

Abstract

An exploration of a dataset consisting of data drawn from the World Of Warcraft in game auction house, to explore if inflation has occurred within the virtual environment. The data was gathered and compiled for the purpose of trying to answer this question. The write up including the loading, cleaning and transformation of data before being loaded into WEKA. Along with detailed steps of secondary cleaning that was required for the dataset. Limited analysis was done with many issues occurring within WEKA in relation to its programming and the dataset provided. Along with suggestions to solutions and alternatives analysis tools.

Contents

Abstract.....	0
Introduction – What is Data Mining?.....	2
The steps of Data Mining	2
Time Series Analysis.....	2
About the Data.....	2
Why is it interesting?	2
Analysis Outcomes	3
Where is this data from?	3
Primary Data Source	3
Secondary Data Source	3
How is the data stored and manipulated	3
Data Processing and Cleaning.....	4
Initial Data Issues	4
Data Processing and Handling	4
Data Filtering.....	4
Initial Cleaning and transformation	5
Dataset Questions.....	6
WEKA and the Data.....	7
MySQL Connection.....	7
First Visualisation	7
Initial Time Series Analysis.....	8
Secondary data cleaning and transformation.....	9
Time Series – Take 2	11
Small Data Set	11
Full Data Set	12
Limited Data Analysis.....	13
Testing filters.....	16
WEKA Issues	16
Processing and Memory limitations	16
Alternatives to WEKA.....	17
Data Findings.....	17
Issues with the Data.....	17
Remedies to Issues.....	17
Dataset Questions.....	17
Conclusion.....	18

Introduction – What is Data Mining?

Data Mining is an analytical process that is used to explore large amounts of data. Typically used in Business it is used to look for patterns and potential relationships within the data. The main use of data mining is for analysts to take existing and meaningful data and potentially gain new knowledge and predict future trends and patterns. (1)

The steps of Data Mining

Data Mining consists of three main steps. Exploring the data, Modelling and validating, and lastly Deploying that model on to new data to generate predictions.

Exploration of data though is not the first step. Within exploration of the data, the gathering and cleaning of data is actually the first step and is the most crucial of all the steps. Unclean data may have inaccuracies within it and as a result produce inaccurate models for later prediction.

Modelling and validation involves trying to build a model that accurately represents the data you are looking to predict. A model maybe built around trying to answer a particular question you have about the data. Whilst validation looks at applying that model to other sample or test sets of data you may have to try and validate that the model is answering the questions being asked.

Lastly deploying the model is the application of the validated model on to new data coming in and using it to make accurate predictions of future data.

Time Series Analysis

Time series analysis is a subset of data mining and looks at extrapolating and understanding data that has a dimension of time applied to it. Trends within this kind of analysis may have outside influences applied to the data, such as day of the week or time of year, as examples. This is a particular favourite analysis method of most businesses as it allows for predictions of stock requirements at particular times of the year. (2) The only limitation to using Time Series Analysis within WEKA is the version. For this analysis WEKA version 3.7.12 was used.

About the Data

The dataset being worked with for this project was a compilation of data from the in-game auction house from the popular and long running massively multiplayer game, World of Warcraft (WoW). WoW is as stated is an online game with an interesting currency system for purchasing and selling of items within the virtual world. With a long standing economy, now passing ten years of operation, some interesting effects have occurred to the in-game currency and market place.

Why is it interesting?

Inflation in economics is a sustained increase in the price of goods and services over time. (3) So why does inflation occur? As a greatly discussed topic in both the real world economy and the WoW economy, no one really knows. There are two main types of inflation Demand-Pull and Cost-Push. Cost-Push is where product prices are increased to maintain profit margins. However the far more likely issue with WoW is Demand-Pull, this is where there is too much money trying to purchase too few goods. As with a game of this nature, money can be obtained through completion of in-game tasks, selling items and looting it from other characters or enemies. With too few goods possible to purchase but plenty of ways to obtain money rapidly, prices for items are going to increase.

The purpose of this dataset is to try and visualise this inflation and where it is occurring. Along with this the data shall try and visualise pricing fluctuations due to time changes. Do prices increase on specific days or specific times?

Analysis Outcomes

What could someone do with the potential results obtained from this analysis? Primarily it could be used to try and target the areas greatly affected by inflation. Although it could also be used by other players to increase inflation further. They could target trading, buying and selling of specific items at certain specific times. The data could potentially allow for someone to increase their monetary gains rapidly. However, at the cost of potentially breaking the economic system entirely if mass amounts of this type of trading occurs. Can a dataset bring about economy collapse?

Where is this data from?

Primary Data Source

The dataset is a compilation of data exported from the WoW in-game auction house. The data is collected from a third party website called WoWuction (4). The data has had to of been compiled over several weeks as the source website only exports data from the website for the previous 24 hours. Each table of information from each day contains the item's unique identifier along with its name. Also included is the price average of that item that day in several formats and how many were available. This data was also combined with another information source to add the category of item to the dataset as well.

Secondary Data Source

The secondary data source comes from Blizzard, WoW's developing company, public application program interface (API) website (5). The site is simply an online resource that allows for third parties or individuals to pull data about any of the game's aspect into either their own websites or down into separate databases. This API is primarily how the WoWuction website gathers its information about the auction houses. The API could have been used to collect the desired data, however seeing as WoWuction was already collecting and presenting the data without a need of a specialised application the data was taken from them. The data that was gathered directly using the API was in-game item data. The three fields that were of interest from within the data were the Item's ID, Item category and the item sub-category. The item ID was used to tie this second data set to the primary set. The category and sub-category data allows for the data to be classified into main item groups, for example Armour, Consumables, Weapons; with the sub-category allowing for further identification. For example in the Armour category there are sub classes of Cloth, Leather, Mail, Plate etc. The required level was also pulled out, this indicates the minimum level the player has to be to use the item. Most consumable items do not have a required level whilst nearly all armour and weapons do. One of the main issues with using the API it turns out there is a limit to the number of polls that can be made to API on each day from a single IP address. As a result the data was also collected over several days.

How is the data stored and manipulated

The data from the primary source was delivered as a CSV file. Although the data could have been processed easily using Microsoft's Excel package, the data was then imported to a MySQL database. This was due to restrictions of handling the data from the secondary source. As the data from the secondary source is gathered in a JSON format. An automated program to poll the Blizzard API was

built which wrote each requested item to an individual JSON file. Which was later then re-opened and had the fields of interest pulled out and written to the MySQL database. The data extracted from these JSON files was the Item ID, Category of Item, the Sub-Category and lastly the items required level.

Data Processing and Cleaning

Data processing and cleansing is important to any data analytics for making sure that no unclean or invalid data makes its way into the analytical tool. Cleaning the data can also be used to refine the data, making it easier to build a model of the data.

Initial Data Issues

Immediate issues with the data gathered is with the primary data coming from WoW auction house. The main issue is that not every single item sells every day creating missing data. Along with that if an item does not sell, but is on the auction house it is still listed in the output data, but with a zero value. This produces some issues with how much of the primary data can be analysed.

Data Processing and Handling

To effectively clean the data, as it is stored in several spread sheets (one for each day) the data was imported into a single database using a MySQL backend. Getting the data into one location so it can be managed is key to more effectively identifying issues with the data. It also allows for faster execution of filtering queries and merging multiple sources. Other advantages to using MySQL are that fact it is not bound to a particular operating system. In this setup it was run on a server with a phpMyAdmin web-based frontend. This allowed for easy access to the data from any machine and location regardless of its specification and operating system. The majority of all the data cleaning was done on the server before being exported to CSV format.

Data Filtering

The original data from both sources have multiple records with not all being relevant to this particular analysis and the questions being asked. Within the primary data source several columns that might be relevant to other data analysis were dropped this case. The realm (game server) that the data is drawn from is the first, in this case comparing item values over several different realms was not being done, this was the first to be dropped. The data and time of each data export is also available in the primary data, although the data when being imported to MySQL was being stored in tables that were named by the date that the data was collected. Making the data and time redundant fields in this case.

The quantity of each item passing through the auction house is also listed. The value at which an item sold within the data is an average of all that items differing value sold for that day, making the quantity field redundant. However it could be used to look for scarcity of items coming through the auction house. That particular question is not really a data mining question and more just analytics.

The primary also carries some other auto-generated calculations from WoWuction in regards to the items value over the past fourteen days. The issue with this data is that items that did not sell on a particular day, their zero value was still used in the calculation of the average. Making the data unreliable.

The secondary data as stated previously comes directly from the Blizzard and is actual in-game data of each item. Each JSON file contains all of the raw attributes that can be seen in game. The majority of the fields were not of interest with only the ItemID, Item category, item sub-category and the required level for each item being kept. Other values stored that might be of interest include weapons and armour stats, item descriptions and item level. The item description is unfortunately

an un-mineable field within time series analysis as does it not change. The Item level also does not change over time but could be used in conjunction with item description to look for word traits the higher an item level.

The item level could also easily replace the required level as a means of examining if the market place favours higher level items. Although as the higher level items tend to require higher required levels, the two are interchangeable resulting in both not being required.

Initial Cleaning and transformation

At the time of starting analysis and getting data into WEKA initially, it was unknown how much cleaning would be required to the data. Testing this the data was only briefly cleaned at first to retain as much information as possible.

The first cleansing performed was to check that all the data had been merged correctly as MySQL merge function operates without data type checks upon migrating data. Resulting in it simply joining data regardless of type.

	A	B	C	D	E	F	G	H	I
1	Item_ID	cat	subcat	Feb-23	Feb-24	Feb-25	Feb-26	Feb-27	Feb-28
2	9317	0	8	White an	0	0	0	0	0
3	118426	0	8	the Jade \$	the Jade \$	the Jade \$	the Jade \$	the Jade \$	the Jade \$
4	94592	2	8	Reborn	Reborn	Reborn	Reborn	Reborn	Reborn
5	94587	2	7	Reborn	Reborn	Reborn	Reborn	Reborn	Reborn
6	94589	2	7	Reborn	Reborn	Reborn	Reborn	Reborn	Reborn
7	94582	2	5	Reborn	Reborn	Reborn	Reborn	Reborn	Reborn
8	94583	2	5	Reborn	Reborn	Reborn	Reborn	Reborn	Reborn
9	94586	2	5	Reborn	Reborn	Reborn	Reborn	Reborn	Reborn
10	94585	2	4	Reborn	Reborn	Reborn	Reborn	Reborn	Reborn
11	94576	2	1	Reborn	Reborn	Reborn	Reborn	Reborn	Reborn
12	94580	2	1	Reborn	Reborn	Reborn	Reborn	0	0
13	94577	2	0	Reborn	Reborn	Reborn	Reborn	Reborn	Reborn
14	86607	0	8	Mark II	Mark II	Mark II	Mark II	Mark II	Mark II
15	40110	0	0	5E+09	0	5E+08	5E+09	1.68E+09	1.68E+09
16	11203	9	8	2E+09	1E+08	2E+09	2E+09	2E+09	2E+09
17	4597	9	6	2E+09	2E+09	2E+09	2E+09	0	2E+09
18	9296	9	6	2E+09	2E+09	2E+09	2E+08	2E+08	5.73E+08
19	13495	9	6	2E+09	2E+09	2E+09	2E+09	0	1.42E+08

Fig.1 – Problem data.

As can be seen above several items in the list had String values where integer values should have been. The String in each of the items above case was the either the item name or part of its description. This issue was not a MySQL issues with these errors actually stemming from the primary source data. The primary data was checked to see if the 13 affected items data could be recovered, however there error was from exporting the data from the initial source. As shown below.

	94574	Pygmy Direhorn		0	0	0	
	94575	The Planar Edge	Reborn	46498650	1	464	
	94576	Lunar Crescent	Reborn	40937899	1	40	
	94577	Black Planar Edge	Reborn	72918600	1	72	
	94578	Mooncleaver	Reborn	0	0		
	94579	Wicked Edge of the Pl	Reborn	1.21E+08	2	121	
	94581	Drakefist Hammer	Reborn	25000000	1		
	94582	Thunder	Reborn	48068543	1	480	
	94585	Dragonstrike	Reborn	1.74E+08	2		
	94586	Stormherald	Reborn	1.21E+08	3	121	
	94587	Fireguard	Reborn	40000000	3		
	94591	Blazefury	Reborn	37397891	1	373	
	94592	Lionheart Executioner	Reborn	1.49E+08	1	149	
	94595	Spawn of G'nathus		5949999	4	594.9999	594

Fig. 2 – Problem data in source.

Along with checking the consistency of the results some results that had the majority of zero (non-existent) records were removed. Some examples are shown below.

1391	2	10	10054200	10054200	10054200	0	10054200	10054200	0	0	0	10054200	10054200	10054200	10054200	0	0	0	0
1413	2	7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1414	2	5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1425	4	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1429	4	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1443	4	0	0	0	0	1950000	4714200	1032500	1000000	0	0	850000	905950	2950000	684300	0	0	770000	75000

Fig. 3 – Examples of removed records.

A solution to getting these null results would have been to gather more primary data. The current set was only from one realm within the game, however if several data worth of realms had been collected, an average could have been drawn out from them all to lessen the number of null values.

Dataset Questions

As the dataset is built around time series and looking for fluctuations within the data, the questions are biased in that direction.

- Is there inflation within the WoW economy?
- Are there pricing fluctuations based on date?
- Do items of particular category sell for more?
- Is one group of players going to favour using the auction house?

The first question is entirely what made the dataset the way it is, due to the targeting of looking for the answer to this question. This raised another question however. Due to the dataset being built specifically to answer this question, will it still be able to answer the question?

Pricing fluctuations such as the time of the week will be looked for, to see if any game playing patterns can be seen within the data. Along with answering the question of when is best to sell items.

Items within particular categories will look for an answer to if an item is worth selling if they are within certain item categories at the auction house. For example, do particular items sell for more?

Lastly looking for the player levels who favour using the auction has a few issues, mainly presumptions. Items within the game have a required level, many common items do not have a required level but many specialised weapons and armours do. If it can be presumed that players are buying items as soon as they can, then we can look for trends based upon the level grouping of players using the auction house.

WEKA and the Data

The first step to using WEKA is getting the data in. WEKA accepts several different file formats, but first a different data input method was tried.

MySQL Connection

WEKA has support for directly connecting it with a MySQL database. Seeing as the data was being stored in a MySQL database, it was worth an attempt at getting it to work. Mainly because this would save having to export the data from the MySQL database to CSV or ARFF format.

However, it turned out that the driver packaged with WEKA is only suitable for connecting WEKA to a Microsoft SQL database. Despite changing the driver to one that was designed to allow Java based applications to connect with MySQL databases, WEKA kept on refusing to connect with the database with it continually throwing the error of not being able to find the remote server which the database was on. As a result data was then exported to CSV for use with WEKA.

First Visualisation

After importing data into WEKA for the first time, the data was looked at initially visually to see any patterns or trends could be seen.

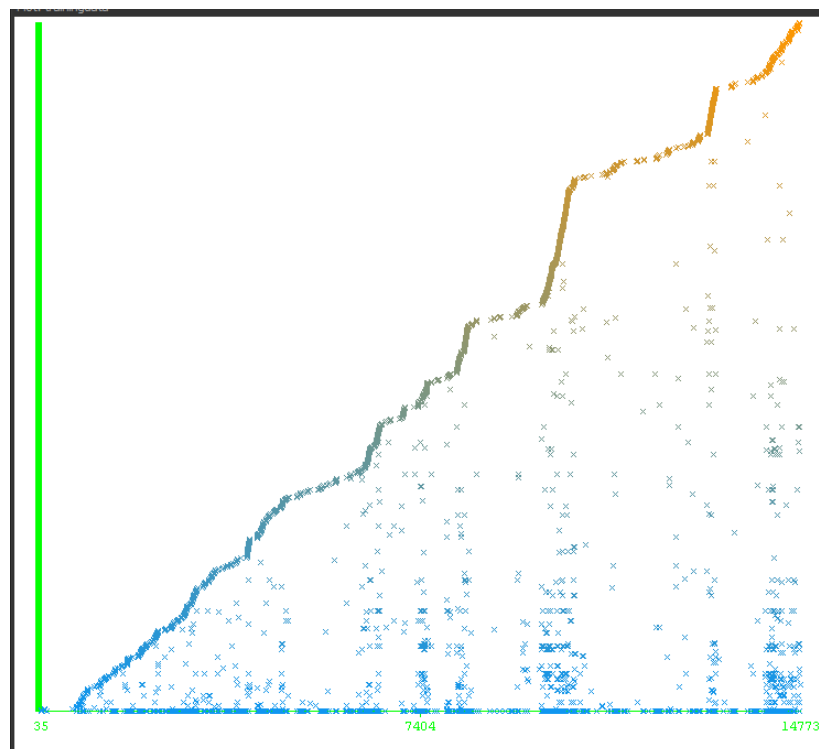


Fig. 4 – Data Visualisation

The above image shows the visualisation of item number (along the x-axis) and the price sold for (along the Y-axis) with this data coming from the first day in the database.

Immediately what can be seen is items with higher ID numbers, which were added later in development and the games life, have higher values. This shows that later content produced for the game was target towards keeping existing players playing.

Also it can be seen quite clearly the number of zero values within each date.

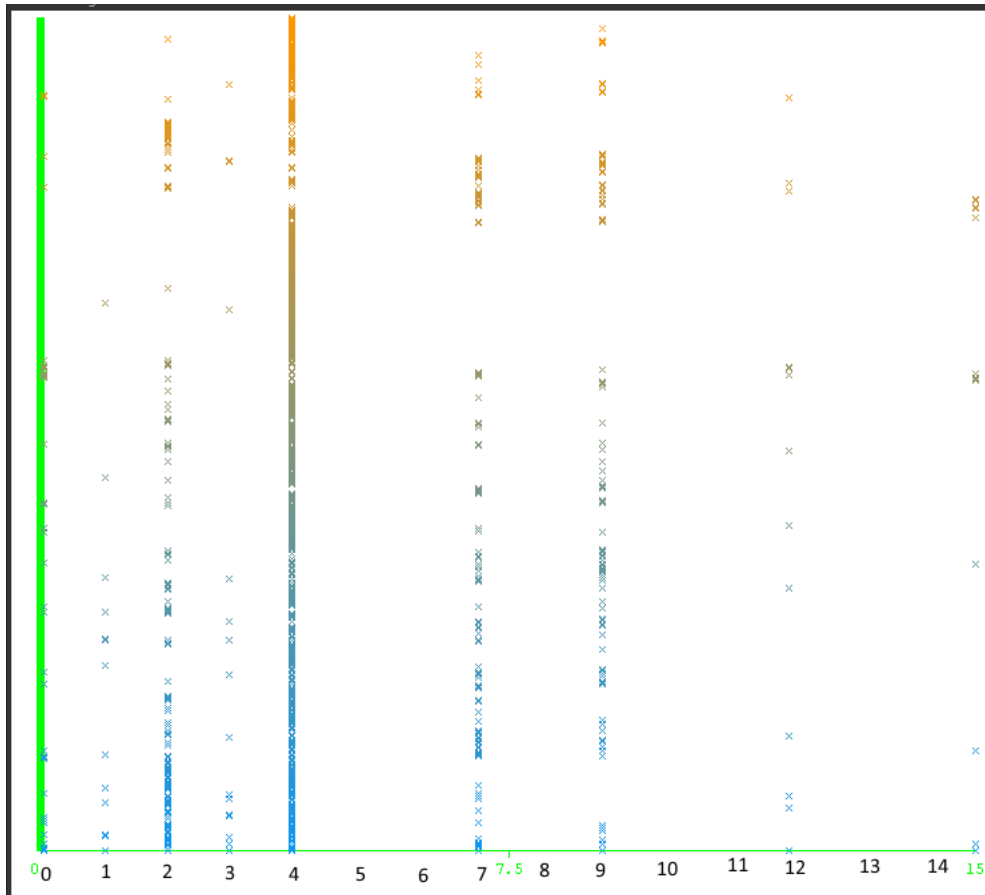


Fig. 5 – Item Category against Price.

The above graph shows the distribution of prices of items categorised into the items category. As can be seen there are no items for sale in select categories, categories 5 and 6 (Reagent and Projectiles) are craft able only items and are not sell-able in the auction house. Whilst item categories 8, 10, 11, 13 and 14 are place holders and no longer actually carry any items. Items within these categories have likely been either removed or migrated to other categories over time, whilst the numeric category reference still exists to prevent game engine complications.

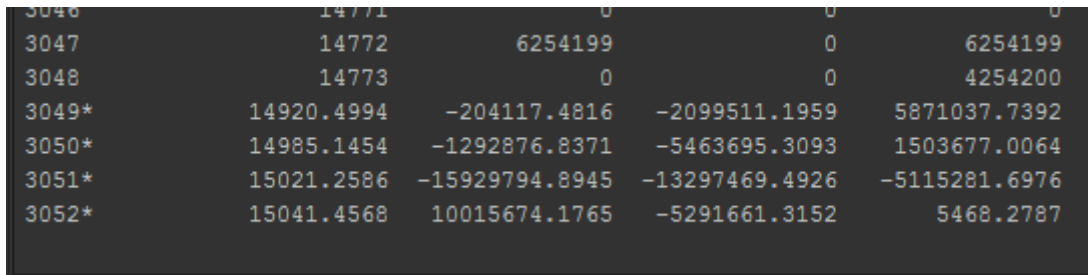
Category number 4 has the most items for sale within it. Category 4 being armour has the widest consistent spread of values across it.

Initial Time Series Analysis

To first use time series analysis within WEKA two packages must first be added via the in-built Package Manager. The 'timeSeriesFilters' and 'timeseriesForecasting' (6). Once installed additional options will be available within 'Classifier' options along with a new 'Forecast' tab at the top of the main window.

Quickly running a set of data through the Forecasting tool however, revealed several issues with the data currently being presented.

Firstly was some very odd negative figures appearing but most importantly that the time series analysis tool works using rows not columns. How the data was setup was that there was a column for the 'ItemID', 'ItemCategory', 'ItemSubCat' along with a column for each days worth of data. Whilst how it needs to be setup is the opposite with a time identifier in one column, in this case our days and the itemID along the top as column headers.



The screenshot shows a table with 6 columns. The first column contains item IDs (3046, 3047, 3048, 3049*, 3050*, 3051*, 3052*). The next three columns contain item category and sub-category information (14771, 14772, 14773). The final two columns contain numerical values, some of which are negative, indicating forecasting issues. The table is displayed on a dark background with light-colored text.

3046	14771	0	0	0
3047	14772	6254199	0	6254199
3048	14773	0	0	4254200
3049*	14920.4994	-204117.4816	-2099511.1959	5871037.7392
3050*	14985.1454	-1292876.8371	-5463695.3093	1503677.0064
3051*	15021.2586	-15929794.8945	-13297469.4926	-5115281.6976
3052*	15041.4568	10015674.1765	-5291661.3152	5468.2787

Fig. 6 – Forecasting issues.

Secondary data cleaning and transformation

Due to issues within the data, secondary cleaning was undertaken. This time round, due to negative values appearing, which is technically impossible as none selling items produce zero or null values not negative values. As a result any record with a zero value was removed. This cut the dataset from 13,332 records long to just 1,305. A much more manage-able size.

The transformation that needed to occur was the change from column based data to row based data, along with standard Java dates being added to the database instead of the current placeholders. Due to this shift the Category, Sub-Category and Required Level columns would need to be dropped.

As the data was already being handled additionally in a CSV format, Excel was used to transform the data from the current state to the new desired format.

This can be achieved simply by copying the data you require and selecting to 'Paste Special' (see Fig.7)

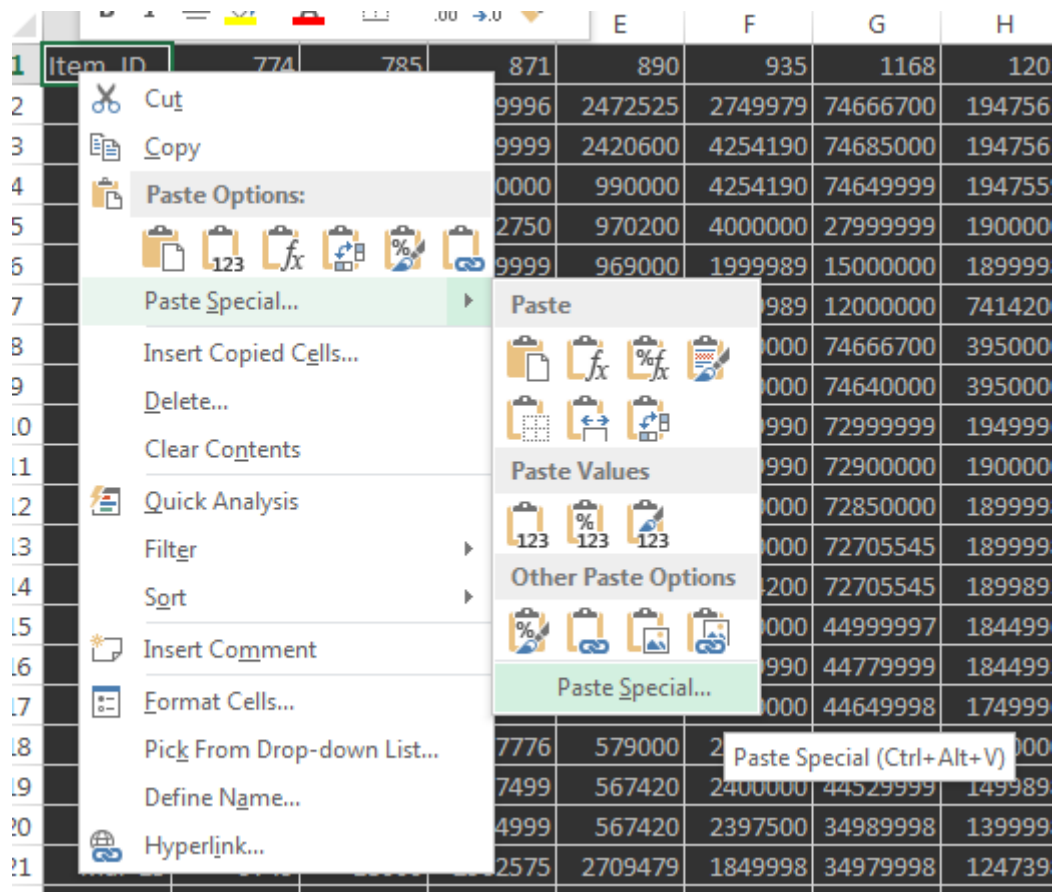


Fig. 7 – Paste Special

From here this produces a new dialogue box, in which the Transpose option is selected (see Fig.8) This results in all the data being transformed or flipped, with rows becoming columns and columns becoming rows.

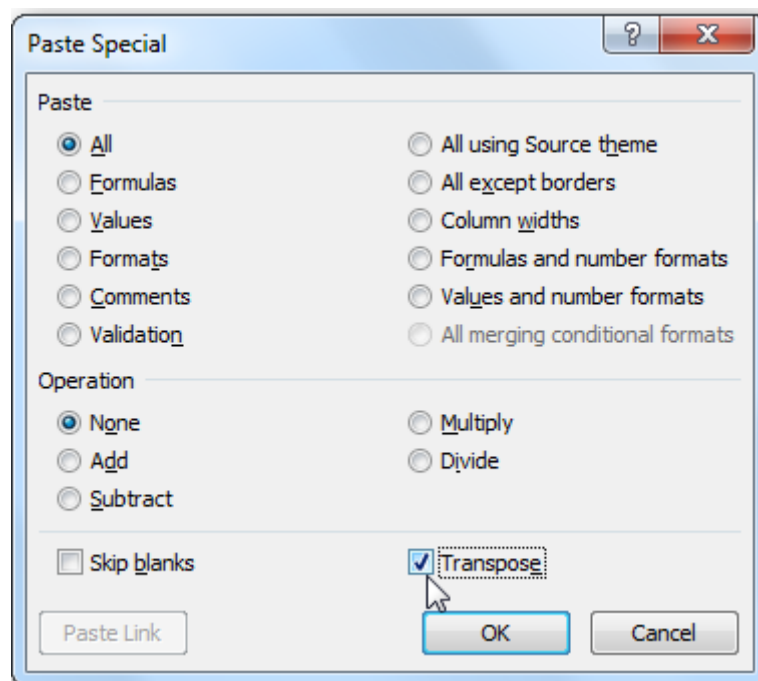


Fig. 8 – Excel Transpose option.

Doing this made the data look as follows, with only the dates needing to be switching to standard Java based dates.

Item ID	774	785	871	890	935	1168	1203
Feb-23	9673	14000	5999996	2472525	2749979	74666700	1947561
Feb-24	8677	6273	5989999	2420600	4254190	74685000	1947561
Feb-25	8316	12400	5990000	990000	4254190	74649999	1947559
Feb-26	7603	11212	5722750	970200	4000000	27999999	1900000
Feb-27	50000	10800	3899999	969000	1999989	15000000	1899998
Feb-28	12399	13800	3899999	3057864	1999989	12000000	7414200
Mar-01	9800	13800	3895000	1000000	4000000	74666700	3950000
Mar-02	14334	9900	3894999	1000000	4000000	74640000	3950000
Mar-03	7000	9499	2900000	1000000	2749990	72999999	1949996
Mar-04	9743	7995	2825062	2175000	2749990	72900000	1900000
Mar-05	12455	5500	2822500	975000	3000000	72850000	1899998

Fig. 9 – Transformed Data without dates.

Time Series – Take 2

Small Data Set

With the data now in the correct format with as many issues with data removed, it was time to see if the Forecasting tool would work.

Firstly the parameters, using a small sample of the data (eight items in total) the chosen parameters were as followed.

The screenshot shows the 'Forecast' tab of a software interface. Under 'Basic configuration', the 'Target Selection' section has four buttons: 'All', 'None', 'Invert', and 'Pattern'. Below these is a table with columns 'No.' and 'Name'. The table contains five rows with checkboxes in the 'No.' column and item IDs in the 'Name' column: 1 [checked] 774, 2 [checked] 785, 3 [checked] 871, 4 [checked] 890, and 5 [checked] 935. To the right, the 'Parameters' section includes: 'Number of time units to forecast' (1), 'Time stamp' (Date), 'Periodicity' (Daily), 'Skip list' (empty), 'Confidence intervals' (unchecked), 'Level (%)' (95), and 'Perform evaluation' (unchecked).

Fig. 10 – Parameters chosen.

However trying to use the Date column within the data threw an error, with the forecaster complaining about 'Priming instances do not appear'. This was believed to be a date format issue, but this did not seem to be the problem. So the 'Time Stamp' variable was changed to '<Use an artificial time index>'. Using this parameter did not throw any errors upon trying to perform the forecasting, however produced some issues with the results. (See Fig.11)

36	9200	25000	1897498	2406710	850000	35000000	7414200
37	9743	7995	2825062	2175000	2749990	72900000	1900000
38	4800	25000	1747500	490000	850000	49999996	1967257
39*	-3709.5741	29995.2608	3671308.7932	-2061932.433	-2454401.4879	94188647.6797	-2104892.5922

Fig. 11 – More data issues.

This time more negative values appeared, which as mentioned previously should not occur nor should be occurring with all zero value records being dropped. The 'Time Stamp' was finally changed to '<none>' and the forecasting re-run. Success! No negative values appeared with the predicted values actually looking like they might fit with the existing data!

32	7599	14000	1897500	849999
33	9214	14000	3490000	495000
34	9800	14000	2227500	727649
35	9819	54230	2229999	693000
36	9200	25000	1897498	2406710
37	9743	7995	2825062	2175000
38	4800	25000	1747500	490000
39*	11525.2632	12727.3947	2825519.5263	1258383.8158

Fig. 12 – Success

Upon successful completion of forecasting the data, a new tab is presented named 'Train Future Pred.'. This new tab shows a graph of the existing data followed by the predictions.

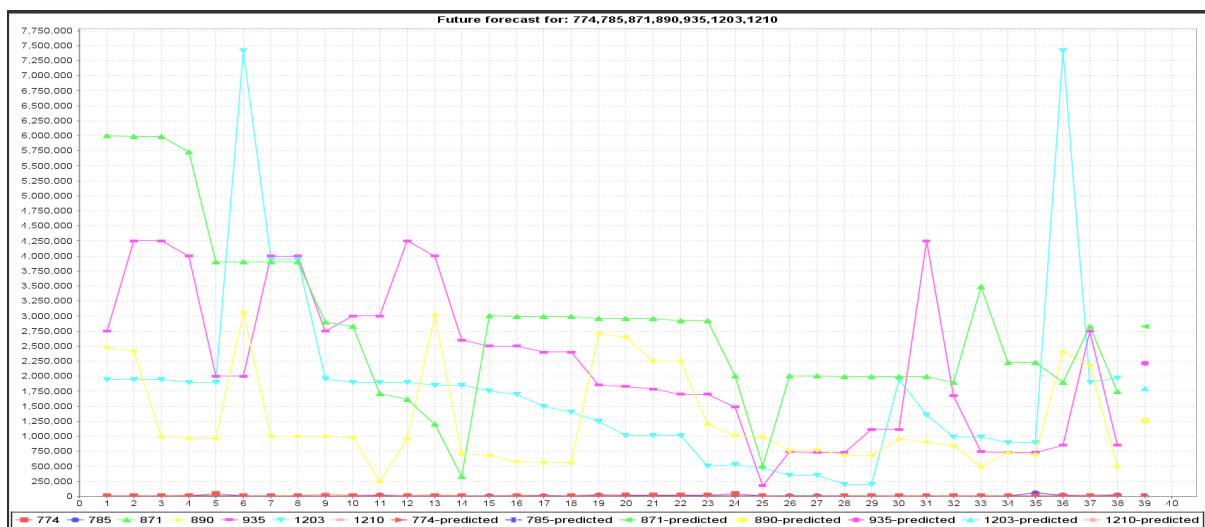


Fig. 13 – Graphing Forecast 1

The data shows the first hints of some time series analysis, with WEKA plotting the data into graph format. A major issues was apparent with this data as one record was omitted from the graph above. Higher random values were making the graph hard to determine any meaningful data. With Fig. 13 items 774 and 785 have low values in comparison to the others and as a result flat line across the X-axis.

Full Data Set

After playing with the data a bit and looking at options available and how they affect the graphing tool, more data was loaded into WEKA to see more of the data. Upon trying to forecast more of the

data, WEKA throws an issue with running out of memory (Fig.14). As WEKA runs on Java, this can simply be remedied by increasing the Heap Size of Java, as suggested by the error debugger.

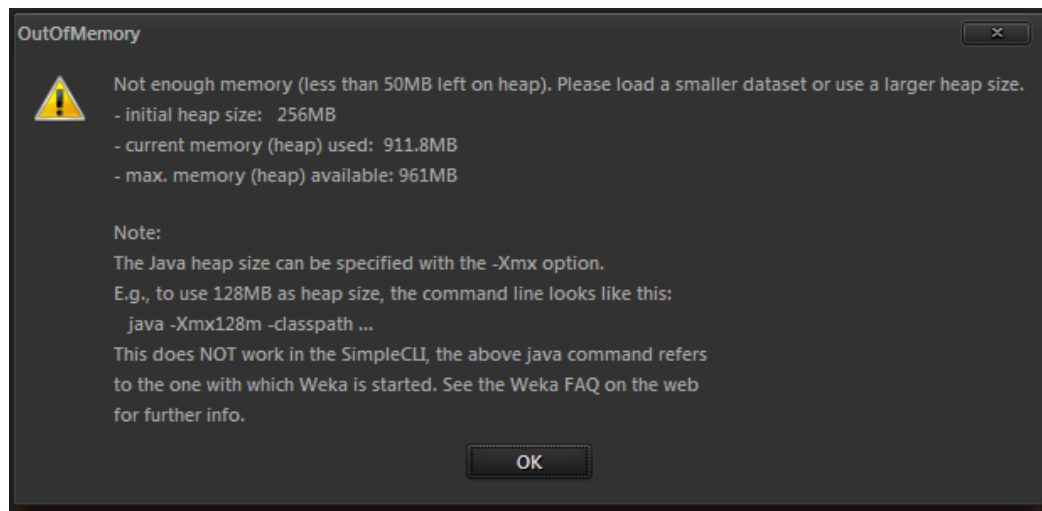


Fig. 14 – WEKA error.

Upon re-launching WEKA using an increased Heap Size set to 4GB. The data was reloaded with basic parameters and only one day to predict, however upon trying to run the data back through the forecaster, WEKA locks up whilst producing no errors. However the console revealed that Java had silently run out of memory, again.

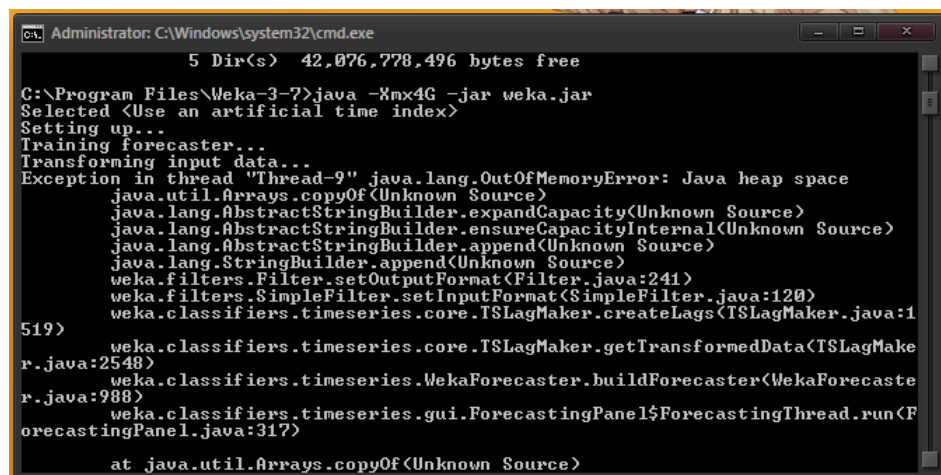


Fig. 15 – More WEKA errors.

Despite a second attempt at assigning almost all the machines RAM to use with WEKA (14GB in total). WEKA continued to crash with complaints of running out of RAM. There appears to be some major issues with the Time Series Forecasting tools and Filters that are available for WEKA, which are likely causing some RAM leaking within the program.

After attempting to find other people who may be experiencing similar issues, nothing came up. So further analysis of the data is extremely limited to small samples of data.

Limited Data Analysis

Although some data analysis can be done visually just by looking at the graphs that WEKA plots, such as the slight downward trend of certain items in Fig. 16. The Time Series tool for WEKA is

primarily built for forecasting future predictions. As WEKA was being a little less than stable, the future prediction tool was also limited. However was still functional enough to do short term predictions.

Firstly, running the standard configuration of the time series forecasting tool utilises a LinearRegression for the Base Learner Configuration. Running this in a standard configuration does predict results however it produces a straight line, with no alteration within the predicted vales.

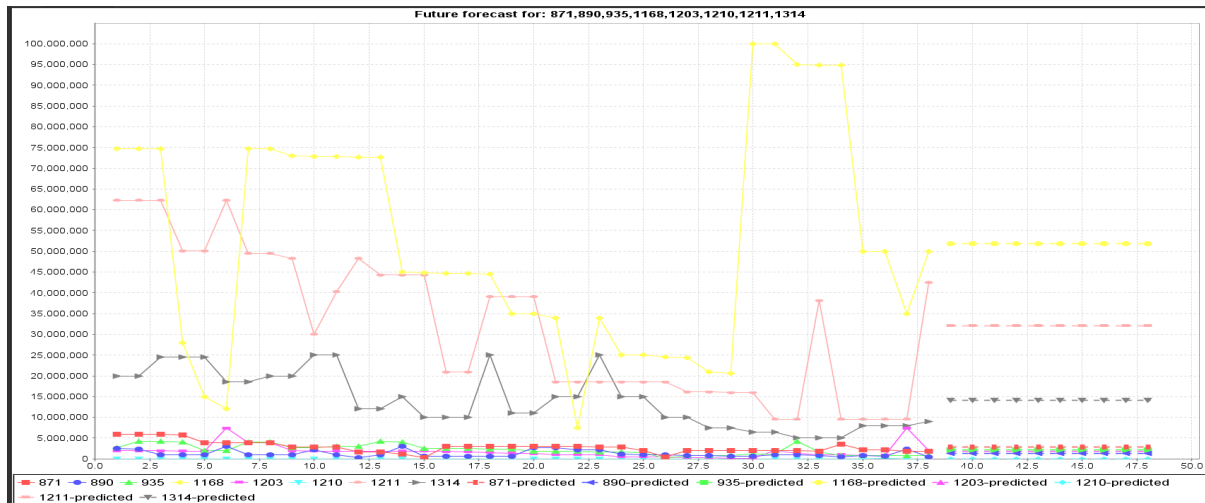


Fig. 16 – Time Series Forecasting of ten values using linear regression.

Changing any of the parameters yielded the same results with a flat line prediction. This would indicate that the Linear Regression filter is not designed to analyse and predict time series data.

So instead of using the linear regression filter, the additionally installed time series filter was utilised instead with hopes that they would show some time series like patterns. The filter is named internally in WEKA as 'HOLTWINTERS' and has various parameters that can be tweaked to aid in data forecasting.

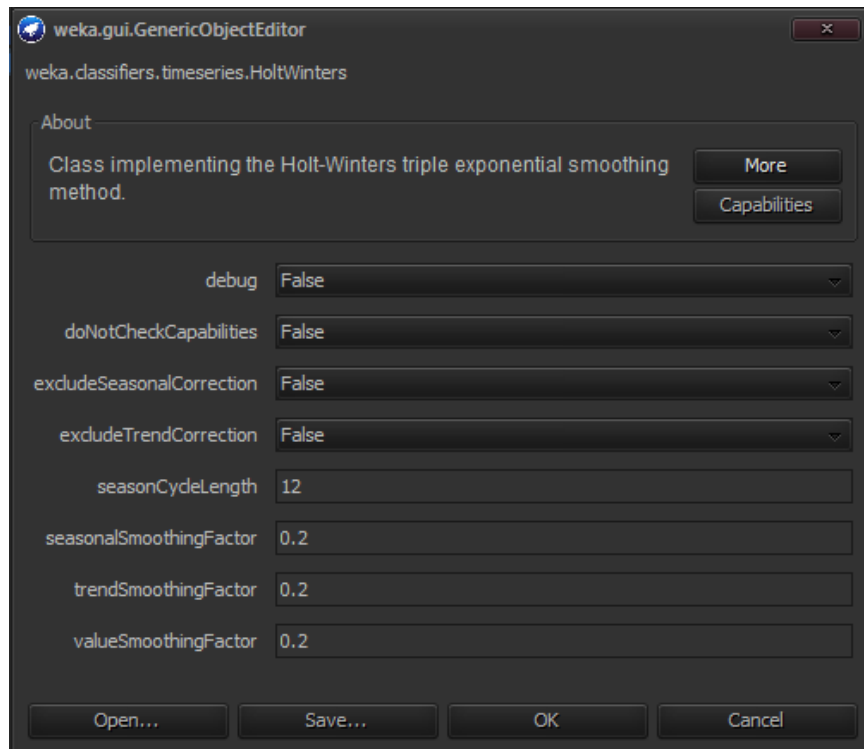


Fig. 17 – 'HOLTWINTERS' Time series parameters.

The seasonal correction parameter is designed to allow for the data prediction to take into account the seasonal effects upon the data. The season operators do not have any effect within this data set due to the limited time span of the data. To effectively use this filter upon any data set would require years worth of data so any seasonal patterns can be analysed and used in the predictions. The filter is not designed for a smaller time scale like that over the time of couple weeks. This also makes the season cycle length parameter unusable as well.

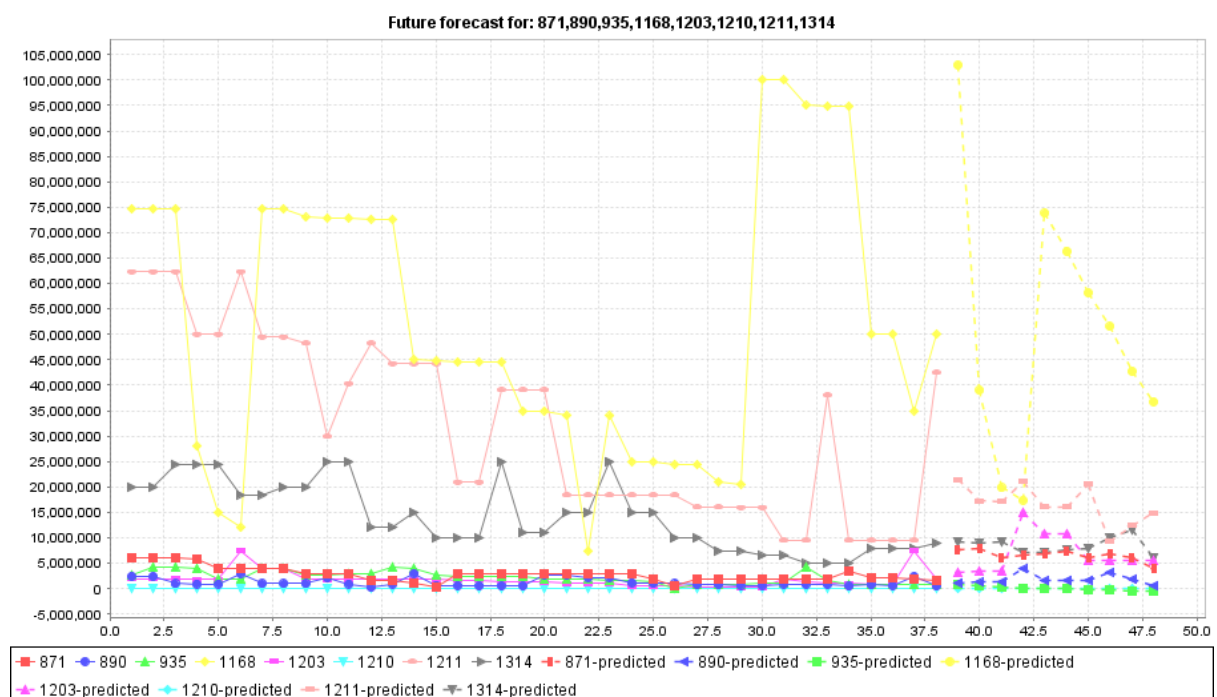


Fig. 18 – Time series filters used on time series forecasting.

As can be seen the above Fig. 18, the time series filters work much better at producing realistic predictions to the data. Each predicted data plot follows the near the linear line that can be imagined to be passing through the data.

The main issues that appear with this Filter and time series analysis is that the prediction tools is not very accurate at taking into account the random spikes in price fluctuations as it also incorporates these into its pricing predictions. This indicates that the data would require pre-smoothing before being input through the data analysis.

Testing filters

To try and hone in the predicted values and to make for an accurate prediction, the final 10 values were removed from the data set. Cutting the total record count down from 38 to 28. This was so the predicted values could be compared to the actual values to try and validate the accuracy of the filters being used.

Doing so however, started to produce an error within the dataset upon trying to predict the values. The forecasting tool started to produce the error shown in Fig. 19.

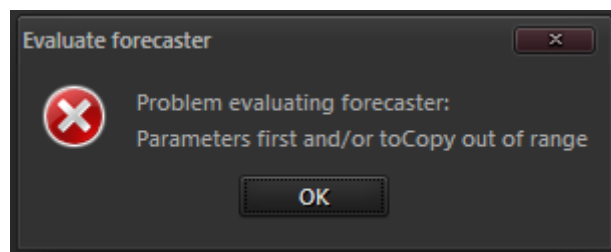


Fig. 19 – Forecasting error.

Upon looking for the cause to this error, nothing was found. This error can only be presumed to be an issue with the forecasting tool itself or the data set does not span a great enough time period to effectively work, causing it to error out as shown.

WEKA Issues

Upon trying to do many of the tasks for time series analysis within WEKA, multiple issues have arisen. With much frustration being caused. Some of these issues may have been caused by the dataset itself, however some are due to the nature of WEKA's programming itself.

Processing and Memory limitations

WEKA is built using the Java Runtime Environment (JRE), this is great for allowing the code to run almost everywhere as the JRE is available for almost all major platforms. This has also caused some noticed issues.

An occasional error that occurred with WEKA is due to the way Java handles threads. Java without thread handling is single threaded and although WEKA has multithreading programmed into it occasionally it would get stuck in a single thread. This wasn't a major issue but may prove some issues with slower clocked processors.

Memory allocation was a major issue with the time series forecasting tool and was one of the major issues with it that prevented full dataset analysis. Java would continually run out of memory regardless of how much was assigned to it. At quantities of RAM, WEKA would error gracefully (Fig.14) however at larger RAM values it would error with indications towards a memory leak (Fig.15) and error un-gracefully causing WEKA to crash.

Although there were other issues with WEKA they are not worth mentioning.

Alternatives to WEKA

An alternative to WEKA that could have been used is Rapid Miner (7). Rapid Miner appears to be a more rounded data analytics tool, with a larger host of features oriented towards business analytics. Although the Time series analysis could not be conducted within Rapid Miner due to time constraints, it has a far greater documentation about this feature with more support for it.

Rapid Miner also has a much tighter integration with R programming, a language designed around statistical computing (8). With knowledge of R and Rapid Miner was used the data analytics may have turned out with better results.

Data Findings

From what data could be analysed it showed that there were some patterns to the data. Higher priced items were on a downward trend whilst lower price/common items were on a fairly level trend that was slightly rising.

Issues with the Data

The data collected was unfortunately not based on a long enough time period to show proper trends from just graphing it or for the time series forecasting tool to make adequate predictions to future data.

The data was also very unclean, with lots of zero values where items either didn't sell or only appeared a few times over the time period data was collected over. This may have had a magnitude of effects upon the validity and accuracy of any predictions drawn from the data, as the data did not represent the entire spectrum of items that are sellable upon the WoW auction house.

Remedies to Issues

As the dataset was built entirely for this project the solution to not having enough data would have been to gather more data. However the amount of data was limited due to the constraints of the project and how much time would have been left for analysis and trouble shooting.

As mentioned earlier the data was collected from a single realm (or game server) whilst there are multiple available. The chosen server was a fairly active server, making it the reason data was collected from it.

Whilst in reality the data should have been collected from several servers and the data averaged out between them all. Doing this would of given a better approximation of data across the whole of the game along with being able to have a larger dataset that represented more of the in-game items.

Dataset Questions

- Is there inflation within the WoW economy?

This was inconclusive due to the issues within the time series tools and filters. However WoWuction has a 14 day trend available on their site. With the short term data showing that more items are on a higher proportion of increasing in value than those that are dropping. See figure 20.

14-day top rising prices		
Item	PMktPrice	Daily Change
Fuzzy Green Lounge Cushion	805 28	+3.50 %
Enormous Crescent Saberfish	3 31	+2.94 %
Hemet's Heartseeker	129 72	+2.33 %
Crescent Saberfish Flesh	1 23	+2.10 %
Blackwater Whiptail Flesh	0 41	+1.99 %
Cinderbloom	3 88	+1.95 %
Flawless Battle-Stone	2469 80	+1.90 %
Safari Lounge Cushion	739 29	+1.90 %
Super Cooling Module	1840 56	+1.73 %
Super Cooling Coolant	237 13	+1.66 %
Salty Squid Roll	17 13	+1.65 %
Primal Fire	91 78	+1.63 %
Blind Lake Sturgeon Flesh	0 95	+1.62 %
Titanium Ore	35 74	+1.61 %
Burnished Inscription Bag	943 22	+1.58 %

14-day top dropping prices		
Item	PMktPrice	Daily Change
Cyclical Power Sequencer	180 99	-4.35 %
Smoldering Robe	632 56	-2.64 %
The Night Blade	468 91	-2.42 %
Cyclical Power Framing	236 46	-2.37 %
Soothsayer's Runes	309 70	-2.03 %
Cyclical Power Housing	318 21	-2.00 %
Gnoll War Harness	696 39	-1.53 %
Truesteel Reshaper	59 70	-1.45 %
Universal Language Filter	11925 15	-1.44 %
Ace of War	242 45	-1.41 %
Draenic Dust	2 63	-1.27 %
Slarkskin	951 60	-1.26 %
Unknown item #113632	5441 69	-1.18 %
Lifelike Mechanical Frostboar	216 25	-1.10 %
Arakkoa Cipher	78 71	-1.08 %

Fig.20 – 14 day price stats from WoWuction (4)

This indicates that there is at least some form of pricing inflation even if it is only in the short term.

- Are there pricing fluctuations based on date?

Again this was inconclusive due to improper functionality of the time series tools. Additional data that may have been interesting to see along with any date based fluctuation would have been pricing fluctuations of items around in-game events.

- Do items of particular category sell for more?

As can be seen within figure 5, there is a higher distribution of items in the armour category. Although they may not sell for more, they sell with the widest range in price along with the greatest quantity. Showing that there is ample supply of a wide range of this item type.

Conclusion

There is potential for this sort of analysis against virtual economies, with the advantage of being able to figure out solutions and test them. However this dataset was inconclusive and did not provide the answers that were being sort after. Data is still being collected for this dataset and maybe revisited as a personal project at some point.

References

1. **Dell.** What is Data Mining. *Statsoft*. [Online] Dell. [Cited: 1 05 2015.] <http://www.statsoft.com/textbook/data-mining-techniques>.
2. —. How To Identify Patterns in Time Series Data: Time Series Analysis. *Statsoft*. [Online] Dell. [Cited: 1 05 2015.] <http://www.statsoft.com/textbook/time-series-analysis#main>.
3. **Federal Reserve.** What is inflation and how does the Federal Reserve evaluate changes in the rate of inflation? *Federal Reserve2*. [Online] 26 01 015. http://www.federalreserve.gov/faqs/economy_14419.htm.
4. **WoWuction.** *WoWuction*. [Online] <http://www.wowuction.com/>.
5. **Blizzard.** WoW API Documents. [Online] <https://blizzard.github.io/api-wow-docs/>.
6. **WEKA.** Time Series Analysis and Forecasting with Weka. *Weka Pentahoo*. [Online] <http://wiki.pentaho.com/display/DATAMINING/Time+Series+Analysis+and+Forecasting+with+Weka>.
7. **Rapid Miner.** Rapid Miner: About Us. *Rapid Miner*. [Online] <https://rapidminer.com/about-us/>.
8. **The R Foundation.** The R Project for Statistical Computing. [Online] <http://www.r-project.org/>.