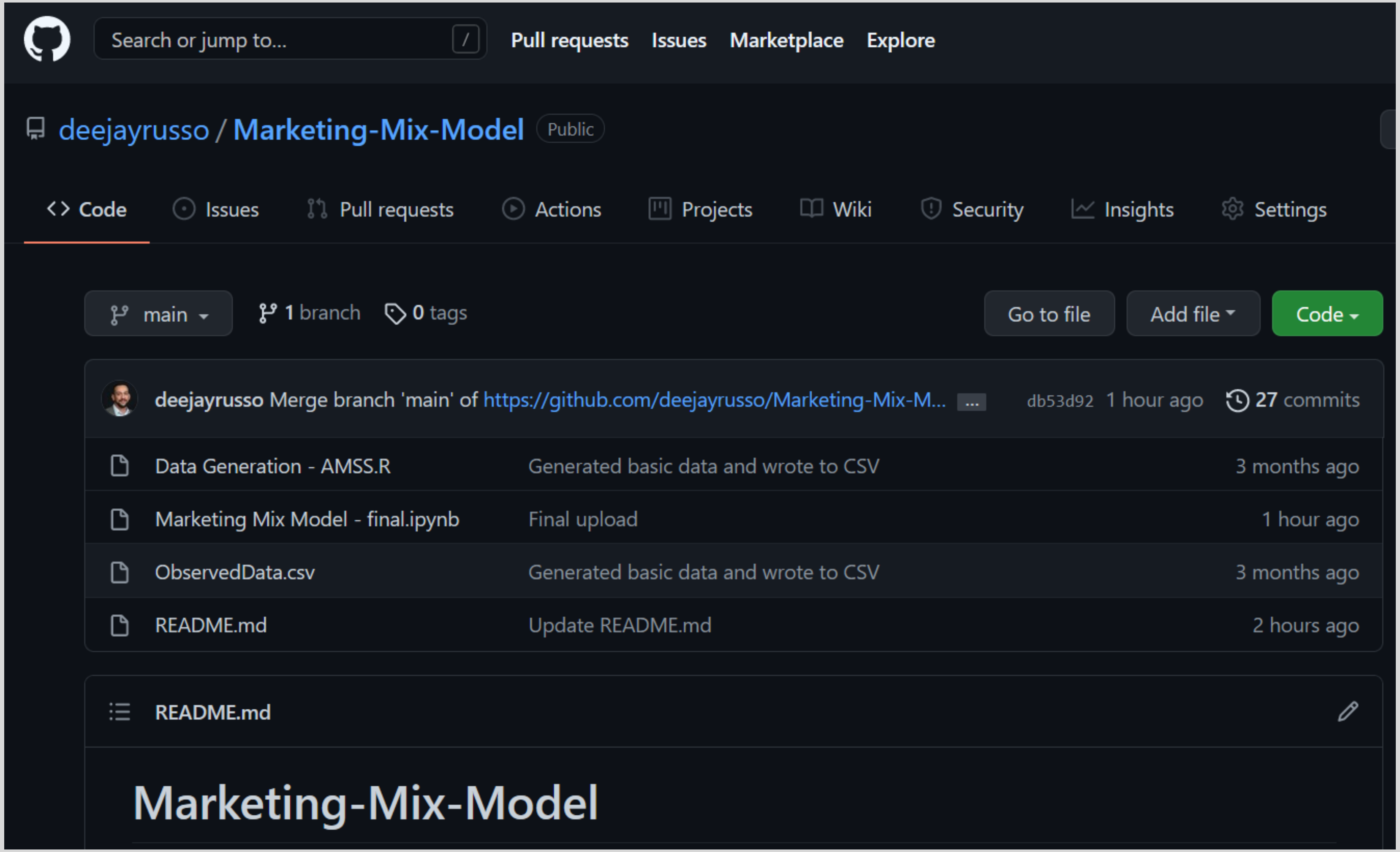
MARKETING MIX MODEL

Dominick Russo
Muma College of Business
ISM6905: Integrative Project
Master of Science, Business Analytics and Information Systems
December 6, 2021

GitHub Repository



Marketing Mix Models

- Advertisers use marketing mix models to measure the effectiveness of various advertising channels on improving a metric [1], such as sales or return on investment (ROI).
- These models use time series data to model an outcome resulting from advertising variables, usually marketing or media spend [1].

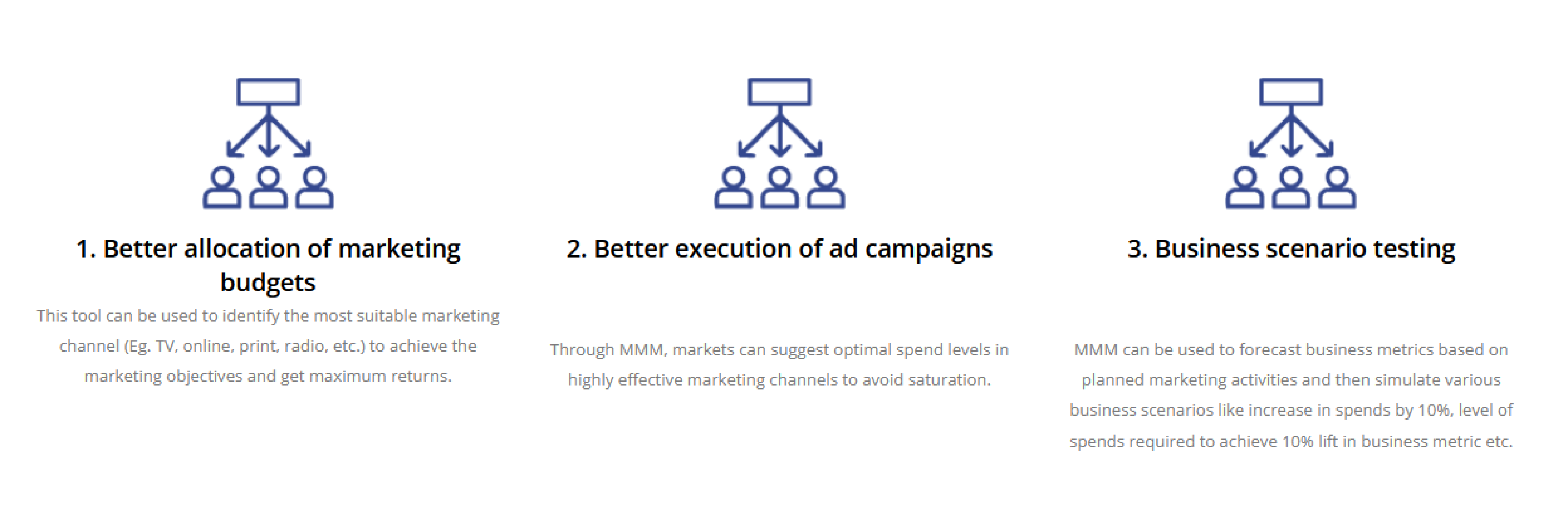


Image source: https://www.latentview.com/marketing-mix-modeling/

Purpose

- The purpose of this project is to explore building a marketing mix model using simulated weekly marketing and sales data. The model will attempt to predict a change in sales volume based on changes in spend for TV and paid search advertisements.
- By using statistical analysis to estimate impacts of marketing tactics and predict future impacts, marketing mix models help [3]:
  - Choose the most effective media environments to use
  - Optimize amount of each media used
  - Maximize marketing spend across all media
    - Prevent spending past diminishing returns
  - Optimize marketing strategy
    - Does running ads continuously or on a flighting schedule work best? For which media?
  - Optimize impact of advertisements and know when they have become “worn out”.

Data

- Generated using the Aggregate Marketing System Simulator (AMASS) [5]
  - Open-source R package provided by Google and available on GitHub [2].
- Variables used:
  - **tv.spend:** Weekly dollar spend on TV advertisement
  - **search.spend:** Weekly dollar spend on paid search advertisement
  - **revenue:** Weekly dollar sales
- Total rows generated: 208 (4 years weekly data).
- 104 rows dropped
  - First 52 rows dropped to allow the simulator time to normalize and produce more consistent data.
  - Next 52 rows dropped after feature engineering to remove null values produced after shifting data to create lag variables, leaving
- Final data: 104 rows (2 years weekly data).
- View data

Method

Tools Used:

- R: data generation
- Python:
  - Data exploration
  - Build/evaluate models

Generate & load data
- Simulated data

Explore data

Feature selection
- Revenue as a function of marketing spend

Feature engineering
- Lag variables
- Rolling averages

Data prep
- 70/30 train/test split

Data pipeline
- Scale
- Normalize

Explore models
- Various regression models:
  - OLS
  - Polynomial
  - Lasso
  - Ridge
  - Elastic Net
  - SGD

Evaluate models
- RMSE
- MAPE
- R^2

EDA Findings

- Revenue:
  - Weekly range: $44MM - $171MM
  - Weekly average: $99MM
- TV spend:
  - Weekly range: $0 - $5.4MM
  - Weekly average: $865K
  - **Key finding:** Spending on < 50% of weeks
- Search spend
  - Weekly range: $0 - $904K
  - Weekly average: $348K
  - **Key finding:** Spending on up to 75% of weeks

	tv.spend	**search.spend**	**revenue**
count	156	156	156
mean	865,385	347,564	99,329,390
std	1,278,889	220,656	32,027,560
min	0	0	44,910,960
25%	0	0	70,138,040
50%	0	353,900	96,423,200
75%	1,515,215	481,178	124,820,900
max	5,446,534	903,994	171,843,900

Feature Engineering

These features were created to account for seasonality and other trends:

- tv.spend, search.spend, and revenue are shifted 52 weeks to create one year lag variables (_lag1y) for seasonality.
- Revenue will also be shifted one week (_lag1w) and three months (lag3m) to explore effects of delayed purchase after interacting with an advertisement.
- To explore advertising trends, rolling averages for both tv.spend and search.spend are calculated for one week (_p1w), three months (_p3m) , and one year (_p1y).

```
# Create lag variables for 1y for revenue and channels to look for seasonality

sale_spend = pd.concat([sale_spend, sale_spend.shift(52).add_suffix('_lag1y')], axis=1)

sale_spend['revenue_lag1w'] = sale_spend['revenue'].shift()
sale_spend['revenue_lag3m'] = sale_spend['revenue'].shift(13)
```

```
# Create rolling average variable for 1y, 3m, and 1w for all channel spend

sale_spend = pd.concat([sale_spend, sale_spend.rolling(1).mean().add_suffix('_p1w'),
                        sale_spend.rolling(13).mean().add_suffix('_p3m'),
                        sale_spend.rolling(52).mean().add_suffix('_p1y')], axis=1)
```

Model Selection

- Models explored:
  - Ordinary least squares regression
  - Polynomial regression (3rd order)
  - Lasso regression
  - Ridge regression
  - Elastic net regression
  - Stochastic gradient descent regressor

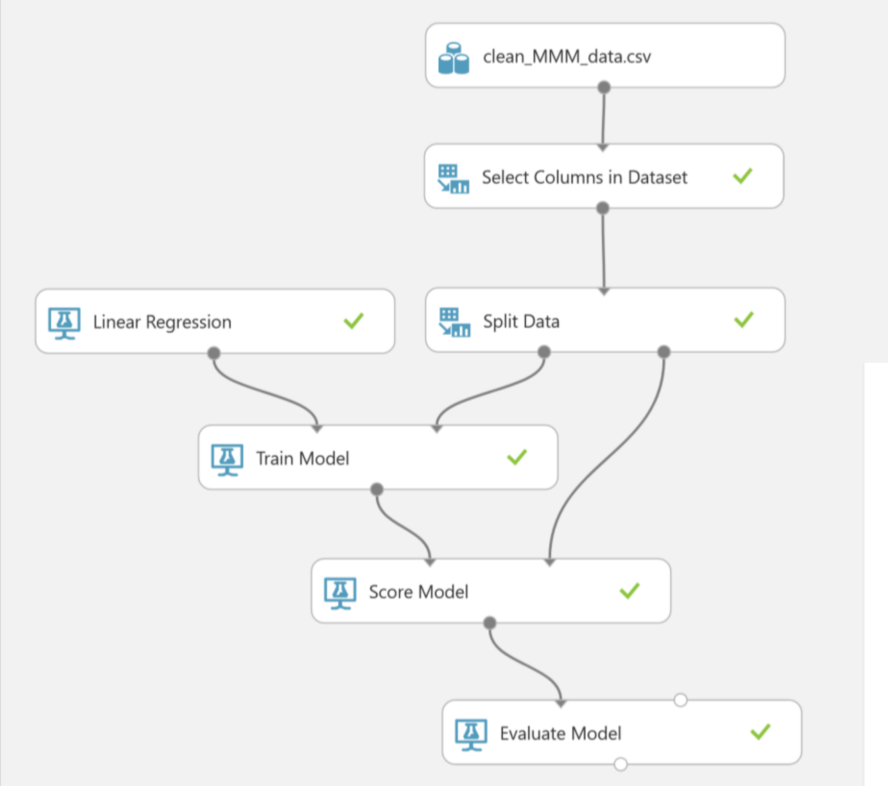
Model Evaluation

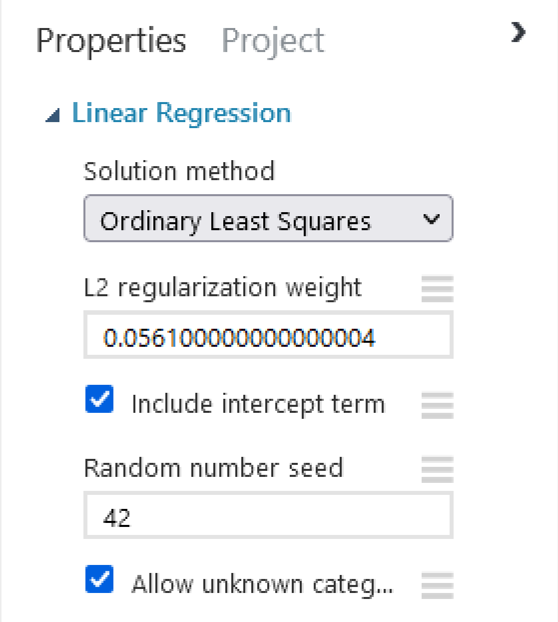
Best: Lasso regression

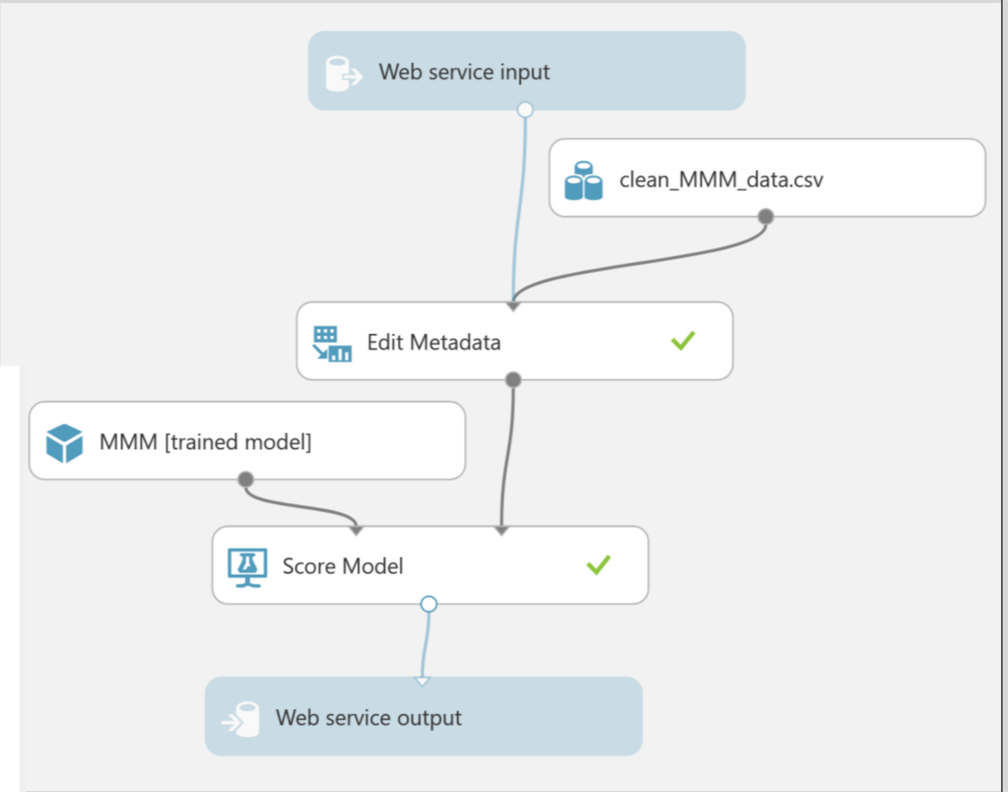
Lowest RMSE: lasso (7,071,077.81)
Lowest MAPE: elastic-net (0.0541)
Highest R^2: lasso (0.95197444)

	lin_reg_lag	elastic_net	SGD	lasso	ridge
0	{'RMSE': 7071079.569844888}	{'RMSE': 7085549.06971255}	{'RMSE': 7503748.643524809}	{'RMSE': 7071077.80695873}	{'RMSE': 7080237.906924615}
1	{'Train MAPE': 0.0515058420916952}	{'Train MAPE': 0.05223132259513465}	{'Train MAPE': 0.05581596683761173}	{'Train MAPE': 0.051505852898839494}	{'Train MAPE': 0.05203242934704791}
2	{'Test MAPE': 0.054165981814058364}	{'Test MAPE': 0.05409778100531319}	{'Test MAPE': 0.056624997491201104}	{'Test MAPE': 0.05416595858404647}	{'Test MAPE': 0.0541155086173926}
3	{'R2': 0.9519744132901219}	{'R2': 0.9517776633583747}	{'R2': 0.9459173720730838}	{'R2': 0.9519744372365733}	{'R2': 0.9518499289479136}

Azure Webservice Example





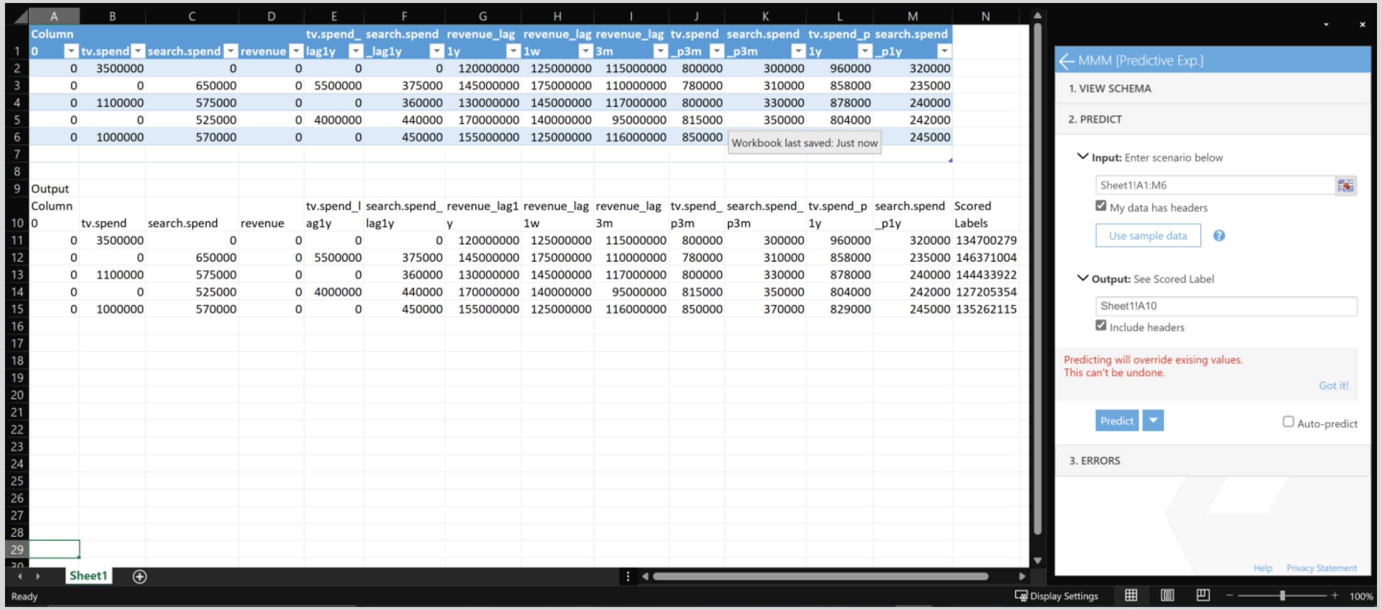


Build similar ridge regression

Deploy trained model as web service

Demo

Play →

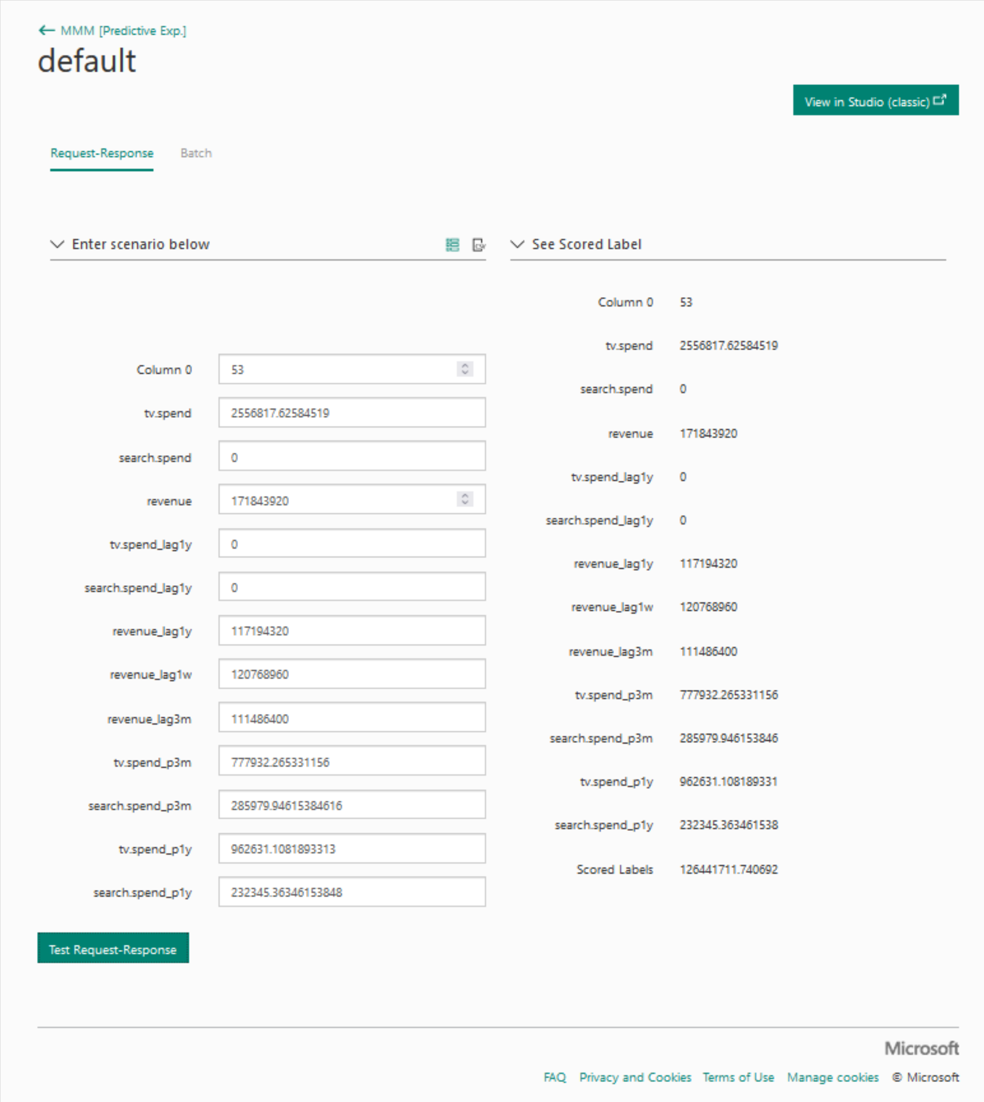


Column 0	tv.spend	search.spend	revenue	tv.spend_lag1y	search.spend_lag1y	revenue_lag1y	revenue_lag1w	revenue_lag3m	tv.spend_p3m	search.spend_p3m	tv.spend_p1y	search.spend_p1y
0	3500000	0	0	0	0	120000000	125000000	115000000	800000	300000	960000	320000
0	0	650000	0	5500000	375000	145000000	175000000	110000000	780000	310000	858000	235000
0	1100000	575000	0	0	360000	130000000	145000000	117000000	800000	330000	878000	240000
0	0	525000	0	4000000	440000	170000000	140000000	95000000	815000	350000	804000	242000
0	1000000	570000	0	0	450000	155000000	125000000	116000000	850000			245000

Workbook last saved: Just now

Output

Column 0	tv.spend	search.spend	revenue	tv.spend_lag1y	search.spend_lag1y	revenue_lag1y	revenue_lag1w	revenue_lag3m	tv.spend_p3m	search.spend_p3m	tv.spend_p1y	search.spend_p1y	Scored Labels
0	3500000	0	0	0	0	120000000	125000000	115000000	800000	300000	960000	320000	134700279
0	0	650000	0	5500000	375000	145000000	175000000	110000000	780000	310000	858000	235000	146371004
0	1100000	575000	0	0	360000	130000000	145000000	117000000	800000	330000	878000	240000	144433922
0	0	525000	0	4000000	440000	170000000	140000000	95000000	815000	350000	804000	242000	127205354
0	1000000	570000	0	0	450000	155000000	125000000	116000000	850000	370000	829000	245000	135262115



Recommendations

- Full interpretation of applying the model to guide decision making
  - Further research into Yeo-Johnson transformations due to different transformations applied to positive and negative numbers [4].
  - Instead, explore using more easily explained transformations (ie: log, square-root, etc.) if error is not increased.
- Exploring the use of an ensemble model
  - Create additional model for lag effects of advertising
  - Create additional model for diminishing returns on advertising spend
  - A base sales model could also be used to simulate lift over base sales
  - Stack additional models with media mix model to explore modeling more complex interactions
- Dashboard or web application for stakeholder interface
  - Deploy model with user interface once models are acceptable for production
  - Options could include using a Model-View-Control (MVC) framework in C# or Model-Template-View (MTV) framework in python (Django)
  - Dashboarding BI tools can also be used (PowerBI, Looker, etc.)
  - User interface should allow for input of spend scenarios to simulate revenue outcomes to help stakeholders optimize marketing budgets across channels.
    - This will help drive decisions to fund each channel sufficiently without overspending past the point of diminishing returns.
  - Interface could also use data entry fields to add new actual spends and revenue to keep data current on a weekly basis.

Sources

- [1] Chan, D., & Perry, M. (2017). Challenges and opportunities in media mix modeling. https://services.google.com/fh/files/misc/challenges_and_opportunities_in_media_mix_modeling.pdf
- [2] Google, Inc. (2017) Google AMSS GitHub page. https://github.com/google/amss/blob/master/vignettes/amss-vignette.Rmd
- [3] Nielsen. (2014). Marketing Mix Modeling: What Marketing Professionals Need To Know. https://www.nielsen.com/wp-content/uploads/sites/3/2019/04/marketing-mix-modeling-what-marketers-need-to-know.pdf
- [4] Yeo, I.-K., & Johnson, R. A. (2000). A New Family of Power Transformations to Improve Normality or Symmetry. Biometrika, 87(4), 954–959. http://www.jstor.org/stable/2673623
- [5] Zhang, S. and Vaver, J. (2017). The Aggregate Marketing System Simulator. *https://research.google.com/pubs/pub45996.html*.