Use of Hidden Markov Models to Predict Change in Brand Household Penetration

Final Project Submission for I526 - Applied Machine Learning, Fall 2016

Peter Annable

Project Overview

This project explores the use of Hidden Markov Models HMM as a way to model and project changes in consumer brand sales. A common measurement of this is household penetration, or the number of households who have purchased the product during the past 12 months. This is commonly used as a business goal as it is a leading indicator gaining new buying households which is a key component to growing the brand's business. A brand leader will often set a 12 month goal for increased household penetration. By using HMM's I hope to project the probability of the penetration goal being reached.

Problem Description

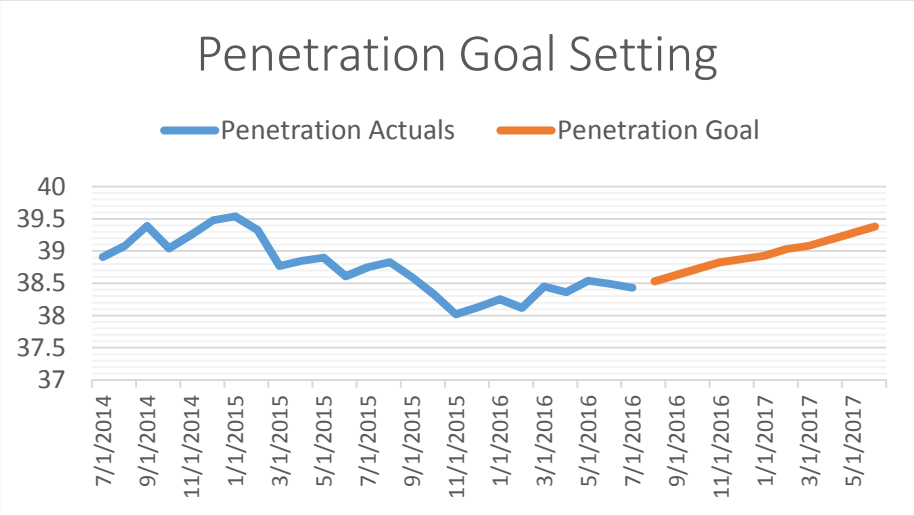
Household Penetration measures the number of households that have purchased the brand at least once during the last 12 months. This performance indicator is driven by several underlying features that are tracked on a monthly basis, described here:

Model Feature	Type	
Category Volume Growth	Percentage	Total growth of category. Often, a brand's growth is driven by growth of the total category.
Volume Share	Percentage	Brand's share of the total category by volume. Brands with a higher share will grow at a higher rate with the category.
Category Price per Stat Over/Under Competition	Index	Measures the brand's product price per statistical unit vs. competition – how much over or under expressed as percentage. Being even or less that the average competition will generally help to drive sales.
Media Planned GRP	Number	Media Gross Ratings Points is the number of impressions to be purchased in a month. It is expressed as percentage of the total target population. i.e. 400 GRP's = 400% of target population. This provides a projection of how much exposure the brand will have on TV and Online Media and therefore how much consumer awareness.

Total Distribution	Number	A number that represents the percentage of stores that have the product. Better product availability helps sales.
Average Price of Gasoline	Dollars	Average price per gallon, is highly correlated to the number of shopping trips. This acts a proxy for how many trips a consumer will make to the store. More trips equals more chances for a sale.

The chart below describes actual penetration data and an annual goal. The goal here is to increase penetration from approximately 38.4 to 39.4 during the 12 months of the fiscal year:

Figure 1 – Penetration Goals



The question here is, do I have a sufficient business plan, and do market conditions support achieving this goal? I will use HMM models to find out, and ideally express the desired change as a probability.

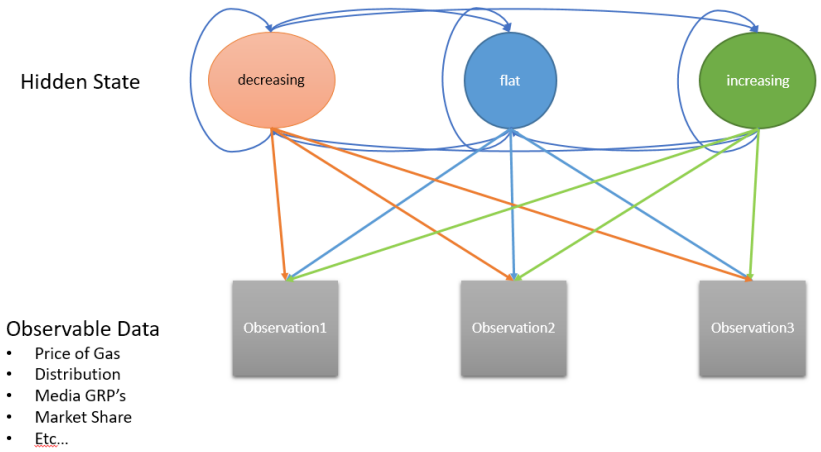
Methodology Overview

For this study, we assume that Household Penetration behaves as a Markov Chain with hidden states. In the simplest case, the states are "increasing" and "decreasing". For a given set of observations, we wish to determine is the resulting state "increasing" or "decreasing". Or in a more complex model would be to determine if Penetration "decreasing", "flat", or "increasing".

During this study, 2 implementations are explored

1. Using a Fitted model, re-applying a stock market regime change method, using only historical penetration change.
2. A fitted model using multiple features from the table above.

Figure 2 – Penetration Change as Hidden Markov Model



Method 1 – Reapplication of Financial Regime Change Model

The first model re-applies code that is described at the end of this presentation:
http://www.slideshare.net/DerekKane/data-science-part-xiii-hidden-markov-models

It is very similar to other stock price forecasting examples, such as the one found at
http://gekkoquant.com/2014/09/07/hidden-markov-models-examples-in-r-part-3-of-4/

The general idea is to predict the future change in stock price based on previous change history using a fitted HMM, using the RHMM package. Using the RHMM package simplifies this task by providing a function call that initializes the model and calculates the states using the Baum-Welch algorithm. Then once I have the model built, I provide a set of future desired observations that correspond to desired change, and using the Viterbi algorithm, project the most likely state change path and probability of occurrence. This will give me an expected final value and its probability of happening, given the desired change history.

Data Set

The data set for this case consists of two files: 1) a set of historical changes to penetration, calculated on previous observations, and 2) a set of desired future changes.

Data Set 1 – Training Data:

This is actual penetration for a common P&G product sold in the United States.

month_end_date	Penetration_Amt	Penetration_Change_PP
10/31/2011	36.4	0

11/30/2011	36.42	0.02
12/31/2011	36.85	0.43
1/31/2012	36.82	-0.03
2/29/2012	37.34	0.52
3/31/2012	37.16	-0.18
4/30/2012	37.36	0.2
5/31/2012	37.67	0.31
6/30/2012	37.58	-0.09
7/31/2012	37.92	0.34
8/31/2012	38.4	0.48
9/30/2012	39	0.6
10/31/2012	38.99	-0.01
11/30/2012	39.12	0.13
12/31/2012	38.8	-0.32
1/31/2013	38.72	-0.08
2/28/2013	38.76	0.04
3/31/2013	39.03	0.27
4/30/2013	39.24	0.21
5/31/2013	39.35	0.11
6/30/2013	39.26	-0.09
7/31/2013	39.01	-0.25
8/31/2013	38.75	-0.26
9/30/2013	38.83	0.08
10/31/2013	39.15	0.32
11/30/2013	39	-0.15
12/31/2013	38.97	-0.03
1/31/2014	38.77	-0.2
2/28/2014	38.4	-0.37
3/31/2014	38.76	0.36
4/30/2014	38.45	-0.31
5/31/2014	38.52	0.07
6/30/2014	38.58	0.06
7/31/2014	38.91	0.33
8/31/2014	39.08	0.17
9/30/2014	39.39	0.31
10/31/2014	39.04	-0.35
11/30/2014	39.25	0.21
12/31/2014	39.48	0.23
1/31/2015	39.54	0.06
2/28/2015	39.33	-0.21
3/31/2015	38.77	-0.56

4/30/2015	38.85	0.08
5/31/2015	38.9	0.05
6/30/2015	38.61	-0.29

DATA SET 2 – GOAL DATA:

In this example, for the same product, I pretend that in the last six months, a new product initiative is launching that is believed will increase penetration by 0.5 points.

month_end_date	goal
7/31/2015	0
8/31/2015	0
9/30/2015	0
10/31/2015	0
11/30/2015	0
12/31/2015	0
1/31/2016	0.1
2/29/2016	0.1
3/31/2016	0.1
4/30/2016	0.1
5/31/2016	0.1
6/30/2016	0.1

R Code

```
library("TTR")
library("xts")
library("quantmod")
#
# Read in Data, convert date for extended time series
#
branddata <- read.csv(file="../USFeTrainData2011-2015.csv", head=TRUE, sep=",")
branddata$month_end_date <- strptime(branddata$month_end_date, format ="%m/%d/%Y")
goaldata <- read.csv(file="../USFeGoalData15-16.csv", head=TRUE, sep=",")
goaldata$month_end_date <- strptime(goaldata$month_end_date, format ="%m/%d/%Y")

gd.xts <- xts(x=goaldata[,"goal"], order.by = goaldata[,"month_end_date"])
pen.goal <- gd.xts
colnames(pen.goal)[1] <- "PenChangeNumeric"

pen.train <- xts(x=(as.numeric(branddata$Penetration_Change_PP)), order.by =
branddata[,"month_end_date"])
colnames(pen.train)[1] <- "PenChangeNumeric"
print (pen.train)
print (pen.goal)
#
# Fit Model with training data using 3 states
#
library("RHmm")
hm_model <- HMMFit(obs=pen.train, nStates=3, control=list(iter=200))
print(hm_model)

#
# Determine probability of future sequence projection
#
print("Probability of Goal Sequence")
VitPathGoal <- viterbi(hm_model, pen.goal)
print(VitPathGoal)
print(VitPathGoal$logViterbiScore)

print(exp(VitPath$logProbSeq))
#
# Show most likely path state
#
pen.predict <- rbind(pen.train, pen.goal)
pen.predict <- cbind(pen.predict,0,0,0,0)
colnames(pen.predict)[2] <- "state"
colnames(pen.predict)[3] <- "change"
colnames(pen.predict)[4] <- "Penetration_Amt"
colnames(pen.predict)[5] <- "PenetrationPredicted"
chg_calc <- matrix(unlist(hm_model$HMM$distribution$mean),nrow=1,ncol=3)

#
# use loops to calculate Penetration values based on initial plus series of changes

for (i in 1:nrow(branddata)) {
  pen.predict$Penetration_Amt[i]=branddata$Penetration_Amt[i]
}
```

```
pen.predict$PenetrationPredicted[1] = pen.predict$Penetration_Amt[1]
VitPathStatesCombined <- c(VitPath$states, VitPathGoal$states)

for (i in 1:nrow(pen.predict)) {
  print(paste("i=",i), quote=FALSE)
  #print(VitPath$states[i])
  #print(hm_model$HMM$transMat[(VitPath$states[i])])
  pen.predict$state[i]= VitPathStatesCombined[i]
  pen.predict$change[i] = chg_calc[pen.predict$state[i]]
  if (i < nrow(pen.predict)) {pen.predict$PenetrationPredicted[i+1]=pen.predict$PenetrationPredicted[i]
+ pen.predict$change[i] }
  if (pen.predict$Penetration_Amt[i] == 0 ) {pen.predict$Penetration_Amt[i] = NA }
  }

#
# For charting purposes, build a time series with just the forecast data, but calculated started from
the last known actual value.
#
pen.forecast <- tail(pen.predict,nrow(pen.goal))
pen.forecast$PenetrationPredicted = branddata$Penetration_Amt[nrow(branddata)]

for (i in 1:nrow(pen.forecast)) {
  #print(paste("i=",i), quote=FALSE)
  pen.forecast$state[i]= VitPathStatesCombined[i]
  pen.forecast$change[i] = chg_calc[pen.predict$state[i]]
  if (i < nrow(pen.forecast))
{pen.forecast$PenetrationPredicted[i+1]=pen.forecast$PenetrationPredicted[i] + pen.forecast$change[i] }
}
#
# Chart the Result
#

chartSeries(pen.predict$PenetrationPredicted, type="line",theme="black",
legend="Forecast",name="Penetration", TA="addTA(pen.predict$Penetration_Amt, on=1, col=4)" )
addTA(pen.forecast$PenetrationPredicted, on=1, col=8, type="line")
addTA(pen.predict[pen.predict[,2]==1,"Penetration_Amt"],on=1,type="p",col=5,pch=25)
addTA(pen.predict[pen.predict[,2]==2,"Penetration_Amt"],on=1,type="p",col=6,pch=23)
addTA(pen.predict[pen.predict[,2]==3,"Penetration_Amt"],on=1,type="p",col=7,pch=24)
```

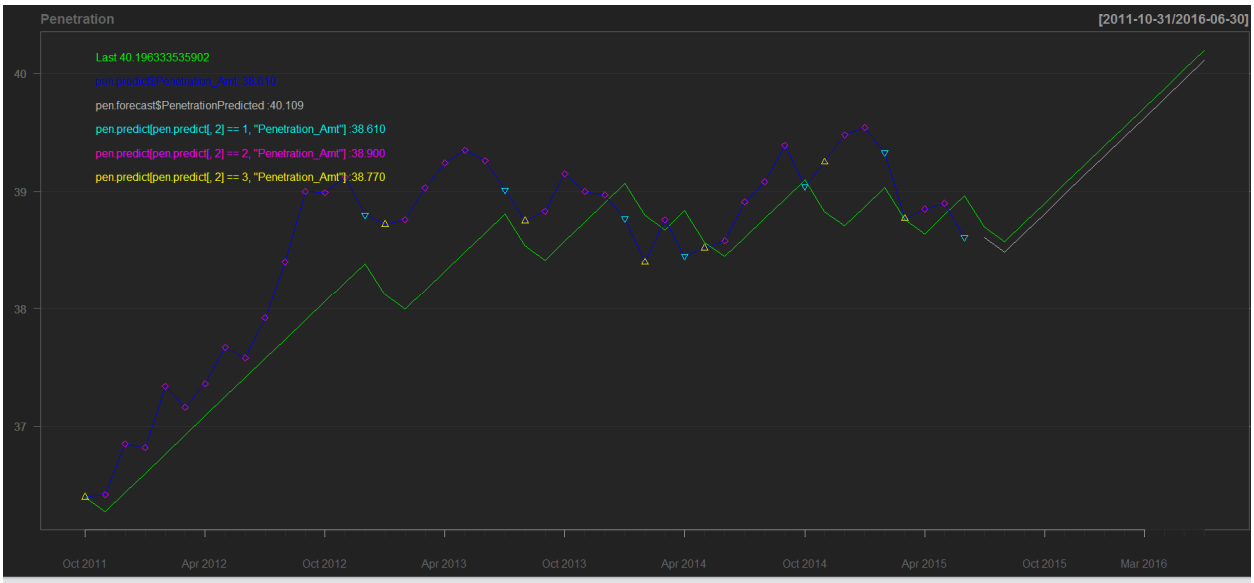
Results

```
Call:
----
HMMFit(obs = pen.train, nStates = 3, control = list(iter = 200))

Model:
------
3 states HMM with univariate gaussian distribution

Baum-Welch algorithm status:
----------------------------
Number of iterations : 127
Last relative variation of LLH function: 0.000001

Estimation:
```

```
-----------

Initial probabilities:
           Pi 1         Pi 2 Pi 3
  9.467675e-317 3.256164e-15    1

Transition matrix:
             State 1      State 2      State 3
State 1 9.508936e-18 9.662133e-06 9.999903e-01
State 2 2.312439e-01 7.687559e-01 1.406958e-07
State 3 2.164484e-34 1.000000e+00 6.170785e-14

Conditional distribution parameters:

Distribution parameters:
              mean         var
State 1  -0.2673519 0.003168474
State 2   0.1622300 0.041422208
State 3  -0.1229553 0.064843839

Log-likelihood: 1.14
BIC criterium: 51.02
AIC criterium: 25.73

> VitPathGoal <- viterbi(hm_model, pen.goal)

> print(VitPathGoal)

$states
 [1] 3 2 2 2 2 2 2 2 2 2 2 2

$logViterbiScore
[1] 3.23699

$logProbSeq
[1] -0.0001135754

[1] "Probability of Goal Sequence: 0.323624519572208"
```

As the chart visual shows, our last actual reading in **blue** was 38.6. When the goal changes were applied which were believe to project an increase of 0.5, we ended a reading of 40.1 (the **gray** line), with a probability of 32%. The **green** line is a calculation of penetration based on the states determine from the training data.

Figure 3 – Chart of Actual and Single-Feature HMM Model-Calculated Penetration, with 12 months forecast penetration.



Method 2 – Extension of Model 1 using Additional Features

The 2nd HMM extends upon the first model by determine if additional features can improve forecasting accuracy.

Data Set

Four features are used: Media GRPs, Volume Share, Gas Prices and Penetration Change. To create a model with weightings of these features that represent their expected impact in real-world use, I created a composite indicator *CI* using the following formula:

CI = .05*(Planned GRPs)/1000 + 1*Volume Share + .20(Log(1/Gas Price $USD))

Example data snippet:

month_end_date	Composite_Indicator	Penetration_Amt
7/31/2014	38.5610434	38.91
8/31/2014	39.34507065	39.08
9/30/2014	41.2589934	39.39
10/31/2014	35.90395141	39.04

…..

Then in the R code, both values are normalized to be between 0 and 1.

```
           Composite_Indicator Penetration_Amt
2014-07-31          0.81351073      0.58552632
2014-08-31          0.86770472      0.69736842
2014-09-30          1.00000000      0.90131579
2014-10-31          0.62984567      0.67105263
```

2014-11-30 0.90704067 0.80921053
….

Code

My R code was essentially the same as above, building a fitted HMM model with 3 states, and using the Viterbi algorithm for path projections. However, there were two critical changes: 1) Normalization of data; 2) providing my own "penetration change amount" to correspond to each of the 3 states. This was required since my data was normalized, the actual expected change for each state was estimated manually based on actual penetration changes.

```
# Normalization of training and goal data:
pen.train$Composite_Indicator = normalize(pen.train$Composite_Indicator)
pen.train$Penetration_Amt = normalize(pen.train$Penetration_Amt)
pen.goal$Composite_Indicator = normalize(pen.goal$Composite_Indicator)
pen.goal$Penetration_Amt = normalize(pen.goal$Penetration_Amt)

normalize <- function(x) {
  x <- sweep(x, 2, apply(x, 2, min))
  sweep(x, 2, apply(x, 2, max), "/")
}

#
# Define the three change states manually:
#
chg_state_amounts <- c(-.17, .03, .12)
```

Results

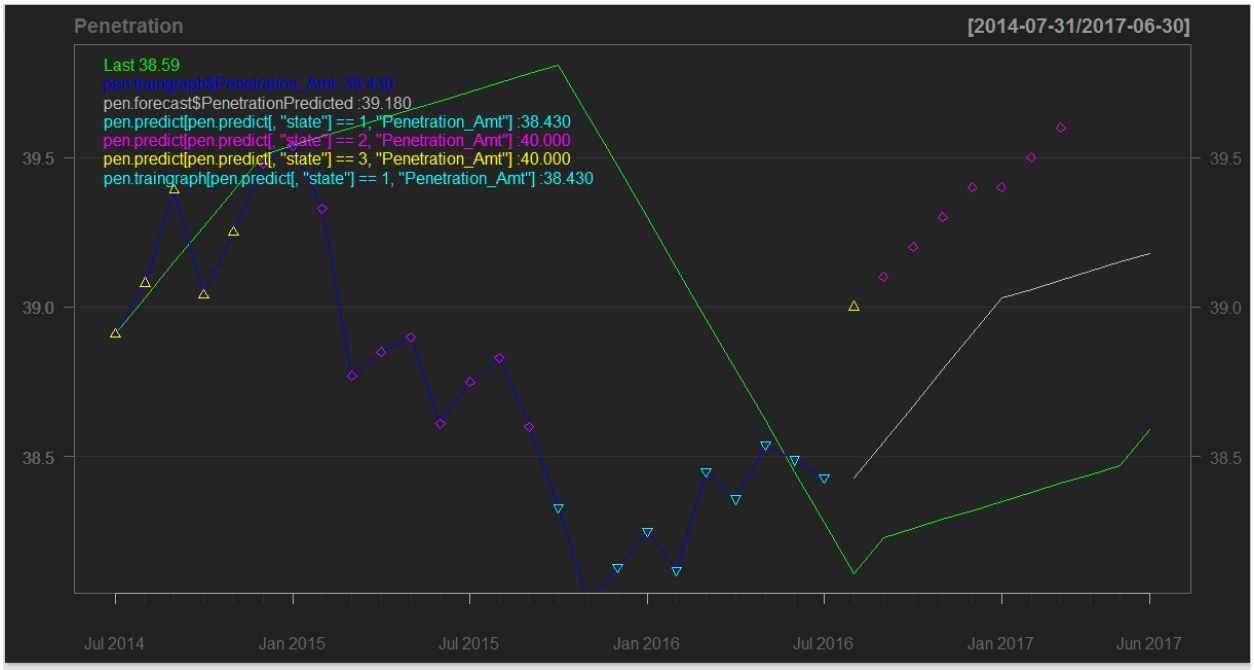
```
source('~/Box Sync/Annable/Grad School Work/Indiana University/I526 Applied Machine
Learning/Final Project/HMM R Project/Pen-Multi-rHMM-v1.R')
           Composite_Indicator Penetration_Amt
2014-07-31          0.81351073      0.58552632
2014-08-31          0.86770472      0.69736842
2014-09-30          1.00000000      0.90131579
2014-10-31          0.62984567      0.67105263
2014-11-30          0.90704067      0.80921053
2014-12-31          0.34191112      0.96052632
2015-01-31          0.19748959      1.00000000
2015-02-28          0.31921784      0.86184211
2015-03-31          0.56852460      0.49342105
2015-04-30          0.23165007      0.54605263
2015-05-31          0.68416733      0.57894737
2015-06-30          0.47260446      0.38815789
2015-07-31          0.41093669      0.48026316
2015-08-31          0.51361842      0.53289474
2015-09-30          0.71338976      0.38157895
2015-10-31          0.34766961      0.20394737
2015-11-30          0.17492621      0.00000000
2015-12-31          0.00000000      0.07236842
2016-01-31          0.06718584      0.15131579
2016-02-29          0.08521128      0.06578947
2016-03-31          0.13054087      0.28289474
2016-04-30          0.49211489      0.22368421
2016-05-31          0.29133928      0.34210526
2016-06-30          0.45667825      0.30921053
2016-07-31          0.44339464      0.26973684
           Composite_Indicator Penetration_Amt
2016-08-31           0.1428571             0.0
```

```
2016-09-30           0.5714286             0.1
2016-10-31           0.0000000             0.2
2016-11-30           0.0000000             0.3
2016-12-31           0.5714286             0.4
2017-01-31           0.4285714             0.4
2017-02-28           0.2857143             0.5
2017-03-30           0.1428571             0.6
2017-04-30           0.2857143             1.0
2017-05-31           0.8571429             1.0
2017-06-30           1.0000000             1.0

Call:
----
HMMFit(obs = pen.train, nStates = 3, control = list(iter = 200))

Model:
------
3 states HMM with 2-d gaussian distribution

Baum-Welch algorithm status:
----------------------------
Number of iterations : 13
Last relative variation of LLH function: 0.000001

Estimation:
-----------

Initial probabilities:
          Pi 1         Pi 2 Pi 3
  3.246137e-275 3.945298e-60    1

Transition matrix:
             State 1      State 2      State 3
State 1 1.000000e+00 6.179682e-24 1.702798e-47
State 2 1.032068e-01 8.967931e-01 1.089105e-07
State 3 5.198045e-41 2.000025e-01 7.999975e-01

Conditional distribution parameters:

Distribution parameters:
  State 1
         mean cov matrix
     0.2628837 0.03468123 0.01385331
     0.1980063 0.01385331 0.01280751

  State 2
         mean  cov matrix
     0.4367800  0.02700191 -0.02272962
     0.6298901 -0.02272962  0.04768362

  State 3
         mean  cov matrix
     0.8436205 0.015132591 0.009595922
     0.7328943 0.009595922 0.012198838


Log-likelihood: 24.67
BIC criterium: 24.69
AIC criterium: -3.35
DimObs=2
$states
 [1] 3 3 3 3 3 2 2 2 2 2 2 2 2 2 1 1 1 1 1 1 1 1 1 1
```

```
$logViterbiScore
[1] 24.30207

$logProbSeq
[1] -0.3709057

attr(,"class")
[1] "viterbiClass"
DimObs=2
$states
 [1] 3 2 2 2 2 2 2 2 3 3

$logViterbiScore
[1] -65.50172

$logProbSeq
[1] -0.0006098438

attr(,"class")
[1] "viterbiClass"
[1] "Probability of Goal Sequence: 0.690108995776666"
```

As the chart visual shows, our last actual reading in **blue** was 38.4 (training data). When future observation data to project an increase of 0.5 is supplied, we ended with a reading of 39.18 (the **gray** line), with a probability of 69%. The **green** line is a calculation of penetration based on the states determine from the training data.

Figure 4 – Chart of Actual and Multi-Feature HMM Model-Calculated Penetration, with 12 months forecast penetration.



Discussion of Results

Model Accuracy

As you can see from the two charts, the first HMM does a better job of creating a model that fits the training data. This is partially due to the fact that my training data is the data type as the HMM States. (penetration change). So the model just needs to identify the state, and calculate mean change amounts for each state, and we can re-create the actual penetration amounts fairly closely. Additional states would make the model even more accurate, but perhaps over-fit the data. Note: I did not know a good way to calculate a confusion matrix for this situation, I assume this could be done with more time to research it.

The more complex model was less accurate vs. the training data. Many of the change states calculated did not correspond properly to the actual data. This was certainly due to my additional features not correctly predicting state change. I tried other training data set weightings, but did not get any result closer than the graph that is shared here. This means I either had incorrect weightings, or my features were not the ones that truly are driving change up or down on my state. I will spend more time testing feature variations, but I ran out of time to complete this before the project was due.

Conclusions

My conclusion is that the HMM process holds promise for providing a solution to this business problem. However, the first model is too simple. I cannot just predict the future based on a set of future observations I hope to have. I need a model like the 2nd where I use features that can predict the state change, and features that have planned or projected values in the future. The planned values in future are something that can be chosen by the business, such as how much media will be aired. Or, it can be data that can be projected with some certainty, such as the price of gasoline.

Another approach could be to create multiple HMM's and use an average. Start with the single feature Model 1, but then use Model 2 to influence the future predictions. This could combine the advantage of better fitting on training data, but with more realistic future predictions. I would like to do this in the future.

Acknowledgements

Prof. Sriraam Natarajan – for the idea to try Hidden Markov Models on this problem.

Devendra Dhami – For his help on ideas and resources

Derek Kane – Data Science Lectures at: https://www.youtube.com/watch?v=j3r9a75zOvM

Bert Huang – Hidden Markov Models: https://www.youtube.com/watch?v=9yl4XGp5OEg

http://gekkoquant.com/2014/09/07/hidden-markov-models-examples-in-r-part-3-of-4/

https://en.wikipedia.org/wiki/Hidden_Markov_model