Final Project

Michael Zoucha

2021-11-20

Table of Contents

Setup

Required Packages

1. tidyverse
    - Includes:
        - ggplot2 for visualization
        - dplyr for data manipulation
        - tidyr for data cleaning
        - readr for data import
        - purr for functional programming
        - tibble for tibble management
        - stringr for strings
        - forcats for factors
2. lubridate
3. report

Introduction

Inventory is the arguably the most important aspect of a sales business to manage, if you don't have goods to sell you can't sell them, but how much excess inventory is too much? Excess inventory, first and foremost, reduces available cash flow for an organization (Gartenstein, 2019). Reduced cash flow can lead requiring loans to cover liabilities and pay interest on that loan, or it can eliminate the ability to take advantage of a capital-requiring opportunity. In recent times, during the uncertainty of the pandemic, reduced cash flow from excess inventory meant the difference between weathering the storm and closing for good (Lazar, 2020). Using machine learning, forecasting sales quantity amounts per item, per location can help give businesses the opportunity to ensure they have enough inventory for upcoming sales, all while reducing excess inventory and the costs associated with that excess.

Research Questions

1. Using historical sales data, can one predict the maximum inventory needed on hand for a particular item in a particular time period? (i.e., for May 2022 we will need 22 of item x, and in June of 2022 we will only need 17)
2. Combining the predictions of inventory needs with up-to-date manufacturing and/or delivery times, can one create a dynamically changing minimum inventory number for each item? (i.e., it's the middle May of 2022 and we have 10 of item x left, we need to order 15 by May 22 to have the stock we need for June.)
3. Can one optimize the relationship between number of orders to a manufacturer (hence, shipping costs) and keeping the absolute minimum number of stock on hand? (i.e., Should we order 15 on May 22, or 5 on May 22 then 5 on June 2, etc.)
4. Can these predictions and optimizations be broken down to the region, district, state, or even city level? (i.e., West Coast needs 12 in May 2022, Midwest needs 5 for May 2022, etc.) (i.e., San Diego needs 5, Los Angeles needs 3, Chicago needs 2, etc.)
5. Can one build price modeling into the forecasting to ensure inventory is purchased at lowest possible average-per-unit price? (i.e., the price of item x is forecast to go up in July of 2022, is it more profitable to buy and hold excess inventory at a lower cost?)
6. Could acquiring regional warehouse capabilities help reduce excess inventory even further? (i.e., San Diego only needs 2 of item x in store, because if they really need it, more is 1 day shipping away)

Approach

By first analyzing historical sales data, a model can be created to forecast future sales amounts per item. Using this forecast model, one can create a dynamically changing minimum level of inventory to trigger new orders instead of creating static over-estimates. This dynamic inventory level will ensure enough product is in stock for upcoming sales, without compromising and overstocking excess inventory just to be certain.

Data

Data comes from the AdventureWorks2019 test database from Microsoft: AdventureWorks2019.bak

1. ProductDetails
   a. ProductID (INT) – Unique product number identifier for database
   b. Name (CHAR) – Unique product description / name
   c. ProductNumber (CHAR) – Unique warehouse stock number
   d. MakeFlag (INT) (BINARY) – Indicates if the good needs assembled in house
   e. FinishedGoodsFlag (INT) (BINARY) – Indicates whether the product is finished in the warehouse
   f. Color (CHAR) – Color of the product, if applicable
   g. SafetyStockLevel (INT) – Desired amount of inventory per item
   h. ReorderPoint (INT) – Level at which inventory should be reordered
   i. StandardCost (FLOAT) – Average cost per item
   j. ListPrice (FLOAT) – Average price per item
   k. Size (CHAR) – Measurement of item, either character or integer
   l. SizeUnitMeasureCode (CHAR) – Unit of measurement used for size
   m. Weight (FLOAT) – Weight measurement of item
   n. WeightUnitMeasureCode (CHAR) – Unit of measurement used for weight
   o. DaysToManufacture (INT) – Days required to process order before shipment
   p. ProductLine (CHAR) – Subclass of products
   q. Class (CHAR) – Subclass of products
   r. Style (CHAR) – Subclass of products
   s. ProductSubcategoryID (INT) – Splits major categories into subcategories
   t. ProductModelID (INT) – Unique product model number
   u. SellStartDate (DATE) – Date of first sale
   v. SellEndDate (DATE) – Date of last sale
   w. DiscontinuedDate (DATE) – Date manufacturer discontinued production
   x. ROWGUID (CHAR) – Used by MS SQL for replication
   y. ModifiedDate (DATE) – Date line was modified
2. ProductCategory
   a. ProductCategoryID (INT) – Unique product category ID
   b. Name (CHAR) – Unique category name
   c. RowGUID – Used by MS SQL for replication
   d. ModifiedDate (DATE) – Date line was modified
3. ProductSubcategory
   a. ProductSubcategoryID – Unique product subcategory ID
   b. ProductCategoryID (INT) – Product category ID associated with each subcategory
   c. Name (CHAR) – Unique category name
   d. RowGUID – Used by MS SQL for replication
   e. ModifiedDate (DATE) – Date line was modified
4. SalesOrderDetail
   a. SalesOrderID (INT) – Unique identifier for each complete sale
   b. SalesOrderDetailID (INT) – Unique identifier for each line item sold
   c. CarrierTracking (CHAR) – Tracking number for order
   d. OrderQty (INT) – Number of each item ordered per order
   e. ProductID (INT) – Unique identifier for the product sold
   f. SpecialOffer (INT) – Discount code
   g. UnitPrice (FLOAT) – Total price for each item sold
   h. UnitPriceDiscount (FLOAT) – Discount amount, if applicable

   i. LineTotal (FLOAT) – Total Price per line item (UnitPrice x OrderQty)
   j. RowGUID – Used by MS SQL for replication
   k. ModifiedDate (DATE) – Date line was modified

5. SalesOrderHeader
   a. SalesOrderID (INT) – Unique Sales Order ID number
   b. RevisionNumber (INT) – Order revision number
   c. OrderDate (DATE) – Date of order
   d. DueDate (DATE) – Date order is due
   e. ShipDate (DATE) – Date order was shipped
   f. Status (INT) – Order status code
   g. OnlineOrderFlag (INT) (BINARY) – Indicates if order was online or in store
   h. SalesOrderNumber (INT) – "SO" + SalesOrderID
   i. PurchaseOrderNumber (INT) – Purchase Order Number
   j. AccountNumber (CHAR) – Customer account number
   k. CustomerID (INT) – Customer ID number
   l. SalesPersonID (INT) – ID of salesperson
   m. TerritoryID (INT) – Territory ID code
   n. BillToAddressID (INT) – Key to store billing address
   o. ShipToAddressID (INT) – Key to store shipping address
   p. ShipMethondID (INT) – Key to store shipping method
   q. CreditCardID (INT) – Key to store credit card info
   r. CreditCardApprovalCode (INT) – Approval code for credit transaction
   s. CurrencyRateID (INT) – Key to store currency used in transaction
   t. Subtotal (FLOAT) – Order subtotal
   u. TaxAmt (FLOAT) – Taxes for order
   v. Freight (FLOAT) – Shipping cost
   w. TotalDue (FLOAT) – Total for order (subtotal + tax + shipping)
   x. Comment (CHAR) – Special order comments, if applicable
   y. RowGUID – Used by MS SQL for replication
   z. ModifiedDate (DATE) – Date line was modified

6. SpecialOffer
   a. SpecialOfferID (INT) – Unique offer identifier
   b. Description (CHAR) – Discount description
   c. DiscountPct (FLOAT) – Discount percentage
   d. Type (CHAR) – Discount type
   e. Category (CHAR) – Discount category
   f. StartDate (DATE) – Promotion start date
   g. EndDate (DATE) – Promotion end date
   h. MinQty (INT) – Minimum quantity required for discount
   i. MaxQty (INT) – Maximum quantity allowed to be discounted
   j. RowGUID – Used by MS SQL for replication
   k. ModifiedDate (DATE) – Date line was modified

7. SalesTerritory
   a. TerritoryID (INT) – Unique territory identifier
   b. Name (CHAR) – Territory name
   c. CountryRegion (CHAR) – Country code
   d. Group (CHAR) – Territory group
   e. SalesYTD (FLOAT) – Aggregate of sales YTD
   f. SalesLastYear (FLOAT) – Aggregate of sales last year
   g. CostYTD (FLOAT) – Aggregate of cost YTD
   h. RowGUID – Used by MS SQL for replication
   i. ModifiedDate (DATE) – Date line was modified

Data Cleaning

1. Parse down data using select function
2. Combine product date frames into one data frame

   a. Merge ProductCategory to ProductSubcategory using a left join on ProductCategoryID
   b. Merge ProductSubcategory to ProductDetails using a left join on ProductSubcategoryID
   c. Rename 'name' columns to give more meaning

3. Combine the rest of the data frames with the newly created product_merged data frame

   a. Merge SalesTerritory to SalesOrderHeader using a left join on TerritoryID
   b. Merge SpecialOffer to SalesOrderHeader using a left join on SpecialOfferID
   c. Merge new SalesOrderHeader_merged data frame to SalesOrderDetail using a left join on SalesOrderID
   d. Merge product_merged to sales_details_merged using a left join on ProductID
   e. Rename columns to give more meaning

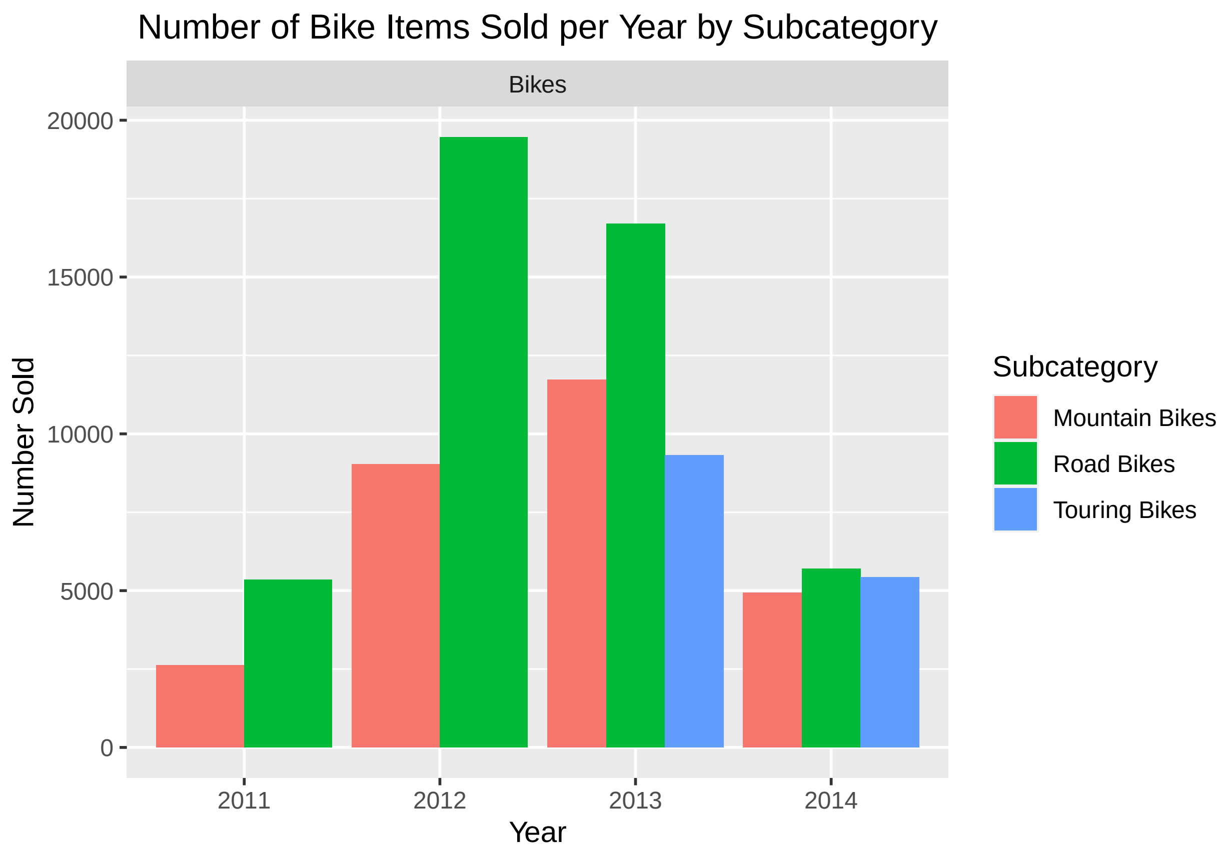
4. Remove unnecessary data frames
5. Add calculated columns to the data frame for further analysis

   a. Year substring of date
   b. Month substring of date
   c. TotalProfit
   d. ItemProfit

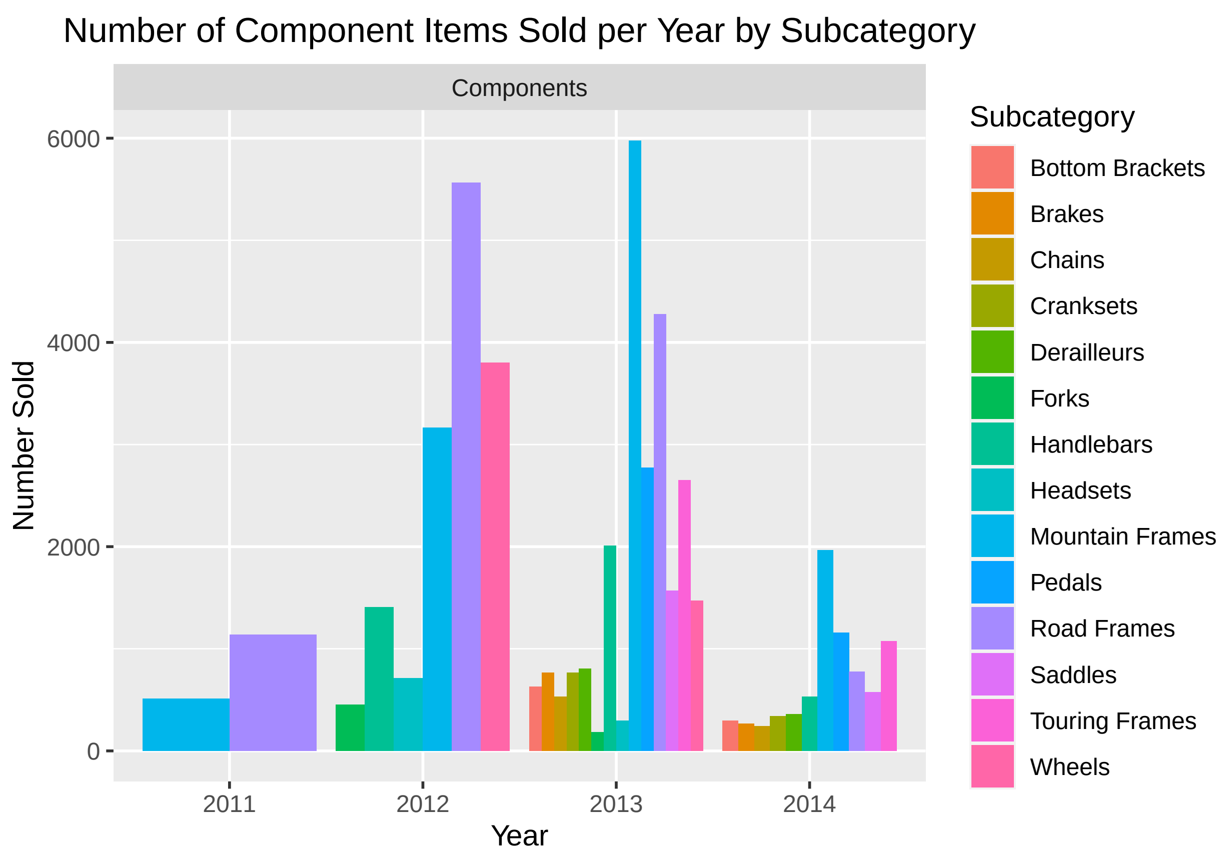
```
## 'data.frame':    121317 obs. of  27 variables:
##  $ TerritoryID         : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ SpecialOfferID      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ SalesOrderID        : int  70638 69525 64365 53623 70474 70584 65572 47986 65170 72935 ...
##  $ ProductID           : int  870 938 860 973 873 969 711 831 998 708 ...
##  $ OrderQty            : int  1 4 1 3 1 1 1 2 4 1 ...
##  $ UnitPrice           : num  4.99 24.29 24.49 1020.59 2.29 ...
##  $ LineTotal           : num  4.99 97.18 24.49 3061.78 2.29 ...
##  $ ProductCategoryID   : int  4 2 3 1 4 1 4 2 1 4 ...
##  $ ProductSubcategoryID: chr  "28" "13" "20" "2" ...
##  $ ProductName         : chr  "Water Bottle - 30 oz." "LL Road Pedal" "Half-Finger Gloves, L" "Road-
##  $ ProductNumber       : chr  "WB-H098" "PD-R347" "GL-H102-L" "BK-R79Y-40" ...
##  $ Color               : chr  "NULL" "Silver/Black" "Black" "Yellow" ...
##  $ StandardCost        : num  1.866 17.978 9.159 1082.51 0.857 ...
##  $ ListPrice           : num  4.99 40.49 24.49 1700.99 2.29 ...
##  $ Subcategory         : chr  "Bottles and Cages" "Pedals" "Gloves" "Road Bikes" ...
##  $ ProductCategory     : chr  "Accessories" "Components" "Clothing" "Bikes" ...
##  $ OrderDate           : chr  "04-14-2014" "03-31-2014" "01-16-2014" "07-31-2013" ...
##  $ DiscountDescription : chr  "No Discount" "No Discount" "No Discount" "No Discount" ...
##  $ DiscountType        : chr  "No Discount" "No Discount" "No Discount" "No Discount" ...
##  $ DiscountCategory    : chr  "No Discount" "No Discount" "No Discount" "No Discount" ...
##  $ TerritoryName       : chr  "Northwest" "Northwest" "Northwest" "Northwest" ...
##  $ CountryRegionCode   : chr  "US" "US" "US" "US" ...
##  $ TerritoryGroup      : chr  "North America" "North America" "North America" "North America" ...
##  $ year                : chr  "2014" "2014" "2014" "2013" ...
##  $ month               : chr  "04" "03" "01" "07" ...
##  $ totalprofit         : num  3.12 25.27 15.33 -185.75 1.43 ...
##  $ itemprofit          : num  3.12 6.32 15.33 -61.92 1.43 ...
```

Exploratory Data Analysis

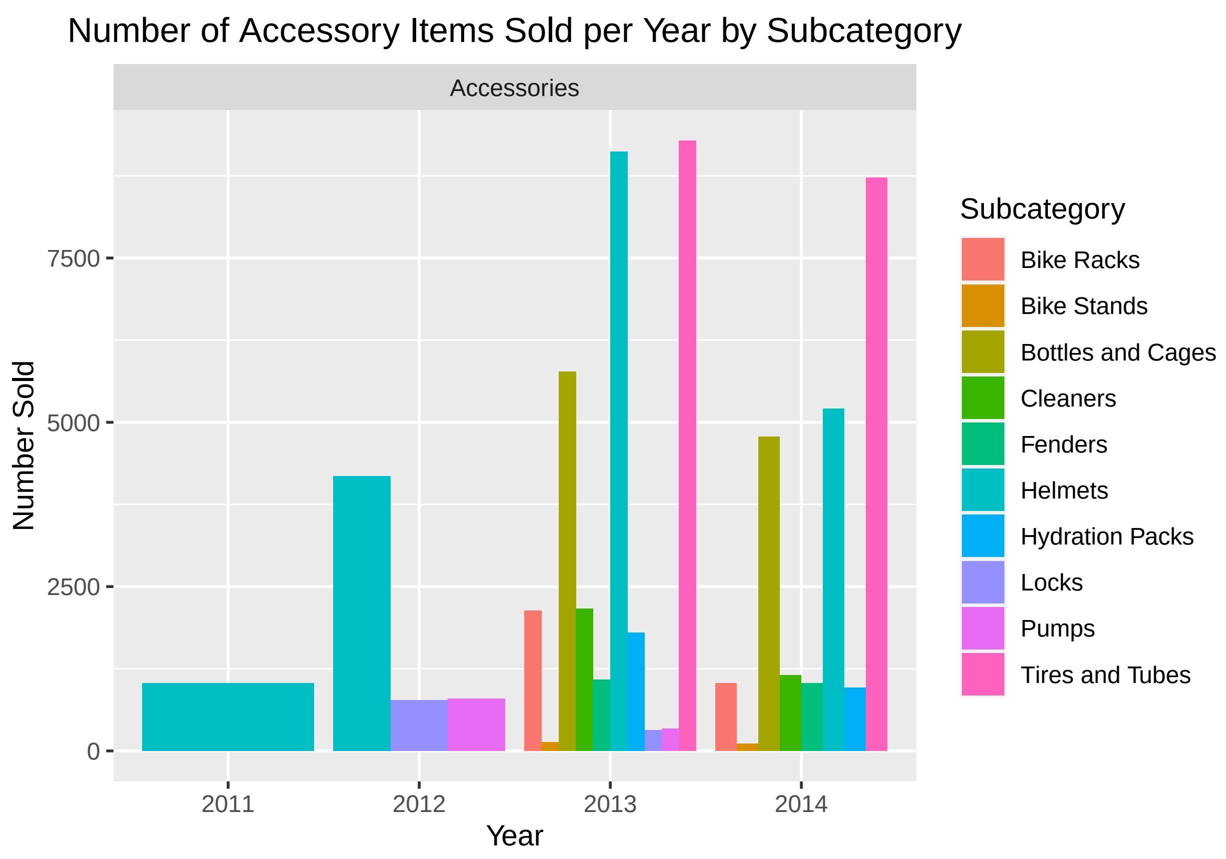
1. Create summary tables for visualizations
    a. Number of Sales by Product, by Year
    b. Number of Sales by Product, by Year and Month
    c. Number of Sales by Product Category, by Year
    d. Number of Sales by Product Category, by Year and Month
    e. Number of Sales by Product Subcategory, by Year
    f. Number of Sales by Product Subcategory, by Year and Month
2. Visualize changes in number sold throughout time by category / subcategory



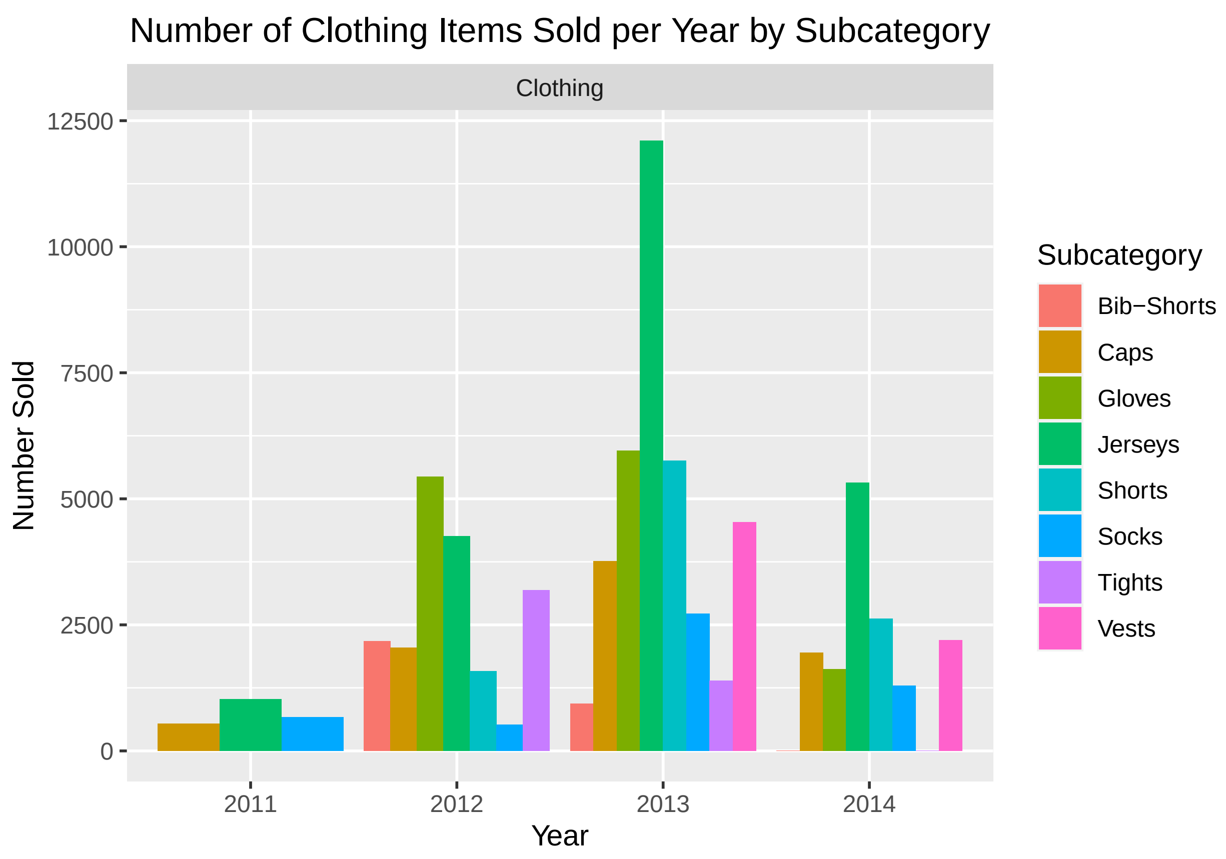
Subcategory	2011	2012	2013	2014
Mountain Bikes	2621	9034	11734	4932
Road Bikes	5342	19460	16698	5696
Touring Bikes	NA	NA	9316	5435



Subcategory	2011	2012	2013	2014
Bottom Brackets	NA	NA	627	294
Brakes	NA	NA	767	268
Chains	NA	NA	533	241
Cranksets	NA	NA	768	339
Derailleurs	NA	NA	804	362
Forks	NA	452	182	NA
Handlebars	NA	1407	2011	532
Headsets	NA	714	295	NA
Mountain Frames	510	3168	5976	1967
Pedals	NA	NA	2774	1157
Road Frames	1137	5564	4277	775
Saddles	NA	NA	1571	574
Touring Frames	NA	NA	2651	1074
Wheels	NA	3802	1471	NA



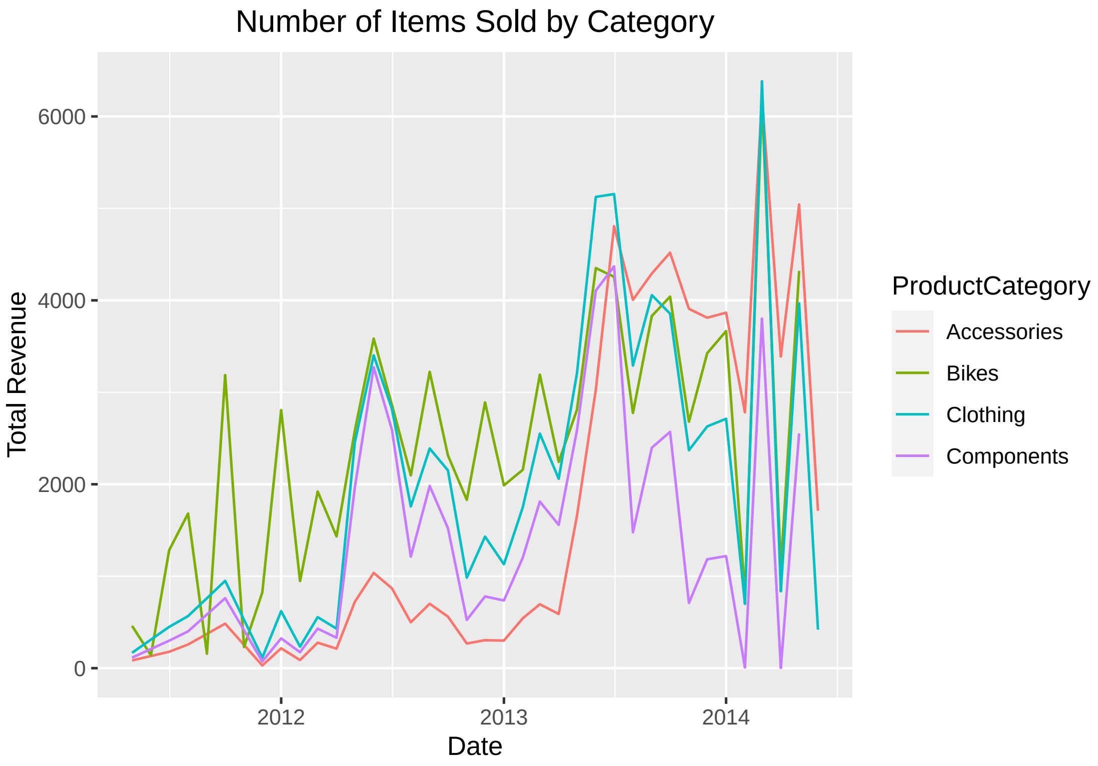
Subcategory	2011	2012	2013	2014
Bike Racks	NA	NA	2137	1029
Bike Stands	NA	NA	136	113
Bottles and Cages	NA	NA	5772	4780
Cleaners	NA	NA	2165	1154
Fenders	NA	NA	1088	1033
Helmets	1032	4184	9116	5209
Hydration Packs	NA	NA	1802	959
Locks	NA	773	314	NA
Pumps	NA	793	337	NA
Tires and Tubes	NA	NA	9286	8720



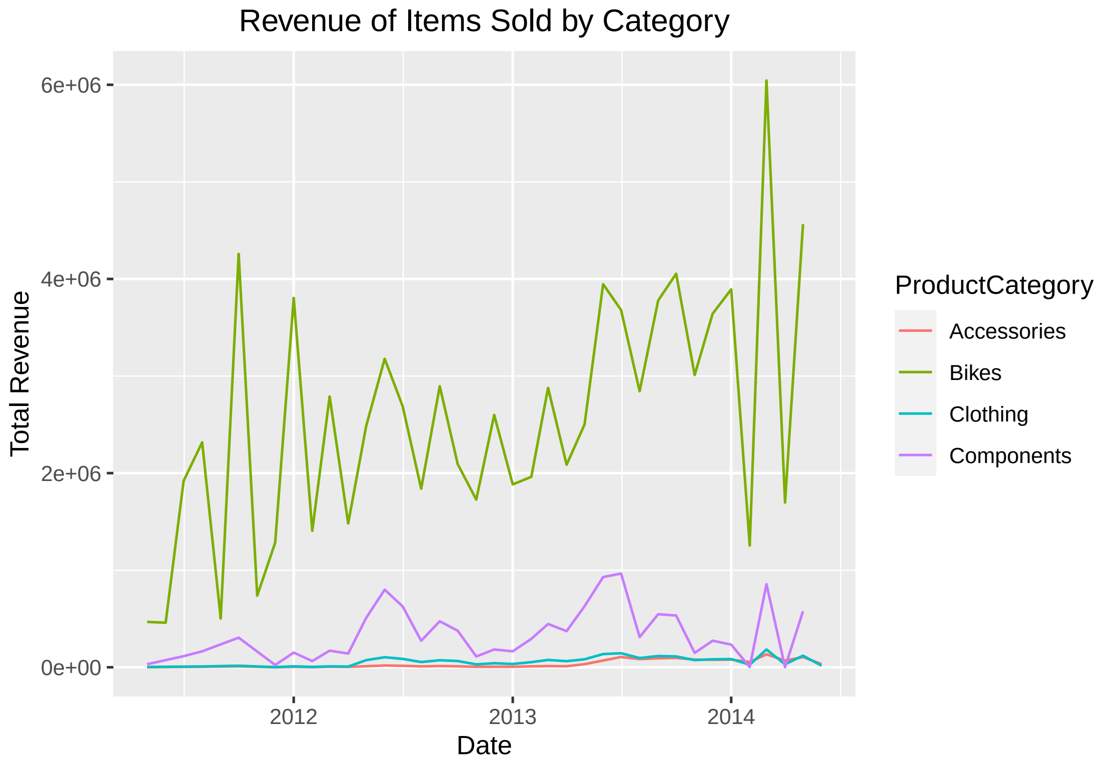
Subcategory	2011	2012	2013	2014
Bib-Shorts	NA	2181	936	8
Caps	545	2048	3768	1950
Gloves	NA	5442	5952	1618
Jerseys	1027	4263	12104	5317
Shorts	NA	1586	5761	2620
Socks	674	523	2724	1296
Tights	NA	3185	1398	6
Vests	NA	NA	4537	2201

Time Series Analysis

1. For basic analysis, visualize the change in number of items sold over time by Category
   a. No Category sells significantly more quantities than the others

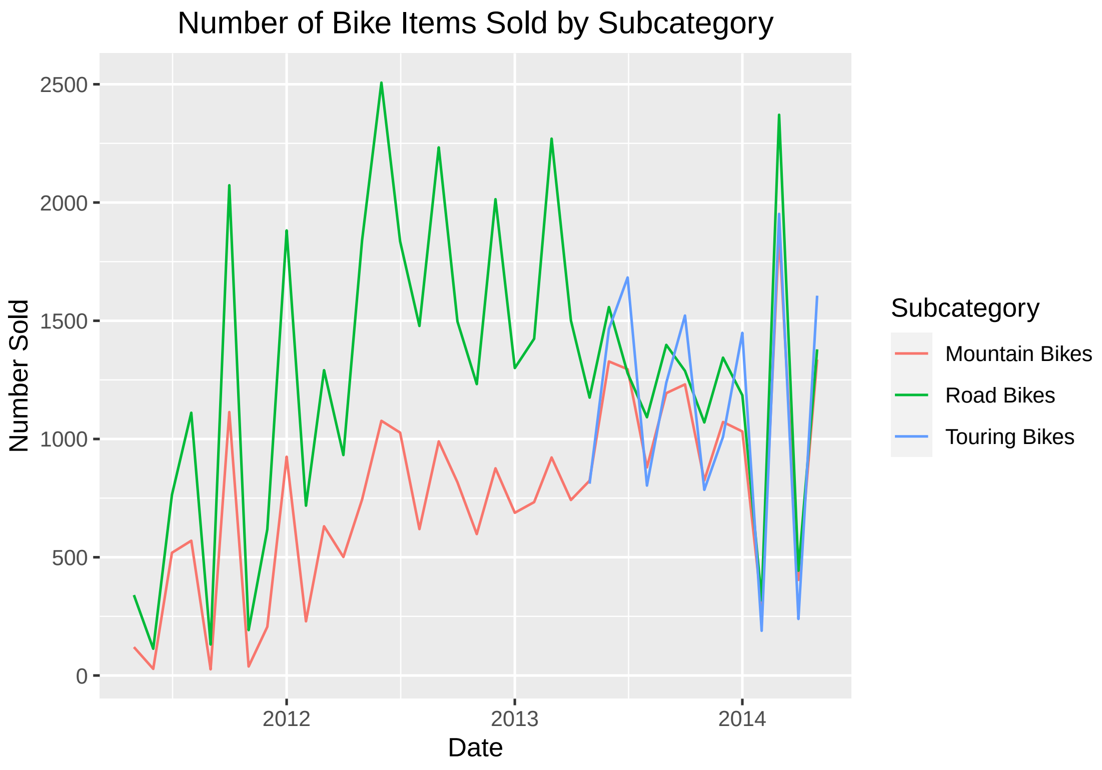


2. To pick a subset to test our regression models, visualize total revenue by Category
    a. The bikes category accounts for significantly more revenue than all other Categories combined
    b. Since this is such a large chunk of the business, the models will start with Bikes

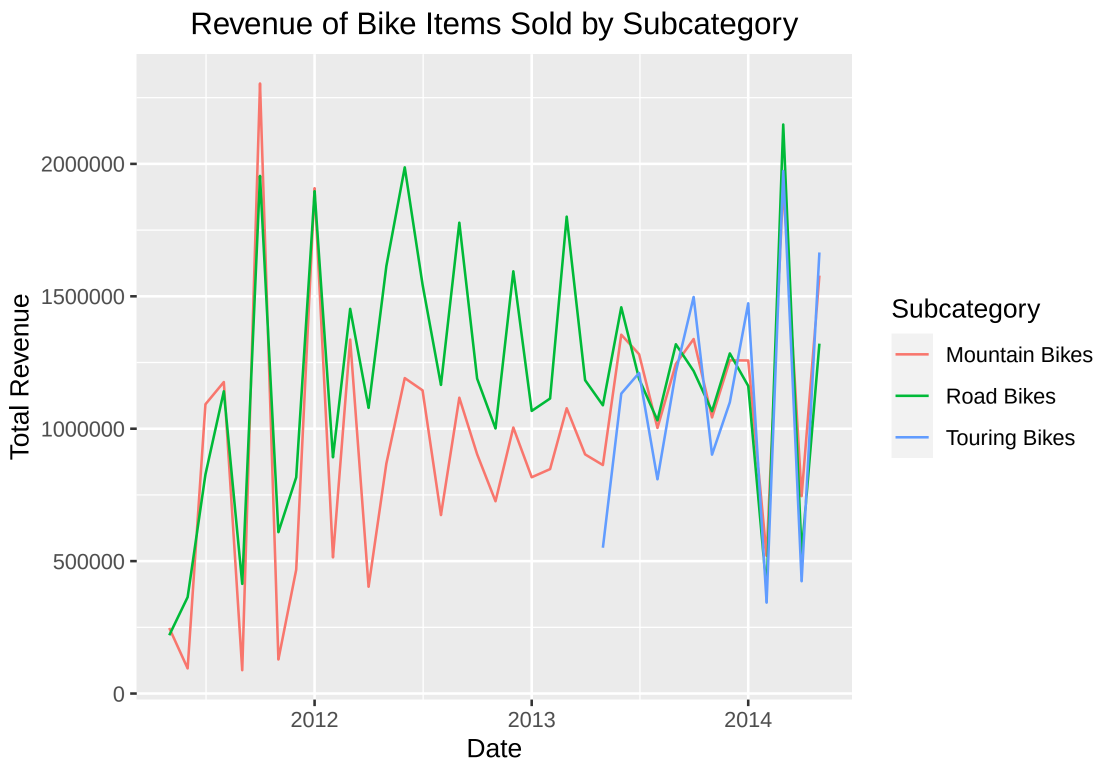


3. Visualize the differences in the types of Bikes sold over time

   a. From mid-2013 on, all subcategories of Bikes are similar in the quantity sold



4. To pick a subset of Bikes to perform regression analysis on, visualize the revenue by Bike type
   a. All subcategories are similar in the amount of revenue
   b. Road bikes have the most complete data and will be the initial test group



Regression Modeling / Forecasting

Using ProductID, year, and month, months_from_start, julian_date, and numeric_date in a linear regression model, we can see that no date columns have no statistically significant outcome on the number of items sold. For this reason, I would suggest using a multivariate time series model, such as ARIMA or exponential smoothing. Dummy binary variables could be created for each of the product ID's to further expand the feature selection of the model. Running these forecasts every month would update the model with the best currently available information. Taking these forecasts, the SafetyStockLevel and ReorderPoint for each item can be updated dynamically based on upcoming needs.

```
##
## Call:
## lm(formula = number_sold ~ ProductID + date + months_from_start +
##     julian_date + numeric_date, data = lm_model_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -77.509 -28.329  -4.402  21.067 177.143
##
## Coefficients: (1 not defined because of singularities)
##                     Estimate Std. Error t value Pr(>|t|)
## (Intercept)        3.631e+04  2.986e+04   1.216    0.224
## ProductID          1.574e-01  2.292e-02   6.866 1.32e-11 ***
## date              -2.410e+00  1.978e+00  -1.218    0.223
## months_from_start  7.328e+01  6.023e+01   1.217    0.224
## julian_date       -6.058e-03  1.300e-02  -0.466    0.641
## numeric_date              NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 38.28 on 806 degrees of freedom
## Multiple R-squared:  0.07754,    Adjusted R-squared:  0.07296
## F-statistic: 16.94 on 4 and 806 DF,  p-value: 2.412e-13
```

Limitations

This analysis does not take into account some basic assumptions about the business and their reordering process. Bulk discounts, for example, may actually save the company money when ordering larger quantities and keeping that excess inventory in the warehouse. It also does not account for the needed optimization between number of orders, inventory cost savings, and delivery costs to factor the best frequency in which to order. To be a truly effective model, it would need to account for order frequency, as well as freight delivery times, to accurately predict how much is needed in inventory for the next ‘x’ months. For example, if the company gets a 20% discount for bulk orders, and the freight for those orders is terribly expensive, it may be more cost effective to ignore the forecasts and stock up on extra inventory less often. Another major assumption made is that items being sold will continue being sold for a decent amount of time. Items that are planned to only be in stock for a short duration will not benefit from forecasting models because of the lack of historical data.

Conclusion

The time series analysis makes it evident the company could benefit from the cost-reductions that a dynamically forecast inventory program could provide. From limiting expansion opportunities to decreasing marketing budgets, excess inventory can make or break a business if not kept in check, small- to medium-businesses especially. From the beginning analysis, a large number of columns had little to no statistically significant effect on the variable(s) in question, and ultimately, to predict for the future. Another thing that was overestimated was the sheer volume of different products, some of which were no longer being sold and some of which had just recently began being sold, which is why analysis was broken down to the most profitable subcategory of the most profitable category.