# Code Explanation – Purdue IP team

## Introduction

This paper explains the code written by the Purdue IP team (2022) – Jackson Bronkema, Diego Victor Carlos Chavez, Asad Husain, Yuxuan Li, Kumar Rahul, and Kshitij Virdi.

The purpose of the code is to forecast the shipping volumes for RH for the next year.

To know more about the project, please refer to the project presentation or poster.

## Install R and RStudio

Throughout this project we used R as our tool of choice to analyse everything. R and its IDE (Integrated Development Environment – a software to manage your files and read/write the code) RStudio are open source and free.

Please see this link on how to install R and its IDE in your workstation:

Install language - <https://cran.r-project.org/>

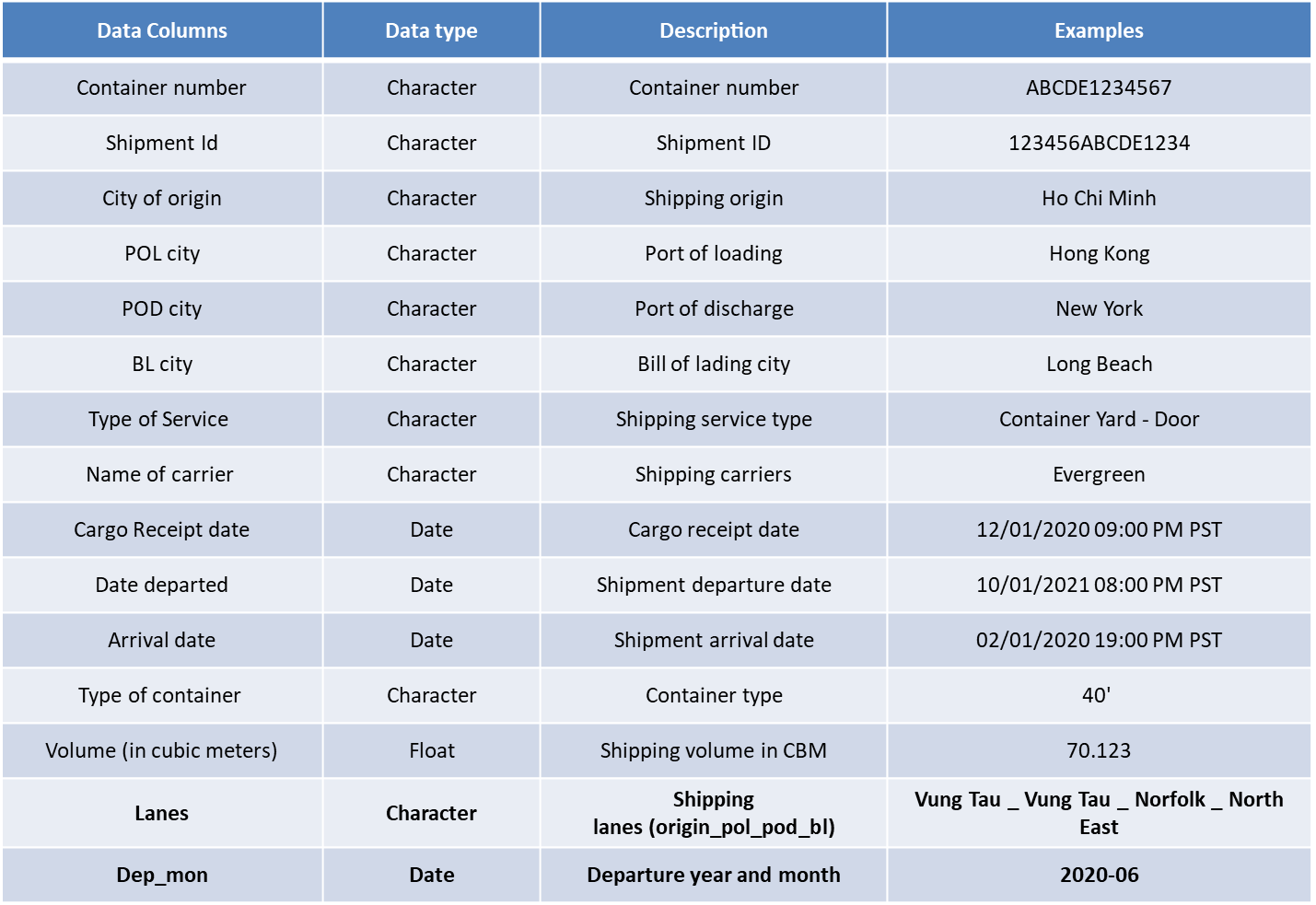
Install IDE (free desktop version) - <https://www.rstudio.com/products/rstudio/download/>

If you need a tutorial, do refer to following YouTube tutorials,   
- Windows: <https://youtu.be/TFGYlKvQEQ4>

- Mac: <https://youtu.be/cCgiR1uwXzU>

## Data

The data file needed to run the code is also attached called ‘ShippingVolumeFull.csv’. This file has data starting from 2016 to 2021. All six-month files were merged to get this data set.

The dataset has the following columns:

## Code

In this section we are going to run through the purpose of each code chunk. To summarise, there are 3 sections in the code.

1. Libraries Used
2. Custom Functions
   1. Data Cleaning function
   2. Lane function
   3. Forecast and plotting function
3. Running the code with functions

To begin, open the code file attached in RStudio. As soon as you open the file, RStudio might suggest installing some packages for libraries that are used in the code which are not by default installed with the software.

For ease of use, do put the data file and R code file in the same folder and remember to set the working directory by choosing ‘To Source File Location’ as shown in the pic below.

Graphical user interface, text, application, chat or text message, email

Description automatically generated

After installing the packages and setting the working directory, you can simply select entire code and press *Run* button or ‘ctrl + enter’ or ‘cmd + enter’ to run the code. It might take a few minutes, depending on the machine, but as soon as the code is run, the result table will populate the screen.

To understand the working of each section, please follow along.

### Libraries Used

Text

Description automatically generated

Here we simply load the required libraries for the program. Each library consist of a few functions that we are using in different sections of the code. The commented part (green) might give some hints as to what purpose does it serve.

### Custom Functions

To repeatedly model the lanes and make the code readable we made functions to solve a mini problem.

Each of those functions are explained below.

#### Data Cleaning Function

Input: -

* Name of the csv file
* num\_lanes – No. of lanes to model. Default number set is 15

Steps: -

* Read the data file
* Delete unwanted columns and rename the ones left for easy access
* Create new column – Complete shipping lane (origin city + pol city + pod city + delivery city – Lane
* Combine certain lanes if there is some external business fact/decision (ex. delivery city for some lanes changed from Oakland to Patterson in 2018 July)
* Change data type of columns
* Filter abnormal rows with less than 50 CBMs as shipped volume (less than 1 container load but marked as FCL)
* Create new column – departure month – Dep\_month
* Subset the columns *Lane, Volume, Dep\_month,* group the dataset by Lane, aggregate the volume by month, arrange the grouped list in descending order of volume
* Drop lanes which don’t have any shipments at and after 2019
* Create a list of data frames with equally spaced months and subsequent volumes for ‘num\_lanes’ top lanes

Output: -

* A list of lane-wise data frames (‘num\_list’ in number) having months and volumes as the two columns

#### Lane Function

Input: -

* List of data frames (output of the datacleaning function)
* (empty) data frame ‘alllanesfinal’ which stores results for all lanes

Steps: -

* Read the output of the previous function (list of data frames)
* Check if the lane has less than 24 months of data and if more than half of the months’ volume is missing, if no proceed to next step, if yes print ‘not enough data to model’ and skip to next lane/ data frame
* Extract the start date and end date of the lane
* Check if the lane has any missing values, if no proceed to next steps, if yes repeat the rest of the steps with mean imputed and no imputed data frames and then choose the one least MAPE score
* Define a timeseries with the data frame with start and end date as extracted in the previous step
* Call the ‘forecastnplot’ function while using the timeseries, timehorizon (default 12), and name of the lane as input and storing the output in a variable
* Add the result of the previous line in the solution data frame ‘alllanesfinal’

Output: -

* Final solutions data frame ‘alllanesfinal’

#### Forecast Function

Inputs: -

* Time series – to model
* Timehorizon – to know how many months to forecast
* Nameofplot – to put as title of the plot

Steps: -

* Defining ets and arima functions to use in the tsCV function later
* Calculating the cross validated error using the tsCV function
* Calculating MAPE from the point errors of both ets and arima models
* Check which MAPE score is lower and follow the following steps for only that model
* Model and forecast the lane for ‘timehorizon’ months
* Aggregate the volume for the next ‘timehorizon’ months and save as contractval (aggregated yearly forecasted volume), save the minimum and maximum contract value in the 95% confidence interval, save the MAPE score as well
* Plot the model

Output: -

* A vector of MAPE value, min contract value, contract value, max contract value

### Running the code with functions

Call the datacleaning function and store the list of dataframes in a variable ‘lanedata’.   
Get the names of all data frames in the list in ‘lnames’.   
Define an empty data frame called ‘alllanesfinal’.  
Run a loop to model and plot all the lanes.