WeatherOrNot: Weather Enriched Short Term Fashion Forecast

Aishwerya Akhil Kapoor, Ogheneovo Ojameuaye, Peshotan Irani

MOTIVATION AND BACKGROUND

Forecasting demand is a crucial issue for driving efficient operations management plans. This is especially the case in the fashion industry, where demand uncertainty, lack of historical data and seasonal trends usually co-exist[1]. As such, demand forecasting is a popular research topic and many models for forecasting fashion products have been proposed in the literature over the past few decades. Recently, there have been a number of studies concerning the impact of weather on demand. Weather patterns influence how people decide, what type of clothing they buy or if to shop at all. People shop for clothing that helps them feel comfortable in the current or expected weather. Seasonal changes also influence fashion trends. Retail companies must, therefore, understand and predict shoppers' behaviour to help better planning. Such demand forecasting helps these companies improve cost efficiencies as it provides reliable intelligence to better plan supply, manage inventory, and more efficiently staff stores. Our focus, therefore, is to model the impact of weather on shopping behaviour, providing the demand forecasting that aids such cost efficiencies.

PROBLEM STATEMENT

Along with our focus of modeling the impact of weather on demand, we aim to discover if certain aspects that make up the daily weather data affect the demand of a particular product. For instance, we explore questions such as the degree of impact of temperature compared to the impact of precipitation on demand. We analyze if cities in proximity and with similar weather conditions also consistently exhibit similar demand patterns. As our product data is hierarchical in nature, we observed these demand changes brought about by weather in different levels of product hierarchy.

Our study remains focused on modelling customer shopping behaviour attributed to aspects of weather.

ASSUMPTIONS

During the process of exploring and modelling the data, we make some assumptions:

- If there is no data for a particular level of hierarchy, we assume that no sale was made.
- A similarity between styles under a particular class in terms of demand
- We assume classes with count over 5 years of less than 0.5% are low selling items

DATA SCIENCE PIPELINE

Our data science workflow was divided into four major components as shown below:

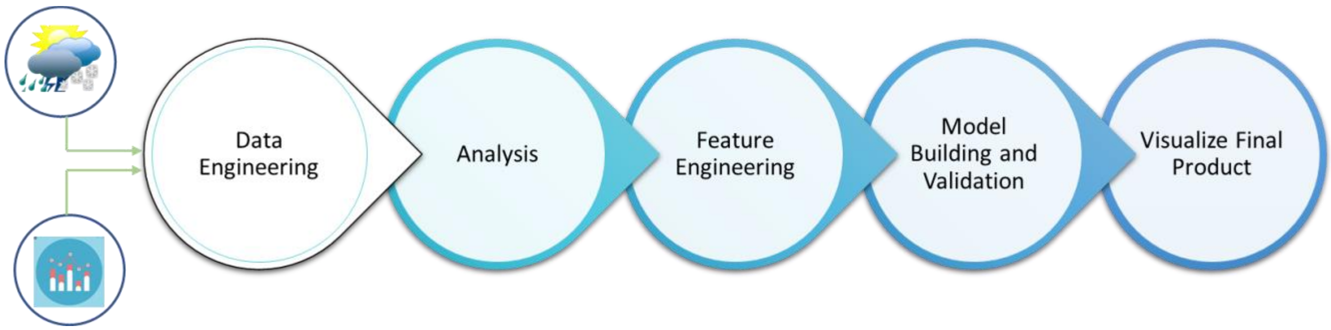


Fig1: The Data Science Pipeline

Data Collection

Three distinct datasets were used for the project; the first one was the

- Transactional data provided by the company,
- Historical weather data scraped through various weather websites datasets
- Created by us by calling a weather API endpoint which was used for spatio-temporal integration.

For our scraping of the webpages we wrote our python scraping script and utilized selenium due to the website's dynamic nature. AWS was used for all data storage.

Exploratory Data Analysis

The aim of performing exploratory data analysis was to understand the sales data better and build our hypothesis. The level of anonymization was quite high, thus EDA helped us make safe assumptions to build our models and form hypotheses to test.

(i) General findings:

Sales data was transactional and product information was in the following hierarchy (top-down):
Category -> Department -> Class -> Style

- ❑ The distribution was very skewed. Almost 90% data belonged to Category 1, whereas 10% was under Category 2.
- ❑ On a city level, the data was imbalanced. Out of 44 cities, 26 cities were from British Columbia, 14 from Alberta and 4 from Saskatchewan.
- ❑ There was a lot of inconsistency at class level of product information
- ❑ Some erroneous records were dropped (where sales were positive and quantity sold was negative, and vice-versa). Total information loss was less than 5%
- ❑ Some cities exhibited no correlation with respect to its some major weather components, e.g. Kelowna

Fig2: Distribution of Data

(ii) Exploration with different windows

Product demand and sales we assessed on provincial as well as city level for three different time frames: daily, weekly, monthly and yearly.

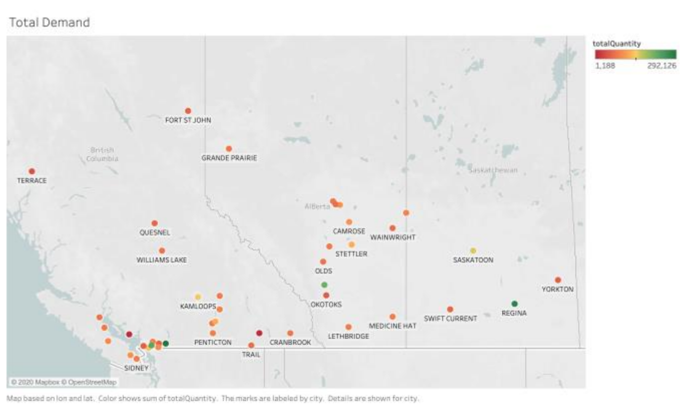
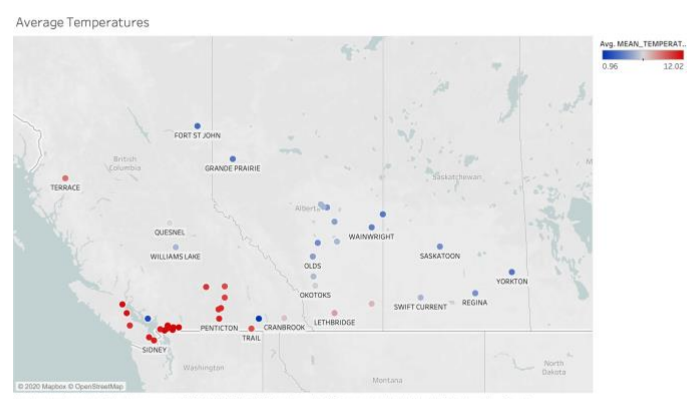
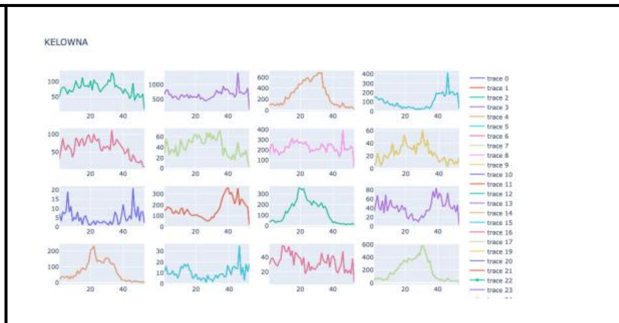
	

Fig3a : Total Demand across three provinces	*Fig3b: Typical average temperatures across three provinces*

(iii) Critical Observations:

- ❏ Temperature and Precipitation were the major aspects of weather that affected shopping patterns each year. We saw more demand with lower temperatures than with high precipitations (rain and snow)
- ❏ Average temperature rising more than 15-degree Celsius discouraged people from shopping.
- ❏ The class level product information is highly seasonal.
- ❏ Speed/Direction of wind did not same to affect demand
- ❏ Demand for Class 3 and Class 4 are inversely proportional to each other.

❏ Department 12 is highly in demand when temperatures start to decrease.

	
Fig4a: Demand trend in each class in Calgary	*Fig4b: Demand trend in each class in Kelowna*

While demand for classes in Calgary and Kelowna follow the same overall pattern, the change in precipitation over the year is not as varying as Calgary's.

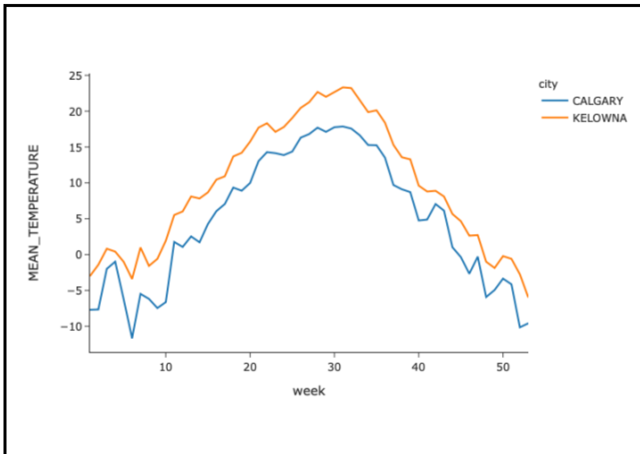
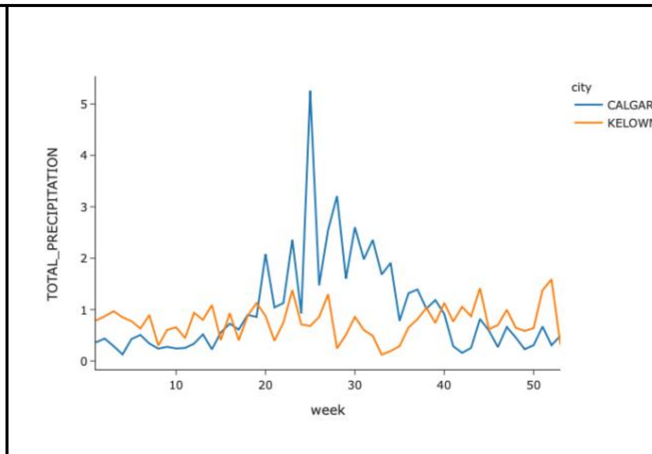
	

Fig5a: Mean Temperatures in Calgary and Kelowna	*Fig5b: Mean Precipitation in Calgary and Kelowna*

Data Engineering

The transactional data provides covers multiple cities while for the weather data, each city consists of multiple stations. To enable the spatio-temporal integration of both datasets, we obtain ground zero coordinates and integrate both datasets using the station closest to that coordinate. The haversine distance algorithm was used to find the shortest distance between coordinates. It was implemented as a KDtree which organizes the points in 3D space. This was done for optimization as this data structure helps with searches over multidimensional arrays.

Imputation

Imputation based on single station statistics was not enough to capture and preserve the trends in the weather dataset. To overcome this limitation, we impute missing data in the weather dataset with data from the same time period in the nearest neighboring station.

To address areas where we still had gaps left, we imputed the values based on average values from the previous and next days.

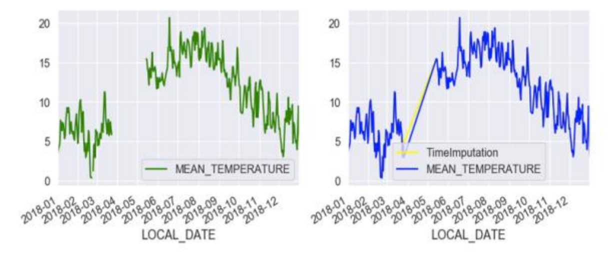


Fig6a: Imputation using linear extrapolation

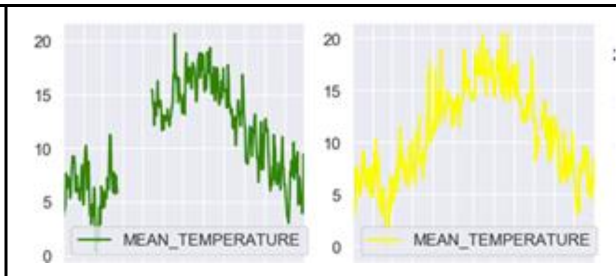


Fig6b: Imputation using the nearest neighbor

Data Wrangling

The data maintained its time series nature based on how granular it was explored. Looking at the daily demand for each city was trivial because at least one item was sold each day. As we explored down the hierarchy, this ceased to apply. In order to maintain the time series nature of the data and preserve its properties, the data was restructured. For example, we ensure that there is an entry for every day, for each city on each class level. Where this data doesn't exist, we create an instance for it and assume that class was not sold on that day and impute zero entries.

Further exploration of the data showed that its distribution was long tail and skewed. We drop some data points as follows:

City Level: We dropped cities with a count of data points less than 1000. We assume the demand in those cities are negligible or the items are not popular.

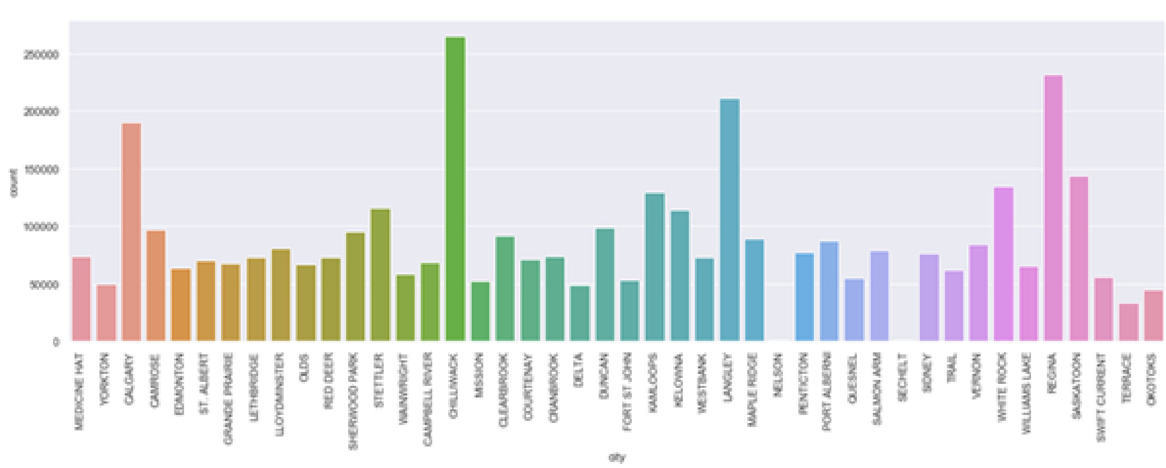


Fig7: Count of demand (target) for each city

Class Level: We drop classes that are extremely rare. These classes accounted for less than 0.5% of the total demand across the entire dataset.

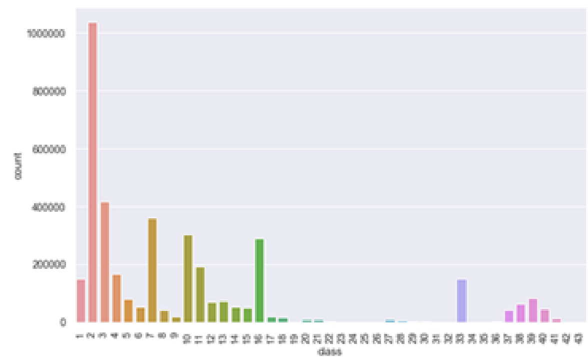


Fig8a: Occurrence of each class in dataset

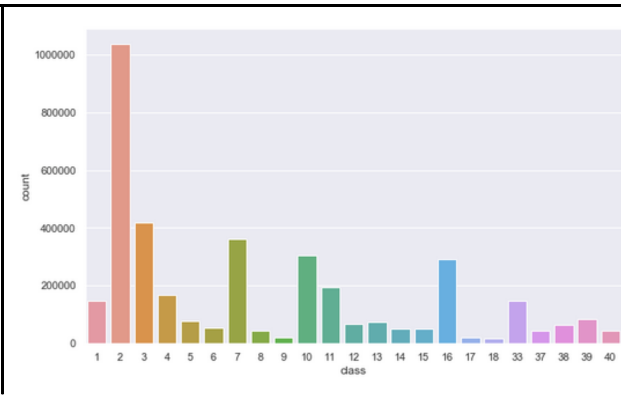


Fig8b: Occurrence of classes in dataset after dropping

While doing this smooths out some of the noise in our data, we also acknowledge that we lose some level of signal as well.

Feature Engineering

This is the second major components of our pipeline and consists of two stages:

- Feature Creation
- Feature Selection

New features are generated to capture temporal properties, spatial properties, seasons, holidays, variations of weather features, smooth out weather-based features to reduce noise as well as create lags of the target. Due to the time series nature of the data, there is an inherent autocorrelation present and we aim to capture this. The duration of the lags is obtained by considering the autocorrelation in the data. This shows a variability with the first 52 weeks with the highest peaks being observed early and later within that period

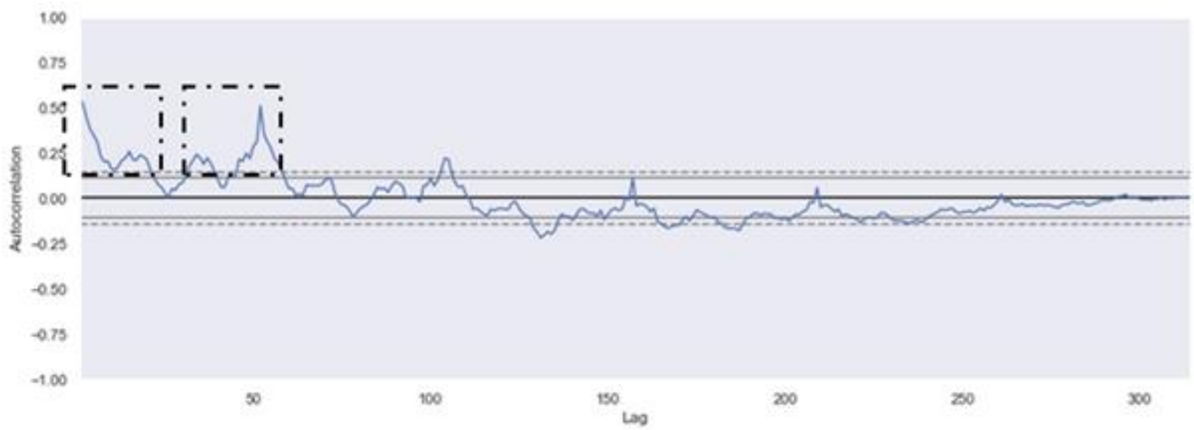


Fig9: Autocorrelation in demand

We then investigate the correlations between features and as a first step, drop features that are highly correlated to handle multicollinearity. Next we adapt the recursive feature elimination which removes the weakest feature recursively eliminating a small number of features per loop.

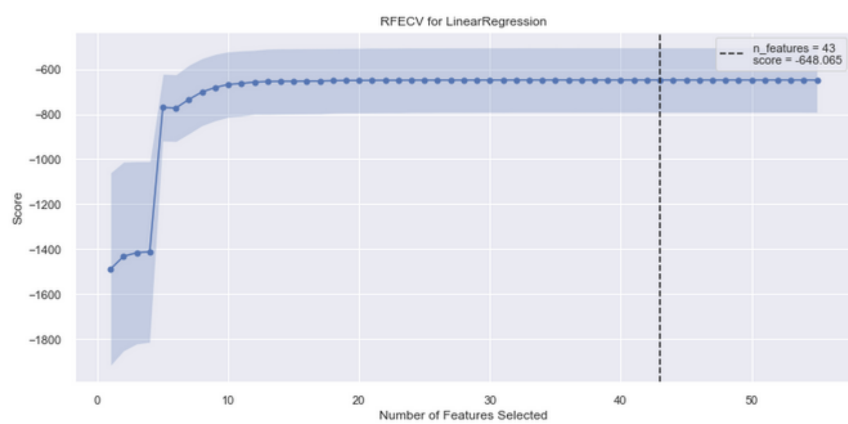


Fig10: Recursive Feature Elimination

Data Modelling and Evaluation

In order to model the data, this phase is split into 2 stages to achieve the goal of this project. Modelling the data to monitor the impact of weather features on demand forecasting and building a final model to forecast demand on the short term.

Demand modeling works in a bottom-up matter. It breaks the demand components into a series of internal and external factors and looks at how each impacts demand to predict future demand. We adapt a step-wise approach and consider different levels of weather information.

- First, we use our model to investigate the location-specificity of the weather effect on historical retail sales

- Next we consider class within the product hierarchy

Demand forecasting software can usually factor in climate and seasonality, but weather data adds a more granular time-sensitive signal. There were some assumptions surrounding our process.

- We choose to model a forecast within the first week because of the variability and uncertainty associated with weather forecasting.
- We represent **demand lags** as features as weather changes often do not immediately translate into demand.
- We choose to explore machine learning algorithms rather than their statistical components because we believe modelling weather-based demand forecast can be complex and introduces more variables into the modelling process. The machine learning approach eliminates the need for multiple statistical assumptions on lags for the different types of products and is able to capture the demand latency.

Other considerations that were taken into account:

- Tree based and Deep learning models are chosen for their ability to model feature interactions even if transient in time, so that they capture non-linear relationship between target and regressor
- Tree based models are trained in non-linear manner, where lagged values of time varying features are used to capture temporal dependencies
- LSTM model was trained without explicitly coding temporal dependencies through lagged features as done with other models.

We also pay attention to other properties of time series data like seasonality and trends and account for these by applying differencing to the demand. We find this doesn't make a real difference while modelling.

City-Level Modelling

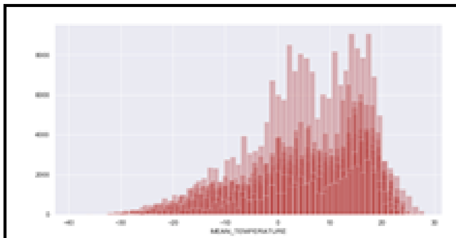
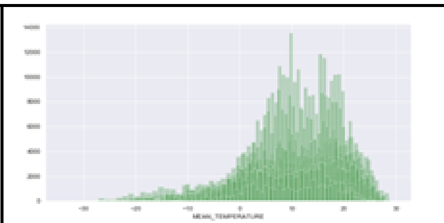
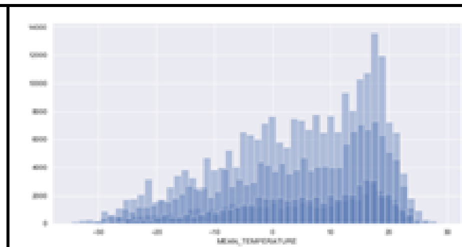
		

Fig11a: Temperature Distribution, AB	*Fig11a: Temperature Distribution, BC*	*Fig11a: Temperature Distribution, SK*

We inspect the distribution of our weather features and target feature to determine how to best capture the effects of weather. We notice while the cities within each province have different spreads (minimum and maximum) for weather features, the distribution is different. We decided to build 3 models (on accounting for each province) to investigate the city level weather effects.

We model the data with and weather features and evaluate our finding based on how our model error changes and more intricately by how our model predicts during different time periods. Various models were tried during this process (Lasso Regression, Random Forest, GBM). We find GMS to model the data better and provide better results.

After exploring the impact of weather features, LSTM models are built to forecast the next 5 days. We train multiple models for each day and evaluate the model by using its learning curves.

Class level Modelling

From our initial analysis, we find that classes can be quite seasonal and hypothesize that this would lead to weather features improving our model's ability to forecast demand. The data is restructured to forecast the demand for each class in each city. We follow a similar approach as the city level modeling and evaluation.

OBSERVATIONS

Observations - City Level Modelling

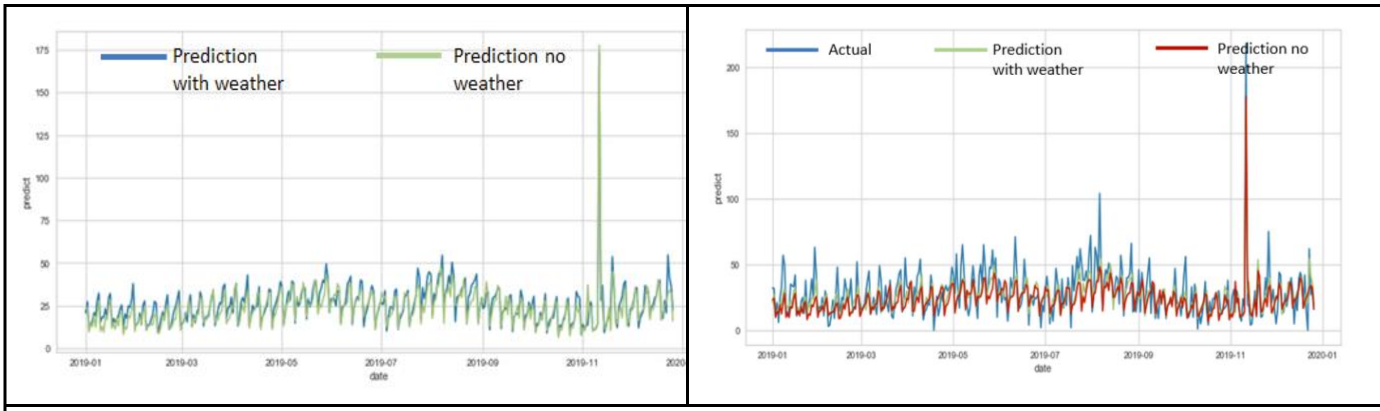


Fig12: Demand Forecasting with and without weather data on a city level

The above figure shows the demand forecasting for CAMPBELL RIVER with and without weather features. It is observed that the forecast with weather features is slightly higher than the forecast without it. Both forecasts follow the same trends. Comparing the actual target, we see that while both models try to capture the overall trend in the data, the predictions from the weather enriched dataset performs slightly better. Though both models are able to capture outliers that might be due shopping events (yearly spike in November), they fail to capture an increase in demand which occurs in the summer months.

	RMSE_ww	RMSE_w	MAE_ww	MAE_w
Validation_BC	20.5987	19.3379	12.0358	11.5983
Validation_AB	18.2484	18.1030	13.4005	13.2503
Validation_SK	18.9074	18.5717	13.5476	13.3775

**ww - without weather*

**w - with weather*

We initially considering using MAPE to evaluate our model but this metric breaks down when you have 0 target values.

Observations: Class-Level Modelling

We observe that building a model for each city-class level resulted in a small dataset where-in the model was not able to predict more detailed trends in the data like daily spikes in demand.

We remedy this by clustering all cities within a province and building a model for each province accounting for class. This leads to better predictions overall which were able to capture short term fluctuation in demand.

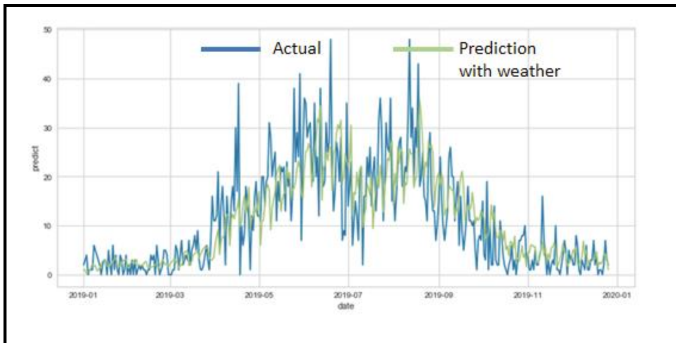


Fig13a: Actual vs Prediction with weather features

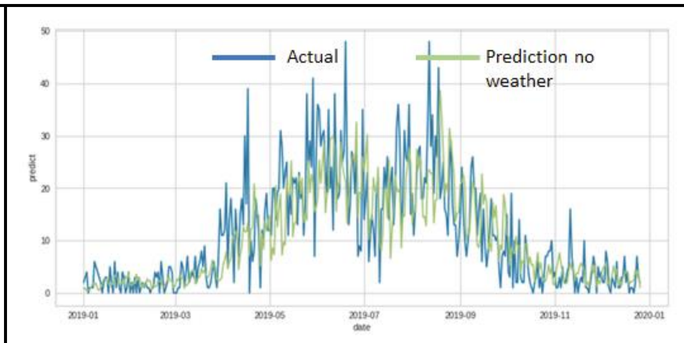


Fig13b: Actual vs Prediction without weather features

During the modelling processes, RFE led to no weather features being selected as a subset. This is suspicious and we went on to model the data both with and without weather features. We find that for a seasonal item, the addition of weather details leads to little or no change in demand forecasting on the short term.

Observation: Weather-based features

We focused our attention on temperature and precipitation, snow and rain because we found other features n our dataset were mostly correlated, so they were removed to avoid collinearity issues in our analysis. We found that while demand lags were the most influencing features during forecasting, temperature and precipitation lags contributed more to the model in terms of weather.

ARCHITECTURE

For our project we decided to build a full stack web-app that utilizes our model and the weather data from an external API service. We decided to go with a blend of cloud as well as local technologies.

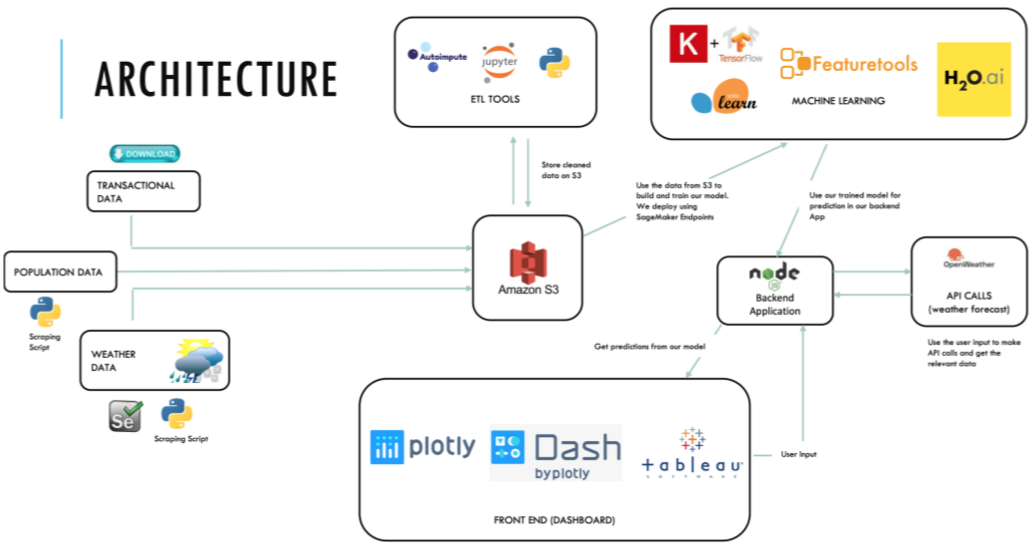


Fig14 : The Architecture

All our datasets, scripts and models were saved to S3. For extraction, transformation and loading (ETL) of the data we utilized Autoimpute and our vanilla pandas and numpy in Jupyter notebooks. Once the ETL was performed and we got the refined data we saved it unto a new S3 bucket. For our modelling and evaluation, we utilized TensorFlow, Kears, ScikitLearn and H2O.ai. We utilized node.js for backend and react.js / material design for our front end. We also spawned-child process to run our python files from node.js. Plotly was used for visualizations and the cloud service from plotly hosts our plots and animations. In the next section we describe our webapp and our work-flow in detail.

WEB-APP

Our idea behind building a webapp was to give the end user as much visual information about the dataset and project as possible, hence we decide to go with an interactive dashboard design. The dashboards contained with the website would give the end-user all the relevant information regarding the data, so even if a person has minimal domain knowledge they would still be able to make sense out of the datasets and contribute towards building an effective model.For the backend part of the project we decide to go with node.js, as node is very scalable has a major performance benefit over python backend.

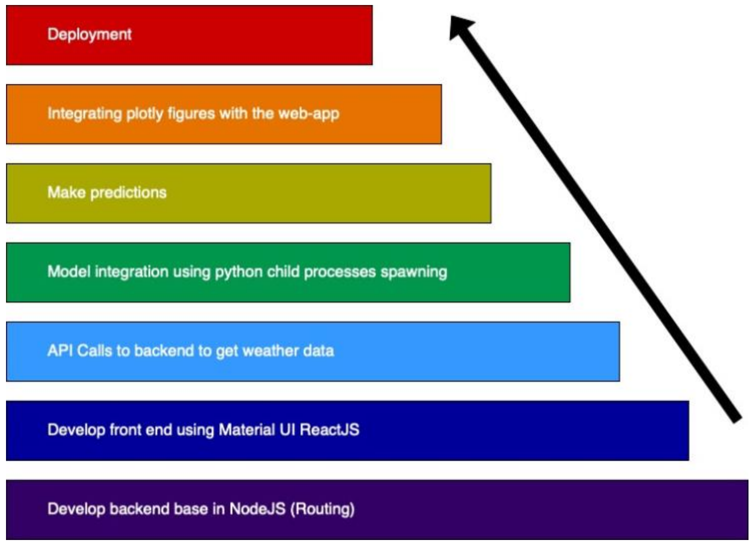


Fig 15: The Process

We also hosted our python-plotly scripts in the cloud and then displayed them unto our website, for the prediction part of the webapp. We make 2 calls in synchronous order: the first call is to the weather API endpoint to extract the useful features, then we utilize that information as input parameters and make a call to our machine learning model to get the predictions. To make a call to our ML model, we need to spawn multiple child processes in the background and this can be done utilizing node's inbuilt spawn module.
Here are some screenshots from our web-app:

	
Fig16a : Homepage	*Fig16b: Trend Analysis Dashboard*

LESSONS LEARNT

This project provided an opportunity to apply everything we had learnt during this term in CMPT733. We got the chance to work with real data and all its nuance. Some of the lessons that stood out for us include:

- Understand data modelling from a business point of view.
- Dealing with missing data and various imputation techniques. We had to chance to use autoimpute and explored various imputation techniques for time series data. We find that if more than one datapoint is missing and multiple datapoints are missing contiguously, methods like statistical imputation should not be used.
- We also got introduced to working and modelling time series data. Modelling time series data with machine learning algorithms can be challenging especially on a multivariant and multiproduct level.
- Deploying a model in real time.

FUTURE WORK

- ❏ Identifying 'seasonality clusters' that exhibit similar seasonal demand patterns.
- ❏ Refining these demand trends around the clusters could yielding more accurate results
- ❏ Further drill down on granularity based on the product hierarchy to perform forecasting
- ❏ Building an Amazon Sagemaker Endpoint to enable calls to model with an API

REFERENCES

1. **Demand Forecasting in the Fashion Industry: A Review** Maria Elena Nenni*, Luca Giustiniano, Luca Pirolo, First Published January 1, 2013
2. https://www.intellias.com/forecasting-demand-and-sales-in-fashion-retail-bet-on-data-not-fortune/
3. Adrianne Pasquarelli. "Weather Woes Spell Red for Retailers," AdvertisingAge, November 23, 2015. http://adage.com/article/cmo-strategy/weather-woes-spell-red-retailers/301461/, accessed November 2, 2016.
4. https://forecasters.org/wp-content/uploads/gravity_forms/7-c6dd08fee7f0065037affb5b74fec20a/2017/07/An-Overview-of-Retail-Forecasting_short.pdf

APPENDIX

Link to git repository: https://csil-git1.cs.surrey.sfu.ca/oojameru/cmpt-project-3er

Product Web-App: https://teamweatherornot.herokuapp.com/