Assignment 2

ML as a Service

Ivan Cheung
Student ID: 13975420

08/10/2023

36120 - Advanced Machine Learning Application
Master of Data Science and Innovation
University of Technology of Sydney

1. Executive Summary

GitHub Repository	https://github.com/ivanutsmdsi/amla2023_at2
Heroku App	https://mysterious-sands-98640-63a0822c91e6.herokuapp.com/

A machine learning approach to consume the entirety of the company's sales records across the country provides an opportunity to gain new insights into our company performance. This consuming of information is not possible through conventional methods due to the large size of the data. From machine learning models, it will be possible to predict:

1) A seven-day forecast on the revenue across the company.
2) An estimate on the revenue for an item within one of our stores on a given day.

The information provided by such predictions will support business operations in coordinating inventory stock lists, promotion of items in key markets and corporate decision making in policy and coordinating staff rostering.

The deployment of a cloud-based service for company stakeholders to consume the predictions will allow store managers across the company easy access to the service, regardless of location. This project has shown that it is possible to deploy such an application to capture company sales records and deploy a cloud predictive engine for stakeholder consumption. While the proof of concept is shown to be viable, further technology investment from the company is required to fully leverage the insights available in our sales data.

2. Business Understanding

a. Business Use Cases

The project aims to provide a machine learning application that can support our supermarket operations through predictive foresight into product revenue and national sales revenue. The application addresses the business opportunities by providing two predictive calculations:

3) A seven-day forecast on the revenue across the company.
4) A predictive estimate on the revenue for an item within one of our stores on a given day.

The application of machine learning raises an opportunity to gain insight into our sales data at a scale not possible through conventional analytical means. With over 3000 items for sale at a given store and around 6000 transactions in a day, the use of a machine learning model allows this high-volume information to be ingested with ease.

b. Key Objectives

The key objectives of this project will be to implement a machine learning application that can ingest company sales information and provide predictions on expected revenue across the company. The predictions from this application will support store managers, in evaluating the best selling items in their stores and trends on item categories across departments, as well as in head office, providing forecasting towards future company revenue.

In order to meet these objectives, the application will be deployed on a cloud service, allowing access to all company stakeholders nationally, regardless of location. The cloud service will support different stakeholder queries through a set of prompts, based on the information required.

3. Data Understanding

The primary data source for this project is the company sales records. This data contains the following information:

item_id	The id code of the item sold. This id code is shared across the corporate stocklist.
dept_id	The department of the store that the item is sold in.
cat_id	The item category.
store_id	The store that recorded the sales record, by store id.
state_id	The state of the store.
d_####	A count of items sold by the given store for the unique item id on a specific date, coded by a numeric value.
sell_price	The price of each quantity of item, by item id, store id and date.

As this data is company information, the collection of this information poses minimal challenges. Data can be collected directly from our point of sale (POS) system. However, the information is collected in two separate tables. The data requires preparation before it can be used for analysis. In addition to our company sales records, a complimentary dataset on calendar events has also been sourced. This events calendar lists events by date on national events (including public holidays), cultural, religious, and sporting events. The inclusion of significant events should help us understand the shift in market patterns and enable the application to adjust predictions accordingly.

4. Data Preparation

a. Preprocessing

Preparing the sales table

Sample sales data:

	id	item_id	dept_id	cat_id	store_id	state_id	d_1	d_2	d_3	d_4	...	d_1532	d_1533	d_1534	d_1535	d_1536	d_1537	d_1538	d_1539	d_1540	d_1541
0	HOBBIES_1_001_CA_1_evaluation	HOBBIES_1_001	HOBBIES_1	HOBBIES	CA_1	CA	0	0	0	0	...	1	1	1	0	1	0	1	0	0	1
3049	HOBBIES_1_001_CA_2_evaluation	HOBBIES_1_001	HOBBIES_1	HOBBIES	CA_2	CA	0	0	0	0	...	0	0	0	1	1	1	0	0	1	0
6098	HOBBIES_1_001_CA_3_evaluation	HOBBIES_1_001	HOBBIES_1	HOBBIES	CA_3	CA	0	0	0	0	...	0	0	0	0	0	0	0	0	0	2
9147	HOBBIES_1_001_CA_4_evaluation	HOBBIES_1_001	HOBBIES_1	HOBBIES	CA_4	CA	0	0	0	0	...	1	1	1	0	1	0	0	0	0	0
12196	HOBBIES_1_001_TX_1_evaluation	HOBBIES_1_001	HOBBIES_1	HOBBIES	TX_1	TX	0	0	0	0	...	0	0	0	0	0	0	0	0	0	1
15245	HOBBIES_1_001_TX_2_evaluation	HOBBIES_1_001	HOBBIES_1	HOBBIES	TX_2	TX	0	0	0	0	...	0	1	1	0	0	0	0	1	1	2
18294	HOBBIES_1_001_TX_3_evaluation	HOBBIES_1_001	HOBBIES_1	HOBBIES	TX_3	TX	0	0	0	0	...	1	1	0	0	1	1	0	1	0	1
21343	HOBBIES_1_001_WI_1_evaluation	HOBBIES_1_001	HOBBIES_1	HOBBIES	WI_1	WI	0	0	0	0	...	0	0	1	0	0	0	0	0	0	1
24392	HOBBIES_1_001_WI_2_evaluation	HOBBIES_1_001	HOBBIES_1	HOBBIES	WI_2	WI	0	0	0	0	...	0	0	0	0	0	0	0	0	2	0
27441	HOBBIES_1_001_WI_3_evaluation	HOBBIES_1_001	HOBBIES_1	HOBBIES	WI_3	WI	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0

The sales records for the company requires no data cleaning. However, the dates are coded in an unfriendly format (d_####). Additionally, sales records for each date are captured per column. The individual columns for each date prevent the datetime being applied as a feature in machine learning applications and require a pivot:

	item_id	dept_id	cat_id	store_id	state_id	d	items_sold
0	HOBBIES_1_001	HOBBIES_1	HOBBIES	CA_1	CA	d_1	0
1	HOBBIES_1_002	HOBBIES_1	HOBBIES	CA_1	CA	d_1	0
2	HOBBIES_1_003	HOBBIES_1	HOBBIES	CA_1	CA	d_1	0
3	HOBBIES_1_004	HOBBIES_1	HOBBIES	CA_1	CA	d_1	0
4	HOBBIES_1_005	HOBBIES_1	HOBBIES	CA_1	CA	d_1	0

However, with 1541 dates on file, the pivot causes the table to expand significantly. From a 30490 x 1547 table to a 46,985,090 x 7 table. This significantly enlarges the table cells by seven-fold. With this pivot, memory challenges are introduced. This is further explored below.

Joining item sell prices

Sample item price data:

	store_id	item_id	wm_yr_wk	sell_price
0	CA_1	HOBBIES_1_001	11325	9.58
1	CA_1	HOBBIES_1_001	11326	9.58
2	CA_1	HOBBIES_1_001	11327	8.26
3	CA_1	HOBBIES_1_001	11328	8.26

The sale price of items for company records is recorded in a separate table. In order to calculate revenue (quantity sold × sell price), a join will need to be performed between the two tables.

However, this table records date by **wm_yr_wk** while the sales table records in the format of **d_##.** To reconcile the two tables together, a bridging calendar table was used.

Joining calendar events

To further enrich our model, significant events across the country were also ingested. These events include cultural, religious, sporting and national events:

	date	event_name	event_type
0	2011-02-06	SuperBowl	Sporting
1	2011-02-14	ValentinesDay	Cultural
2	2011-02-21	PresidentsDay	National
3	2011-03-09	LentStart	Religious
4	2011-03-16	LentWeek2	Religious

This events table was joined with the sales and item prices table by the date column, also using the bridging calendar table. However, this table introduced duplication issues. Due to some dates having multiple significant events, such as 05/05/2013 (Cinco De Mayo & Orthodox Easter) as well as 15/06/2014 (Father's Day & NBA Finals). To prevent duplicating primary table data, events were aggregated into a per day count by event type. This process reduces the detail within the events data but allows multiple events per day to be measured quantitatively:

date	event_cultural	event_national	event_religious	event_sport
2011-02-06	0	0	0	1
2011-02-14	1	0	0	0
2011-02-21	0	1	0	0
2011-03-09	0	0	1	0
2011-03-16	0	0	1	0

Memory challenges as a result of large datasets

Initial feature set sample:

item_id	dept_id	cat_id	store_id	state_id	datediff	revenue	event_cultural	event_national	event_religious	event_sport
HOBBIES_1_001	HOBBIES_1	HOBBIES	CA_1	CA	28	0.0	0	0	0	0
HOBBIES_1_002	HOBBIES_1	HOBBIES	CA_1	CA	28	0.0	0	0	0	0
HOBBIES_1_003	HOBBIES_1	HOBBIES	CA_1	CA	28	0.0	0	0	0	0
HOBBIES_1_004	HOBBIES_1	HOBBIES	CA_1	CA	28	0.0	0	0	0	0
HOBBIES_1_005	HOBBIES_1	HOBBIES	CA_1	CA	28	0.0	0	0	0	0

Putting the sales data, item pricelist and calendar events into an enriched feature set for machine learning proved difficult due to the initial pivot step expanding the data to 46 million rows. This feature set required over 5GB in memory to store, reaching the memory limits of the project

machine. The excessive memory requirements made additional feature engineering stages difficult. To address these issues, a number of steps were explored:

1. dropping features

The first attempt to reduce memory utilisation was to reduce quality of features to save on memory. Two approaches were explored;

1) converting date to a number,
2) and using label encoding over one hot encoding.

In both cases, a reduction in feature richness proved to negatively impact model performance too severely to be practical. This approach was abandoned.

2. cleaning variables in python environment

Proactive deletion of variables after they have been used and no longer required proved effective in maintaining a low memory usage throughout the python environment. This approach has been implemented across the entire application and can be seen in the app script and python notebooks.

3. reducing variable type to converse memory allocation

The most efficient approach which was able to bring down the feature table from 5GB to management levels was the downcasting of features to lower memory alternatives. This approach had significant gains on memory conservation. The following type changes were carried out:

Feature	Initial Type	Downcast Type
item id	object (string)	category
dept id	object (string)	category
cat id	object (string)	category
store id	object (string)	category
state id	object (string)	category
revenue	float64	float32
event cultural	float64	uint8 (unsigned int)

event national	float64	uint8 (unsigned int)
event religious	float64	uint8 (unsigned int)
event sport	float64	uint8 (unsigned int)
day of week	int32	uint8 (unsigned int)
month	int32	uint8 (unsigned int)
year	int32	uint16 (unsigned int)
Total Memory Allocation	**5 GB**	**807 MB**

Enriching date feature for training

After addressing the memory limitation issue, further enrichment of the features was possible. To leverage the most out of the sales date, the context was broken up into 4 columns:

- day of week,
- day of year (forecast use only),
- month,
- and year.

Packaging the clean dataset

The entire process was packaged into a single notebook for the two predictive models and is available within the folder /notebooks/engineering/.

b. Feature engineering for model fitting

One Hot Encoding

To enable categorical features to be fitted to models, encoding categorial features into numeric values is required. The nominal approach utilises One Hot Encoding (OHE). However, OHE generates an additional column (feature) for each unique value within a feature. In the sales data there are 3049 unique item ids. The other categorical features also range from 10 - 3 values. The large number of item ids prevented OHE being a viable approach, due again to excessive memory requirements.

Ordinal Encoding

To address the limitations in OHE on categorical values, Ordinal Encoding was used instead (OE). Ordinal Encoding does not perform as well as OHE but also does not expand the feature set with additional columns. OE was considered a viable alternative to limit the memory requirements.

The categorical features were encoding using either Ordinal or One Hot Encoding:

Ordinal Encoding: item id, department id and store id

One Hot Encoding: category id and state id

5. Modeling

Linear Regression and SGD Regression

The first regression approaches attempted were a linear regressor (as a proof of concept) and a Stochastic Gradient Descent (SGD) regressor. The linear regressor proved to be too simplistic to produce an accurate prediction and was quickly discarded. While SGD regressor struggled to train on the full feature set and would result in out of memory errors.

Due to further memory limitations encountered while training models, new approaches were required.

Warm start and partial fit

In researching ways to control memory limitations, two possible solutions were identified. The use of either 'warm start' or 'partial fit' functions in certain models. Warm fit allows a model to be retrained with new data, creating new patterns (trees or iterations) while retaining the model's existing patterns. Partial fit allows data to be ingested in partitions and prior to the training finalisation. Both options are model dependent with models support one, both or none. As the SGD regressor did not support either option, a new regression approach was required.

Random Forest regressor

no. estimators (trees)	5 + an additional 5 per batch
max depth	30
warm start	True

Random Forest (RF) was selected as it supports warm start. Under warm start, each new fit is assigned to the new estimators (trees) and trees generated from previous fittings are retained. This allowed the feature set to be trained in batches, allowing the full feature set to be fitted to the model.

While RF was successful in fitting all the data, the model file stores each tree generated in memory. After training the RF regressor on the full dataset, the resulting model was at 2 GB. The size of this model was unable to be uploaded to Heroku. As a result an alternative model was required.

Gradient Boosting regressor

no. estimators (iterations)	5 + an additional 5 per batch
learning rate	0.5
max depth	8
warm start	True

Gradient Boosting (GB) regression was selected in favor of RF due to the small size of the trained model and that this regressor supported warm start. In a warm start GB, the lessons learned from previous iterations are retained and new lessons are generated with the fitting of each partition. The trained model file retains only the final iteration, as a result the model maintained a manageable size of 1 MB. This allowed the model to be uploaded to Heroku for deployment.

Histogram Gradient Boosting regressor (National Revenue Forecasting)

max iterations	10000
learning rate	0.1
max depth	None
min sample leaf	1
early stop	True
no. iterations with no change	500

As the forecast model for national revenue required less sales breakdown and had reduced features, memory limitations were not a factor. A more robust regressor could be used to improve model performances. A Histogram Gradient Boosting (HGB) regressor was selected from a number of possible options, due to the testing performances of this model vs other options.

6. Evaluation

Evaluation Metrics

The performance of each model was evaluated based on an 80/20 test split (80% of data trained and 20% reserved for evaluation). The core metrics used for each model was the prediction root mean square deviation (RMSE) and r2 score against the true revenue data. The RMSE score evaluates the average difference between the predicted revenue and the true revenue and the r2 score provides a proportional variance in the predicted vs actual as a decimal.

Results and Analysis

Model	RMSE	r2 score
Individual Item Revenue Models		
Random Forest	6.999	0.40563
Gradient Boosting	8.424	0.13911
National Revenue Model		
Histogram Gradient Boost	3204.55	0.23463

While the Random Forest model performed better than the GB model, it was unable to be used by the project team due to storage limitations described above.

Business impact and benefits

The two models developed by this project allow the company to predict on item sales and national company revenue in support of business operations. By having access to predictions on which items will have high revenue on a given date, store managers can plan accordingly to capitalise on this information, including promotion of the product and managing stock volumes to support expected sales. The national revenue forecast will help support head office decision makers in setting corporate policy around rostering (matching staff overhead with expected revenue), and strategic decisions, particularly around significant events. The information provided by this project enhances existing business decisions, by introducing future-focused information into the mix alongside historic financial accounting and sales records.

Data privacy and ethical concerns

At this time, all of the data ingested is either corporate data (sales records) or public information (significant events). The sales information has been de-identified and does not contain any customer information, product information beyond product category and department and cannot be used to discriminate against an individual or corporate partners. The project team do not see any privacy or ethical concerns in consuming this data and leveraging insights from the predictions.

7. Deployment

The trained models have been saved and deployed into the company's Heroku account, leveraging this platform service to support a ML as a service approach. The deployment has been made possible through the company's GitHub account:

GitHub Repository	https://github.com/ivanutsmdsi/amla2023_at2
Heroku App	https://mysterious-sands-98640-63a0822c91e6.herokuapp.com/

To redeploy the application, the following steps are required:

1. Ensure all model changes are saved into the models/ folder

2. Commit changes, including new model files into git

2. Push the new commit to the Heroku service

Heroku will automatically restart and deploy the new version if a new commit has been detected. The service reads the **heroku.yml** and **Dockerfile** on the project parent folder to determine the build requirements.

8. Conclusion

This project has shown that it is possible to train a machine learning model to ingest company sales records to provide predictive insights into sales performance to store managers and corporate stakeholders. The project is able to deploy the model into a cloud-based service, to support users in using the application at their convenience, across the entire country on any internet enabled device.

While the proof of concept showed that there is an opportunity to enrich our insights through this approach, much more progress can be gained. In particular, the project was highly limited by the memory available on the development machine. As a result the final models are not as accurate as the data allows and the capacity for enrichment of the data for model training has not been fully explored. With a commitment from the company for additional resources, the predictive accuracy of the service can be greatly improved, offering even greater benefits back to the company stakeholders.