Marketing Mix Modeling Algorithm Comparison

Timothy Cauley

Northeastern University - August 2023
MS in Data Science Capstone Phase 2 Project

Abstract

The goal of this project was to create MMMs that showcase their 4 use cases: forecasting (with <10% error), produce realistic attribution and ROAS values, and optimize budget allocation to produce a >10% increase in predicted revenue. The inputs for MMMs are various marketing, macroeconomic, and time series variables, and the output is revenue. The data used was 4 years worth of weekly simulated data made by Meta. The machine learning algorithms implemented were OLS Regression, Bayesian Structural Time Series, XGBoost, Shapley Value Regression using a Feed Forward Neural Network, and Robyn (a semi automated solution created by Meta). As expected, the Robyn model performed the best, and BSTS performed very well. Surprisingly, the Neural Network also performed well despite no existing research on deep learning solutions for MMMs. Overall, while the models forecasted and optimized budgets well, they did not produce consistent ROAS numbers, which adds to the belief that MMMs may not be a perfect solution to every marketing analytics problem that they try to solve at once.

Introduction

In the past few decades, businesses have turned more to data science in order to make quality, informed decisions. In the marketing department, there are many potential use cases of advanced analytics, such as churn, customer segmentation, forecasting, single source attribution, and budget optimization. One tool that can be used to solve some of these problems is a Marketing Mix Model (MMM).

An MMM is a regression model that takes inputs such as media spend on different channels, internal business numbers such as distribution, and market/macroeconomic numbers such as consumer price index, and has the output of sales or revenue for the inputs time period. One use case of an MMM is forecasting - seeing projected revenue for different media spend. Another is attribution - seeing what percent of sales were attributed to different media channels. Viewing return on ad spend (ROAS), which is the ROI specifically for different advertisements, is another use case of these models. Finally, optimizing budget allocation for ad spend by finding the distribution of a given budget that maximizes revenue is another use case for such models. The purpose of this project was to create MMMs that showcase all 4 of these use-cases.

The machine learning algorithms implemented were Ordinary Least Squares (OLS) Regression, Bayesian Structural Time Series (BSTS), XGBoost, and Shapley Value Regression using a Feed Forward Neural Network. In addition, the data was used to train an MMM using Robyn, a semi-automated R package made by Facebook to create MMMs. The goal of these models were to be able to forecast accurately with a provisional goal of less than 10% test error, to produce realistic ROAS and contribution values, and to be able to optimize budget allocation with a provisional goal of producing 10% higher predicted revenue with the same total budget for the test period.

Background

Prior to training an MMM, some transformations are first done to the data. First, an Adstock transformation is done to the media spend variables. An adstock transformation is a form of an exponentially weighted moving average meant to mimic the lagging effect that advertising has on consumer decisions.

$$AdStock(x_i) = x_i + r * AdStock(x_{i-1})$$

When a consumer views an advertisement, they may not purchase the item immediately, but that advertisement does have an effect on their purchase consideration and brand awareness, which will play a large impact on their eventual purchase. The second transformation done was applying log plus one to the media spend variables. This is done to mimic the diminishing returns to advertising spending.

OLS Regression is a simple yet effective supervised machine learning algorithm used to predict continuous numeric values. It has a closed form solution of minimizing the sum of squared residuals, with the coefficients β being calculated with the following formula:

$$\beta = (X^T X)^{-1} X^T y$$

Bayesian Structural Time Series (BSTS) is a time series analysis technique that combines Bayesian inference with structural time series modeling. It decomposes a time series to capture time series capturing trends, seasonality,

regression effects, and errors, as shown in the equation below:

$$y_t = u_t + \gamma_t + \beta^T x_t + \epsilon_t$$

where u_t are the trends at time t, γ_t is the seasonality at time t, $\beta^T x_t$ is the regression effect at time t, and ϵ_t is the error at time t. It is trained by first defining prior beliefs about the distribution and parameters, and then iteratively updating these distributions using Bayesian inference to obtain the posterior distributions. These distributions are then sampled using Markov Chain Monte Carlo methods to generate predictions and estimate uncertainty [7].

XGBoost is a supervised machine learning algorithm that can be used to predict both classification or continuous numeric values. It uses gradient boosting where it builds an ensemble of weak learners, in this case regression trees, that attempt to learn from the mistakes of the previous learners by following the negative gradients of the loss function of previous learners in an attempt to minimize residuals. The predictions of these models are then combined together for a final prediction [6].

Shapley Value Regression is an additive analysis done on top of a regression model. Its purpose is to determine the contribution each variable had towards the final output. It is based on a game theory concept called Shapley values, also known as SHAP (SHapley Additive exPlanations) values, created by mathematician and economist Lloyd Shapley who won the Nobel Prize for this creation.

The goal of Shapley Value Regression is to fairly approximate the contribution that each player (media channel) has had to the game (revenue) by looking at the marginal effect it has on the output when it is and is not present. A SHAP value for variable i is calculated using the following equation:

$$\sum_{S \subseteq N \setminus \{i\}} \frac{1}{n!} * (|S|!\,(n - |S| - 1)! * (v(S \cup \{i\}) - V(S))$$

where S is all possible coalitions of variables, $V(S)$ is the total output for coalition S, and n is the total number of variables contributing to the output [5].

Since Shapley Values can be calculated for any type of regression (or classification) model, it was chosen to use a Feed Forward Neural Network as the base model. A Feed Forward Neural Network is a deep-supervised machine learning algorithm that is a neural network with no cycles. They have multiple hidden layers with weights and biases and are trained using a technique called backpropagation which minimizes the mean squared error.

The source of the data was from an R package made by Meta called Robyn, which was also used to train a model. Robyn "is an experimental, ML-powered and semi-automated Marketing Mix Modeling (MMM) open source package". Largely speaking, the machine learning methods being done by the package is undisclosed, but they do mention that it is incorporating "Ridge regression, multi-objective evolutionary algorithm for hyperparameter optimisation, and gradient-based optimisation for budget allocation" [13].

Related Work

The bulk of the inspiration for this project comes from a 2019 PhD thesis from Linköping University titled "Marketing Mix Modeling: A comparative study of statistical models" by Richard Wigren [8]. Wigren compared the effectiveness of OLS, Generalized Least Square Regression (GLS), Shapley Value Regression (using OLS as a base model), BSTS, and XGBoost in forecasting revenue and accurately attributing sales, but stopped short of implementing these models to optimize budget allocation. They found that BSTS overall performed the best, but Shapley Values were the best in terms of attribution. This project aims to extend their work to look at how well the models optimize budget, as well as compare these models to a simple neural network, and to Meta's semi automated solution. This project excluded GLS as the final model was extremely similar to the final OLS model and did not add any additional insights.

While there has been other prior research showing the effectiveness of Bayesian approaches to MMMs [3, 4, 8], zero prior research was found regarding the use of deep learning for MMMs, which is why it was chosen to try a simple Feed Forward Neural Network. The lack of research into the use of deep learning for MMMs is likely due to limited data availability making it difficult to train complex algorithms. Additionally, interpretability is a large component for MMMs in order to gain insights into the variables, which is more difficult with deep learning. Finally, overfitting is already a large problem with the current state of MMMs [2], which would be amplified with the use of more complex models such as neural networks.

Project Description

The data used for this project is a simulated dataset made by Meta to model their MMM package, Robyn. Academic research on MMMs is most commonly done using simulated data as it would require confidential data to be able to train a model using real life data. Originally the R package "dammmdatagen" was going to be used to generate simulated data, but due to little to no documentation and far too much noise in the generated data, it was decided to move on from the package.

The Robyn data comes with 4 years worth of weekly level data, including 11 variables. The first 3 years of data

were used for training and the final year was held out as a test set. There were 5 different media channels that this company spent advertising money on: Facebook, Out of Home (OOH), Search, TV, and Print. 18 additional variables were added, including three macroeconomic metrics (Consumer Price Index, Consumer Confidence Index, and Gross Domestic Product) [10, 11, 12], 4 holiday variables due to clear spikes (New Years, Mothers Day, Fathers Day, and Christmas), and dummy variables for each month.

The first step of the project consisted of exploratory data analysis by looking at visuals and relationships amongst the variables. Looking at the data, it is clear that this is a product that sells the most during the winter, and a company that spends the bulk of their advertising on OOH advertising.

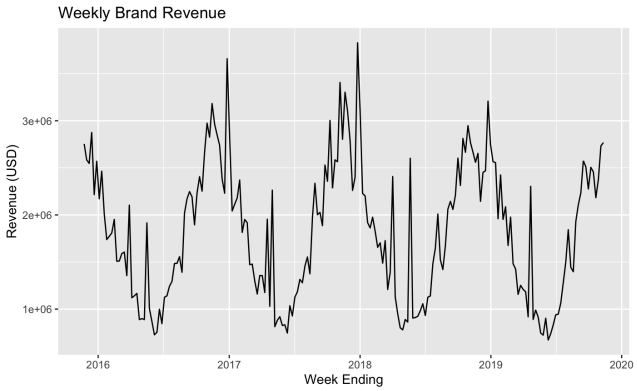

Figure 1: Revenue by Week

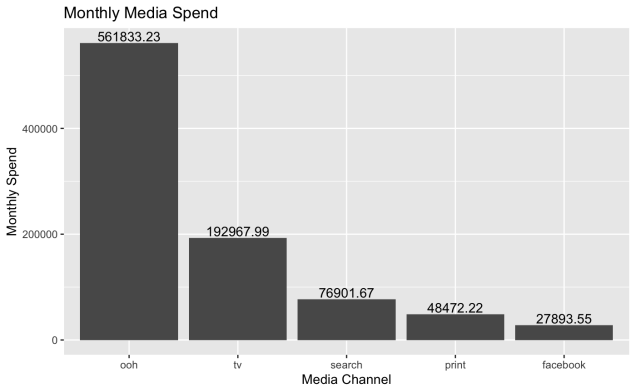

Figure 2: Average monthly media spend by channel

When looking at the correlation amongst the variables in the data, outside of the relationship between Search Spend and Search Clicks, and Facebook Spend and Facebook Impressions, all of the strongest relationships are with revenue.

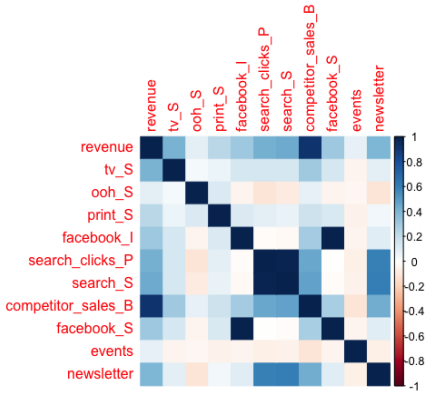

Figure 3: Correlation matrix of main features

Adding lag and decay to the variables increased the correlations even more, with TV Spend having a 0.73 correlation with revenue. The three macroeconomic variables all had a very low correlation with revenue, which makes sense as it is fake data and was not necessarily made trying to model a specific period of time or a specific country's economy.

When training the MMMs, the first step was finding the optimal Adstock rate for each of the five spend variables if it made sense to use one for the model. This was done by finding the set of rates that minimized the training error. The range of rates that were tested were chosen based on guidance given by Meta regarding where rates for different media channels commonly should be [9]. No adstock rate was applied to the final BSTS or Feed Forward Neural Network models as they learned better without the adstock applied. Additionally, a log plus one transformation was done on all of the media spend variables. The final adstock rates used can be seen in the table below:

	TV	FB	Print	Search	OOH
OLS	0.7	0.1	0.1	0	0.1
XGBoost	0.5	0.2	0.1	0.2	0.4
Robyn	0.44	0.15	0.16	0.13	0.39

Figure 4: Adstock rates used in final models

Feature selection was done using forward and backward stepwise feature selection, making sure the base model included all 5 media spend variables. In some cases, grid search and cross validation were also used to tune the hyperparameters and coefficients of the models. The training mean average percent error (MAPE), R^2, graphs of residuals, and autocorrelation were examined for all models to ensure the models were properly fitted to the

data. Once the models were trained, their forecasting ability was judged based on their MAPE on the test set.

For OLS and BSTS, the percent contribution of each variable was calculated by dividing the product of the relevant regression coefficients and the sum of their training values with the total predicted revenue for the training data. For XGBoost and the Feed Forward Neural Network, the percent contribution of each variable was calculated by dividing the variable's Shapley Value with the sum of all Shapley Values. The ROAS of each media variable was calculated by multiplying the percent contribution with the total predicted revenue, and dividing by the amount spent on that variable.

For OLS and BSTS, the optimal budget allocation for the test set was calculated by assigning each dollar to the channel that maximizes that dollar's marginal return. For XGBoost and the Feed Forward Neural Network, the optimal budget allocation for the test set was calculated by using a method called proportional budget allocation, where each channel's budget was proportional to their contribution to revenue during the training period. Different methods were done due to the inability to easily calculate marginal return for each dollar in the XGBoost and Feed Forward Neural Network models, as well as the final Feed Forward Neural Network model not including a log transformation on the data.

Once the optimal budget allocation was determined, the weekly spend values for each channel was allocated by ensuring the distribution of total monthly media spend matched the spending timeline in the test set.

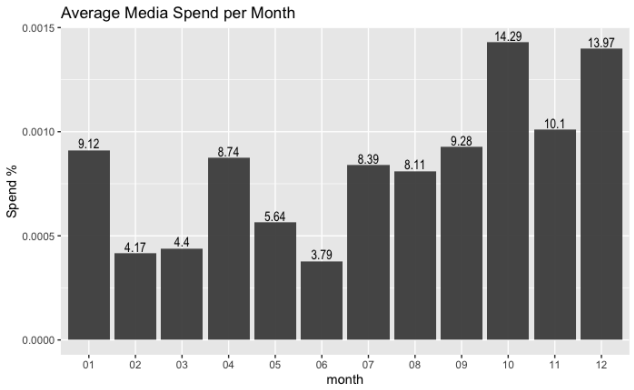

Figure 5: Distribution of media spend by month

The relevant transformations were done on the new media spend variables in the test set, and the model was used to generate new revenue predictions for the test set. The sum of the weekly predictions was compared to the actual revenue during the test period, as well as the original predicted revenue during the test period.

The Robyn model was trained by following Meta guides on how to use the package [14]. The package has pre built functions in order to choose Adstock Rates and tune all hyperparameters. The model was chosen out of the options given by minimizing the training MAPE. The test set used by Robyn was slightly different from the test sets, as Robyn uses the final 30 weeks compared to the final 52 weeks used for the rest of the models. Since they also use the 30 weeks prior to the test set as validation data, none of the test set that was used in the other models was used to train the Robyn model.

Empirical results:

The first model that was trained was the OLS regression model. This was trained using the caret package in R. Forward and backward stepwise regression was used, returning 11 features. The best model had an R^2 of 0.93, a training error of 6.35%, a testing error of 7.88%, and did not use any regularization. Its residuals were randomly scattered about zero and no autocorrelation.

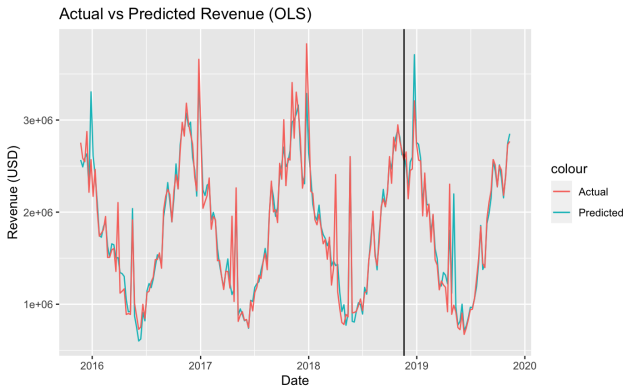

Figure 6: OLS forecast results

More details regarding media contribution and ROAS for all models can be seen in Figure 13, but overall advertising was attributed to 32.2% of training revenue. The optimal budget allocation gave 89% of the budget to TV, 10% of the budget to print, and 1% of the budget to search. This resulted in a predicted optimal revenue of $112,413,783.00, which was 21.1% higher than the actual test period revenue, and 17.5% higher than the predicted test period revenue with the original budget allocation.

The second model that was trained was BSTS. This was trained using the bsts package in R. The best model used all features and had local linear, seasonal, and regression components trained on 10,000 iterations. It had an R^2 of 0.92, a mean training error of 10.16% and a mean testing error of 7.42%. Its residuals were randomly scattered about zero and no autocorrelation.

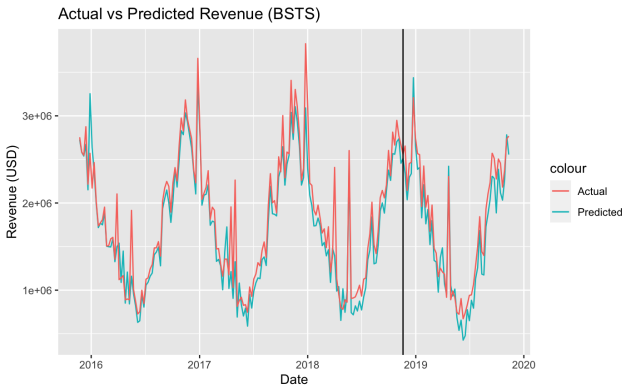

Figure 7: BSTS forecast results

Overall advertising was attributed to 4.3% of training revenue. The true model estimate could be higher as media is likely contributing to the 11.5% contribution coming from the trend component, however it cannot be broken out into specific categories. The optimal budget allocation gave 99% of the budget to TV, <1% of the budget to print, and <1% of the budget to Facebook. This resulted in a predicted optimal revenue of $100,213,277.00, which was 7.9% higher than the actual test period revenue, and 7.5% higher than the predicted test period revenue with the original budget allocation.

The third model trained was XGBoost. This was trained using the xgboost package in R. The best model considered only the same 11 features selected during OLS feature selection, and used grid search with 5 fold cross validation to tune the hyperparameters. It had a max depth of 3, a learning rate of 0.05, a subsample of 0.8, minimum child weight of 2, and both L1 and L2 regularization. It had an R^2 of 0.93, a training error of 8.04% and a testing error of 10.49%. Its residuals were randomly scattered about zero and no autocorrelation.

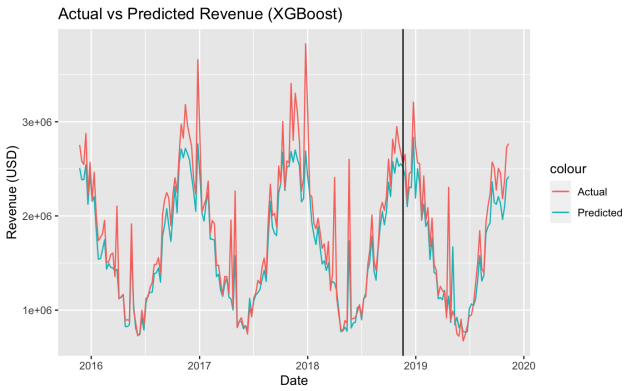

Figure 8: XGBoost forecast results

Overall advertising was attributed to 19.4% of training revenue. The optimal budget allocation gave 56.2% of the budget to TV, 24.2% of the budget to Facebook, 10.1% of the budget to Print, 4.2% of the budget to Search, and 5.3% of the budget to OOH. This resulted in a predicted optimal revenue of $98,978,954.00, which was 6.6% higher than the actual test period revenue, and 15.4% higher than the predicted test period revenue with the original budget allocation.

The next model that was trained was the Feed Forward Neural Network. This was trained using the TensorFlow library in Python. Due to the small amount of data, the architecture of the model was quite simple, with one dense layer of 64 nodes with L2 regularization, relu activation functions, "adam" optimization function, and mean squared error loss function. The training data was randomly split into a 9-1 training and validation data split, and used all features. It had an R^2 of 0.86, a training error of 7.45% and a testing error of 7.03%. Its residuals were randomly scattered about zero and no autocorrelation.

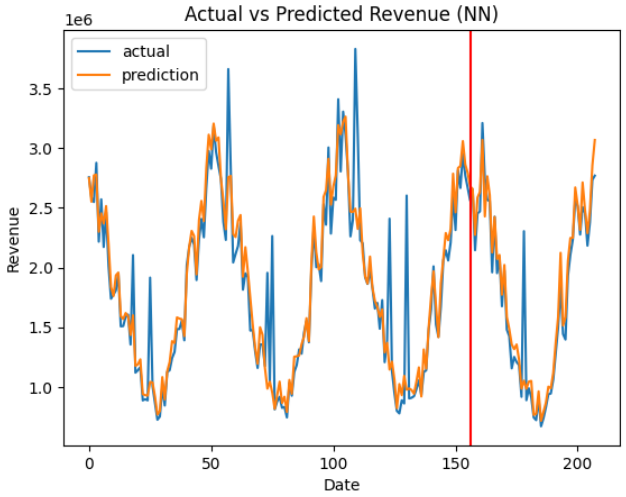

Figure 9: Neural Network forecast results

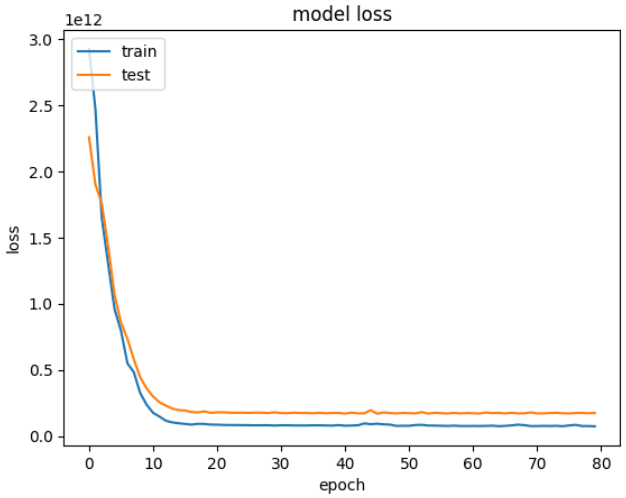

Figure 10: Training and validation loss by epoch

Overall advertising was attributed to 8.8% of training revenue. The optimal budget allocation gave 52.8% of the budget to OOH, 20.9% of the budget to TV, 18.4% of the budget to Facebook, and 7.9% of the budget to Print. This resulted in a predicted optimal revenue of $141,546,962.00, which was 52.5% higher than the actual test period revenue, and 47.8% higher than the predicted test period revenue with the original budget allocation.

The final model trained was the Robyn model. This was trained using the Robyn package in R. The package does all of the preprocessing, feature selection, and training on its own with few hyperparameters needed to be manually tuned. It automatically incorporates a 72-14-14 train-validation-test split, and outputs multiple model options. The best model was chosen based on training and validation MAPE. It had an R^2 of 0.95, a training error of 5.24%, a validation error of 3.85%, and a testing error of 5.17%.

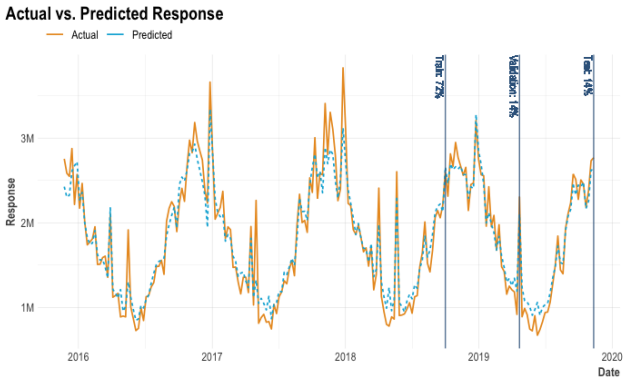

Figure 11: Robyn model results

Overall advertising was attributed to 8.2% of training revenue. The optimal budget allocation gave 53.7% of the budget to OOH, 25.5% of the budget to TV, 11.7% of the budget to Search, 5.74% of the budget to Print, and 3.28% of the budget to Facebook. This resulted in a predicted optimal revenue of $166,099,314.00, which was 78.9% higher than the actual test period revenue, and 70.6% higher than the predicted test period revenue with the original budget allocation.

A summary of model fit, forecasting performance, and optimized budget can be seen below:

	R^2	Train MAPE	Test MAPE	Opt. Budget
OLS	0.94	6.4%	7.9%	+21.1%
BSTS	0.92	10.3%	7.4%	+7.9%
XGBoost	0.93	8.0%	10.5%	+6.6%
FNN	0.86	7.4%	7.0%	+52.5%
Robyn	0.95	5.2%	5.2%	+78.9%

Figure 12: Summary of model results

It is entirely unsurprising that Robyn performed the best, as the data used for the project was crafted with the exclusive purpose of showing how well their package performs (a package that's been in development for years made by a trillion dollar company). XGBoost performed the worst in terms of forecasting, which is also unsurprising as that was the same case in Wigren's paper.

It was surprising that the Feed Forward Neural Network performed the best. As previously stated, there has been little to no research on the use of deep learning for neural networks. Combining that with the fact that there were only 156 weeks of training data led to the belief that a neural network would not generalize well to the test set, but it did. In terms of forecasting, its only fault was that it was unable to capture any of the spikes in sales that occurred around New Year's and Mother's Day. This was surprising as all other models captured that relationship. Looking at the train/val loss curves, while training was stopped at 80 epochs as the training loss began decreasing faster than the validation loss, it was still not drastic. It is therefore possible that the model could have benefited from more epochs and eventually have learned the spikes in sales that occur on certain holidays.

The ROAS and percent contribution for each media channel for each model can be seen below:

	TV	FB	Print	Search	OOH
OLS	1718 (32%)	-852 (-2%)	469 (2.3%)	46 (<1%)	-0.7 (-10%)
BSTS	154 (4.1%)	0.4 (~0%)	0.7 (~0%)	2.5 (~0%)	-0.1 (~0%)
XGBoost	380 (11%)	1202 (3%)	268 (2%)	96 (<1%)	11 (1%)
FNN	31 (<1%)	2658 (9%)	264 (1.8%)	-229 (-1.8%)	-16 (-1.5%)
Robyn	44 (1.1%)	7 (<1%)	92 (<1%)	46 (<1%)	44 (4%)

Figure 13: Summary of model ROAS and contributions

Looking across media channels and models, there are little trends to be found, other than most media channels do not have a very good ROAS (ROAS of 100 means every $100 spent generates $100 of revenue). This in and of itself is not necessarily unusual or concerning. Although

advertising may not break even in the short term, the long term effects are worth it as removing all advertising would greatly diminish brand awareness, which would greatly diminish sales in the long run [1].

However, the fact that there are not consistent estimates of ROAS for each media channel brings upon a larger problem regarding MMMs and their usage. There has been plenty of research on the problems with MMMs as a whole from a data science perspective. It is not unusual for MMMs to return ROASs that are not consistent with internal expectations based on prior lift studies, and it is likewise not unusual for companies to force ROASs in MMMs to meet their predetermined expectations rather than letting the data speak for itself. Much of this problem comes from the difficulty of determining single source attribution for sales in most companies, due to the fact that consumers are complex and have countless different sources affecting their purchasing decisions [2].

Conclusions / Future Directions

The models made in this project were able to fit the data well and forecast the training set with <10% error. The performance of the different algorithms mimicked what prior research has seen, with BSTS and OLS performing well, and XGBoost performing poorly. Combining deep learning methods with the game theory concept of Shapley Values showed great performance in terms of forecasting and budget optimization, but, like all other models, the ROASs and contributions were a bit wonky. This all leads to my general belief that MMMs are not quite in a place where they are proper data science solutions. Two big reasons for this are a lack of data, and trying to do too much with one model. Companies only having a few years worth of weekly level data makes it difficult to properly adhere to standard data science guidelines, especially when they are trying to use 2-4x more features than the general "~10 observations per feature" rule. Additionally, trying to both accurately state media channels historical ROASs and accurately forecast is a tricky problem. In order to describe a media channel's historical performance as accurately as possible requires the models to be very fit to the training data, but that can be detrimental for generalizing to new data.

All of this is what leads to some questionable methods done by some major MMM providers. Some ignore what the data and models are saying ROASs are and force them to what they believe they should be, hindering the models ability to learn and going against the point of machine learning. Some also utilize no test set or cross validation, and train the model using 100% of the data, deceptively advertising their forecasting ability with their model's training error. These sorts of methodologies lead me to believe that companies are jumping to MMMs as a machine learning solution before they have enough quality data and the systems in place to do so.

On the brighter side of MMMs, I believe once companies acquire more years worth of online marketing data, these sorts of methodologies will become a thing of the past. Based on the performance of the FNN, and the trends of most problems solved using machine learning, I believe that the combination of deep learning and additional explanatory analysis will be the future of MMMs.

If I had more time on this project, I would like to further explore the possibilities of utilizing deep learning for MMMs. Additionally, I would like to further explore exactly what the Robyn package is doing behind the scenes and how to better replicate their results. Advice for future students taking the course would be to make sure they are utilizing high quality data, and the right data for the problem they are trying to solve. Attempting to make sense of the randomly generated data from the "dammmdatagen" package rather than immediately trying to find higher quality data was a mistake. Additionally, I would recommend keeping track of how much time you spend on the project. I found it helpful to log my hours every time I worked on the project to ensure I spent 10-15 hours a week to keep up with the workload.

GitHub / Instructions

https://github.com/timcaul/marketing_mix_modeling

The GitHub for this project has 3 folders: one with all code from the project, one with all the data required to run the code, and one with the paper and presentation. The only alteration needed to run the code is to change the source of the data in the read_csv lines to your local directory.

Sources

[1] Abraham, M. (1990). *Getting the Most Out of Advertising and Promotion.* Harvard Business Review.

[2] Chan. D. (2017). Challenges and Opportunities in Media Mix Modeling. Google Inc.

[3] Jin, Y. (2017). Bayesian Methods for Media Mix Modeling with Carryover and Shape Effects. Google Inc.

[4] Karlsson, J. (2022). *Bayesian Structural Time Series in Marketing Mix Modelling.* KTH Royal Institute of Technology

[5] Pandey, A. (2020). *Explainable AI: Application of Shapley Values in Marketing Analytics.* Medium

[6] Samudrala, A. (2018). *Unveiling Mathematics behind XGBoost.* Medium.

[7] Scott, S. (2014). *Predicting the Present with Bayesian Structural Time Series.* Inderscience Publishers Ltd: 4–23.

[8] Wigren, R. (2019). *Marketing Mix Modeling: A comparative study of statistical models.* Linköping University

[9] *Analysts guide to MMM.* (2021). Meta. facebookexperimental.github.io/Robyn

[10] *Consumer Confidence Index.* (2023). OECD.

[11] *Consumer Price Index.* (2023). US Bureau of Labor and Statistics

[12] *Gross Domestic Product.* (2023). US Bureau of Economic Analysis

[13] *Robyn.* (2021). Meta. facebookexperimental.github.io/Robyn

[14] *Robyn Demo.* (2021). Meta. facebookexperimental.github.io/Robyn