**Getting Started in R**

* Download R
  + R program: <https://cran.rstudio.com/>
    - Download 64-bit if your computer is 64-bit, but you’ll need to make sure that your Java is also 64-bit (see Gardner’s link below for more info)
  + R Studio (User-friendly GUI): <https://www.rstudio.com/products/rstudio/download/>
* Download Github
  + <Using Github>
* Learning R
  + Take Matt Gardner’s in-person R course: <https://wiki.vip.corp.ebay.com/display/~magardner/R+Jumpstart+Two+Day+Training+Course+Introduction>
  + Online R course (used as pre-work in Gardner’s class): <https://www.codeschool.com/courses/try-r>
* eBayTD: Matt Gardner’s package that pulls data from SQL queries in R
  + Follow instructions in this link: <https://wiki.vip.corp.ebay.com/display/Analytics/Connecting+to+Teradata+from+R+at+eBay+-+ebaytd+R+package+not+RJDBC+RODBC>
  + Matching 64-bit R w/ 64-bit Java is important: if you don’t have 64-bit Java, uninstall your current version and then install the 64-bit
  + Email Jay Weiler if you have any questions: [jaweiler@ebay.com](mailto:jaweiler@ebay.com)
* R packages
  + Packages are necessary for many functions to work
  + Packages that need to be loaded are seen at the top of each script
    - Ie. library(plyr)
    - **Before this package can be loaded, you need to download it via the command install.packages(‘plyr’)**

**3 forecasting splits**

*Each model has the same structure but has different splits*

* Verticals (within B2C)
  + Home & Garden
  + Electronics
  + Parts & Accessories
  + Fashion
* B2C/C2C
* DE aggregate

**Basic Structure**



**Do.R**

*Main execution file*

1. Do.R runs the entire code and calls the all subscripts
2. Manual Inputs
   1. testDays: number of days to forecast, enter a multiple of 7 days to avoid DoW effect
   2. trainEnd: last date for GMV input—defaults to T-3 to avoid missing data from DW
3. Other settings
   1. trainStart(startYear, startDoY): <>
   2. testStart, testEnd: <forecasting section>

**Update.R**

*Update PET table*

1. Automatically updates PET table data to T-3
   1. Removes 14 latest dates in the existing table
   2. Adds back those last 14 dates + data until T-3
   3. eg. If today is August 30 and data goes to August 15, then update.R will remove Aug 1-Aug 15 and add in Aug 1-Aug 27
2. Original PET table query is *createActualsTable.sql* if you need to recreate the PET table
3. PET table dates are automatically calculated and entered into the SQL queries
4. **Initiation: Before you run this or *Load.R* for the first time, you need to download the *ebayTD* package and enter credentials. See under ebayTD under ‘Getting Started in R’**

**Load.R**

*Pull PET data into R*

1. Pulls data from PET table into R
2. Vertical forecasting filters for B2C
3. Variables
   1. *vertList*: list of top 4 verticals
   2. *vertName*\_gmv: creates individual GMV datasets by vertical
4. ddply is used to restructure the data into the necessary ‘shape’
   1. In this case, *vertsGMV* takes in the *actual0.lim* dataset and formats the set to be date x vertical x sum(GMV\_PLAN)

**Variables.R**

*Create variables for regression and forecast*

1. This script creates the independent variables needed for the regression and forecast
   1. Because most variables are holidays, they cannot be hardcoded as the variables are date-dependent
2. *DFend*: the end date for variable creation, defaulted as 2 years out from the latest actuals date
   1. Actuals = past data points with GMV available
3. *weeklyGrowth*: site growth variable-- increasing by 1 each week to remove seasonality
4. *woy*: week of year variable: 1-52 for each week-- seasonality variable
5. *dow*: day of week variable (Monday-Saturday) **+ holiday marker**
   1. Holiday marker is if the date is a holiday as the DoW is no longer relevant— whether Christmas occurs on a Wednesday or Sunday does not matter; all that matters is that it’s a holiday
6. *holidays*
   1. The holiday is a forced variable in the regression, but the days surrounding them are not; more explanations later
   2. Some holidays are available in the holiday() function from the timeDate package and the rest are derived from either x-days around Easter (ie. DECarnival) or occur at the same date each year (ie. DE Labor Day)
   3. Holidays: Easter, Pentecost, DEAscension, DECorpusChristi, DEGermanUnity, DEChristmasEve, DENewYearsEve, ChristmasDay, NewYearsDay, DECarnival, DELaborDay, BoxingDay, AllSaints, AllSaintsA (Day before All Saints), MothersDay, PentecostB, Walpurgisnacht
7. *paydates*
   1. Germans are paid either at the end of the month (31st or 30) or the beginning of the month
   2. Because credit card penetration is low, spending occurs around these dates when they have more money in their bank account
8. *7d naming convention*:

**Reg\_lnGMV.R**

**ARIMA.R**

**Forecast.R**

**Other Info**

* Trace=TRUE: as a setting for auto.arima() in arima.R and step() in reg\_lnGMV.R shows the steps, or ‘thinking process’, of the algorithmic stepwise regression and ARIMA processes

**Troubleshooting**

Updating in parallel—wait until each load.R is completed to avoid constant re-writing of the PET table

Troubleshooting:

* If forecast is wildly off, check that the input GMV numbers are correct. It’s possible that in the PET table process, the GMV numbers are loaded incorrectly. If the numbers are incorrect, rewrite the PET table using the ’createActualsTables’ SQL query under the SQL folder
* If past GMV numbers are confirmed to be correct, check the following:
  + Regression vs ARIMA
    - These are the two parts of the forecast. ARIMA determines the level of GMV, while regression determines the distribution of the GMV across days
  + Coding of holidays

Common known warnings <>

**Next Steps**

Possible areas for improvement

1. Fix the sinusoidal errors <chart>