Reinforcement Learning for Flight Ticket Prices

Niki Agrawal, Ramnik Arora
nikhar@stanford.edu, rarora@stanford.edu

Problem + ML Framework

Problem: Can we use Reinforcement Learning to help a customer decide the optimal time to purchase a flight ticket?

Models Constructed:

1) Baseline
2) Q-learning
3) DQN

ML Framework:

Assume a customer decides to purchase a ticket for a particular flight at time = X hours before departure. The optimal time to purchase the ticket t_{opt} is:

- in the range [X hours before dep., 4 hours before dep.]
- time at which we achieve minimum flight price until departure

Reinforcement Learning Framework:

- Timestep t_o = customer looking to buy a ticket on a particular flight X hours before departure.
- State s = [hours before departure, other flight features]
- Using flight features, decide whether action = wait to purchase or action = buy ticket now
- If action is buy, next state s' is terminal
- If action is wait:
  - If flight sells out/takes off, s' is terminal
  - Else, s' is [X-4 hours before departure, other flight features]

Data

- Google flights pricing data in the form of JSON files using the Google QPX Express API service.
- Collected every four hours, date range of 3/2016-3/2017 for one-way flights on the SFO → NYC route.
- For a given (flight, date-time) pair, the information we use includes:
  - Flight ID, Source Airport, Destination Airport, Departure Date/Time, Carrier, Current Ticket Sale Price
- **Data Preprocessing:**
  - Json format (almost 300GB) → Amazon RDS to setup a database and parse appropriate data
  - Constructed label (of buy versus wait) for each (flight, date-time) pair
  - Normalized features for DQN neural network
- **Training Data:** 586 flights, 97,848 data points (65% of the flights)
- **Dev Data:** 103 flights, 30,451 data points (10% of the flights)
- **Test Data:** 149 flights, 51,945 data points (25% of flights)

Models

Baseline: Customer always selects "buy." Reflects a common trend for flight prices to increase close to purchase point, avoids ticket selling out.

Q-Learning:

- Tabular representation of (state, action) → Q-value developed during training
- Each state belongs to an equivalence class: same flight_uid (carrier + flight number), same hours before departure, but different departure dates
- Single iteration over training data in each equivalence class

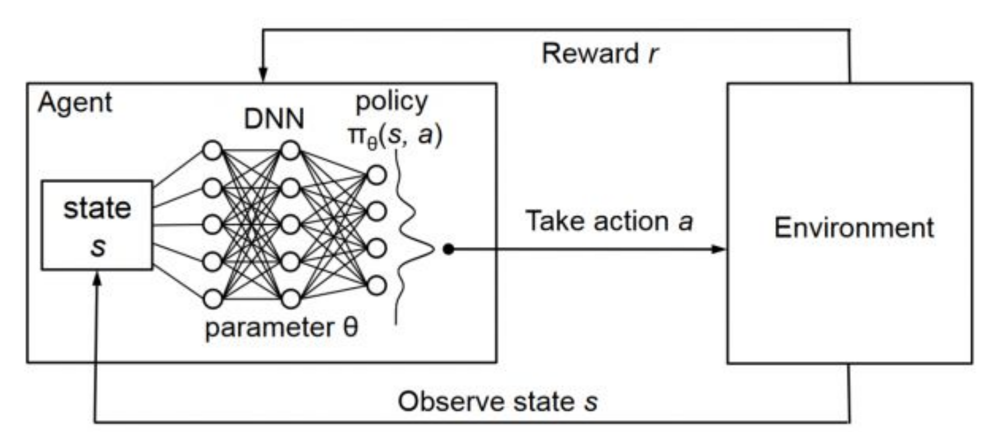
$$Q(a,s) = Avg_{s^* \sim s}(R(s^*, a) + max_{a'}(Q(a', s')))$$

- Reward Function:
  - For action = wait:
    - If flight sells out, Reward = -300,000
  - If flight does not sell out, Reward = 0
- For action = buy: Reward = -current sale price

DQN

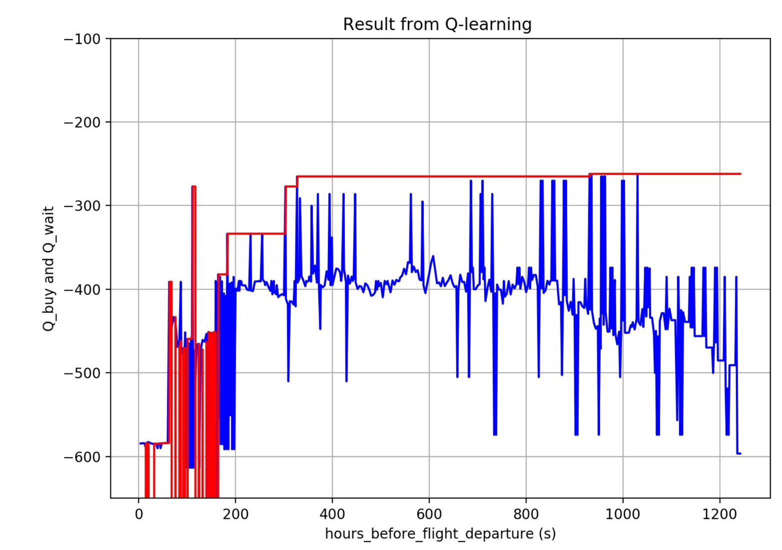
$$Q(a,s) = R(s,a) + \gamma max_{a'}(Q(a', s'))$$

- State → neural network → Q-value
- 2 hidden layers: sigmoid activation followed by ReLU, Output layer: linear
- State = [current sale price, hours before departure, booking_code_count]
- Reward Function: If flight sells out, Reward = -previous flight price - $300



Results

		Percent Savings	Maximum Savings	% Wait Decisions	% Correct Wait Decisions	Total Tickets	Total Flights
	Baseline	0.00%	16.23%	0.00%	0.00%	7402	51
Train	Q-Learning	-15.44%	16.23%	86.14%	36.34%	7402	51
	DQN	0.00%	16.23%	0.00%	0.00%	7402	51
	Baseline	0.00%	26.25%	0.00%	0.00%	2720	10
Dev	Q-Learning	-2.24%	26.25%	93.46%	33.40%	2720	10
	DQN		26.25%	0.00%	0.00%	2720	10
	Baseline	0.00%	23.77%	0.00%	0.00%	3183	10
Test	Q-Learning	10.17%	23.77%	94.85%	31.00%	3183	10
	DQN	0.00%	23.77%	0.00%	0.00%	3183	10



		Percent Savings	Maximum Savings	% Wait Decisions	% Correct Wait Decisions	Total Tickets	Total Flights
	Baseline	0.00%	16.23%	0.00%	0.00%	7402	51
Train	Q-Learning	-15.44%	16.23%	86.14%	36.34%	7402	51
	DQN	0.00%	16.23%	0.00%	0.00%	7402	51
	Baseline	0.00%	26.25%	0.00%	0.00%	2720	10
Dev	Q-Learning	-2.24%	26.25%	93.46%	33.40%	2720	10
	DQN	0.00%	26.25%	0.00%	0.00%	2720	10
	Baseline	0.00%	23.77%	0.00%	0.00%	3183	10
Test	Q-Learning	10.17%	23.77%	94.85%	31.00%	3183	10
	DQN	0.00%	23.77%	0.00%	0.00%	3183	10

Conclusions & Future Work

- Access to high quality flight data for multiple routes and over long time-periods is incredibly important to be able to train the data. It'll be interesting to extend the work to more routes, with more data.
- Deep Q-Networks can capture more nuanced states, but they are difficult to train and needs more parameter tuning.
- Users generally purchase a ticket on a route, not an individual flight. The current setup doesn't capture the majority use case.
- Currently, the agents interaction doesn't change the environment. However, in actual ticket purchase problems, the agents behavior can lead to tickets being sold out etc.

References

1. To Buy or Not to Buy: Mining Airfare Data to Minimize Ticket Purchase Price (https://www.stat.berkeley.edu/~aldous/157/Papers/etzioni.pdf)
2. On Optimizing Airline Ticket Purchase Timing (https://dl.acm.org/citation.cfm?id=2733384)
3. Computational Complexity of Air Travel Planning (http://www.ai.mit.edu/courses/6.034f/psets/ps1/airtravel.pdf)