicted travel time will be added to the actual departure datetime to derive the predicted ETA. The accuracy of the predicted ETA will also be analyzed using MSE and MAPE.

Sample calculations based on the sample data:

Origin: QINGDAO departs: 14-OCT-2014 09.43.34PM

Destination: NINGBO arrives: 16-OCT-2014 08.56.48AM

ETA: 16-OCT-2014 02.00.00 PM

Estimated travel time: 1 day 16 hrs 16 mins or 40.27 hours

Actual travel time: 1 day 11 hrs 13 mins or 35.22 hours

Absolute Percentage Error: 14.33%

Error: -5.05 hours

In summary, the data on speed = 0 (or <= 0.1 for some cases), position (latitude/longitude coordinates) and destination (which can be port code, port name, international port code, etc.) are important columns that will derive the origin port, departure datetime, destination port, arrival datetime and estimated time of arrival (ETA). These derived columns are used to calculate the accuracy of the ETA and the predicted ETA. Moreover, these data columns are also sources of noise data and will be handled in the data wrangling in details.

4. Data wrangling

4.1 Destination

The data on destination is a free text and does not follow a specific rule. The values that vessels may send include port code, port name, United Nations (UN) location codes, country port to destination country and abbreviations.

**Sample data and their meaning:**

Port code: SIN is Singapore or SIN-EBGA is Singapore eastbound

Port name: SHANGHAI

UNLOCCODE: SG SIN is SG country code plus SIN port code for Singapore

Country port to destination country: JP YKK SN is country code JP=Japan plus YKK=Yokohama plus SN is country code for Senegal. Another sample: HKHKG > CNYTN is Hong Kong, Hong Kong and bound to China Yantian.

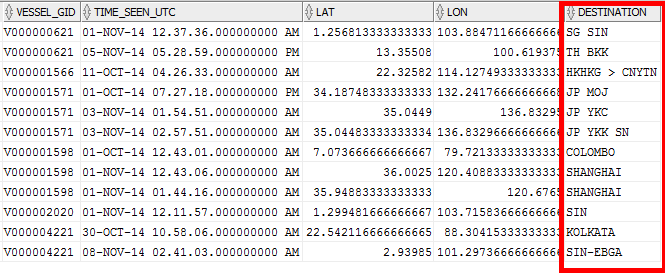


Figure 4. Sample mixed data format for destination column

4.2 Noise data on destination

Since the data on destination is a free text, it is a cause of many noise data. When the destination is not properly cleaned, the study will give a different result since the whole dataset for a given voyage will be incomplete and the number of data samples will be low. Below are some reasons on why the data on destination needs a careful cleaning.

1. There are spaces in between words like ZHANG JIA GANG should be ZHANGJIAGANG
2. Multiple spaces in between words like HONG KONG is same as HONG KONG
3. Misspelled port name like PYONGTAEK and should be PY**E**ONGTAEK
4. Abbreviations like SPORE (Singapore), HK is Hong Kong and J.ALI is Jebel Ali

Sample data below shows some values that need data cleaning.

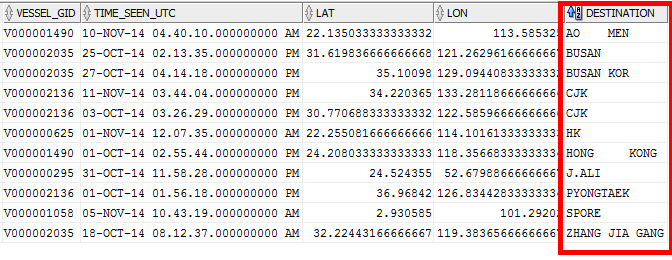
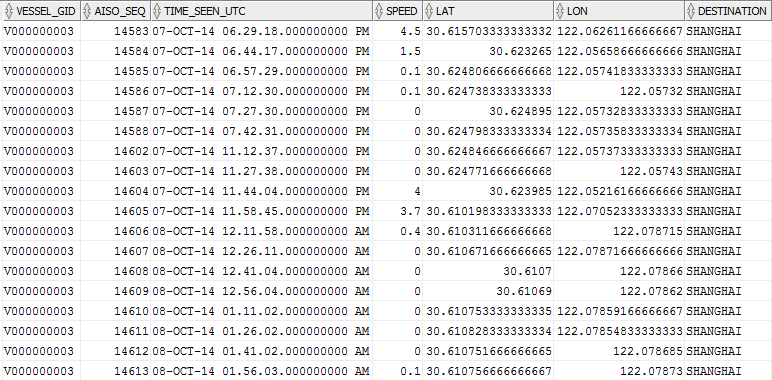


Figure 5. Sample noise data on destination

4.2 Vessel stopping at the middle of voyage

When the vessel reported a stop (speed = 0 or .1), it doesn’t mean that it arrives at the destination. During port congestion, the vessels are stopping and waiting near the port for further instructions when to enter the terminal. As seen in the sample data below, the vessel slows down until it stops on 07-OCT-2014 07.27.30 PM then continued to sail after 4 hours on 07-OCT-2014 11.44.04 PM. Also, it is also not uncommon for a vessel to stop more than once during the voyage especially going to busy ports like SHANGHAI (CHINA), NINGBO (CHINA), BUSAN (KOREA), JEBEL ALI (UAE), ROTTERDAM (NETHERLANDS), HONG KONG (CHINA) and SINGAPORE.

Figure 6. Vessel stopping in the middle of the voyage due to port congestion

4.3 Updated ETA after departing at origin port

When vessel is behind or ahead of schedule for some other reason, it will report another ETA and will be recorded via AIS. However, this change will not be considered in this study since the predicted ETA will be done at the start of the voyage and will not be changed even if the voyage is behind or ahead of schedule.

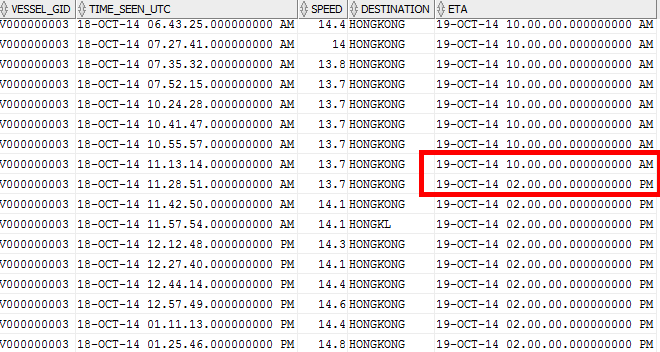


Figure 7. A change on ETA reported after start of voyage

Thus, the ETA 19-OCT-2014 10.00.00 AM at the start of the voyage will be used as the estimated travel time rather than 19-OCT-2014 02.00.00 PM.

4.4 SQL

The succeeding sections will discuss the sql codes that will 1) extract the dataset for voyages that has destination to NINGBO or SHANGHAI; 2) ensure that there is no skipped transmitted data; 3) ensure that the voyage is complete and the ship actually arrived at SHANGHAI.

Step 1: Create a working table

Create a working table which contains the columns needed in this study and then also transform timestamp date format of ETA and TIME\_SEEN\_UTC into date format for easier date calculations later.

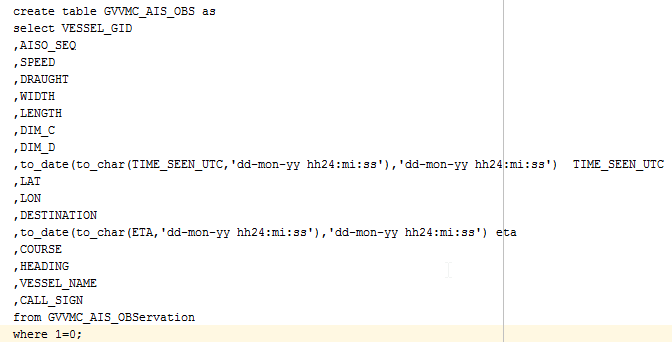


Figure 8. SQL to create a working table GVVMC\_AIS\_OBS

Then import the data from an integration environment to a development environment via sql loader utility program that loads a total of 105M records. This will help in running the queries on a local database rather than connecting remotely.

Step 2: Remove duplicate records and create an index

This step will remove records that are duplicate and create an index for efficiency in running the queries.

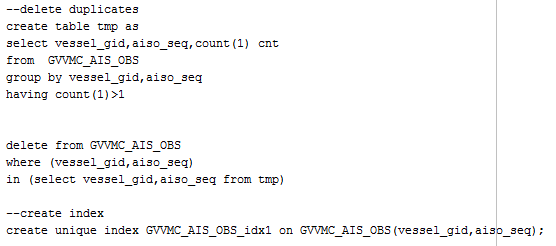


Figure 9. Remove duplicate records and create index

Step 3: Extract voyages that start in SHANGHAI and destination to NINGBO and combine with voyages that start in NINGBO and going to SHANGHAI. The sequence number (AISO\_SEQ) should be consecutive and destination is either port name or country code plus port code (CN SHA or CN NBO).

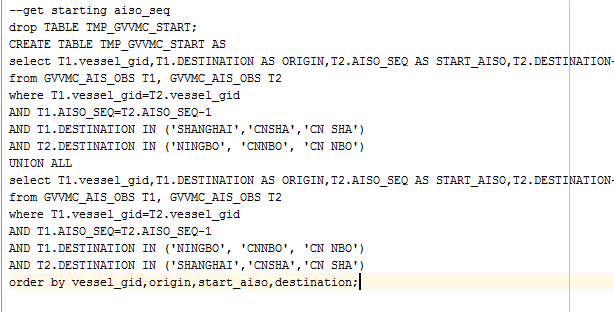


Figure 10. Extract voyages that start in SHANGHAI and going to NINGBO and vice-versa

Step 4: Ensure that the voyage data is complete where AIS data transmitted should be 50 records or more.

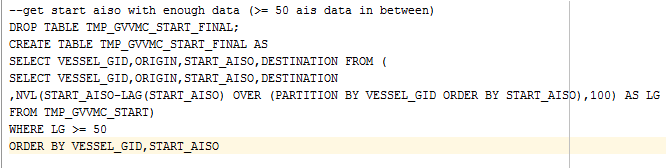


Figure 11. Ensure that data transmitted is complete, that is, 50 records or more

Step 5: Extract the ending aiso sequence number or the record at the end of the voyage.

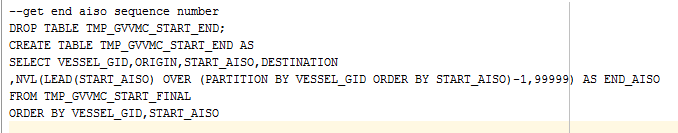


Figure 12. Get the ending sequence number of the voyage

Step 6: Some transmitted ais data on the ending voyage has a difference destination. If the destination is changed when the vessel is approaching the destination port, then this voyage is incomplete. Thus, these voyages will be filtered and will not be included in the dataset.

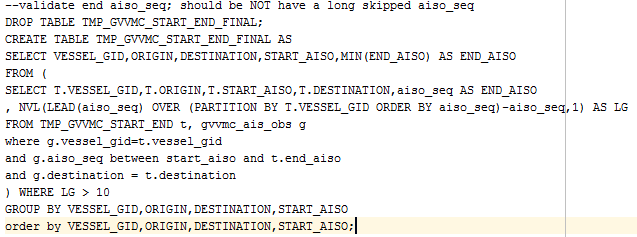


Figure 13. Validate the ending aiso sequence number and ensure that there is no skipped sequence number

Step 7: To ensure that the vessel arrives at the destination port, check the latitude and longitude and should be within 0.5 from the latitude and longitude of SHANGHAI and NINGBO ports.

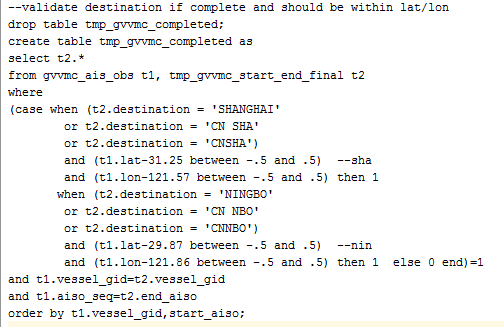


Figure 14. Check that the vessel arrives at SHANGHAI and NINGBO ports within 0.5 from specified latitude and longitude

Step 8. When a vessel arrives at the port, it will continue to send data of its position but speed = 0. We will remove these data since we will do the prediction only when the vessel is moving and not stationed at the port.

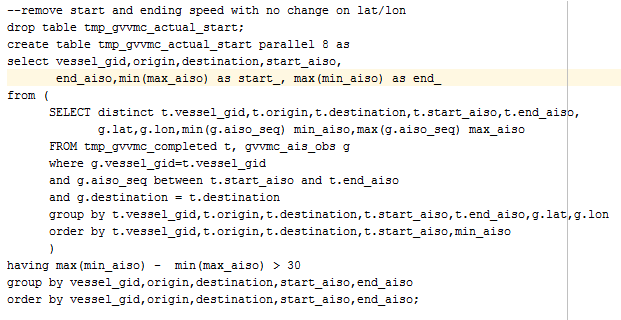


Figure 15. Clean the data with no change on the position of the vessel upon arrival at the port

Step 9. Create the table with the cleaned data

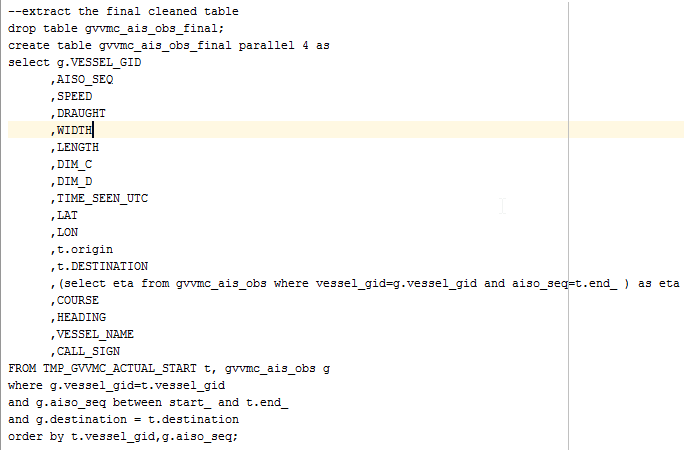


Figure 16. Create the final and cleaned dataset table

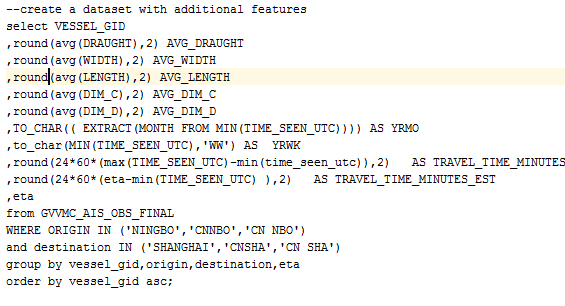
Step 10. Create a dataset with additional features like month and week number of the start date of the voyage and the target variable travel time in minutes

Figure 17. Create a dataset for this study

Below are the features that will be considered in this study:

AVG\_DRAUGHT – this the vertical distance between the waterline and the bottom of the vessel. The larger the value of draught, the heavier the cargo onboard the vessel

AVG\_WIDTH – this is the measure of the width of the vessel

AVG\_LENGTH – this is the length of the vessel

AVG\_DIM\_C – this is the horizontal distance of the AIS transmitter from the edge of the vessel

AVG\_DIM\_D – this is the vertical distance of the AIS transmitter to the front of the vessel

MONTH – the numerical value of the month (1 – 12) when the vessel starts its voyage to SHANGHAI

WEEK\_NUMBER – the week number of the date at the start of the voyage

TRAVEL\_TIME\_MINUTES\_EST – the estimated travel time from AIS data

TRAVEL\_TIME\_MINUTES – recorded travel time of the vessel in minutes. This is the target variable of this study.

5. Statement of the Problem

Estimating or predicting vessel ETA is only a portion of the entire containerization shipment cycle from booking at origin city to return of empty container at destination city. A booking is an agreement between a carrier and a customer to reserve a space for a cargo on an intended vessel to transport a container from origin city to a destination city. Important information on the booking includes shipper’s (sender of cargo) contact info, consignee (receiver of cargo), cargo description (what is inside the container) and weight (for state regulations on road weight limit). Upon creation of the booking, the ETA at the destination will be estimated based on the travel time for outbound, sea/ocean and inbound. The outbound travel time is from the time customer picks up the empty container until it is loaded onboard a vessel. Sea or ocean travel time is when the vessel departs from the origin port until it arrives at the port of unloading. Inbound travel time is from the time the container leaves the port and arrives at the customer location or warehouse and the empty container is returned to shipping company.

**Parts of containerization shipment cycle:**

OUTBOUND: Pick-up empty container -> return the full container and load onto a ship

*SEA/OCEAN TRAVEL TIME: Vessel departs from origin port -> ship arrives at destination port*

INBOUND: Unload container to truck/rail then leaves the destination port -> arrives at customer site and returns empty container



Figure 8. Containerization Shipment Cycle from outbound, sea time and inbound

This project focuses on the prediction of sea or ocean travel time when the vessel departs the origin port and arrives at the destination port. Thus, this study will be a part of an end to end prediction that integrates the entire predictions in the supply chain demand wherein the customer confirms the booking, picks up an empty container, carrier delivers the container to end customer site until customer returns the empty container.

Specifically, this study aims to

1. Improve the estimation of arrival time of vessels from NINGBO, China to SHANGHAI, China using machine learning. This choice of origin and destination port is because this port pair is most common and both ports are two of the busiest in the world.
2. Investigate patterns of vessel routes at sea for voyages when there is a delay versus voyages that have no delays.
3. Determine on how to measure seasonality (date when the booking is made), vessel size (length, width) and number of cargo onboard the vessel as features in estimating vessel arrival time.

6. Approach taken

Data is taken from development database of my company which were collected from 2014 and 2015. After the data is analyzed and cleaned via SQL codes, the data exploration is done in iPython notebook using python scripts (see data explorations below). Then the data is transformed to get more features like ETA month and week number and number of stops that are then fed to the algorithm. The algorithm is written and run in R language. Once the model is created, evaluated the tuned, the model is then reconstructed into python code and saved into a file for user interface (UI).

**Calculation of predicted travel time of vessel arrival at destination port**

Once the booking is created with the customer, the intended vessel will be included in the booking confirmation document. The predicted travel time will be added to the planned vessel departure and the predicted arrival time will be displayed.

**Current estimate:**

Planned vessel departure date + (estimated travel time) = Estimated vessel arrival date

**Predicted:**

Planned vessel departure date + (predicted travel time) = Predicted vessel arrival date

**Machine Learning Techniques**

The machine learning techniques used in this study are Gradient Boosting machine (GBM) and regression using xgboost (linear) and random forest models written in R. However, random forest and GBM models obtain the best mean absolute percentage error (MAPE) so the codes for both models are kept and included on this report.

RandomForest

<https://github.com/anonyXmous/CapstoneProject/blob/master/Using_RF.R>

Regression using gradient boosted machine (GBM) <https://github.com/anonyXmous/CapstoneProject/blob/master/Using_GBM.R>

Below are some of the methods to explore the data and study feature engineering that can be taken from the dataset.

**Google maps API**

I used a python application program interface (API) to google maps [[5]](http://nbviewer.jupyter.org/github/anonyXmous/CapstoneProject/blob/master/Capstone-Maps.html) to plot the latitude / longitude and explore the path where the vessel travels from NINGBO and arrives at SHANGHAI port. This is helpful in analyzing the pattern of the path when the vessels are having delays and comparing when a vessel is not having delays.



Figure 18. Codes to draw the scatterplot for vessel path via google maps

**Correlation Heatmap**

A basic correlation matrix and heatmap is also used to check the correlation relationship between the features. If a correlation is +/- 0.80 or more then these two variables are highly correlated and one variable will be enough to use in the model.

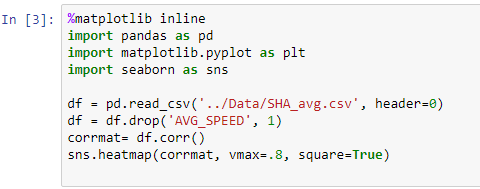


Figure 19. Codes for correlation heatmap using seaborn

**User Interface (UI) using Flask**

Through a user interface (UI) written in Flask, a user can explore features such as draught, length and width of the vessel and dimension affect the prediction result. The UI runs in Flask and the model is re-written from R into python. Sample screen shots are shown below.

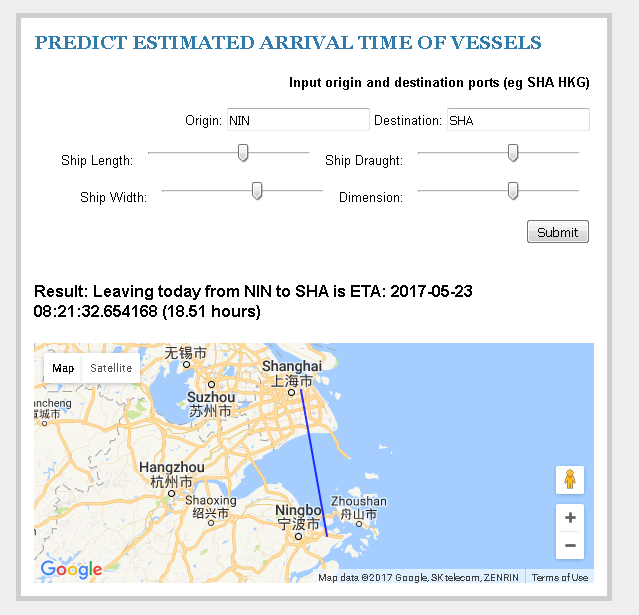


Figure 20. Sample screen shot of user interface to illustrate the effects of ship length, draught, width and dimension (dim\_c and dim\_d) in the predicted travel time from NINGBO to SHANGHAI

7. Findings

Below are the findings of this study:

1. Current Transit time has an **average of 32 hrs.** with minimum of 10hrs and maximum of 83hrs (3.5days) from **Ningbo to Shanghai in 2014 to 2015.**  The current mean absolute percentage error (MAPE) is 84% and mean absolute error (MA) equals 19 hours while this study is 14% and 4.3 hours, respectively.

Mean absolute percentage error (MAPE)

**MAPE Value**

Current ETA 84%

Predicted ETA 14%

Mean absolute error (MAE)

**MAE Value**

Current ETA 19hrs

Predicted ETA 4hrs

1. **Predicted ETA difference is within +/- 10 hrs** from actual travel time while current ETA is between -30 hrs to +10 hrs

As seen on the scatter plot below, this study reduced the range of the ETA prediction from minus 30hrs to +10hrs into plus and minus 10 hrs (half).

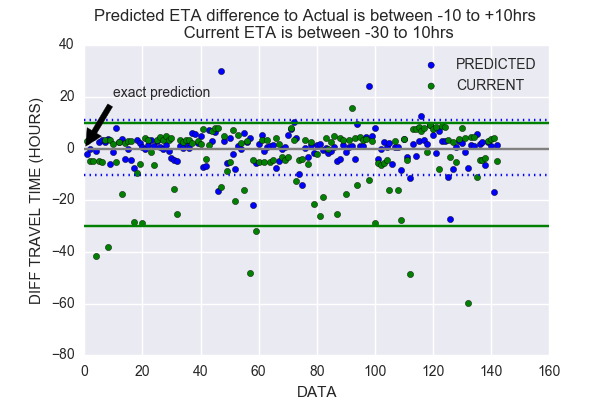


Figure 23. Predicted ETA is between +/- 10 hours from actual while current ETA is between -30 and +10 hours from actual travel time

1. **Seasonality** has a **small effect** on the prediction

Relative importance analysis is widely used to supplement multi variate regression analysis. The goal is to partition the explained variance among the predictors to better understand how each variable can explain the regression equation. However, we must also check that the predictors are not highly correlated to each other since this will make the indicators unreliable since one predictor can be removed in the regression equation.

Based on the results for regression analysis using gradient boosting (GBM) model, draught and ship length are top factors in predicting the travel time (or arrival time). Seasonality (in terms of year and week YEAR\_WK) is fall below 12% in relative influence factor.

**variable rel.inf**

AVG\_DRAUGHT 26.61848

AVG\_LENGTH 23.93271

AVG\_WIDTH 14.48452

YEAR\_WK 12.02791

AVG\_DIM\_D 11.71585

AVG\_DIM\_C 11.22053

This finding on the seasonality (YEAR\_WK) shows that it is a not a good predictor in the model and this answers the problem on how to detect or measure seasonality as a factor in estimating vessel’s arrival time.

1. Going to **Shanghai so busy, a vessel** has 4 or more stops **56% of the time.**

When a vessel is without delay, it will transmit around 40 AIS messages which is equivalent to around 10 hour travel time.



Figure 21. Speed of vessels without delay

Below graph shows the speed of the vessel with approximately 4 hours of delay.

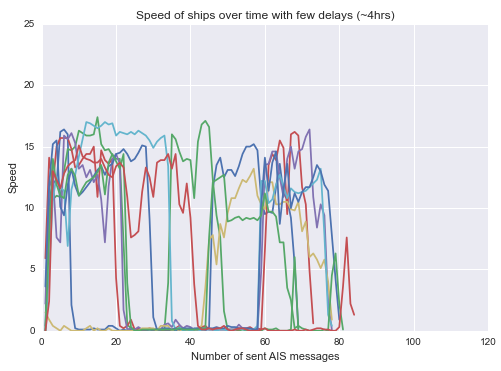


Figure 22. Speed of vessels with few delays (approx. 4 hours)

Below shows the average travel time from vessels.

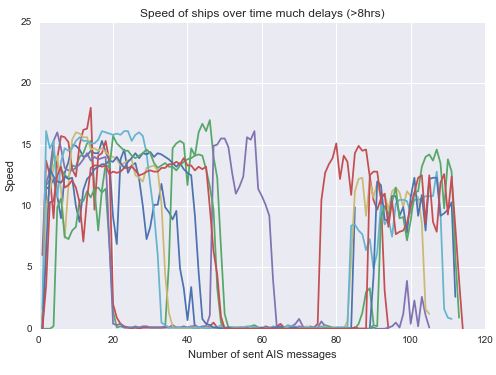


Figure 23. Speed of vessels with many delays (8 hours or more)

1. Path of vessels with or without delays

The first dataset is for voyages without stops and highlighted in green while those voyages with few and many stops are highlighted with yellow and red. This shows a different path when a vessel is delayed or with stops (red or yellow) versus when a vessel is on-time (green).

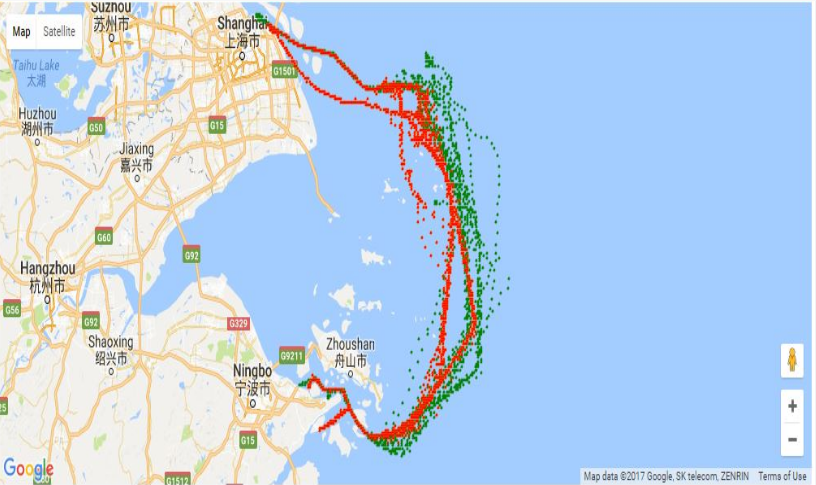


Figure 24. Path of the vessel when there is a stop or delay (red or yello) and when there is no delay (green). Yellow dots are not shown since it is overshadowed by red.

You will notice that when the vessel has no stops (no delays), it is going to a path which is on the farther out of the sea while a vessel with few or more stops are taking the path closer to islands.

1. Correlation heatmap

The correlation heatmap below shows that travel time (target variable) is highly correlated to number of stops which is expected. ETA year week number and month are also trivially highly correlated. Other variables such as ship width and length are lowly correlated to each other. See figure on the next page on the correlation heatmap.

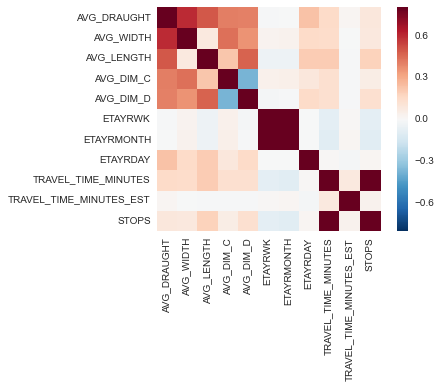


Figure 25. Correlation heatmap among features. An expected high correlation is observed between number of stops and travel time (target variable) and ETA year week number and month

8. Ideas for future research

This study can further extend to the following areas for future research

1. Real-time consideration on the prediction that updates the predicted travel time even when the vessel is in the middle of the voyage (similar to the way a GPS app would work). To get the predicted arrival of the vessel to the destination port, this study will add the predicted travel time to the planned departure date of the vessel based on the pre-planned schedule of the vessel. However, during the course of the voyage, the vessel may encounter delay due to factors like port congestion or other port conditions. If this happens, updating the predicted ETA will make it more accurate rather than using the ETA when the vessel departs from the origin port.
2. We can also extend this study to use other regression techniques like polynomial, ridge, lasso and elasticnet. Future research can explore these techniques and potentially improve the accuracy of the prediction. These techniques also can provide some advantages compared to linear models like random forest and gradient boosting.
3. Research further other factors (or features) that will make the prediction more accurate. Other factors such as weather conditions, ocean tide and terminal congestion can be added in the model to make it more accurate. Data can be accessed externally from public dataset or subscription from marine transportation websites.

9. Recommendations

The following recommendations are offered for related research in the field of logistics and supply chain demand predictions.

1. Integrate this sea/ocean travel time prediction with predictions related to outbound and inbound shipment cycle. This effort will enable a more accurate end to end prediction on the delivery date of the shipment when a customer creates a booking. Other predictions include predicting if a customer confirms or cancels the booking, predicting the traffic conditions when the shipment is in-transit and predict the time when a customer returns the empty container. <real-time> update in the model once traffic data is given
2. Apply a data validation on the AIS data in particular to the column: Destination. Since the format is in free text, it is a source of data quality issues like misspelling, multiple spaces in between words and abbreviations. Once the data on destination is clean, more data can be added in the study and will enable more accuracy on the predictions.
3. This study concentrated on NINGBO to SHANGHAI ports since this is the busiest ports in the world. It is suggested to create other models to study other major ports such as HONG KONG, SINGAPORE, ROTTERDAM, LOS ANGELES, BUSAN, etc. Each origin and destination pairs will have one model and will be used separately depending on the departing port and arrival port of the vessel.
4. Take an entity resolution[[6]](http://www.umiacs.umd.edu/~getoor/Tutorials/ER_VLDB2012.pdf) approach to get better idea of what other values of destinations exist. This task is to cluster the values of destination that corresponds to the same entity [[7]](http://www.datacommunitydc.org/blog/2013/08/entity-resolution-for-big-data). Since the data for destination is a free text and very prone to spelling errors, abbreviations and other input errors, we can match those different destination values that corresponds to the same destination name. This will help in cleaning the data and put more records in the dataset.

**Appendix:**

**Regression using random forest and gradient boosted machine (GBM)**

Link: <https://github.com/anonyXmous/CapstoneProject/blob/master/Using_GBM.R>

rm(list=ls())

gc(reset=TRUE)

setwd("D:/Rfiles/")

options(scipen=999) #prevent scientific notation

randomSeed = 123

set.seed(randomSeed)

library(gbm)

library(caret)

mape5 <- function(actual, preds) {

mape = 0

err = vector(mode="numeric", length=length(actual))

for(i in 1:length(actual)) {

err[i] = ifelse(actual[i] == 0, NA, (abs((actual[i] - preds[i])/actual[i])))

mape = mape + err[i]

}

return (mape/length(err))

}

LogLossBinary = function(actual, predicted, eps = 1e-15) {

predicted = pmin(pmax(predicted, eps), 1-eps)

- (sum(actual \* log(predicted) + (1 - actual) \* log(1 - predicted))) / length(actual)

}

######################## MAIN ################

fname = 'c:/Users/bacoyjo/NIN\_SHA.csv'

# target variable

target\_var = c('TRAVEL\_TIME\_HOURS')

# continous var

cont\_var = c( 'AVG\_DRAUGHT', 'AVG\_WIDTH', 'AVG\_LENGTH', 'AVG\_DIM\_C', 'AVG\_DIM\_D','ETAYRWK')

# index var

index\_var = c('VESSEL\_GID', 'TRAVEL\_TIME\_MINUTES')

ff\_input <- read.csv(fname, stringsAsFactors = F)

ff\_input$TRAVEL\_TIME\_HOURS = ff\_input$TRAVEL\_TIME\_MINUTES/60.0

ff\_input\_select = ff\_input[,c(cont\_var, target\_var, index\_var)]

for(i in c(cont\_var, target\_var))

ff\_input\_select[,i]<-as.numeric(ff\_input\_select[,i])

str(ff\_input\_select)

# remove outliers in the dataset

low\_perct = quantile(ff\_input$TRAVEL\_TIME\_MINUTES, .25) - 1.5\*IQR(ff\_input$TRAVEL\_TIME\_MINUTES)

high\_perct = quantile(ff\_input$TRAVEL\_TIME\_MINUTES, .75) + 1.5\*IQR(ff\_input$TRAVEL\_TIME\_MINUTES)

ff\_input = ff\_input[ff\_input$TRAVEL\_TIME\_MINUTES > low\_perct & ff\_input$TRAVEL\_TIME\_MINUTES< high\_perct,]

# split data into training/testing set

trainIndex <- createDataPartition(ff\_input\_select$TRAVEL\_TIME\_HOURS, p = .8, list = F, times = 1)

training <- ff\_input\_select[trainIndex,]

training$TRAVEL\_TIME\_HOURS = training$TRAVEL\_TIME\_MINUTES/60.0

testing <- ff\_input\_select[-trainIndex,]

testing$TRAVEL\_TIME\_HOURS = testing$TRAVEL\_TIME\_MINUTES/60.0

##############GBM MODEL ####################

gbmModel = gbm(formula = TRAVEL\_TIME\_HOURS ~ AVG\_DRAUGHT+ AVG\_WIDTH+ AVG\_LENGTH+ AVG\_DIM\_C+ AVG\_DIM\_D+ETAYRWK ,

distribution = "laplace",

data = training,

n.trees = 300,

shrinkage = .05,

cv.folds = 5,

n.minobsinnode = 5)

gbmTrainPredictions = predict(object = gbmModel,

newdata = training,

n.trees = 300)

gbmTestPredictions = predict(object = gbmModel,

newdata = testing,

n.trees = 300)

head(data.frame("Actual" = testing$TRAVEL\_TIME\_HOURS,

"PredictedProbability" = gbmTestPredictions))

summary(gbmModel, plot = FALSE)

bestTreeForPrediction = gbm.perf(gbmModel)

mape\_err = mape5(as.vector(testing$TRAVEL\_TIME\_HOURS),as.vector(gbmTestPredictions))

sprintf("MAPE (percent): %2.0f", (mape\_err)\*100)

################# L G O C V ###################

dataSubsetProportion = .2

randomRows = sample(1:nrow(training), floor(nrow(training) \* dataSubsetProportion))

trainingHoldoutSet = training[randomRows, ]

trainingNonHoldoutSet = training[!(1:nrow(training) %in% randomRows), ]

trainingHoldoutSet$RowID = NULL

trainingNonHoldoutSet$RowID = NULL

trainingHoldoutSet$Model = NULL

trainingNonHoldoutSet$Model = NULL

gbmForTesting = gbm(formula = TRAVEL\_TIME\_HOURS ~ AVG\_DRAUGHT+ AVG\_WIDTH+ AVG\_LENGTH+ AVG\_DIM\_C+ AVG\_DIM\_D+ETAYRWK,

distribution = "laplace",

data = trainingNonHoldoutSet,

n.trees = 500,

shrinkage = .01,

n.minobsinnode = 2)

summary(gbmForTesting, plot = FALSE)

gbmTestPredictions = predict(object = gbmForTesting,

newdata = testing,

n.trees = 500)

mape\_err = mape5(as.vector(testing$TRAVEL\_TIME\_HOURS),as.vector(gbmTestPredictions))

sprintf("MAPE (percent): %2.0f", (mape\_err)\*100)

################# WITH CROSS VALIDATION ###################

gbmWithCrossValidation = gbm(formula = TRAVEL\_TIME\_HOURS ~ AVG\_DRAUGHT+ AVG\_WIDTH+ AVG\_LENGTH+ AVG\_DIM\_C+ AVG\_DIM\_D+ETAYRWK,

distribution = "laplace",

data = trainingNonHoldoutSet,

n.trees = 500,

shrinkage = .1,

n.minobsinnode = 5,

cv.folds = 5,

n.cores = 1)

bestTreeForPrediction = gbm.perf(gbmWithCrossValidation)

gbmTestPredictions = predict(object = gbmWithCrossValidation,

newdata = testing,

n.trees = 500)

mape\_err = mape5(as.vector(testing$TRAVEL\_TIME\_HOURS),as.vector(gbmTestPredictions))

sprintf("MAPE (percent): %2.0f", (mape\_err)\*100)

**RandomForest:**

**Link:** <https://github.com/anonyXmous/CapstoneProject/blob/master/Using_RF.R>

rm(list=ls())

gc(reset=TRUE)

setwd("D:/Rfiles/")

options(scipen=999) #prevent scientific notation

library(h2o)

library(leaps)

library(kernlab)

library(caret)

library(readr)

library(dplyr)

library(lubridate)

library(Hmisc)

library(ggplot2)

library(Metrics)

library(DMwR)

library(methods)

library(reshape2)

library(dummies)

library(xgboost)

library(e1071)

library(doParallel) #Register workers for parallel run

mape5 <- function(actual, preds) {

err = vector(mode="numeric", length=length(actual))

for(i in 1:length(actual)) {

err[i] = ifelse(actual[i] == 0, NA, (abs((actual[i] - preds[i])/actual[i]))\*100.0)

}

return (err)

}

# GridSearch regression

GridSearch\_regression <- function(model.label,

Xtrain,

Ytrain,

Xtest,

GridObject,

ControlObject,

Importance = FALSE,

Verbose = FALSE)

{

if (model.label == "knn") { ### KNN\_Reg

if(!missing(GridObject)) {

knnGrid = GridObject

} else {

knnGrid = expand.grid(c(.k = 1:20))

}

model <- caret::train(x = Xtrain, y = Ytrain,

method = "knn",

preProc = c("center", "scale"), # or preProcess = 'range'

tuneGrid = knnGrid,

trControl = controlObject,

importance= Importance,

verbose = Verbose)

} else if (model.label == "svmRadial") { ### KNN\_Reg

if(!missing(GridObject)) {

svmGrid = GridObject

} else {

svmGrid =expand.grid(.C = c(1, 10, 100, 500, 1000),

.sigma = c(0.001, 0.01, 0.1))

}

model <- caret::train(x = Xtrain, y = Ytrain, method = "svmRadial",

preProc = c("center", "scale"), # or preProcess = 'range'

tuneGrid = svmGrid,

trControl = controlObject,

importance= Importance,

verbose = Verbose)

} else if(model.label == "randomForest") { ### RANDOM FOREST ####

if(!missing(GridObject)) {

rfGrid = GridObject

} else {

rfGrid = expand.grid(mtry = c(1,2,3,5,7,9))

}

model <- caret::train(x = Xtrain, y = Ytrain,

method = "rf",

tuneGrid = rfGrid,

ntrees = 300,

#max\_depth = 7, min\_child\_weight = 5,

do.trace = 100,

trControl = controlObject,

importance = Importance,

verbose = Verbose)

} else if (model.label == "xgbLinear") {

print("Start xgbLinear")

if(!missing(GridObject)) {

xgbGrid = GridObject

} else {

#xgbGrid = expand.grid(nrounds=1000,

xgbGrid = expand.grid(nrounds=500,

eta = c(0.01, 0.05, 0.1, 0.3), # step size shrinkage

lambda = c(0), # L2 Regularization

alpha = c(1)) # L1 Regularization

}

model <- caret::train(x = as.matrix(Xtrain), y = as.numeric(Ytrain), method = "xgbLinear",

tuneGrid = xgbGrid,

nthread = 6,

eval\_metric = "rmse",

subsample = 0.8,

preProcess = c("center","scale"), # scale feature

#preProcess="pca", # another scale feature

trControl = controlObject,

importance= Importance,

verbose = Verbose)

} else if(model.label == "svmPoly") {

print("Start svmPoly")

if(!missing(GridObject)) {

xgbGrid = GridObject

} else {

svmPolyGrid = expand.grid(C = c(10, 100, 200),

scale = c(0.01),

degree = c(2,3,4))

}

model <- caret::train(x = Xtrain, y = Ytrain, method = "svmPoly",

verbose = T,

preProc = c("center", "scale"), # or preProcess = 'range'

tuneGrid = svmPolyGrid,

trControl = controlObject,

importance= Importance,

verbose = Verbose)

} else if(model.label == "pcr") {

print("Start pcr")

if(!missing(GridObject)) {

pcrGrid = GridObject

} else {

pcrGrid = expand.grid(.ncomp = 1:10)

}

pcrGrid

model <- caret::train(x = Xtrain, y = Ytrain, method = "pcr",

verbose = T,

preProc = c("center", "scale"), # or preProcess = 'range'

tuneGrid = pcrGrid,

trControl = controlObject,

importance= Importance,

verbose = Verbose)

}

#trellis.par.set(caretTheme())

plot(model)

model$results # results of training

model$bestTune # tuning parameters

return(model)

}

mape5 <- function(actual, preds) {

err = vector(mode="numeric", length=length(actual))

for(i in 1:length(actual)) {

err[i] = ifelse(actual[i] == 0, 100.0, (abs((actual[i] - preds[i])/actual[i]))\*100.0)

}

return (err)

}

###################################### main ######################################

fname = 'd:/files/SHA\_NIN.csv'

# target variable

target\_var = c('TRAVEL\_TIME\_HOURS')

# continous var

cont\_var = c('AVG\_DRAUGHT', 'AVG\_WIDTH', 'AVG\_LENGTH', 'AVG\_DIM\_C', 'AVG\_DIM\_D','ETAYRWK','STOPS')

# index var

index\_var = c('VESSEL\_GID', 'TRAVEL\_TIME\_MINUTES')

ff\_input <- read.csv(fname, stringsAsFactors = F)

# remove outliner

low\_perct = quantile(ff\_input$TRAVEL\_TIME\_MINUTES, .25) - 1.5\*IQR(ff\_input$TRAVEL\_TIME\_MINUTES)

high\_perct = quantile(ff\_input$TRAVEL\_TIME\_MINUTES, .75) + 1.5\*IQR(ff\_input$TRAVEL\_TIME\_MINUTES)

cat("BEfore remove outlier,", dim(ff\_input)[1], '\n')

ff\_input = ff\_input[ff\_input$TRAVEL\_TIME\_MINUTES > low\_perct & ff\_input$TRAVEL\_TIME\_MINUTES< high\_perct,]

cat("After remove outlier,", dim(ff\_input)[1], '\n')

location\_status\_count <- ff\_input %>%

select(AVG\_DRAUGHT, AVG\_WIDTH, AVG\_LENGTH, AVG\_DIM\_C, AVG\_DIM\_D, ETAYRWK, TRAVEL\_TIME\_MINUTES) %>%

group\_by(LOCATION, STATUS) %>%

summarise(meanETA=mean(DETA),stdETA=sd(DETA), support=n()) %>%

as.data.frame() %>%

arrange(desc(meanETA)) %>%

filter(support>30)

dim(ff\_input)

names(ff\_input)

describe(ff\_input)

str(ff\_input)

# Generate features

# convert to hours

ff\_input$TRAVEL\_TIME\_HOURS = ff\_input$TRAVEL\_TIME\_MINUTES/60.0

ff\_input$VOLUME = log(ff\_input$AVG\_LENGTH\*ff\_input$AVG\_WIDTH\*ff\_input$AVG\_DIM\_C)

ff\_input\_select = ff\_input[,c(cont\_var, target\_var, index\_var)]

for(i in c(cont\_var, target\_var))

ff\_input\_select[,i]<-as.numeric(ff\_input\_select[,i])

str(ff\_input\_select)

pairs(ff\_input\_select)

# split data into training/testing set

set.seed(123)

trainIndex <- createDataPartition(ff\_input\_select$TRAVEL\_TIME\_HOURS, p = 0.8, list = F, times = 1)

training <- ff\_input\_select[trainIndex,]

testing <- ff\_input\_select[-trainIndex,]

Ytrain = training[,target\_var]

Ytest = testing[,target\_var]

curr\_tt = ff\_input[-trainIndex,]$TRAVEL\_TIME\_MINUTES\_EST

training[,target\_var] = NULL

testing[,target\_var] = NULL

testing\_index = testing[,index\_var]

training[,index\_var] = NULL

testing[,index\_var] = NULL

names(training)

names(testing)

cat('\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* models \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n')

cl<-makeCluster(6)

registerDoParallel(cl)

controlObject <- trainControl(method = "cv", number = 10, returnResamp = "all", search = "grid", verboseIter = TRUE, allowParallel = TRUE)

#controlObject <- trainControl(method = "repeatedcv", repeats = 3, number = 10, returnResamp = "all", search = "grid", verboseIter = TRUE, allowParallel = TRUE)

xgbGrid = expand.grid(nrounds= c(5, 10, 100),

#max\_depth = c(2,4,6,8,10,14), #TEST

eta = c(0.01, 0.05, 0.1, 0.3), # learning rate

lambda = c(0), # L2 Regularization

alpha = c(1)) # L1 Regularization

rfGrid = expand.grid(mtry = ceil(c(0.1,0.25,0.5)\*length(names(training))))

model='xgbLinear'

Grid = xgbGrid

Importance = TRUE # FALSE

model\_fit = GridSearch\_regression(model,

Xtrain = training,

Ytrain = Ytrain,

Xtest = testing,

GridObject = Grid,

ControlObject = controlObject,

Importance = Importance,

Verbose = FALSE)

plot(varImp(model\_fit))

if(Importance) {

imports <- varImp(model\_fit)$importance %>%

mutate(names=row.names(.)) %>%

arrange(-Overall)

}

ggplot(model\_fit) + theme(legend.position = "top")

pred = predict(model\_fit , newdata = testing)

err = c(regr.eval(Ytest, pred, stats=c('mape','mae','rmse')), 'R2'=caret::R2(Ytest, pred, form='traditional'))

mape\_err = mape5(as.vector(Ytest),as.vector(pred))

ma\_err = abs(pred-Ytest)

eval = cbind(testing\_index, ACTUAL\_TT=(Ytest),PRED\_TT=pred, MAPE=mape\_err, MA = ma\_err)

result = list("err" = err, "imports" = imports, "eval" = eval)

result

####################################

model='randomForest'

Grid = rfGrid

model\_fit\_rf = GridSearch\_regression(model,

Xtrain = training,

Ytrain = Ytrain,

Xtest = testing,

GridObject = Grid,

ControlObject = controlObject,

Importance = Importance,

Verbose = FALSE)

plot(varImp(model\_fit\_rf))

if(Importance) {

imports <- varImp(model\_fit\_rf)$importance %>%

mutate(names=row.names(.)) %>%

arrange(-Overall)

}

ggplot(model\_fit\_rf) + theme(legend.position = "top")

pred = predict(model\_fit\_rf , newdata = testing)

err = c(regr.eval(Ytest, pred, stats=c('mape','mae','rmse')), 'R2'=caret::R2(Ytest, pred, form='traditional'))

mape\_err = mape5(as.vector(Ytest),as.vector(pred))

ma\_err = abs(pred-Ytest)

eval = cbind(testing\_index, ACTUAL\_TT=(Ytest),PRED\_TT=pred, MAPE=mape\_err, MA = ma\_err)

result = list("err" = err, "imports" = imports, "eval" = eval)

eval = cbind(testing\_index, ACTUAL\_TT=(Ytest),PRED\_TT=pred, MAPE=mape\_err, MA = ma\_err, CURR\_TT=curr\_tt)

eval

err

stopCluster(cl)

References:

1. <http://blog.rbwlogistics.com/the-5-biggest-supply-chain-challenges/>
2. <http://www.re-transfreight.com/blog/top-supply-chain-issues-trends-for-2017>
3. <https://logisticsviewpoints.com/2017/03/16/supply-chain-2017-important-transportation-trends-and-challenges/>
4. <https://en.wikipedia.org/wiki/Automatic_identification_system>

5) <http://nbviewer.jupyter.org/github/anonyXmous/CapstoneProject/blob/master/Capstone-Maps.html>

6) <http://www.umiacs.umd.edu/~getoor/Tutorials/ER_VLDB2012.pdf>

7) <http://www.datacommunitydc.org/blog/2013/08/entity-resolution-for-big-data>