Math 625.492
Final Project Submission

Richard N. Shea

May 15, 2018

Introduction

In this project I've taken Walmart sales data from a closed (invite only) Kaggle [1] competition and apply to three models discussed in Koller [2] and Bishop [3]. These models include Linear Regression, Markov Chains, and Hidden Markov Models.

The most difficult concepts lie in the HMM framework using the R programming language. In general, R packages are written by researchers with specific domains in mind. The closest domain that contains well written packages for Time Series modelling are financial modelling packages such as 'depmixS4' and 'HMM'. Thus, the majority of the labor involved is getting the data to work with the functions contained in these packages using Zucchini [4] as a foundation. The overall all goal (besides learning the course concepts) is to see if HMM is worth applying to univariate time series forecasting.

This project is broken into three parts; Data, Modelling, and Discussion. All relevant formulas and R code has been placed in the Appendix following References. However, it is best to use the GitHub page at https://github.com/foobash/492.

The biggest store and smallest store along with MAPE was chosen, because these are big picture numbers/scenarios with Purchasing Managers [5].

Part I. Data

Cleansing:

Rounded sales and removed (a few) zero sales data for the algorithms in R to converge better.

Pre-processing:

For the Linear and HMM, doing a min/max normalization as follows was needed for their respective R packages.

The data is then normalized as follows:

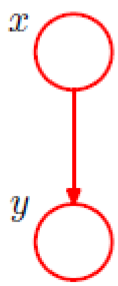
$$sale = \frac{sale - \min(all\ sales)}{[\max(all\ sales) - \min(\text{all sales})]}$$

Note: Must convert back before running forecast metrics on test data.

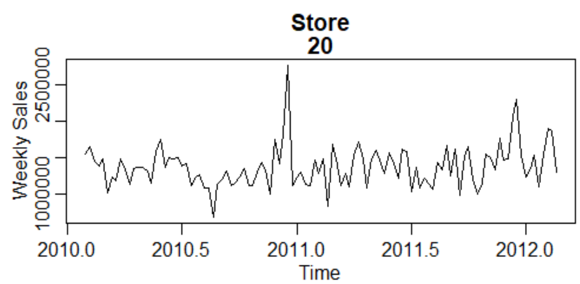
The above was done in Microsoft Excel and Access.

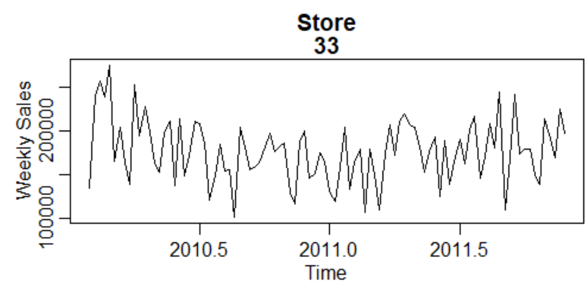
Part II. Modelling

Linear Regression

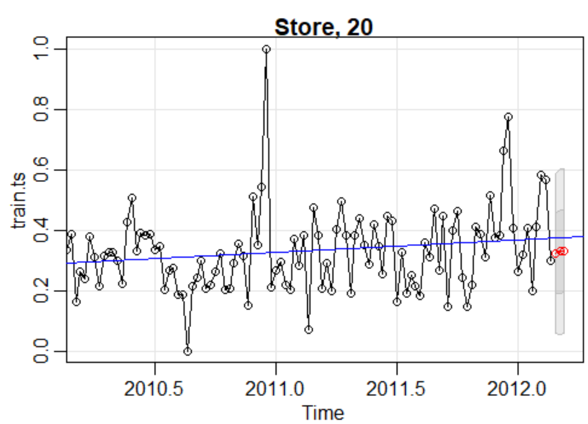


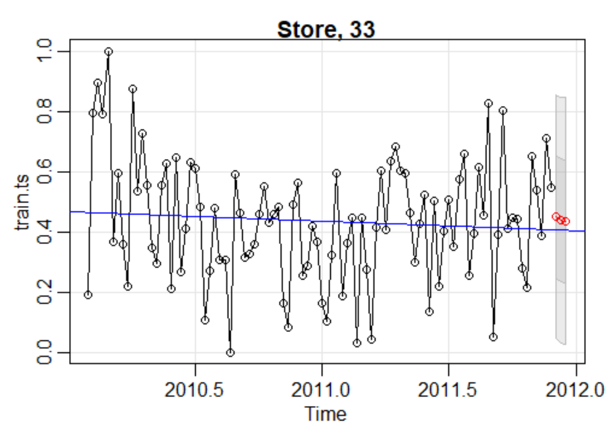
For Linear Time Series analysis in R, the *Applied Statistical Time Series Analysis package* ('astsa') package appears to be widely used and accepted [6, 7, 8]. The 'astsa' exploits the lm() regression function in R because the time series attributes are stripped from the variables before the regression is done using the lm() function alone.





Fitting a regression on the above normalized time series along with a forecast horizon of three weeks gives the following:





Store Stats:

	Min	**1st Q**	**Median**	**Mean**	**3rd Q**	**Max**	**SD**
Store 20	683903	1144325	1353591	1373589	1504607	2773216	286869.3
Store 33	101765	153332	177815	177594	202281	275689	35749.67

Model Stats:

lm(store ~ time)	**AIC**	**BIC**	**LogLik**
Store 20 (unnormalized)	3022	3029	−1510
Store 20 (normalized)	−119	−114	62
Store 33(unnormalized)	2288	2294	−1142
Store 33 (normalized)	−28	−23	16

Linear Forecast Metrics:

MAPE	**Lag 0**	**Lag 1**	**Lag 2**
Store 20	19.31%	11.60%	11.74%
Store 33	16.96%	10.25%	9.18%

MAE	**Lag 0**	**Lag 1**	**Lag 2**
Store 20	262123	157669	160207
Store 33	30544	18418	16444

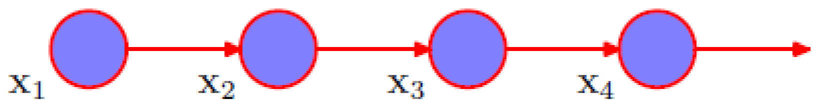
Naïve			
Store 20	Estimated	Actual	MAPE
Lag 0	197044	180117	9.40%
Lag 1	197044	177893.7	10.08%
Lag 2	197044	177601.7	10.37%

Naïve			
Store 33	Estimated	Actual	**MAPE**
Lag 0	1305556	1357478	3.82%
Lag 1	1305556	1369957	4.26%
Lag 2	1305556	1372956	4.48%

Observe that Store 33 (lowest selling store) did much better in error. This is most likely due to the negative slope in the forecast creating a pessimistic forecast in the low selling setting relative to the optimistic forecast for the highest selling store, Store 20.

Part II. Modelling

Markov Chain



The R package ‘markovchain’ is very useful in setting up an intuitive sales model. That is, we separate the sales values by Low [0]/High [1]. The line in the said will be the Median value as done by the Society of Actuaries [9,10].

For example, in splitting Store 20, we have the following sequence to train:

1 1 1 1 1 0 0 0 1 0 0 0 1 1 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 1 1 1 1 0 0 0 0 0 1 0 1 0 1 1 0 0 0 1 1 1 0 1 1 1 0 1 1 0 1 1 0 1 0 0 0 0 1 0 1 0 1 0 1 1 0 0 0 1 1 0 1 1 1 1 1 1 0 0 1 0 1 1 1 0.

We then have the following sequence matrix:

$$\begin{bmatrix} Trans & \mathbf{0} & \mathbf{1} \\ \mathbf{0} & 32 & 22 \\ \mathbf{1} & 23 & 31 \end{bmatrix}$$

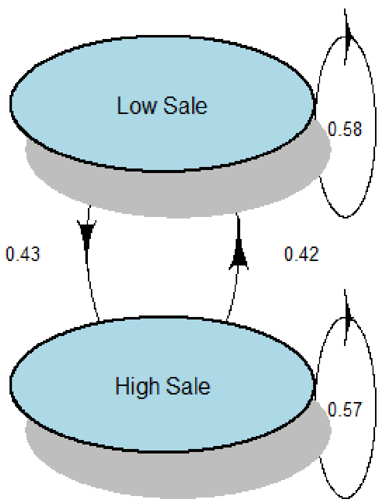
Then setting the markovchainFit() function to solve by the Method of Likelihood Estimation with 90% confidence outputs the following probability matrix.

$$\begin{bmatrix} Trans & \mathbf{0} & \mathbf{1} \\ \mathbf{0} & 0.58 & 0.42 \\ \mathbf{1} & 0.43 & 0.57 \end{bmatrix}$$

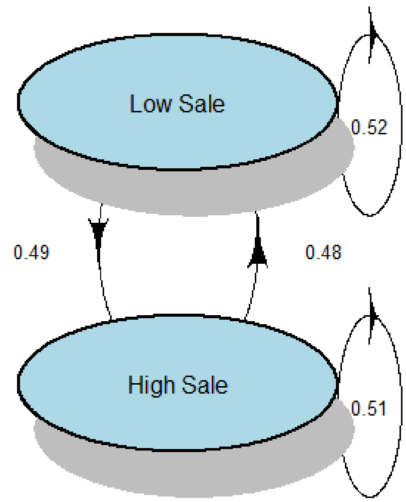
Observe that since we only chose two (pseudo) states, this matrix will converge fast.

Thus, we have the following graphical model for Store 20;

Store 20 Transistion Matrix Week 1



Store 20 Transistion Matrix Week 2



Applying the same to Store 33 and running a forecast horizon of 3 weeks gives the following results:

Stats:

	Median Sales	**Mean of Low Sales**	**Mean of High Sales**
Store 20	1353591	1164643	1582535
Store 30	177814	149404	205784

Model:

Store	***LogLik***	**Confidence**
20	-72.80	0.90
30	-65.80	0.90

Week_1		
Store 20	**Low Sale**	**High Sale**
Low Sale	0.58	0.42
High Sale	0.43	0.57

Week_2		
Store 20	**Low Sale**	**High Sale**
Low Sale	0.52	0.48
High Sale	0.49	0.51

Week_3		
Store 20	**Low Sale**	**High Sale**
Low Sale	0.51	0.49
High Sale	0.50	0.50

Week_1		
Store 33	**Low Sale**	**High Sale**
Low Sale	0.48	0.52
High Sale	0.51	0.49

Week_2		
Store 33	**Low Sale**	**High Sale**
Low Sale	0.50	0.50
High Sale	0.49	0.51

Week_3		
Store 33	**Low Sale**	**High Sale**
Low Sale	0.50	0.50
High Sale	0.50	0.50

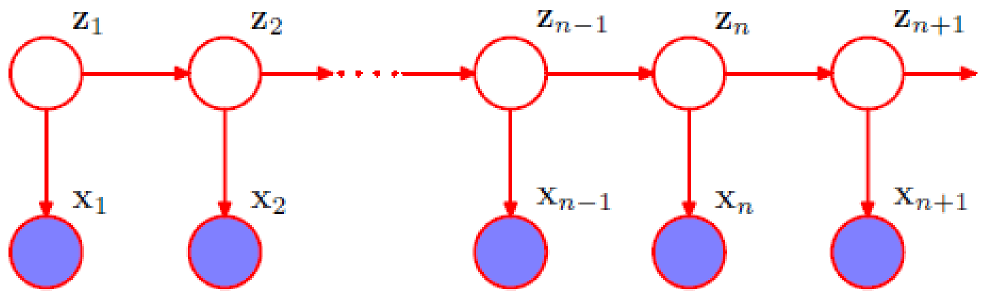
MC Forecast Metrics:

Store 20				
Week 1 (Lag 0)				
Forecast	Estimate	Actual	MAE	**MAPE**
Low	1340158	1619602	279444	17.25%
High	1402841	1619602	216761	13.38%
Week 2 (Lag 1)	Estimate	Actual	MAE	**MAPE**
Forecast				
Low	1366485	1423172	56687	10.62%
High	1375887	1423172	47285	8.35%
Week 3 (Lag 2)	Estimate	Actual	MAE	**MAPE**
Forecast				
Low	1370434	1538238	167804	10.72%
High	1371844	1538238	166394	9.17%

Store 33				
Week 1 (Lag 0)				
Forecast	Estimate	Actual	MAE	**MAPE**
Low	178721.8	134830	279444	32.55%
High	177030.4	134830	216761	31.30%
Week 2 (Lag 1)	Estimate	Actual	MAE	**MAPE**
Forecast				
Low	177842.3	134830	43012.3	32.23%
High	177893	134830	43063	31.62%
Week 3 (Lag 2)	Estimate	Actual	MAE	**MAPE**
Forecast				
Low	177868.7	257258	79389.3	31.77%
High	177867.1	257258	79390.9	31.37%

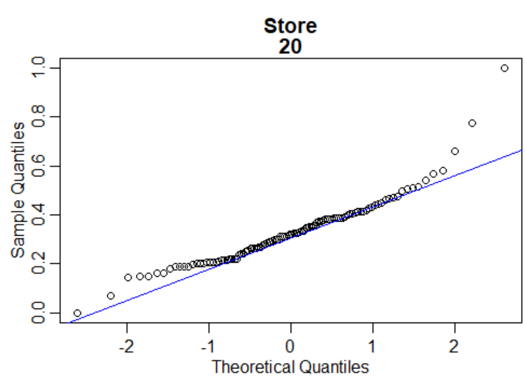
Part II. Modelling

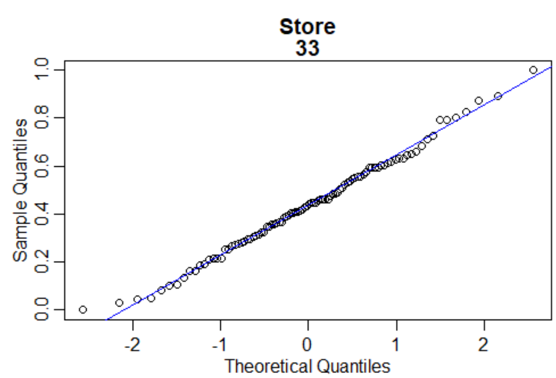
Hidden Markov Model



The Dependent Mixture Models ('depmixS4') appears to be the best at choosing a model (# of states) effectively. However, this package was geared (and tuned) more towards financial data using multivariate states. The Regularized Autoregressive Hidden Semi Markov Model package ('rarhsmm') allows one to turn off the autoregressive and semi features. Doing so, allows the EM and Viterbi to be ran across our sales data smoothly.

The following QQ_plots give motivation to use a Gaussian Markov Model:





Running the 'depmixS4' package on our sales data:

Store 20			
depmixS4	**Loglik**	**AIC**	**BIC**
2 States	80.41	-136.52	-117.75
3 States	80.42	-132.85	-95.30
4 States	86.15	-126.30	-64.61
5 States	90.18	-112.35	-21.16
6 States	96.11	-98.22	27.84

Store 33			
depmixS4	**Loglik**	**AIC**	**BIC**
2 States	17.84	-21.67	-3.72
3 States	21.63	-15.25	20.65
4 States	28.69	-11.37	47.61
5 States	37.43	-6.85	80.34
6 States	40.61	12.77	133.30

We can see the fitting starts falling apart after 3 states for Store 33. Thus, we will stick to forecasting in 2 and 3 state models.

For example: Using the Store 20 three state (Gaussian) Emissions matrix gives.;

$$\begin{bmatrix} Emissions & 1 & 2 & 3 \\ 1 & 51.78\% & 46.30\% & 1.92\% \\ 2 & 0.00\% & 76.25\% & 23.75\% \\ 3 & 10.25\% & 36.35\% & 53.40\% \end{bmatrix}$$

Inference Task;

With the best fitted observation sequence (Viterbi):

```
3 3 3 3 3 2 2 2 2 2 2 2 2 2 2 2 3 3 3 3 3 3 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 3 3 1 1 2 2 2 2 2 2 2 2 2 3 3 2 2 2 3 3 3 2 3 3 3 2 3 3 2 3 3 2 2 2 2
2 2 2 2 3 2 3 2 3 3 2 2 2 3 3 3 3 3 3 1 1 2 2 2 3 2 3 1 1 2
```

The smoothing function, smooth.hmm() gives us;

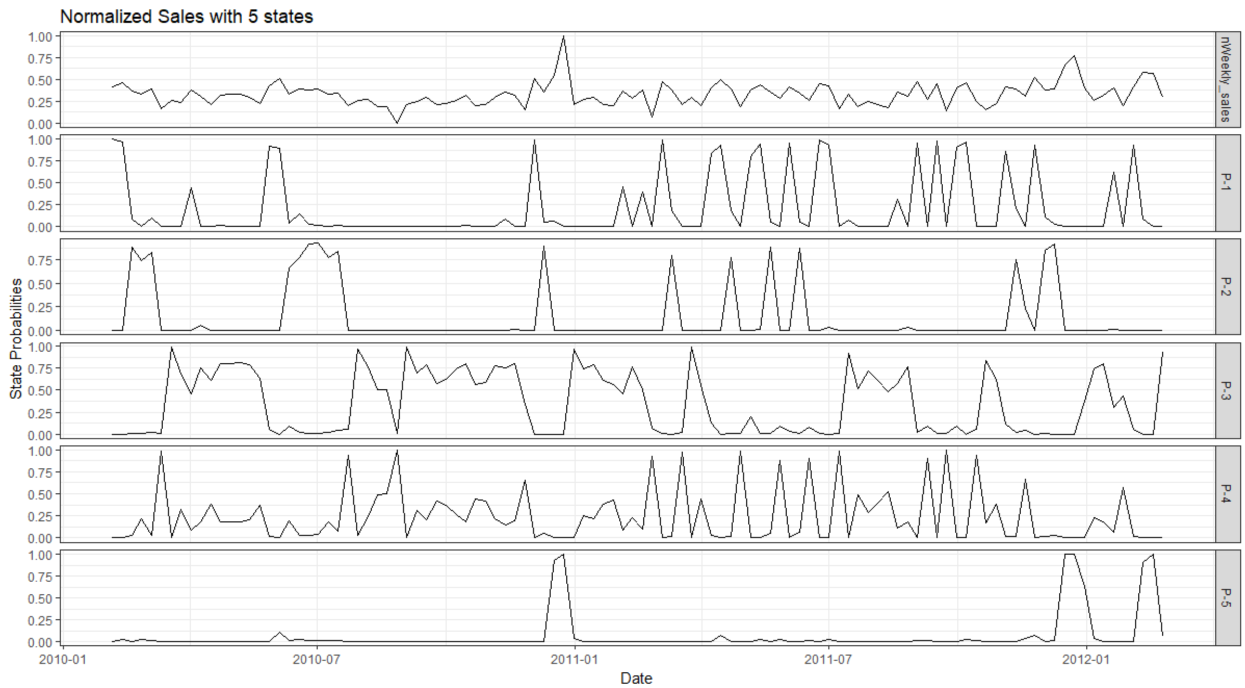
For each week, the probability of being in each state:

```
> head(sp1)
             [,1]         [,2]         [,3]
[1,] 5.911562e-39 4.098582e-30 1.0000000000
[2,] 2.199889e-02 2.408064e-02 0.9539204643
[3,] 1.607623e-02 2.096186e-01 0.7743051388
[4,] 1.097246e-02 4.338135e-01 0.5552140528
[5,] 1.894663e-02 3.977174e-01 0.5833360154
[6,] 5.941080e-03 9.937335e-01 0.0003254566
```

…….. ……………… …………….. ……………

```
> tail(sp1)
              [,1]         [,2]       [,3]
[103,] 0.008952864 0.4022642784 0.58878286
[104,] 0.015786434 0.9736950104 0.01051856
[105,] 0.016471271 0.0095388305 0.97398990
[106,] 0.936294188 0.0002332959 0.06347252
[107,] 0.974668337 0.0029721297 0.02235953
[108,] 0.112432299 0.8692136134 0.01835409
```

Probability of being in up to 5 States for Store 20.



After applying this to Store 33 as well, we have the following forecast results for 1 week out.

Store 20	Estimated	Actual	MAE	**MAPE**
State 2	1388888	1619602	230714.5	14.25%
State 3	1334159	1619602	285443.4	17.62%

Store 33	Estimated	Actual	MAE	**MAPE**
State 2	173933.8	149572	24361.82	16.29%
State 3	174686.2	149572	25114.16	16.79%

Conclusion

As we can see, the Linear model is faster to setup, quicker to update, and beats the MC and HMM in MAPE and MAE. There maybe other metrics that Linear will lose to, but most purchasing managers use MAPE as a big picture number. So, one would need to show a better MAPE before moving forward with another model in practice.

In the univariate sense, it doesn't appear worth while to move forward with predicting sales with Markovian Models. This is because the independent Markov assumption probably doesn't hold. However, it might be worth investigating sales data on the multivariate level. Temperature didn't appear to be a factor throughout the Stores when testing the lm() function, but maybe it's dependent within store departments.

R Commentary:

It was very difficult to work the Viterbi functions due to the financial design of their implantation. There doesn't appear to be an R algorithm out there that predicts more than one step ahead without moving the whole time series. This makes sense for the financial world, not for the purchasing world because the supply chain. However, towards the end of the project, it doesn't appear that difficult to change the package from the GitHub Repo…since R is open source!

References

[1] Kaggle: Your Home for Data Science https://www.kaggle.com/

[2] Daphne Koller & Nir Friedman Probabilistic Graphical Models
ISBN-10: 0465031242

[3] Christopher M. Bishop Pattern Recognition and Machine Learning
ISBN-10: 0-387-31073-8

[4] Walter Zucchini Hidden Markov Models for Time Series: An Introduction Using R, Second Edition
ISBN-10: 1482253836

[5] Demand Planning, LLC, based in Boston, MA http://demandplanning.net/MAPE.htm

[6] Shumway, R.H. & Stoffer, D. S. (2000). Time Series Analysis and Its Applications. New York: Springer.
ISBN 978-3-319-52452-8

[7] Soren Hojsgaard Graphical Models with R
ISBN-10: 1461422981

[8] Brett Lantz Machine Learning with R, Second Edition
ISBN-13: 978-1-78439-390-8

[9] Society of Actuaries - *Hidden Markov Models and You Parts 1 and 2*.

1. https://www.soa.org/library/newsletters/forecasting-futurism/2013/july/ffn-2013-iss7-norris.aspx

2. https://www.soa.org/library/newsletters/forecasting-futurism/2013/december/ffn-2013-iss8-grossmiller.aspx

[10] Society of Actuaries - *Hidden Markov Model for Portfolio Management with Mortgage-Backed Securities Exchange-Traded Fund* *https://www.soa.org/research-reports/2017/2017-hidden-markov-model-portfolio-mgmt/*

[*] GitHub Repo - https://github.com/foobash/492

Appendix

Mean Absolute Percentage Error (MAPE):

$$\mathrm{M} = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{A_t - F_t}{A_t}\right|,$$

where A_t is the actual value and F_t is the forecast value.

Mean Absolute Error (MAE):

$$\mathrm{MAE} = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j|$$

Aikaike Information Criterion: $\mathrm{AIC} = -2\ln L + 2K$

Bayesian Information Criterion: $BIC = -2\ln L + K\ln(N)$

N: number of datapoints; K: number of free parameters

Filtering $p(\mathbf{x}_k \mid \mathbf{z}_{1:k})$

Smoothing $p(\mathbf{x}_t \mid \mathbf{z}_{1:k}) \quad 0 \le t < k$

Prediction $p(\mathbf{x}_{k+t} \mid \mathbf{z}_{1:k}) \quad t > 0$

Most Likely Sequence

$$\arg\max_{\mathbf{x}_{1:k}} p(\mathbf{x}_{1:k} \mid \mathbf{z}_{1:k})$$