Dynamic Marketing Policies:
Constructing Markov States for Reinforcement Learning

Appendix

May 2020

Yuting Zhu
MIT

Duncan Simester
MIT

Jonathan Parker
MIT

Antoinette Schoar
MIT

Contents

All Tests of Length Two

All tests with length two include:

$q_5 = "0000" = \{a^1 = Not\ mail, o^1 = Not\ buy, a^2 = Not\ mail, o^2 = Not\ buy\}$
$q_6 = "0001" = \{a^1 = Not\ mail, o^1 = Not\ buy, a^2 = Not\ mail, o^2 = Buy\}$
$q_7 = "0010" = \{a^1 = Not\ mail, o^1 = Not\ buy, a^2 = Mail, o^2 = Not\ buy\}$
$q_8 = "0011" = \{a^1 = Not\ mail, o^1 = Not\ buy, a^2 = Mail, o^2 = Buy\},$
$q_9 = "0100" = \{a^1 = Not\ mail, o^1 = Buy, a^2 = Not\ mail, o^2 = Not\ buy\}$
$q_{10} = "0101" = \{a^1 = Not\ mail, o^1 = Buy, a^2 = Not\ mail, o^2 = Buy\}$
$q_{11} = "0110" = \{a^1 = Not\ mail, o^1 = Buy, a^2 = Mail, o^2 = Not\ buy\}$
$q_{12} = "0111" = \{a^1 = Not\ mail, o^1 = Buy, a^2 = Mail, o^2 = Buy\}$
$q_{13} = "1000" = \{a^1 = Mail, o^1 = Not\ buy, a^2 = Not\ mail, o^2 = Not\ buy\}$
$q_{14} = "1001" = \{a^1 = Mail, o^1 = Not\ buy, a^2 = Not\ mail, o^2 = Buy\}$
$q_{15} = "1010" = \{a^1 = Mail, o^1 = Not\ buy, a^2 = Mail, o^2 = Not\ buy\}$
$q_{16} = "1011" = \{a^1 = Mail, o^1 = Not\ buy, a^2 = Mail, o^2 = Buy\}$
$q_{17} = "1100" = \{a^1 = Mail, o^1 = Buy, a^2 = Not\ mail, o^2 = Not\ buy\}$
$q_{18} = "1101" = \{a^1 = Mail, o^1 = Buy, a^2 = Not\ mail, o^2 = Buy\}$
$q_{19} = "1110" = \{a^1 = Mail, o^1 = Buy, a^2 = Mail, o^2 = Not\ buy\}$
$q_{20} = "1111" = \{a^1 = Mail, o^1 = Buy, a^2 = Mail, o^2 = Buy\}$

Higher length tests can be constructed similarly.

Dealing with Estimation Errors When Finding Core Tests

To manage the effect of predicted probabilities' estimation errors in finding core tests, we take a relatively conservative approach to calculate the rank and find linearly independent columns of the submatrices in the system dynamic matrix.

1. If the removal of a column causes the rank of the resulting matrix to decrease, that column belongs to the independent columns. We use this criterion to define the linearly independent columns, whose corresponding tests are used for one-step expansions.
2. A separate issue arises in obtaining the rank of the resulting matrix before and after the deletion of each column. Again, we use a relative conservative approach proposed by Golub and Van Loan (2013). The method uses singular value decomposition and uses a singular value cutoff σ_{cutoff}. The estimated rank is the number of singular values above the cutoff σ_{cutoff}. For a $(m \times n)$ matrix A, with $m \geq n$, $\sigma_{cutoff} = \varepsilon ||A||_{\infty}$, where ε is the average error in the matrix entries. The ε term is calculated using Chebyshev's inequality. Chebyshev's inequality delivers a bound on the maximal error. We use this error bound as the average error. In other words, the average error is upper bounded with a given certainty. We choose 0.95 as the certainty measure, which implies that with probability 0.05 the error estimate is worse than ε.

An Example Illustrating the 5-Step Core Tests Algorithm

We illustrate the proposed five-step algorithm for finding core-test by assuming we know the true probabilities in the system dynamic matrix. In practice, we replace the true probabilities with estimated probabilities using the method described in the second subsection of Section 5. In each step we focus on all probabilities that we can estimate in the system dynamic matrix. In our example we will assume that action takes two values: $a = \{Not\ Mail, Mail\} = \{0,1\}$ and observation takes two levels $o = \{Not\ Buy, Buy\} = \{0,1\}$.

1. In the first row of the system dynamic matrix we delete history rows whose corresponding tests have zero probabilities at zero-length history (in the first row). The example below reports hypothetical values for all tests up to length two, and transposes the first row of the matrix (to simplify presentation):

	$\emptyset$
$\boldsymbol{q_1} = \mathbf{"00"}$	0.4
$\boldsymbol{q_2} = \mathbf{"01"}$	0.4
$\boldsymbol{q_3} = \mathbf{"10"}$	0.1
$\boldsymbol{q_4} = \mathbf{"11"}$	0.1
$\boldsymbol{q_5} = \mathbf{"0000"}$	0.15
$\boldsymbol{q_6} = \mathbf{"0001"}$	0.15
$\boldsymbol{q_7} = \mathbf{"0010"}$	0.05
$\boldsymbol{q_8} = \mathbf{"0011"}$	0.05
$\boldsymbol{q_9} = \mathbf{"0100"}$	0.15
$\boldsymbol{q_{10}} = \mathbf{"0101"}$	0.15
$\boldsymbol{q_{11}} = \mathbf{"0110"}$	0.05
$\boldsymbol{q_{12}} = \mathbf{"0111"}$	0.05
$\boldsymbol{q_{13}} = \mathbf{"1000"}$	0.05
$\boldsymbol{q_{14}} = \mathbf{"1001"}$	0.05
$\boldsymbol{q_{15}} = \mathbf{"1010"}$	0
$\boldsymbol{q_{16}} = \mathbf{"1011"}$	0
$\boldsymbol{q_{17}} = \mathbf{"1100"}$	0.05
$\boldsymbol{q_{18}} = \mathbf{"1101"}$	0.05
$\boldsymbol{q_{19}} = \mathbf{"1110"}$	0
$\boldsymbol{q_{20}} = \mathbf{"1111"}$	0

We would delete corresponding histories $h_{15} = "1010"$, $h_{16} = "1011"$, $h_{19} = "1110"$ and $h_{20} = "1111"$. More precisely, in later steps, we do not consider these history rows.

2. We will focus on the submatrix containing all tests up to length one. In our example, this submatrix takes the following form:

	$q_1 = \text{"00"}$	$q_2 = \text{"01"}$	$q_3 = \text{"10"}$	$q_4 = \text{"11"}$
$\emptyset$	0.4	0.4	0.1	0.1
$h_1 = \text{"00"}$	0.375	0.375	0.125	0.125
$h_2 = \text{"01"}$	0.375	0.375	0.125	0.125
$h_3 = \text{"10"}$	0.5	0.5	0	0
$h_4 = \text{"11"}$	0.5	0.5	0	0
$h_5 = \text{"0000"}$	0.375	0.375	0.125	0.125
$h_6 = \text{"0001"}$	0.375	0.375	0.125	0.125
$h_7 = \text{"0010"}$	0.5	0.5	0	0
$h_8 = \text{"0011"}$	0.5	0.5	0	0
$h_9 = \text{"0100"}$	0.375	0.375	0.125	0.125
$h_{10} = \text{"0101"}$	0.375	0.375	0.125	0.125
$h_{11} = \text{"0110"}$	0.5	0.5	0	0
$h_{12} = \text{"0111"}$	0.5	0.5	0	0
$h_{13} = \text{"1000"}$	0.375	0.375	0.125	0.125
$h_{14} = \text{"1001"}$	0.375	0.375	0.125	0.125
$h_{17} = \text{"1100"}$	0.375	0.375	0.125	0.125
$h_{18} = \text{"1101"}$	0.375	0.375	0.125	0.125

We calculate the rank of this matrix, which is two. We can also identify the linearly independent columns, which may not be unique. For example, q_1 and q_3 is one set of linearly independent columns. See also the discussion in the next section of the Appendix titled: "Dealing with Estimation Errors When Finding Core Tests".

3. We expand the submatrix to include the union of all length-one tests, the linearly independent tests found so far, and all one-step extensions of these independent tests. Suppose we take q_1 and q_3 as the linearly independent columns. The one step extensions of q_1 are $\{q_5, q_9, q_{13}, q_{17}\}$ and the one-step extensions of q_3 are $\{q_7, q_{11}, q_{15}, q_{19}\}$. As a result, the submatrix takes the same rows as the previous submatrix, while the columns become $\{q_1, q_2, q_3, q_4, q_5, q_7, q_9, q_{11}, q_{13}, q_{15}, q_{17}, q_{19}\}$.

4. We calculate the rank and linearly independent columns of this new submatrix.

5. We repeat Steps 3 and 4 until the rank does not change.

Proof of Result 1

Let $H(L)$ denote the total number of tests with length L, and $G(L)$ denote the number of L-length tests with zero probability at zero history. Suppose the first zero probability test at zero history has $\underline{L}$ length. Our goal is to prove $G(L+1)/G(L) > H(L+1)/H(L)$ for $\geq \underline{L}$. We focus on $L \geq \underline{L}$ because $G(L+1)/G(L)$ is undefined for $L < \underline{L}$.

For a test q', we define $q'ao$ as one-step *expansion*, and aoq' as one-step *extension*. Thus, if a test q' has zero probability at zero history, all one-step expansions of test q' ($q'ao$), and all one-step extensions of test q' (aoq') have zero probability at zero history. The intuition is that if we imagine an infinite dataset, zero probability of test q' at zero history means that we will not observe any pieces of q' in this infinite dataset. Thus, any pieces of $q'ao$ and aoq' will not exist in this infinite dataset.

We next focus on $H(L+1)/H(L)$ and $G(L+1)/G(L)$. Suppose n is the number of all possible length-one tests. Then, we know that $H(L+1) = nH(L)$ for all L. Deriving the explicit form of $G(L+1)/G(L)$ is more complicated. Instead, our goal is to show that $G(L+1)/G(L) > n$ for all $L \geq \underline{L}$.

We first establish that $G(L+1) \geq nG(L)$. For each zero-probability test q' with length L at zero history, there are n one-step extensions that also have zero probability. Moreover, for different $q'' \neq q'$ with length L, there are no repetitions between aoq' and aoq''. This implies that $G(L+1) \geq nG(L)$. We next show that $G(L+1) \geq nG(L)+1$. If this is the case, we know that $G(L+1)/G(L) > n$, and so the number of zero probability tests (at zero history) scales faster than the total number of tests: $G(L+1)/G(L) > H(L+1)/H(L)$.

To establish that $G(L+1) \geq nG(L)+1$, it is sufficient that there exists a one-step expansion of a zero probability test $q'(q'ao)$ that does not replicate any of the one-step extensions of the zero probability length L tests at zero history. This holds as long as $G(L) < H(L)$, or equivalently, as long as not all tests with length L have zero probability at zero history. We next prove this argument.

If a zero probability length L test has a non-repetitive one-step expansion that does not replicate any of the one-step extensions of the zero probability length L tests (at zero history), it means that this one-step expansion will replicate one of the one-step extensions of the non-zero probability length L tests. That is, we need a full match of two length $L+1$ tests. We will initially focus on $L = \underline{L}$.

Define $p^{\underline{L}}$ as a length $\underline{L}$ test with zero probability (at zero history). Define $p^{\underline{L}-1}$ as the length $\underline{L}-1$ test that contains the last $\underline{L}-1$ elements of $p^{\underline{L}}$. Because $p^{\underline{L}}$ is the shortest test with zero probability, test $p^{\underline{L}-1}$ has positive probability (at zero history). In particular, we denote $p^{\underline{L}} = \widehat{ao}p^{\underline{L}-1}$. We know there is a one-step expansion $p^{\underline{L}-1}ao$ that has non-zero probability. This is an L-length test with non-zero probability (at

zero history). We also know that the "$\widehat{ao}$" one-step extension of $p^{\underline{L}-1}ao$ ($\widehat{ao}p^{\underline{L}-1}\ ao$) has zero probability because $p^{\underline{L}}$ has zero probability. We are done with $\underline{L}$ length tests.

For $\underline{L}$ + 1 length tests, we know that there is a test $p^{\underline{L}-1}aoa'o'$ that has non-zero probability, and that the test $\widehat{ao}p^{\underline{L}-1}\ aoa'o'$ has zero probability. We can continue this for all $L > \underline{L}$. This implies that $G(L+1) \geq nG(L) + 1$ for all $L > \underline{L}$. We are done with the proof.

Proof of Result 2

By deleting all history rows whose corresponding tests have zero probability at zero history, we delete history rows with all zero entries because of the impossibility of the existence of such history.

Suppose at iteration step k, a set $\hat{Q}$ of tests are returned, and at iteration step $k+1$, the algorithm stops because of unchanged rank. This means that columns corresponding to the tests in $\hat{Q}$ are a basis for all one-length tests and one-step extension of tests in $\hat{Q}$. This is guaranteed by the way we expand the submatrix. Formally speaking, for all history h (including the deleted ones), all tests $\hat{q} \in \hat{Q}$ and all length-one action-observation possibilities ao, there exists parameter vectors $m_{ao\hat{q}}$ and m_{ao} such that

$$p(ao\hat{q}|h) = p(\hat{Q}|h)^T m_{ao\hat{q}}$$

and

$$p(ao|h) = p(\hat{Q}|h)^T m_{ao}.$$

Thus, we are able to continuously update the state representation by computing the prediction of any test at any history based on the state-update function provided at the end of Section 4. Thus, tests in $\hat{Q}$ are a basis for all tests' predictions at all histories, and the proof is complete.

Example: Using the Raw Experimental Data to Design the Simulated Environment

For each customer in each experiment, the raw experimental data includes: whether the customer received the email, and the profit earned (from that customer). We group the profit outcomes into five discrete buckets. We offer an example (three customers and one experiment) in the table below:

Customer ID	Email (Experiment 1)	Outcome Bucket (Experiment 1)	Profit (Experiment 1)
A	0	3	xxx
B	1	5	xxx
C	0	1	xxx

Notes. An example illustrating the raw experimental data. For the "Email" column, 0 indicates that the customer did not receive the email, and 1 indicates the customer did receive the email. The Outcome Bucket has five levels: {1,2,3,4,5}. For confidentiality reasons we cannot disclose the actual profits (the entries are real number that can be negative).

To generate the simulated MDP environment from the real data, we perform the following mapping. Suppose we order the experiment in chronological order. The data variable "Email" in Experiment *t* corresponds to the action in Period *t* (a_t); "Outcome Bucket" in Experiment *t* maps to the state in Period *t* (s_t); and the "Profit" corresponds to the reward in Period *t* (R_t).

Customer	a_1	s_1	R_1	a_2	s_2	R_2
A	0	3	xxx	1	1	xxx
B	1	5	xxx	1	2	xxx
C	0	1	xxx	0	1	xxx

Notes. An example mapping the raw experimental data to the MDP framework.

With this mapping, we construct an MDP environment with five states and two actions: $s = \{1,2,3,4,5\}$ and $a = \{0,1\}$. We then can calculate the rewards $R^M: S \times A \to \mathbb{R}$, transition probabilities $P^M: S \times A \to S$, and policy $\pi^M: S \to A$. We describe the details of how to calculate these three quantities in the second subsection of Section 7.[1]

[1] In particular, the rewards are the midpoint of the profit earned in each of the discrete outcome buckets. If the outcome in period *t* corresponds to the third outcome bucket, the rewards for that customer in period *t* is equal to the midpoint of the rewards in that bucket. The transition probabilities for each state-action pair are calculated by counting the proportion of times a customer in that state that received that action (mail or not mail) transitioned to each of the other state in period *t* + 1. Because the actions (mail and not mail) were randomized, the policy is a stochastic policy, and reflects the percentage of times a customer in that state received each action.

Value Function Estimates

Value Function Estimates Under the Current Policy: Deviations in Transition Probabilities

Distortion (α)	True	Benchmark with two STDs	Our Method with two STDs
0	15039	14951 (14741,15161)	15097 (14838,15356)
0.1	14986	13422 (13241,13602)	15143 (14816,15470)
0.2	14891	13170 (12992,13348)	15004 (14671,15338)
0.3	14778	12931 (12754,13107)	14712 (14382,15042)
0.4	14742	12996 (12818,13174)	14873 (14535,15211)
0.5	14652	12761 (12586,12937)	14623 (14291,14955)
0.6	14583	12611 (12438,12785)	14610 (14274,14945)
0.7	14504	12563 (12389,12737)	14613 (14275,14951)
0.8	14448	12560 (12386,12734)	14586 (14251,14920)
0.9	14359	12325 (12155,12495)	14519 (14186,14852)

Notes. The table reports the value function estimates with two standard deviations under the current policy for the true state spaces, benchmark state spaces and the PSR states produced by our proposed method using true probabilities. The non-Markov distortions are distortions in the transition probabilities.

Value Function Estimates Under the Current Policy: Deviations in Rewards

Distortion (α)	True	Benchmark with two STDs	Our Method with two STDs
0	15039	14951 (14741,15161)	15097 (14838,15356)
0.1	15016	14735 (14542,14928)	15114 (14832,15397)
0.2	14993	14519 (14341,14696)	15098 (14818,15377)
0.3	14969	14303 (14138,14468)	15046 (14767,15325)
0.4	14946	14086 (13931,14242)	15044 (14770,15318)
0.5	14923	13870 (13721,14019)	15038 (14765,15312)
0.6	14899	13654 (13507,13801)	14941 (14708,15174)
0.7	14876	13438 (13289,13587)	14860 (14616,15104)
0.8	14853	13222 (13067,13377)	14888 (14640,15136)
0.9	14829	13005 (12841,13170)	14859 (14603,15114)

Notes. The table reports the value function estimates with two standard deviations under the current policy for the true state spaces, benchmark state spaces and the PSR states produced by our proposed method using true probabilities. The non-Markov distortions are distortions in the rewards.

Value Function Estimates Under the Optimal Policy: Deviations in Transition Probabilities

Distortion (α)	True	Benchmark with two STDs	Our Method with two STDs
0	15189	14865 (14454,15276)	15027 (14396,15657)
0.1	15134	13271 (12906,13636)	15033 (14264,15802)
0.2	15038	13327 (12959,13695)	15297 (14498,16097)
0.3	14924	13100 (12748,13451)	15487 (14735,16239)
0.4	14889	12744 (12402,13085)	14983 (14197,15769)
0.5	14798	12866 (12518,13214)	14774 (14031,15518)
0.6	14737	12636 (12295,12977)	14523 (13804,15241)
0.7	14669	12917 (12565,13269)	14956 (14149,15762)
0.8	14628	12880 (12530,13231)	14924 (14193,15655)
0.9	14558	12496 (12156,12836)	14411 (13667,15154)

Notes. The table reports the value function estimates with two standard deviations under the optimal policy for the true state spaces, benchmark state spaces and the PSR states produced by our proposed method using true probabilities. The non-Markov distortions are distortions in the transition probabilities.

Value Function Estimates Under the Optimal Policy: Deviations in Rewards

Distortion (α)	True	Benchmark with two STDs	Our Method with two STDs
0	15189	14865 (14454,15276)	15027 (14396,15657)
0.1	15168	14648 (14271,15026)	15184 (14604,15765)
0.2	15148	14439 (14087,14791)	15167 (14592,15741)
0.3	15127	14214 (13887,14541)	15076 (14483,15670)
0.4	15107	14054 (13748,14361)	15103 (14538,15668)
0.5	15086	13831 (13536,14125)	15152 (14572,15733)
0.6	15066	13607 (13317,13896)	14782 (14320,15245)
0.7	15045	13383 (13090,13676)	14686 (14198,15173)
0.8	15024	13159 (12855,13464)	14924 (14424,15425)
0.9	15004	12935 (12612,13259)	14913 (14399,15428)

Notes. The table reports the value function estimates with two standard deviations under the optimal policy for the true state spaces, benchmark state spaces and the PSR states produced by our proposed method using true probabilities. The non-Markov distortions are distortions in the rewards.

Are the States Markov: Proportion of States that Satisfy the Markov Property

Distortion (α)	Deviations in Transition Probabilities		Deviations in Rewards	
	Benchmark (%)	Our Method (%)	Benchmark (%)	Our Method (%)
0	100	100	100	100
0.1	0	98.21	0	90.00
0.2	0	98.28	0	90.00
0.3	0	100	0	80.00
0.4	0	100	0	88.89
0.5	0	98.21	0	88.00
0.6	0	98.28	0	83.33
0.7	0	100	0	84.62
0.8	0	100	0	84.62
0.9	0	98.28	0	84.62

Note: The table reports the % of states that satisfy the Markov property for the benchmark state spaces and the PSR states generated by our proposed method using true probabilities. The second to the third columns report the outcome under deviations in transition probabilities. The fourth to the fifth columns report the outcome under deviations in rewards.

Estimation Errors: Varying the Magnitudes

We fix the proportion of entries in the system dynamic matrix that deviate from their true probabilities at 10%. We vary the magnitude of deviations from 10% to 50%, in 10% increments: $\vartheta \in \{0.1,0.2,0.3,0.4,0.5\}$. For each deviation, there is a 50% probability of a positive deviation (ϑ), and a 50% probability of a negative deviation (ϑ).

Magnitude (ϑ)	Current Policy (%)			Optimal Policy (%)		
	Designing Core Tests	**Grouping Customers**	**Both**	**Designing Core Tests**	**Grouping Customers**	**Both**
0.1	3.19	11.88	10.27	2.49	3.00	5.20
0.2	3.19	14.97	26.45	2.49	7.55	19.82
0.3	3.19	11.80	39.13	2.49	9.26	30.06
0.4	3.19	11.84	39.13	2.49	9.36	30.06
0.5	3.19	11.52	64.53	2.49	5.10	50.36

Notes. The table reports the error rates of value function estimates for the PSR states produced by our proposed method when introducing errors in the conditional probabilities. Across the different rows in the table we vary the magnitude of errors in each entry where we introduce errors (ϑ). The second and fifth column report results where we only use deviated probabilities in finding core tests. The third and the sixth column report results where we only use deviated probabilities in grouping customers into different states. The fourth and the seventh column report results where we use deviated probabilities in both stages.

Estimation Errors: Value Function Estimates

Value Function Estimates under Different Proportions of Probability Errors

Proportion (τ)	Current Policy			Optimal Policy		
	Designing Core Tests	Grouping Customers	Both	Designing Core Tests	Grouping Customers	Both
0.1	14559	13670	12848	14811	15261	12732
0.2	14559	14312	9467	14811	15073	11067
0.3	14559	16317	6857	14811	17519	8222
0.4	14559	15862	5779	14811	14994	7553
0.5	14559	14724	5276	14811	13990	6093

Notes. The table reports the exact values of value function estimates for the PSR states produced by our proposed method when introducing errors in the conditional probabilities. Across the different rows in the table we vary the proportion of entries in which we introduce error (τ). The second and fifth column report results where we only use deviated probabilities in finding core tests. The third and the sixth column report results where we only use deviated probabilities in grouping customers into different states. The fourth and the seventh column report results where we use deviated probabilities in both stages.

Value Function Estimates under Different Magnitudes of Probability Errors

Magnitude (ϑ)	Current Policy			Optimal Policy		
	Designing Core Tests	Grouping Customers	Both	Designing Core Tests	Grouping Customers	Both
0.1	14559	13253	13495	14811	14733	14399
0.2	14559	12787	11061	14811	14042	12179
0.3	14559	13264	9154	14811	13782	10623
0.4	14559	13258	9154	14811	13768	10623
0.5	14559	13306	5334	14811	14415	7540

Notes. The table reports the exact values of value function estimates for the PSR states produced by our proposed method when introducing errors in the conditional probabilities. Across the different rows in the table we vary the magnitude of errors in each entry where we introduce errors (ϑ). The second and fifth column report results where we only use deviated probabilities in finding core tests. The third and the sixth column report results where we only use deviated probabilities in grouping customers into different states. The fourth and the seventh column report results where we use deviated probabilities in both stages.

Model Free Evidence: Constructing RFM States

We use an example to help illustrate how we construct the RFM states.

Customer	a_1	R_1	a_2	R_2	a_3	R_3
A	1	xxx	0	xxx	1	xxx
B	0	xxx	0	xxx	0	xxx
C	0	xxx	1	xxx	1	xxx

Notes. An example of simulated training data. The action $a_t = \{Not\ Mail, Mail\} = \{0,1\}$ represents whether each customer receives the email (targeting action) in Period *t* in this simulated world. $R_t \epsilon R$ represents the profit outcome for each customer in Period *t*.

We calculate the "Recency", "Frequency" and "Monetary Value" for each customer at each time period (using the historical observations for that customer). The historical data at Period *t* is $a_1, R_1, a_2, R_2, \dots, a_{t-1}, R_{t-1}$. Each dimension of the RFM states takes the following values:

Recency: takes three levels, ranging from 1 to 3, indicating the number of periods since the last purchase.

Frequency: takes three levels, ranging from 0 to 2, indicating the number of purchases in prior periods.

Monetary Value: takes five levels, ranging from 1 to 5, indicating the average size of prior purchases (we first average prior purchases, and then discretize this average).

This yields a state space with 45 states. We evaluate the RFM variables for each customer at each time period, and assign the customer into the corresponding RFM state.

Model-Free Evidence: Varying the Size of the Training Data

We report five sets of results according to how many observations we include in the training data. We allow the size of the training data to vary from 20,000 to 100,000 customers in increments of 20,000. However, for all five replications we hold the size of the validation sample fixed at 36,262 customers.

Model-Free Evidence: Error Rates of Length-Four Profits

Number of Customers in Training Sample	Proposed Method	Full History	RFM	Single State
20,000	85.73% (0.87%)	85.57 % (0.87%)	88.39% (0.86%)	151.17% (0.99%)
40,000	85.38% (0.87%)	85.31% (0.87%)	88.18% (0.86%)	151.82% (1.00%)
60,000	85.08% (0.87%)	85.35% (0.87%)	88.19% (0.86%)	151.58% (1.00%)
80,000	85.42% (0.87%)	85.31% (0.87%)	88.13% (0.86%)	151.40% (1.00%)
100,000	85.17% (0.86%)	85.08% (0.86%)	87.97% (0.86%)	151.32% (0.99%)

Notes. The table reports the average absolute error between the actual and predicted length-four profits for the 36,262 customers in the validation sample. Bootstrapped standard errors are in parentheses.