mosaic
DATA SCIENCE

http://mosaicdatascience.com/

info@mosaicdatascience.com

Data Science Case Study

Automated Cooking Prediction & Optimizer | Deep RL

Mosaic built an automated cooking prediction solution for a leading quick service food chain using deep reinforcement learning.

Industry

Restaurants

Use Case

Automating Cooking Operations

Techniques

Reinforcement Learning, Deep Learning & Adoption & Scale

Outcome

Less product waste, shorter wait times, and higher meal quality

‘FAST FOOD’ CONSUMER EXPECTATIONS & CONTROLLING LABOR COSTS

Restaurant labor is a growing point of concern for many quick service restaurants around the world. From the challenges of hiring and retaining high quality employees to the reality of increased minimum wage legislation across many marketing in the United States, the industry is under pressure to innovate. Couple this with the highly competitive nature of fast food restaurants competing for consumer market share, and these chains needs to find any competitive advantage available to them.

THE PROMISE OF MACHINE LEARNING & ARTIFICIAL INTELLIGENCE | AUTOMATED COOKING PREDICTION

One of the largest quick service restaurant chains came to Mosaic facing some of these same challenges. The firm felt there was an opportunity for automation in their kitchen operations, specifically at the grill and fry stations. The restaurant chain had made investments in drive-thru sensors, added in-store cameras, and aggregated historical sales data, and they realized they needed outside machine learning & AI expertise to analyze all these disparate data sources.

The chain’s ultimate vision was to have one common automation solution for their numerous locations that met the brand’s standards for food quality and consistency while providing a compelling return on investment. They felt a custom solution was much more effective than an out-of-the-box software tool for many reasons:

- The sheer amount of predictive & prescriptive modeling
- Ownership of the intellectual property, saving millions in licensing fees
- Minimized costs for development and deployment at numerous locations

Mosaic, a leading AI consulting company, met with the restaurant at the beginning of the project to layout a ML-driven approach to tackling this problem. At the start of the engagement, Mosaic collaborated with stakeholders to determine that historical sales, drive-thru traffic, weather, and in-store traffic would help facilitate the automatic direction of cooking processes. The output of this modeling effort would be to produce an accurate short-term sales and production forecast. With this information in their hands, the company could cut down on customer wait time, combat waste, and serve more customers.

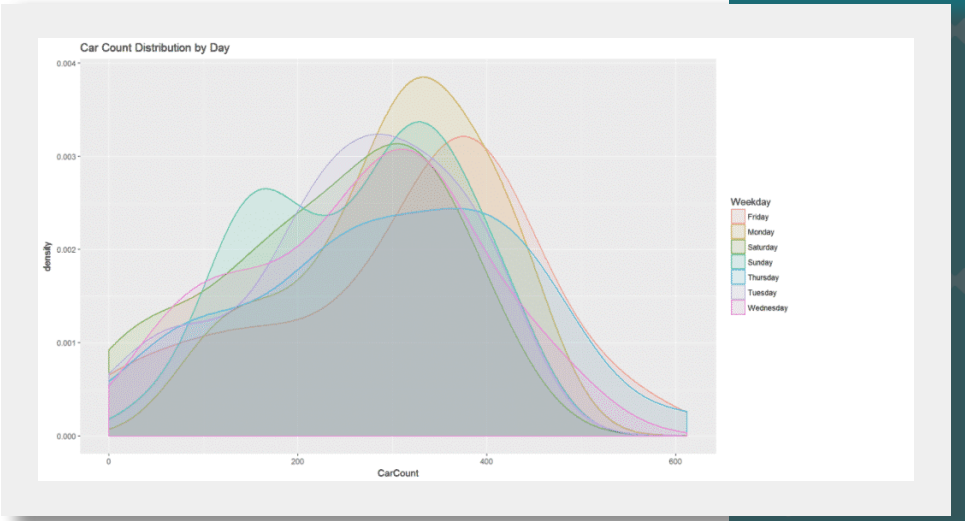
To test this automated cooking solution efficiently and effectively, the chain planned to deploy in their live, test restaurant near corporate headquarters. This allowed the solution to be tuned for maximum efficiency.

CAPACITY DEMAND PROBLEM STATEMENT

In conjunction with customer stakeholders, Mosaic developed a modeling plan for the automated cooking prediction solution. The solution included development and testing of machine learning & deep learning models that enabled appropriate decisions to be made in-store when to prepare food. The goal was to minimize waste (having too much food prepared), minimize lost orders (potential customers see a long line of cars and drive to the competition), and ensure that when orders are placed, the customer experience is optimized (they don't have to wait too long for the food, and the food is fresh).

The solution was composed of very complex models with many detailed and intricate moving parts. Management and integration of these moving parts was critical to maximize model accuracy and to ensure the models worked across the chain's extensive footprint. The proposed production system needed to operate at or very near capacity during routine peak demand periods. Accurate forecasting using sensor and other data, and operational decisions are key to maximizing throughput and revenue in these situations, since significant additional effective capacity and throughput can be realized if the right operational decisions are made before demand reaches capacity. If the need for maximum capacity is not anticipated in advance, nothing can be done to salvage performance of the peak period.

The customer prioritized meeting customer demand – having the correct amount of food prepared for incoming orders – over minimizing waste. Any sort of machine learning & AI approach needed to account for this requirement. As you can see below, demand varied throughout the week according to consumer behavior.



How would a data scientist go about solving this problem? While many approaches might have been tried Mosaic decided on the application of deep reinforcement learning.

DEEP REINFORCEMENT LEARNING

Context

A major requirement for reinforcement learning (RL) is a suitable training environment, which in most cases simply is not available. Specifically, it needs either a simulation environment or the ability (and willingness) to let an agent train in a live environment. In this case, the client had already developed a custom simulated kitchen environment using commercial simulation software. There was significant direct interaction between Mosaic and the simulator developers in the process of defining a simulator API for the RL agent to interact with. From there, Mosaic developed a Python wrapper for the simulator API to facilitate the RL training process.

RL FORMULATION

An RL problem must be formulated as a Markov Decision Process, which defines the environment as a series of States, Actions, Rewards, and probabilistic transitions into new States. There are also considerations about whether this problem can be framed as episodic or continuous in nature, and how a time-step should be defined. In this case, the state included information like which restaurant, day of week, time of day, recent same-day sales, etc. The action space was the policy for each food product containing a refill level and a cook-to quantity. The reward was a function of factors like stock-out and waste. And time-steps of 30 minutes were used within an episode that represented one day. All the above factors represent ways to frame a problem, in much the same way a supervised learning problem can be framed in many ways, yielding better or worse performance. The "best" way to frame a problem is mostly a matter of context and the goal of the business objective. In other words, it depends on what you want to achieve.

TRAINING THE AGENT

There are many RL algorithms and variations out there. In this case, given that there was a continuous (vs. discrete) action space, we opted for an algorithm called Deep Deterministic Policy Gradient[1] (DDPG). At a high level, DDPG – being a deep learning, model-free approach to RL – uses a deep neural network to approximate a policy function. In the case of DDPG and as a result of "tricks" and best practices developed by researchers in recent years, there are four neural networks under the hood training the agent. More specifically, only one of the neural networks (the local actor) represents the actual agent that will ultimately be used to generate recommended actions for the business, while the other three are used only in training.

Training can be a lengthy, iterative process of tuning hyperparameters and debugging before landing on a configuration that converges (i.e. an agent that learns over time).

The performance of the agent will also vary depending on how the reward function is defined. So, slightly different reward functions yielded different performance.

For example, the trained agent, when compared to the existing best policy manually developed by the business, was able to reduce stock-out by 30% without increasing waste and while maintaining the same level of throughput. By tweaking the reward function, the agent could also reduce stock-out by 70%, while letting waste increase by only 40%. In other words, the desired performance could be guided by the goals of the business.

DEPLOYMENT

To enable the delivery of the trained agent in a production setting, it was deployed as a REST API in a Flask application. Mosaic also developed a Spark streaming module to process point-of-sale data and agent cooking instructions to track current inventory and waste count. All software tools and dependencies for this project were open source, including Python, PyTorch, Spark, and Flask.

RESULTS

In the end, the automated cooking prediction solution provided new options for managing the restaurant's business more efficiently through the balancing of opposing goals. The innovative use of reinforcement learning was the breakthrough needed for this to happen.

Endnotes

1. https://spinningup.openai.com/en/latest/algorithms/ddpg.html