Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence

Optimal Airline Ticket Purchasing Using Automated User-Guided Feature Selection

William Groves and **Maria Gini**
Department of Computer Science and Engineering
University of Minnesota, USA
{groves, gini}@cs.umn.edu

Abstract

Airline ticket purchase timing is a strategic problem that requires both historical data and domain knowledge to solve consistently. Even with some historical information (often a feature of modern travel reservation web sites), it is difficult for consumers to make true cost-minimizing decisions. To address this problem, we introduce an automated agent which is able to optimize purchase timing on behalf of customers and provide performance estimates of its computed action policy based on past performance. We apply machine learning to recent ticket price quotes from many competing airlines for the target flight route. Our novelty lies in extending this using a systematic feature extraction technique incorporating elementary user-provided domain knowledge that greatly enhances the performance of machine learning algorithms. Using this technique, our agent achieves much closer to the optimal purchase policy than other proposed decision theoretic approaches for this domain.

1 Introduction

The conventional wisdom of airline ticket purchasing is that it is generally best to buy a ticket as early as possible to avoid the risk that the price may increase. As prices do generally increase dramatically before a flight's departure, it seems generally correct. However, the earliest purchase strategy only occasionally achieves the optimal lowest cost ticket. This paper proposes a model for estimating the optimal policy for future departures. The ultimate application of this model is to autonomously make daily purchase decisions on behalf of airline ticket buyers to lower their costs.

This kind of optimal airline ticket purchasing from the consumer's perspective is challenging principally because buyers have insufficient information for reasoning about future price movements. Prices can vary significantly on a daily basis, and consumers have no information about pricing behaviors of particular routes and airlines.

The airline ticket domain is characterized by adversarial risk in two contexts: the adversarial relationship between buyers and sellers, and the competitive relationships between the airlines providing service. We assume buyers are seeking the lowest price on their travel, while sellers are seeking to keep overall revenue as high as possible to maximize profit. Simultaneously, each seller must consider the price movements of its competitors to ensure that its prices remain competitive to achieve sufficient (but not too high) demand.

A central challenge in airline ticket purchasing is in overcoming the information asymmetry that exists between buyers and sellers. Airlines can mine significant databases of historical sales data to develop models for expected future demand for each flight. Demand for a specific flight is likely to vary over time and will also vary based on the pricing strategy adopted by the airline. For buyers without access to historical price information, it is generally best to buy far in advance of a flight's departure, per the conventional wisdom. However, this is not always best since airlines will adjust prices downward if they want to increase sales.

Given a corpus of historical data and the proposed learning approach, it is possible to compute policies that do much better than the earliest purchase strategy. The success of the proposed method depends on several novel contributions:

1. We leverage a user-provided hierarchical structure applied to the features in the domain and use automated methods to decide which features to include or prune. This enables us to compute efficiently a more optimal feature set than using existing feature selection methods which use only information from the data set.
2. We capture temporal trends in the model by allowing time-delayed observations to supplement or replace the most recently observed value for each included feature. The time-delayed observations are called *lagged features* in the text.

These novel aspects are applicable to many real-world multivariate domains, but this paper demonstrates the power of this technique on the domain of ticket price prediction. This paper extends the existing literature on airline ticket price prediction as well. When comparing to a deployed commercial system, the proposed model is more informative than the output of the Bing Travel "Fare Predictor", the best commercial system currently available for airline fare prediction. The model here is extended to accommodate preferences such as the number of stops in the itinerary or the specific airline to use. This prediction task is more difficult than predicting only the lowest cost ticket but it is more useful for actual airline

passengers. An additional benefit is that the model can provide insights into the domain: the importance of individual variables can be assessed by their presence or absence in the computed optimal model.

2 Background and Related Work

Airlines determine the prices to offer for each flight through a process called yield management which is designed to maximize revenue given constraints such as capacity and future demand estimates (for an overview, see [Belobaba, 1987; Smith *et al.*, 1992]). Mismatches between airplane size and passenger demand are equalized through pricing, which can adjust demand. Choosing optimal pricing on an entire airline network is complex because there are instances (in hub-and-spoke networks) when sacrificing revenue on a particular flight can increase overall revenue.

Some work has been done for determining optimal purchase timing for airline tickets. Our work has been inspired by [Etzioni *et al.*, 2003], where several purchasing agents attempt to predict the optimal purchase time of an airline ticket for a particular flight. The agents determine the optimal purchase time within the last 21 days prior to departure for specific flights in their collected data set. The authors compute the purchasing policy (a sequence of wait/buy signals) for many unique simulated passengers with a specific target airline, target flight, and date of departure to satisfy. The optimal policy (the sequence of buy/wait signals that leads to the lowest possible ticket price) is used as a benchmark for each simulated passenger and the cost of each alternative purchasing agent is computed. The aggregate result shows that, given these purchasing criteria, it is possible to save a significant amount when purchasing. We understand that Bing Travel's "Fare Predictor" is a commercialized version of the models in [Etzioni *et al.*, 2003], so real-world results from this form a benchmark for our results.

Our work extends the state-of-the-art because we directly compute a policy for finding the minimum cost ticket of *any* flight from *any* airline given a route and departure date. This is a more difficult problem because the aggregate minimum price varies less than the price of an individual flight from an individual airline. Our paper goes beyond their work in several ways: the model is not limited to a single flight, the purchases are up to 60 days before departure (instead of 21 in the existing work), the model is compared against realistic financial benchmarks (including buying as early as possible), and the model provides a regression estimate for the expected best price between the current date and departure.

The exclusively data-driven feature selection techniques are also directly applicable to this domain [Molina *et al.*, 2002]. [Hall, 2000] presents the CFS algorithm (Correlation-based Feature Selection) to perform a filter-based feature selection using a merit heuristic (normalized Pearson's Correlation). The algorithm starts with an empty feature set and uses forward best-first search to incrementally add features. Wrapper-based methods employing search (such as best first search (BFS)) using an underlying machine learning algorithm have also been employed [Kohavi and John, 1997]. Both techniques are included in the results for benchmarking.

	Example 1	Example 2
Quote Date:	13 May 2011	13 May 2011
Origin City:	SEA	NYC
Destination City:	IAD	LAX
Departure Date:	20 May 2011	20 May 2011
Return Date:	25 May 2011	25 May 2011
No. of itineraries returned:	1135	1304
No. of airlines quoting:	9	13

Table 1: Airline price quote specifications for all airlines from specific 5-day round trips. The exact dates and cities shown are for illustration purposes only.

3 Data Sources

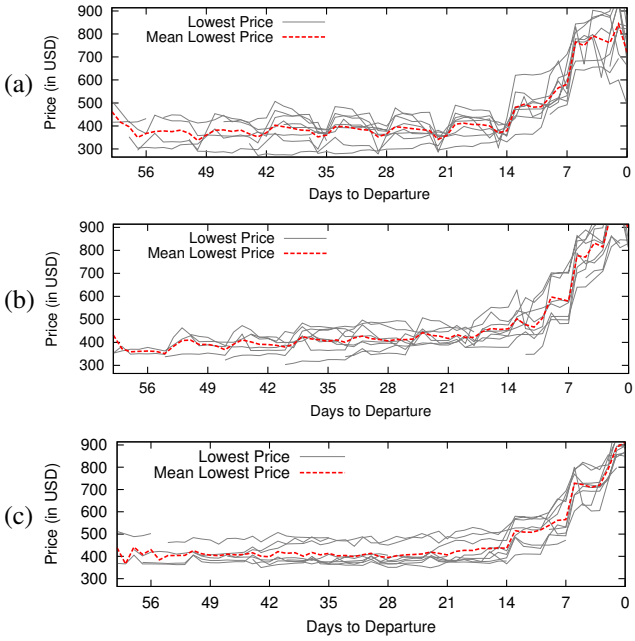
The data for our analysis was collected as daily price quotes from a major travel search web site between Feb. 22 and Jun. 10, 2011 (109 observation days). A web spider was used to query for each route and departure date pair in our study, so the results are representative of what a customer could observe in the market. This set is split sequentially into 3 data sets with the following lengths: 48, 20, and 41 days. The three periods are utilized as the training set, calibration set, and test set, respectively. Each query returned ≈1,200 unique round-trip itineraries from all airlines; most queries returned results from more than 10 airlines. Example queries are in Table 1. Feature values are consistently available aggregates computed from each day's itineraries. For consistency, the queries were initiated at the same time each day.

Bing Travel, a popular travel search web site, has a "Fare Predictor" tool that provides a daily buy/wait policy recommendation for many routes and departure dates. These recommendations were obtained daily from the site for the test set period and are directly compared with our agent's policy.

Pricing Patterns in Historical Data. There are strong cyclic patterns in the time series of prices. For example, Figure 1 shows the mean lowest price quoted by all airlines for a specific origin-destination pair for 2 months of 5-day round trip itineraries departing on (a) Thursdays and (b) Mondays. The Thursday to Tuesday itinerary time series shows a regular drop in prices for Tuesday, Wednesday, and Thursday purchases (*days-to-departure modulo* $7 \in \{0, 1, 2\}$), while the (b) series shows significant increases for Thursday, Friday, and Saturday purchases. As expected, both exhibit price increases in the last few days before departure (*days-to-departure* ≤ 7) but series (b) exhibits this increase earlier in the time series. We posit that the majority of business flights would be Monday to Friday itineraries, and thus demand for series (b) flights would be less sensitive to price increases than leisure flights. The weekly depression in costs in (a) may be due to market segmentation: customers buying mid-week are more price sensitive than weekend purchasers.

The pricing behaviors exhibited for other origin-destination pairs also differ. A high traffic origin-destination pair such as the New York City to Los Angeles route (shown in (c)) exhibits much weaker cyclic patterns. We conjecture that strategic pricing is likely to have a much greater observed effect for routes that have relatively few (2-3) competing

Figure 1: Mean lowest price from all airlines (a) for NYC to MSP 5-day round trip flights having Thursday departure and Tuesday return (Th-Tu), (b) for NYC to MSP M-F, or (c) for NYC to LAX Th-Tu itineraries. Each solid line series indicates quotes for a different departure (8 departure dates in each graph). A dotted series indicates the mean.



airlines than for routes with a large number of competitors.

4 Proposed Model

When constructing prediction models for real-world domains, practical complexities must be addressed to achieve good prediction results. Typically, there are too many sources of data (features). Limiting the set of features in the prediction model is essential for good performance, but prediction accuracy can be lost if relevant inputs are pruned. This is even more acute in situations where the number of observations is limited, often a feature of real-world domains. To meet this challenge, we construct a prediction model that involves the following distinct steps, which we then describe in detail:

1. Feature Extraction – The raw data observed in the market are aggregated into a fixed length feature set.
2. Lagged Feature Computation – A lag scheme is computed using a hierarchy of the features that incorporates some domain knowledge.
3. Regression Model Construction – Using the augmented feature set generated from the lag scheme, a regression model is generated using partial least squares (PLS) regression.
4. Policy Computation – A search of decision threshold parameters is done to minimize calibration set cost.
5. Optimal Model Selection – For each candidate model computed using the previous steps, the one which performs best on the calibration set is chosen. The final performance is estimated on the test set.

Table 2: Raw features by feature class for each quote day on a specific departure day and route. The number of features in some classes (EACH-A, EACH-S) will vary based on the number of airlines quoting the route. The counts given are specific to the MSP-NYC route (92 total raw features). Features are named as "airline-statistic-#ofStops": i.e. ALL-min-A = minimum price quoted by any airline, ALL-min-0 = minimum price quoted by any airline for non-stop flights only, DL-min-A = minimum price quoted by a specific airline (DL = Delta Airlines), and Quote DoW is Mon = boolean variable (1 if quote is retrieved on Monday).

Class	Size	Variable List
Det	8 vars	Days-to-departure, Quote DoW is Mon, Quote DoW is Tue, ..., Quote DoW is Sun
All-A	3 vars	ALL-min-A, ALL-mean-A, ALL-count-A
All-S	9 vars	ALL-min-0, ALL-mean-0, ALL-count-0, ALL-min-1, ALL-mean-1, ALL-count-1, ALL-min-2, ALL-mean-2, ALL-count-2
Each-A	18 vars	DL-min-A, DL-mean-A, DL-count-A, ..., OTHER-min-A, OTHER-mean-A, OTHER-count-A
Each-S	54 vars	DL-min-0, DL-mean-0, DL-count-0, DL-min-1, DL-mean-1, DL-count-1, DL-min-2, DL-mean-2, ...

Feature Extraction. The large number of itineraries (>1000) in each daily query made some data aggregation necessary. The features extracted for prediction are aggregated variables computed from the (large) list of quotes observed on individual query days. For each query day, there are possibly many airlines quoting flights for a specific origin-destination and date combination. This is possibly due to strategic decisions of the airline or due to lack of available capacity. We limit the number of airlines used for distinct features by focusing on airlines that quote for a specific route more than 40% of the query days. Also, each airline may present itineraries that contain non-stop segments or segments with one or more stop. We divide the quotes by number of stops into three bins: non-stop round trips, round trips with a maximum of one stop in each direction, and round trips with 2 or more stops in either direction. For each bin, three features are computed: the minimum price, mean price, and the number of quotes. Additionally, these three features are computed for the union of all three bins. So for each airline, 12 features are computed on each quote day. For airlines not exceeding the 40% criteria, their itineraries are combined into a separate "OTHER" category placeholder. Finally, these same 12 aggregates are generated for all itineraries and are placed in the "ALL" airlines category. Boolean variables indicating the query's weekday are added. A *days-to-departure* (number of days between the query and departure dates) value is computed based on the departure date.

A listing of the features for each query day is shown in Table 2. Each of the 92 features is in a class based on its specificity using the feature class hierarchy in Figure 2.

(a) maximal scheme

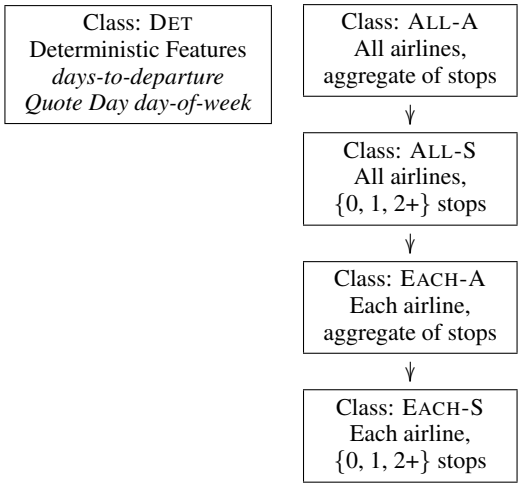
Class	Lagged Offsets 0	1	2	3	4	5	6	7
DET	•							
ALL-A	•	•	•	•	•	•	•	•
ALL-S	•	•	•	•	•	•	•	•
EACH-A	•	•	•	•	•	•	•	•
EACH-S	•	•	•	•	•	•	•	•

(b) minimal scheme

Class	Lagged Offsets 0	1	2	3	4	5	6	7
DET	•							
ALL-A	•							
ALL-S	•							
EACH-A	•							
EACH-S	•							

Table 3: Basic lag schemes

Figure 2: Lag scheme class hierarchy for product price prediction. Arrow denotes a *subset* relationship (i.e. class ALL-A should have an equal or greater set size than class ALL-S).



Lagged Feature Computation. Using only the most recent values (92 features for the NYC → MSP route) as the entire feature set may provide reasonable prediction results in some domains, but such a model cannot predict trends or temporal relationships present in the data. This simple scheme is shown in Table 3(b) and in the results as *Minimal Lag Scheme*. The need to represent temporally-offset relationships (such as weekly cycles or trends) motivates adding time-delayed observations to the feature set as well. We refer to this as the addition of *lagged features*. For instance, if the cost of a route on day $t-7$ is representative of the price available on day $t+1$, the 7 day delayed observation should have a high weight in the model. A regression model which includes all time-delayed instances up to a depth of n days of all features can produce good results, but the inclusion of many variables into the model can result in poor performance. Performance of a model of this construction ($n = 7$) is shown in Table 5 as the model *Full Lag Scheme* and in Table 3(a).

By examining all combinations (a small number) of feature classes it is possible to automatically tune the feature vector to achieve better results. Another constraint is added: more specific feature classes will not have more time delayed instances than more general classes. We posit that time-delayed observations from the target variable (such as the all airline minimum price in *Class* ALL-A) are likely to be most predictive. Time-delayed observations from other more-specific feature classes *may also be* but are less likely to be predictive. It is by this principle that the hierarchy and strict ordering of lagged data additions are based. By constraining the classes so that the less information-dense classes contribute fewer features, we prevent the inclusion of extraneous, irrelevant features.

Table 4: Optimal lag schemes of a domestic route and an international route for a 5-day trip with Monday departure.

(a) New York → Minneapolis

Class	Lagged Offsets 0	1	2	3	4	5	6	7
DET	•							
ALL-A	•	•	•	•	•	•	•	•
ALL-S		•	•	•	•	•	•	
EACH-A				•				
EACH-S								

(b) New York → Minneapolis (non-stop only)

Class	Lagged Offsets 0	1	2	3	4	5	6	7
DET	•							
ALL-A	•	•	•	•	•			
ALL-S				•				
EACH-A				•				
EACH-S				•				

(c) New York → Hong Kong

Class	Lagged Offsets 0	1	2	3	4	5	6	7
DET	•							
ALL-A	•	•	•	•	•	•	•	•
ALL-S								
EACH-A								
EACH-S								

(d) New York → Hong Kong (non-stop only)

Class	Lagged Offsets 0	1	2	3	4	5	6	7
DET	•							
ALL-A	•	•	•	•	•	•	•	•
ALL-S	•	•	•	•	•	•	•	
EACH-A	•	•	•	•				
EACH-S	•	•	•	•				

Next, time lagged data is reformatted to form the augmented feature set, which is called a *lag scheme expansion*. A search of possible configurations is performed to find the best performing configuration for a target (the optimal choice may be different for each route).

The inclusion of a little domain knowledge using feature classes and a class hierarchy to constrain the search for high-performance feature sets allows for significant reductions in the number of possible feature set configurations. The number of lag schemes as formulated with the hierarchy in Figure 2 for a maximum time delay of 7 days is 8517. Without the constraints between classes, there are configurations of the 5 feature classes if constrained to possible time delays of $\{\varnothing, 0, 1, 2, \ldots, 7\}$, but many configurations will be uninteresting variants. Finally, without the hierarchy and constraints between classes, there are $\approx 10^{82}$ configurations of the 92 original features.[1] Using both the feature classification and the constraint hierarchy allows for a greater variety of "interesting" lag schemes to be tested for the same search effort.

While a domain expert could design a high-performance feature set, the automated lag scheme search should contain a

[1] 84 price features, and 8 deterministic features (days-to-departure and quote weekday) $= 2 * (2^7) * (9^{84})$

Table 5: Model results comparison on a single route data set for the lowest cost itinerary on any airline. All itineraries are 5 days (Monday to Friday, or Thursday to Tuesday). Cities are NYC (New York City) and MSP (Minneapolis, MN, USA).

Feature Selection Type	Learning Method	Learning Method Output	NYC-MSP Mon-Fri	NYC-MSP M-F nonstop	NYC-MSP Tu-Th	NYC-MSP Tu-Th nonstop
			(mean cost (in \$), efficiency (as % of optimal savings))			
No Feature Selection	Earliest Purchase	Buy/Wait	$(317, 0.00\%)$	$(414, 0.00\%)$	$(309, 0.00\%)$	$(374, 0.00\%)$
	Optimal	Buy/Wait	$(268, 100\%)$	$(341, 100\%)$	$(263, 100\%)$	$(301, 100\%)$
	Bing Travel	Buy/Wait	$(308, 2.56\%)$	N/A	$(306, 0.903\%)$	N/A
	PLS w/ Minimal Lag Sch.	Regression	$(314, 6.87\%)$	$(384, 41.0\%)$	$(294, 34.0\%)$	$(354, 26.7\%)$
	PLS w/ Full Lag Scheme	Regression	$(300, 34.1\%)$	$(398, 22.4\%)$	$(316, -13.4\%)$	$(345, 38.7\%)$
Off-the-shelf Methods	PLS w/ Correlation-based FS (CFS)	Regression	$(313, 7.93\%)$	$(413, 0.223\%)$	$(308, 0.331\%)$	$(371, 2.98\%)$
	PLS w/ Wrapper Feat. Sel. (BFS)	Regression	$(317, -1.21\%)$	$(416, -3.56\%)$	$(310, -2.32\%)$	$(369, 5.70\%)$
Lag Scheme Feature Selection	Decision Tree	Buy/Wait	$(288, 58.8\%)$	$(388, 35.3\%)$	$(289, 42.9\%)$	$(382, 56.8\%)$
	nu-SVR	Regression	$(295, 45.1\%)$	$(396, 24.5\%)$	$(289, 42.9\%)$	$(338, 48.3\%)$
	Ridge Regression	Regression	$(293, 49.9\%)$	$(383, 42.0\%)$	$(316, -13.5\%)$	$(372, 2.71\%)$
	Decision Tree	Regression	$(284, 65.5\%)$	$(375, 52.0\%)$	$(280, 62.0\%)$	$(334, 54.4\%)$
	PLS Regression	Regression	$(280, 75.3\%)$	$(365, 66.8\%)$	$(276, 72.5\%)$	$(330, 59.9\%)$

configuration similar to what a domain expert can build. Also, the results of the optimal lag scheme search can elicit some surprising relationships in the data. Table 4 shows optimal lag schemes for several targets. It is interesting to note that non-stop targets in Table 4(b, d) benefit from a larger feature set (both in temporal depth and feature class breadth).

Regression Model Construction. Mathematically, PLS regression deterministically computes a linear function that maps a vector of the input features x_i into the output variable y_i (the label) using a vector of weights $\bar{w}$. Several implementations of PLS exist [de Jong, 1993; Martens and Næs, 1992]; each with its own performance characteristics. We use the orthogonalized PLS, Non-Integer PLS (NIPALS), implementation in [Wold *et al.*, 1983]. PLS was chosen over similar multivariate techniques including multiple linear regression, ridge regression [Hoerl and Kennard, 2000], and PCR [Jolliffe, 1982] because it produces better performance than the others and has the ability to adjust model complexity. Other machine learning algorithms can also be used in place of PLS. Experiments using support vector regression (nuSVR), ridge regression, and decision trees are also shown for comparison.

Policy Computation and Evaluation. An obvious approach to choosing a good regression model (lag scheme and trained machine learning model) is to use the model with the highest prediction accuracy, but it may not be the model that generates the lowest average cost policy. Instead, we propose to grade the models by measuring the cost that results from following the computed policy recommendation. To use the regression output (an expected future price) to compute an action policy, we introduce the concept of a decision threshold function. Given $\hat{e}_t$, the model estimate future price at time t, the current observed price p_t and the current number of days-to-departure d_{dtd} (an integer), the current action policy $r_t \in \{\text{BUY}, \text{WAIT}\}$ is computed by Equation 1.

$$r_t = \begin{cases} \text{BUY} & : \hat{e}_t > p_t * (c + (1/30) * s * d_{dtd}) \\ \text{WAIT} & : \text{otherwise} \end{cases} \quad (1)$$

The two parameters c and s are expressed as decimals. Intuitively, c can be thought of as an adjustment in the likelihood of a buy signal. Values of $c > 1.0$ correspond to a policy that is only likely to emit BUY when the current price is far below the expected future price. This situation indicates the current price is a bargain for the customer. The parameter s corresponds to the percent change in the threshold per 30 days of advance purchase (0.02 corresponds to a 2% change in the threshold at $d_{dtd} = 30$). Values of $s > 0.0$ generate a policy more likely to WAIT when far from departure. When a departure is far in the future and $s > 0.0$, the agent is more likely to wait until a highly favorable (low) price appears before deciding to purchase. Adjusting these two parameters can be thought of as determining the optimal level of risk depending on the current price and the degree of advance purchase. The range of c and s values searched was $[0.7, 1.3]$ and $[-0.1, 0.1]$, respectively, in increments of 0.01.

We use this two-parameter approach to make decisions, because it is simple, works well, and provides an intuitive understanding of the policy computation. This is not to rule out more sophisticated approaches, such as reinforcement learning. We leave exploration of this aspect for future work.

Optimal Model Selection. The proposed search of lag schemes is exhaustive, but due to the feature class hierarchy, the number of configurations is relatively small and can be fully explored. A model is constructed for each potential lag scheme: first, for each lag scheme a pricing model is generated using the training set data, then the decision threshold parameters (c and s) are calibrated on the out-of-sample calibration set that results in the lowest average ticket price. Performance is measured by scoring the model on the test set.

5 Experimental Results

The experiments were designed to estimate real-world costs of using various prediction models to develop a purchase policy. A survey of the literature revealed that: airlines assume a relatively fixed rate of purchases until a flight is full, and most tickets for a flight are sold within 60 days of departure [Belobaba, 1987]. Using these facts, we measure performance as the cost of following the purchase recommendations for a specific departure once for purchases between 1 and 60 days before departure ($dtd \in \{1, 2, \ldots, 60\}$). This measure involves hypothetically purchasing an itinerary precisely 60 times for each purchase algorithm under test (but some purchases may

	Model					
	Optimal	Our Model	Linear Reg. (LR)	Ripper	Earliest Purch.	Latest Purch.
Method Origin:	baseline	this paper	[Etzioni *et al.*, 2003]		baseline	baseline
HOU → NYC	100.0	70.9	7.12	-26.8	0.00	-261
MSP → NYC	100.0	73.9	3.09	-45.2	0.00	-227
NYC → CDG (Int'l)	100.0	63.2	3.00	-74.8	0.00	-295
NYC → CHI	100.0	54.6	-4.73	-222	0.00	-626
NYC → HKG (Int'l)	100.0	56.9	10.5	-141	0.00	-161
NYC → MSP	100.0	64.9	5.74	-121	0.00	-289
SEA → IAD	100.0	69.4	4.91	-25.7	0.00	-190
Mean Efficiency	100.0	64.8	4.21	-93.8	0.00	-293
Savings Margin	11.0	7.25	0.514	-10.4	0.00	-32.3

Table 6: Percentage-based performance comparison of various decision theoretic approaches across a combination of 7 (domestic and international) routes by 3-digit airport code. All values are in %. Savings Margin computed as % of earliest purchase.

be deferred for a few days based on the model output). Each of the 60 purchases is called a *purchase episode*.

Performance Benchmarks. The naïve purchase algorithm, called *earliest purchase*, is to purchase a ticket once for each day in the α day range. Its purchase episodes terminate with a purchase event on the first day of the episode and mean cost would be equal to the mean of prices across the α day period. The lowest achievable cost is called the *optimal cost* and is based on purchasing for each of the α episodes at the lowest price between the beginning of the episode and its departure date. The comparison methodology involving simulated purchases is similar to that used in [Etzioni *et al.*, 2003].

In Table 5, the results of estimated costs for several purchasing policies based on purchasing 5-day (Monday or Thursday departure) round trip itineraries from NYC to MSP (265 simulated purchases per column) are shown. The results show how costs vary based on preferences such as a customer requiring non-stop itinerary as well. We also compare our policy result against the cost of following the buy/wait recommendation from Bing Travel's "Fare Predictor."

Table 6 shows there is, on average, a possible 11% savings to be achieved over *earliest purchase*. We denote this percentage as the *savings margin*. Our method of a lag scheme search coupled with PLS Regression and a decision threshold achieves consistently closer to the optimal action sequence than any of the other methods compared. The 74% efficiency achieved by PLS (in Table 5) represents a savings of 8% less than the *earliest purchase* strategy for the NYC→MSP route.

Bing Travel Performance Comparison. It is surprising however that Bing Travel is not able to achieve a greater savings margin on the Any Airline target. We posit that this is due to a risk averse approach taken by the algorithm: it is more likely than our method to advise immediate purchase.

This assertion can be validated by looking at the distribution of buy and wait signals computed for each day by the various policy generators: in the NYC→MSP M-F route, the optimal policy has only a 15% proportion of buy signals. It is noteworthy that the best models constructed with our method emits a similar proportion of wait signals: in the NYC→MSP M-F route, the model with the lowest average cost ($280) only emits a buy signal on 34% of the days. Bing's model has a much higher proportion of buy signals: in the same route, the Bing model emits a buy signal 83% of the days.

Multi-route Comparison. To show that this technique is generalizable to other routes (including international routes), we provide performance statistics of 7 routes in Table 6. The proposed method achieves an average of 69% of the optimal savings which represents an average cost savings of 7.25% when compared to the earliest purchase strategy. Given the high cost of airline tickets, this represents a significant savings. For the purposes of comparison with existing approaches,we provide results of two decision theoretic methods from [Etzioni *et al.*, 2003]: Ripper and LR. Those models use a smaller number of features compared to our model and do not leverage the competitive relationships between airlines when making predictions. We believe prediction approaches should consider price competition between airlines.

6 Conclusions and Future Work

To our knowledge, these results represent the state-of-the-art in ticket price prediction using consumer-accessible data.

This investigation shows that, given publicly-observable information, it is possible to predict airline ticket prices to systematically reduce costs. We believe that there is a significant market for this these kinds of models in the hands of consumers. In particular, reliable price models can assist buyers in determining the range of expected prices for an itinerary.

This feature selection technique also has wide applicability to other multivariate domains where basic domain knowledge is common but not utilized. Building the feature class hierarchy requires only basic domain knowledge to be successful; but, greater expertise in hierarchy construction will improve efficiency by focusing the feature selection search. The constraints also contribute by preventing overfitting (evident from the poor performance of off-the-shelf feature selection approaches) which can occur with many features and few training instances. The inclusion of lagged features in the model captures temporal relationships among features and improves the predictions. This method also contributes in facilitating domain understanding: by examining the relative performance of candidate lag schemes, domain knowledge can be extracted: the significance of individual features can be determined by observing their presence in the lag scheme.

References

[Belobaba, 1987] Peter P. Belobaba. Airline yield management. An overview of seat inventory control. *Transportation Science*, 21(2):63–73, 1987.

[de Jong, 1993] Sijmen de Jong. Simpls: An alternative approach to partial least squares regression. *Chemometrics and Intelligent Laboratory Systems*, 18(3):251–263, 1993.

[Etzioni *et al.*, 2003] Oren Etzioni, Rattapoom Tuchinda, Craig Knoblock, and Alexander Yates. To buy or not to buy: mining airfare data to minimize ticket purchase price. In *SIGKDD Conf. on Knowledge Discovery and Data Mining*, pages 119–128, 2003.

[Hall, 2000] Mark A. Hall. Correlation-based feature selection for discrete and numeric class machine learning. In *Int'l Conf. on Machine Learning*, pages 359–366, 2000.

[Hoerl and Kennard, 2000] Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 42(1):80–86, 2000.

[Jolliffe, 1982] Ian T. Jolliffe. A note on the use of principal components in regression. *J. of Royal Statistical Society. (Applied Statistics)*, 31(3):300–303, 1982.

[Kohavi and John, 1997] Ron Kohavi and George H. John. Wrappers for feature subset selection. *Artif. Intell.*, 97(1-2):273–324, 1997.

[Martens and Næs, 1992] Harald Martens and Tormod Næs. *Multivariate Calibration*. John Wiley & Sons, July 1992.

[Molina *et al.*, 2002] Luis Carlos Molina, Lluís Belanche, and Àngela Nebot. Feature selection algorithms: A survey and experimental evaluation. In *IEEE Int'l Conf. on Data Mining (ICDM)*, pages 306–313. IEEE Computer Society, 2002.

[Smith *et al.*, 1992] Barry C. Smith, John F. Leimkuhler, and Ross M. Darrow. Yield management at American Airlines. *Interfaces*, 22(1):8–31, 1992.

[Wold *et al.*, 1983] S. Wold, H. Martens, and H. Wold. Multivariate calibration problem in chemistry solved by the PLS method. *Matrix Pencils*, 973(18):286–293, 1983.