# TrendCart

Integrating Viral Food Trend Predictions into Grocery Stores Demand Forecasting

**By: Sapan Patel, Akaash Dash**

# Table of Contents

**Motivation**.....3
**Related Works**.....5
**Dataset**.....6
Instacart Basket Dataset.....6
Grocery Dataset.....7
Trends Dataset.....8
Recipe Ingredients Dataset.....9
**Project Architecture**.....10
Underlying ML Model Training.....10
Trending Recipe Scraper.....11
Combined Predictive Analysis.....12
**Implementation Details**.....13
**Results**.....14
Underlying ML Model Training.....14
Trending Recipe Scraper.....14
Combined Predictive Analysis.....15
**Assumptions / Limitations**.....17
**Skills Learned**.....18
**Technical Skills**.....19
**Changes from Proposal**.....20
**References**.....21

# Motivation

The retail industry is going through a sudden, yet rapid evolution. The exploding popularity of short-form content, and platforms such as TikTok and Instagram, have changed the way people view products and shopping. This new era of digital ‘word of mouth’ has created a new retail ecosystem known as social commerce, whose opportunity is set to grow to $1.2T by 2025 [1]. In the different parts of this social commerce ecosystem, many have already jumped ahead of the curve and created effective ways to take advantage of this new-normal. However, one significant area of it has yet to be fully revolutionized - grocery stores.

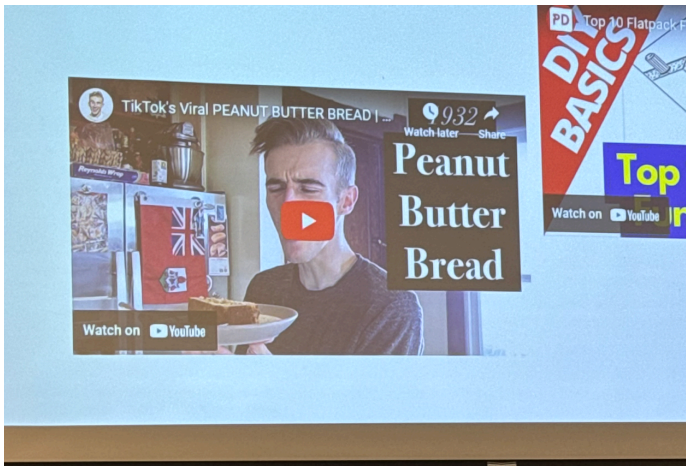
In the rapidly evolving landscape of the retail industry, grocery stores are set to be impacted significantly. It is seen that 83% of Americans engage with food or recipe content on social media, a staggering 89% admit that such content influences their cooking and purchasing decisions [2]. Many grocery stores have noticed this, and taken to social media to try and boost their social commerce presence in an attempt to get more customers and generate increased profit [3]. Yet, this attempt to join into the social commerce space is entirely situationally unaware. Grocers are not retailers who need to attract customers, but rather cornerstones of daily life that are going to be impacted by social commerce without actually needing to participate in it. What grocery stores should be focusing on is not who comes in their store, but what they buy.

Food and recipe content on social media consists of transitory trends, but these are not just for show. Viral recipes can drive temporary surges in demand for specific ingredients or products that are featured in viral content. Often, the quick burst of demand can cause stores to sell out of those items [4]. Unfortunately, the transient nature of these trends often results in missed opportunities for retailers. When stores sell out of trending items, they are unable to meet consumer demand, leading to potential revenue loss and customer dissatisfaction. To fully take advantage of these trends, grocery stores must be able to react to trends before they explode or as they explode and quickly react. They must also be able to know their position on the trend, as jumping on too late or matching trend demand even as the trend is dying can lead to oversupply and therefore a loss.

Recognizing this, we aim to develop TrendCart, a grocery store forecasting tool that is able to predict demand based not only on seasonality and historical sales, but also viral food trends as they are growing. The reason we want to build the project as a tool in an inventory management system and not as a standalone system is because trends impact stores differently. Depending on the current sales and inventory patterns, a trend can lead to a different shift in demand than another store.

Traditional demand forecasting systems often rely on historical sales data and seasonality to forecast demand, which cannot handle unpredictable viral trends, creating a cyber-physical gap. Our project, however, is designed to track and respond to these trends before they reach their peak, enabling store owners to adjust their inventory in real-time and avoid stockouts of in-demand items. This not only allows stores to improve inventory waste and turnover, but it also increases profits of store owners. This project provides an innovative blend of traditional retail data with modern social media analysis which can be used as a base for similar concepts.

While basing sales of TikTok might seem dubious, the impact is seen clearly in the younger generation. The majority of the younger generation is on TikTok and they frequently use TikTok as a source for lifestyle features and purchases [17]. For specifically food purchases, it is well known that most of the younger generation bases what they cook based on social media [18]. This has come to the point where we even analyze these trends in class:

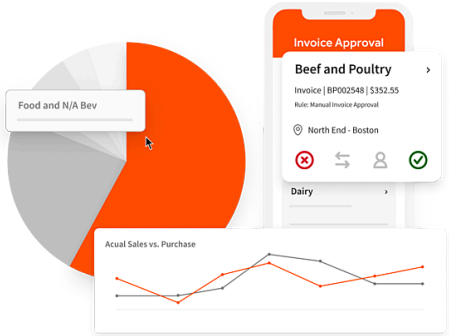



Anecdotally, we saw firsthand even books written entirely about TikTok recipes in bookstores:

This impact of culinary styles is not a trend but an entire shift in consumer behavior. Consumers now base most of their food choices based on social media, and it is a factor that cannot go unrecognized.

# Related Works

The most obvious related solutions in this problem space are traditional demand forecasting systems and inventory management systems. Two of the main pre-existing solutions are MarkT POS [11] and Toast inventory management [12]. While these do contain some data analytics, they only use internal data. They will predict when you will run out of certain ingredients or dishes based on past and current rate. However, they don't incorporate external data to predict demand for specific ingredients, which is the main advantage of our solution.




There are also other similar solutions that target other areas. For example Kalatzis, et. al surveyed frequent google trend keywords and their correlation to social media to show the correlation between the two [5]. They used google trends and social media engagements to extract data and make predictions. Schoen et. al. did similar work in other areas as well. These authors tested the correlation between google trends and real life results [6]. They correlated google trends and twitter engagements with real life ticket sales for a newly released movie.

While these concepts are similar in theory to our solution, the main differentiating factor is the application area difference. However, all of these, including ours, aims to bridge the cyber-physical gap between trends on the internet (cyber) and real life (ingredient sales).

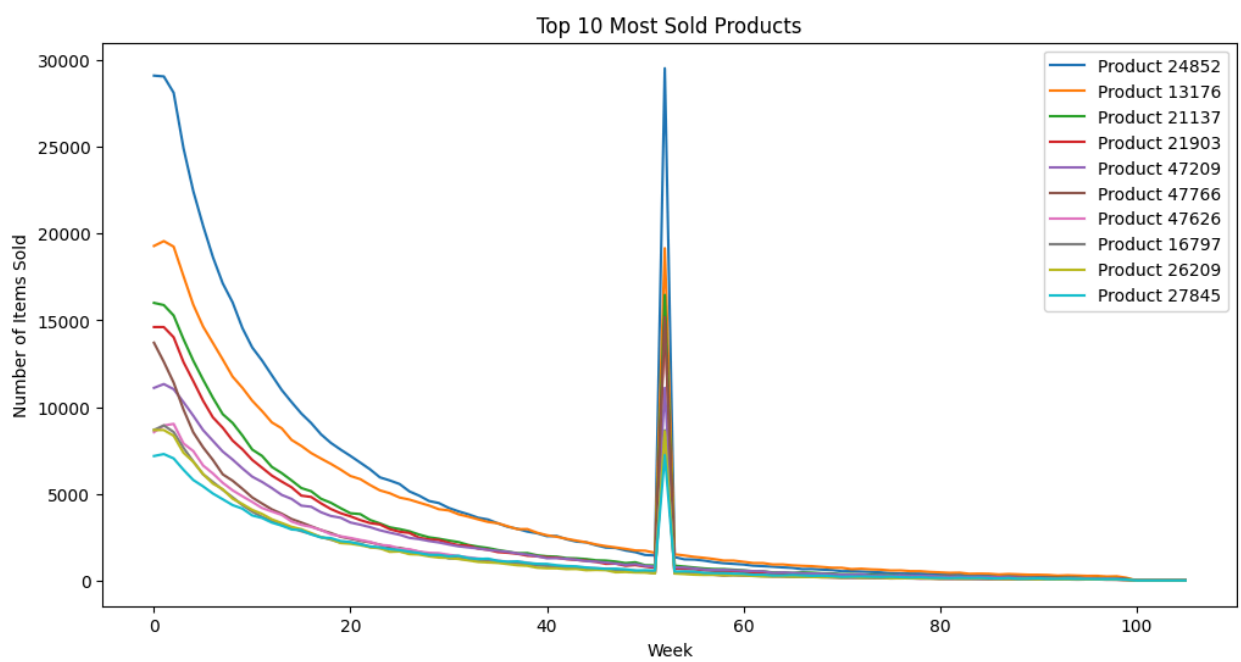
# Dataset

## Instacart Basket Dataset

We initially planned to use the Instacart Market Basket Analysis, but when exploring the data, we realized it was not a good fit for our project [9]. The first mismatch we observed was with the granularity of data. The dataset had a lot of extraneous data that would be good for some tasks, but not for out tasks. This includes information such as aisle, row, department, etc. Additionally, the products were listed as individual item products, and not by categories. This means that three different brands of, for example, pickled jalapenos would be listed as three different products. The separation of similar products led to there being 49688 individual products in the dataset. While this level of granularity makes sense for a grocery store and for an inventory management system of a grocery store, we are aiming to create a working version of our product before expanding to a full system, so the data was too granular for our current use case.

The second issue we ran into was that orders were not timestamped. While it initially appeared that orders were labeled by days from the start of measurement within the dataset, this was not the case. The dataset was created with the intention of exploring products bought together and products with related or repeated purchases, so orders were labeled by days since previous purchase by the same customer. This made it extremely difficult to place orders in relation to each other, entirely defeating the ability to see seasonality or timeseries purchases.

In an attempt to salvage the data, we tried to assume orders were placed one week after each other and explore what that would look like, but we were yielded the following results:



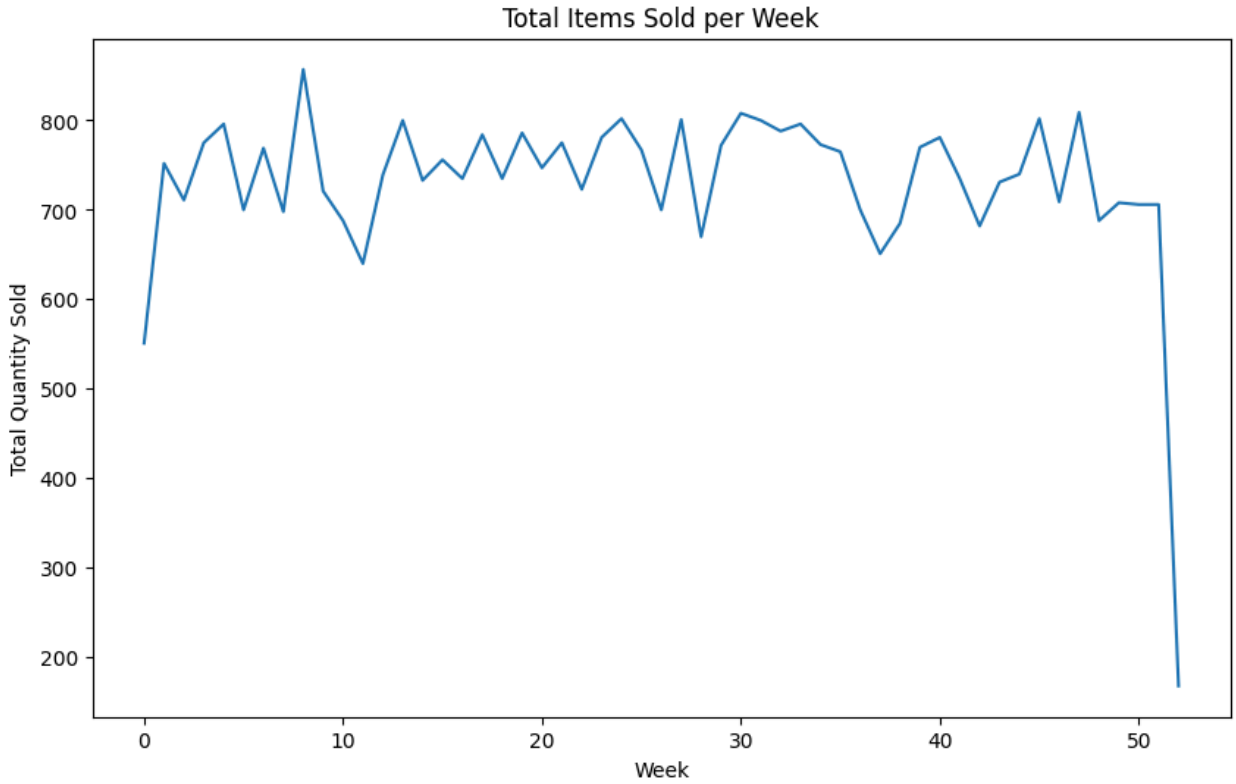

This does not match any realistic time series of purchases, so we decided to seek out alternative datasets.

## Grocery Dataset

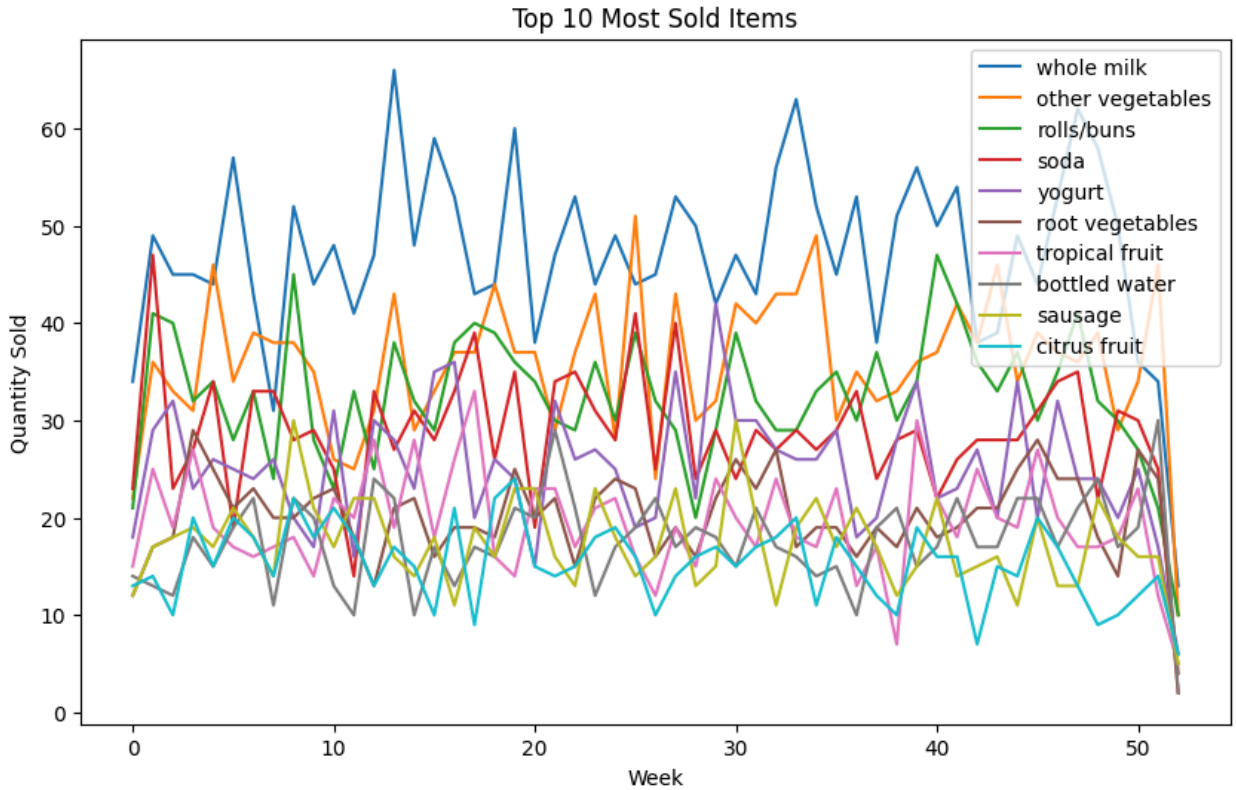
After looking on Kaggle, we found the Groceries dataset [10]. The dataset appeared to match our desired properties - less granularity of products and a time series basis. The dataset consisted of sales at a small grocery store and provided a purchase date and product for each purchase. Purchases were split up so that each product in a purchase had its own individual entry, but this was fine with us since we were aggregating by product. However, there were no product IDs.

The first step was to create a library of product IDs. We counted the unique products and gave them each a unique ID number. This resulted in having 167 products. After looking closer, we noticed that products were not always individual products, but were sometimes categories. For example, sometimes we had entries such as rice, but other times had categories such as berries. While this would not impact the prediction model explicitly, there were two consequences. The first is that products that might have different seasonality within them (such as berries) were grouped together, losing granularity that might be valuable to a grocery store. In addition, this would make integrating trending recipes difficult as we would not be able to directly search up the ingredient. Instead, we would need a way to categorize ingredients into the appropriate category to label them with product IDs.

Next, we needed to create a timeseries. We decided to use a week as the time interval for our analysis. This is in line with actual grocery stores as restocks from suppliers are unlikely to be done daily, but rather over a longer time period. Too frequent restocks would lead to getting burned by overhead costs, but too infrequent restocks would lead to lack of inventory. We decided upon a week as a good time granularity. We then aggregated the products by week and created a time series. We then graphed sales over a year:



We observed that the sales on the very first and very last weeks of the year are outliers, with significantly less sales than the other weeks. Upon further investigation, this is because the first and last weeks of the year are partial weeks with fewer days, but are calculated as full weeks. To avoid this from skewing the data, we drop these weeks in our training. To see how individual products perform, we graph the top ten most sold products:




This graph shows the same pattern of first and last week outliers, confirming the hypothesis. However, the individual product sales look good and show seasonality. This makes the transformed dataset perfect for our task.

## Trends Dataset

The groceries dataset allows for sales prediction, but the crux of our project is the inclusion of trends data in demand forecasting. To do this, we first needed to identify the source of demand. In the modern day, most viral trends emerge from Instagram and TikTok. This is because these platforms have dedicated user bases and short-form content, making them the perfect source for virality [13]. Unfortunately, neither of these platforms have legitimate and accessible APIs, if even having one at all. Because of this, we needed to find an alternative way to identify viral trends.

The conclusion we came to is that there must be some positive correlation between viral trends and Google searches. The reasoning behind this is that if someone sees a recipe viral on TikTok, they will likely search up the recipe online as well [5]. If recipes become viral from other sources, they must be searched and will likely show up as well. This is a major assumption that should be noted. To find trending recipes from Google, we can rely on the Google trends API.

We can query Google trends API for the largest week-over-week increases in topic searches to identify viral searches in a topic. For us, we choose the topic of “recipe”. However, the API is very limited. We can only query for changes over a time range, so we cannot query multiple weeks at once. When trying to query for the weeks in the time period of the groceries dataset, we frequently ran into rate limits. It appears that Google heavily rate limits their trends API, preventing us from getting a dataset for trends. To still include trending recipes in our forecasting, we rely on querying the requested week at inference time and including it in there.

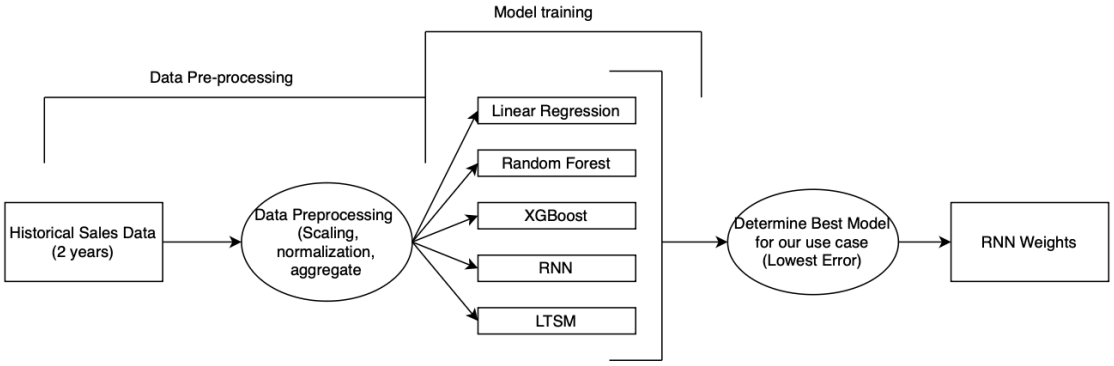
## Recipe Ingredients Dataset

Once a viral recipe has been identified, we then need to find the ingredients of the recipe. Spoonacular is a very well known food-related information provider, and they have a well documented and free API [8]. Included in their API is a way to search for recipes in their database and obtain the ingredients. In case a recipe is not available in their database, they have an API endpoint for which users can provide a link to a recipe and they will scrape and return the ingredients. Because of the tools provided, we do not create our own recipe ingredients dataset, but rather we use the Spoonacular API to rely on their datasets.

One point to note is that SpoonacularAPI requires multiple requests per recipe, and they allow very limited API requests per day. Raising the limits is very expensive, creating another roadblock if we were to have created a historical trending recipe dataset.

# Project Architecture

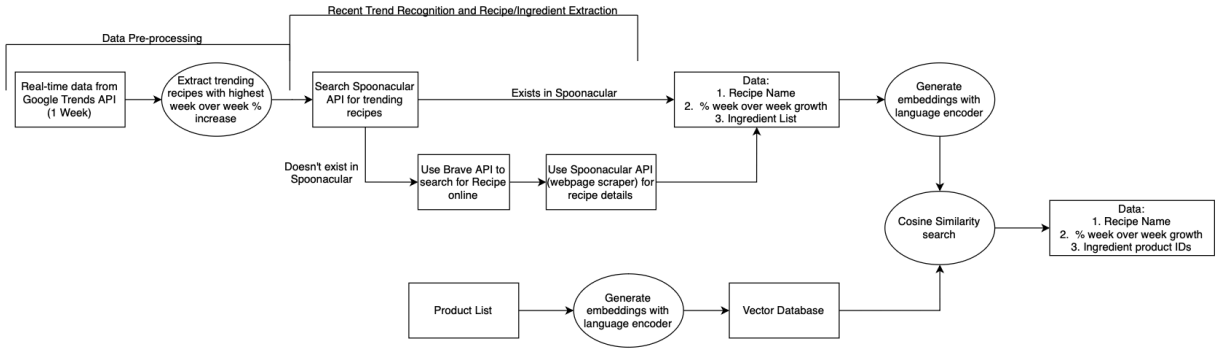
## Underlying ML Model Training




The first part of TrendCart is to create a sales forecasting model. The first step is to preprocess our data. We start with the transformed groceries dataset which has aggregation and scaling applied as mentioned in the dataset section. We then normalize the data using a MinMaxScaler from Scikit-learn [14]. We normalize the data because the volume of sales might be different across stores and years, but the underlying seasonality and trends should remain the same. This is a very important step because it improves the efficiency and quality of the models during training. If we did not pre-process the data in this manner, the outliers would skew the predictions too largely. We don't want to use a Gaussian normalizer as that would put some data into the negatives and possibly revert to larger negatives when denormalizing. Thus, we use a MinMaxScaler as it keeps the scaled values proportional to each other while remaining in a positive range.

After the data has been preprocessed, we train the models. Instead of picking one initial model, we train several models to identify if one does a better job of trend recognition than the others. Candidate machine learning models are linear regression, random forest, and xgboost. All of these models are known for their predictive abilities. However, we anticipate that some of the trends may be nonlinear, so we include deeper models such as RNN and LTSM. After training and testing the models, we found that the RNN model had the best performance.

## Trending Recipe Scraper

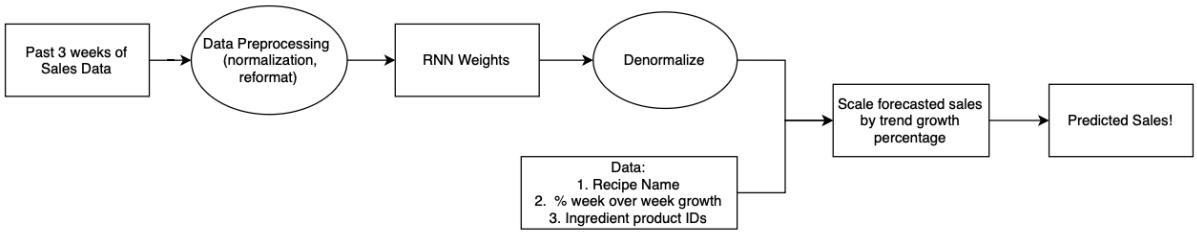



The next component of our project is to gather ingredients that will have a spike in purchases due to recent virality. As explored in the dataset section, we can observe viral recipes using the Google Trends API. We query this API for ‘recipes’ and find recipes with large positive week-over-week change. We are returned with the name of the recipes and the percentages of change. From these, we sort and select recipes with a change greater than 50%.

The next step is to get the ingredients of the recipe. For this, we refer to the Spoonacular API and Brave search API as needed [15]. We search the Spoonacular database for the recipe. If it is found, we then extract the ingredients. If it is not found, we must find the ingredients in an alternative way. Brave search engine provides a free API and we use this to search the web for the trending recipe and select a relevant recipe webpage. Spoonacular API provides an endpoint to obtain recipe ingredients from a webpage. We provide our relevant recipe webpage to this endpoint and then obtain the ingredients.

However, these ingredients are not directly indexable into products. Our products used in the predictive model are sometimes presented as categories instead of individual products. In order to correctly find the right label, we must find which product closest resembles each ingredient. To do this, we use a language encoder model and cosine similarity search [16]. The language encoder model is used to generate embeddings, or vector representations of products and ingredients. Because this is only an encoder model, we do not have to worry about predictive outputs and therefore hallucinations. Cosine similarity is industry standard for finding closest relevance between language embeddings, and we follow this practice [16]. We first use our encoder to generate a vector database of all products. Then, for each ingredient, we create an embedding and use cosine similarity on our vector database to find which product is closest in relevance. From this, we are able to label each ingredient in trending recipes with a matching product ID.

## Combined Predictive Analysis




We now integrate the ML architecture and trending recipe scraper into TrendCart. We utilize the real-time sales data of the past three weeks and start by preprocessing the data. Similar to the ML architecture training, we have to normalize and reformat the data into usable and efficient data. Then, we use the saved RNN weights to create a preliminary prediction. After that, we denormalize the preliminary prediction to match the scale of the input data. Finally, we augment the scaled forecasted sales per product using the week over week percent growth. In other words, for each ingredient in a viral recipe, we scale the matching product sales forecast by the week-over-week growth.This results in the predicted sales for the upcoming week taking into account viral recipes. The stores can use this prediction to increase stock of ingredients that are trending up in the recent week, real-time.

# Implementation Details

TrendCart was implemented fully using Python and its libraries. Python was the most suitable tool to use for our use case due to many reasons such as its ease of use, integration with data analysis libraries, API requests libraries, and extensive machine learning libraries.

All the past data from the sales is formatted in csv files and we converted those into .npy files since we are working with Python and the numpy library is very efficient at working with data at this scale. However, the data from the API's (Google Trends API, Spoonacular API, and Brave API) are formatted in JSON since that is the format returned when making HTTP requests with them. All the data preprocessing is done using the pandas library since it is the most compatible Python library to import csv files and work with them. The pandas library includes many functions that were used in the pre-processing step outlined in the architecture above. Since we are training many different models, we did not use one single library for implementing them. Instead we use multiple: from SciKit-Learn we implemented Linear Regression and Random Forest Models, from TensorFlow, we implemented LSTM and RNN, and finally we used XGboost. While all this is very useful, there is no value provided if it is not understandable and easy to access the information. Therefore, we use matplotlib to plot the graphs and visuals using the predictions and data.

# Results

## Underlying ML Model Training

From our training, we obtain the following results for the machine learning models:

```
Linear Regression R^2 Score: -0.9981017506615912
Random Forest R^2 Score: -0.15533955861504445
XGBoost R^2 Score: -0.6429575482844548
```

As we can see, all of the machine learning models had negative r-squared values. This indicates that the models could not explain any of the variability. Not only this, but the values were negative, meaning the predicted values perform worse than using the average as a predicted value.

This indicates that a traditional machine learning model will not work. It is likely that the seasonal data is non-linear or follows other patterns that a deeper model would be needed to pick up. We then look at the results of our deeper models:

```
LSTM Mean Squared Error: 0.05794665217399597
```

```
RNN Mean Squared Error: 0.06031983718276024
```

We see that both models had very good mean squared errors. This indicates that they are able to capture the patterns in the data and make accurate predictions. Both models had similar scores, but LSTMs are better with long-term dependencies and RNNs are better with short term dependencies. Because our training data is smaller and our inference data is only three weeks, we determined that short-term dependencies are more relevant and we chose the RNN model.

## Trending Recipe Scraper

When using the described approach, we are successfully able to gather all trending recipes and accurately classify them. The pictures below show the results of ingredient scraping and then product ID matching.

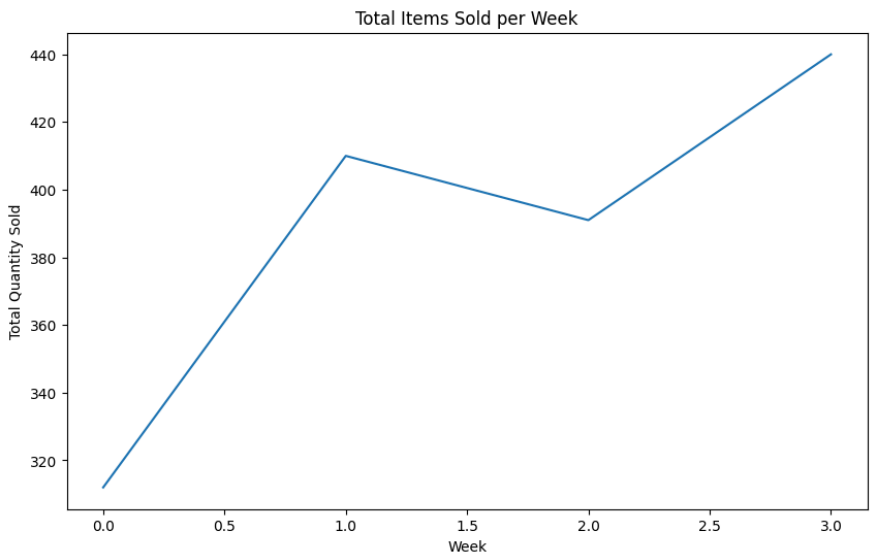
```
{
    "name": "pimento cheese recipe",
    "growth": 200,
    "ingredients": [
        "cheddar cheese",
        "cream cheese",
        "pimento peppers**",
        "mayonnaise",
        "garlic powder",
        "onion powder",
        "ground cayenne pepper",
        "jalape\u00f1o pepper",
        "pepper",
        "salt"
    ]
},
{
    "name": "hummingbird food recipe",
    "growth": 50,
    "ingredients": [
        "carrot",
        "baking soda",
        "spices",
        "all purpose flour",
        "salt",
        "olive oil",
        "vanilla",
```

```
{
    "recipe": "pimento cheese recipe",
    "growth": 200,
    "ingredients": [
        129,
        66,
        163,
        72,
        113,
        60
    ]
},
{
    "recipe": "hummingbird food recipe",
    "growth": 50,
    "ingredients": [
        1,
        34,
        35,
        4,
        69,
```

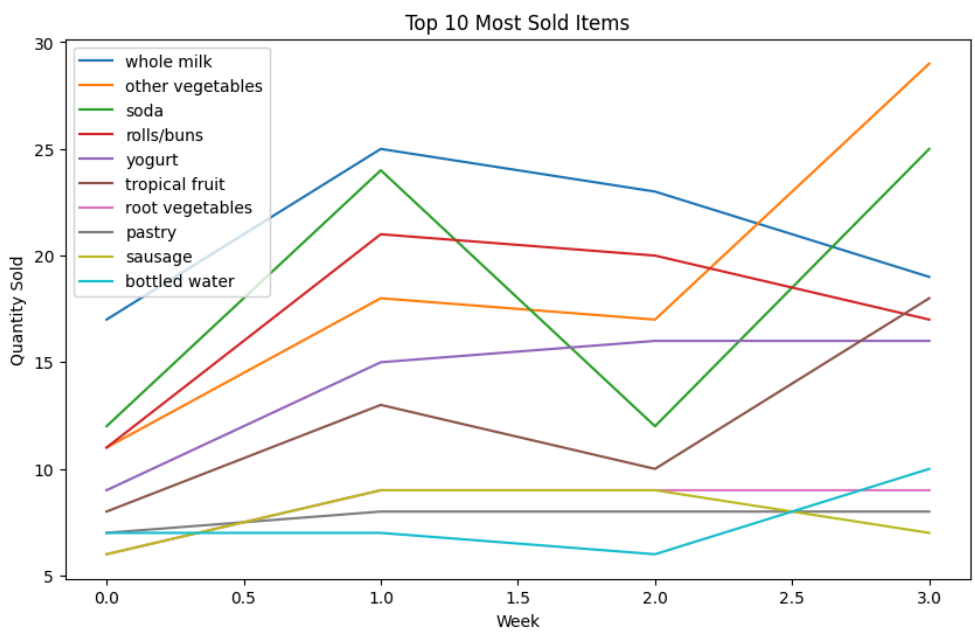

From this, we can see that all recipes with week-over-week growths of over 50% are gathered. Additionally, we are able to successfully scrape ingredients either by using the Spoonacular database or via web recipe ingredient scraping. The picture on the left shows that product matching is successful via the cosine similarity search. Manual evaluation of a few samples indicates that the matched product IDs are accurate to the listed ingredients.

## Combined Predictive Analysis

To test our combined forecasting model, we simulate how the pipeline would be used in a full system. To create a prediction, the pipeline would use the date of the current week and sales of the previous three weeks. We take the week to be the present day current week and create three weeks of simulated sales by indexing the relatively past three weeks and sampling from the distribution of the two years historical data in the same weeks. We then put these into the pipeline and obtain a prediction. To evaluate the predictions, we evaluate relative to the previous three weeks. The first evaluation is to observe the total number of sales compared to previous weeks. The following graph shows this information (where the previous weeks are 0,1 and 2, and the predicted week is week 3):




The graph indicates that the predicted number of sales are within a reasonable expectation of the previous sales. This is indicative of a successful prediction. There is also some upwards trend from the previous week which could be a result of seasonality or viral trend scaling. Next, we want to see if seasonality or virality are accurately captured - that is, they apply to some products while not to others. We plot the top ten most sold items to see this:




From this graph, we can see that some products grow while some go down, indicative of successful capturing of seasonality. The sales per week of products change based on seasonality and may either go up or down, which is accurately represented. We also note that product sales change at different rates. This is indicative of successful capturing of virality. Some products are scaled more than others, showing that they are likely a part of viral trends (specifically those with high growth rates). Overall, we would evaluate the pipeline to be successful based on these results.

# Assumptions / Limitations

There are several limitations of this project that are based on assumptions of conditions. First, we assume that Google Trends captures true virality. Google Trends gauges interest in specific topics over time based on search query volumes. However, some trends may not reflect in search data due to their circulation through other mediums. Additionally, Google Trends data is aggregated and may not capture localized spikes in interest that could significantly impact a single grocery store or geographic area.

Secondly, we assume that Google Trends virality translates to real world sales. While a spike in Google Trends might indicate increased public interest, it does not guarantee a corresponding increase in product sales. Factors such as seasonality, local preferences, economic conditions, and the difference between interest and purchasing intent can influence the translation of virality into physical sales. This assumption is a form of cyber-physical gap and introduces a risk of overestimating the impact of viral trends on demand, leading to potential overstocking and waste.

Third, we assume good conditions of the world and no black swan events. This assumption fails to account for unforeseen sudden conditions that impacted the likes of Southwest and others. The model will be built on the assumption of static conditions, and may lack the flexibility to adapt to these changes, potentially leading to inaccurate forecasts and recommendations. This in essence means that our project is limited to ‘virtual-time’ to some extent.

Fourth, time constraints and expectations in actually implementing the project heavily impacted the scale in which we were going for. Throughout the timeline of the project, based on feedback from the teaching team, we focussed on the backend portions of our initial project, which are inevitably the more innovative problems. However, including a front-end dashboard that makes it easy for non-technical users to understand the data and analysis done would have a huge impact.

Finally, we predict that the models will struggle with knowledge obsolescence. Machine learning models are only as good as the data they are trained on. Long-term changes in consumer preferences can lead to shifts over time. This leads to the risk that the forecasting capabilities will become outdated, leading to suboptimal inventory decisions.

# Skills Learned

There are many meta-skills that we obtained throughout the journey of this project. The first is applying theory in practice. A lot of the models, fine-tuning, and exploration processes used in this project were topics that we had learned in classes. Figuring out how to use the learned knowledge to create tangible products was a skill that we gained.

We then learned to have situational awareness. This primarily comes in the form of identifying assumptions. When starting this project, we had many assumptions of what data and technologies exist and how things work, but these were quickly proven wrong. We had to gain the skill of identifying assumptions in our own knowledge. Also, we created this project on a smaller scale, so many assumptions are included. Being able to identify these assumptions allows us to see what the weak points of our project are and where it would fail if attempted to be used in the real world. Some of these assumptions include the cyber-physical gap and knowledge obsolescence. The entire concept of the project pertains to the cyber-physical, as we are aiming to bridge the gap between the cyber trends and physical sales. However, we also recognize that our project displays its own forms of cyber-physical gaps. We also recognize that trends and patterns change, so any models or pipelines we create may suffer from knowledge obsolescence over time. These situational awareness skills can be extrapolated to other areas of our career.

The next meta skill is trend recognition. In creating this project, we had to identify trends in society to see what projects would be most successful in the current day. This involved looking at the technology push of not only systems enabling deployment, but the research and steps involved. This also involved looking at application pull - many recent projects can show us the trend of applications and what people are looking for. On top of this, our project relies on being able to identify where in the trend cycle you are in. If you are aware of a trend, then you can take advantage of it on either the rise or the wane. This skill is extremely important to our careers, and implementing this project allowed us to get practice with it.

Lastly, we learned to manage our expectations. When creating our pre proposal and proposal, we overestimated what we were capable of and the time that was available to us. Feedback from the professor and the TAs emphasized this point and said that we should scale back to focus on the core components. This skill is extremely important, as when tasks are assigned in the workplace, we must have enough self-awareness and situational awareness to know how much time a task will actually take us to complete.

# Technical Skills

The implementation of the Grocery Sales Forecasting and Inventory Management Dashboard encompasses a comprehensive skill development plan for us. In the realm of data acquisition and preprocessing, we refined out abilities in cleaning and integrating diverse datasets, master the art of API integration using tools like ‘pytrends’ and Spoonacular, and gain proficiency in web scraping techniques employing libraries such as BeautifulSoup and Spoonacular.

Moving into data analysis and machine learning, we delved into machine learning model development, experimenting with both traditional approaches like Linear Regression or random forest and advanced techniques like LSTM. Feature engineering skills were honed for extracting meaningful insights from large, extensive datasets, and we grasped the intricacies of hyperparameter tuning to optimize model performance.

# Changes from Proposal

We were given a lot of feedback to have more realistic expectations and focus on the core product of viral trend recognition and forecasting. Based on the feedback we received, we decided to focus on the core product of viral trend recognition and adjusted our expectations accordingly. We concentrated our efforts on real-time trend detection and sales prediction, as these are the more innovative challenges we set out to solve. These problems are highly relevant to bridging the cyber-physical gap through trend prediction and utilizing machine learning for sales prediction based on "pseudo" real-time data. To streamline our approach, we dropped the website component and instead focused on choosing, picking, and making good data. We also prioritized creating appropriate models and generating representative results.

# References

[1] Accenture. “Why the Future of Shopping Is Set for a Social Revolution.” *Accenture.com*, Accenture, 2 Jan. 2022, www.accenture.com/us-en/insights/software-platforms/why-shopping-set-social-revolution.

[2] Hamstra, Mark. “Sure, TikTok Is Fun. But Can It Drive Sales?” *Supermarket News*, 14 Dec. 2022, www.supermarketnews.com/online-retail/sure-tiktok-fun-can-it-drive-sales.

[3] Evans, Michelle. “How Gen Z Inspired Grocers to Reinvent Their Social Presence.” *Forbes*, 29 Nov. 2022, www.forbes.com/sites/michelleevans1/2022/11/29/how-gen-z-inspired-grocers-to-reinvent-their-social-presence/?sh=639ba5484692. Accessed 10 Feb. 2024.

[4] Youn, Soo. “Trader Joe’s Sold out of Kimbap, Its Latest Viral Offering, Thanks to TikTok.” *NBC News*, 6 Sept. 2023, www.nbcnews.com/news/asian-america/trader-joes-kimbap-tiktok-rcna103403.

[5] Kalatzis, Nikos & Roussaki, Ioanna & Matsoukas, Christos & Paraskevopoulos, Marios & Papavassiliou, Symeon & Tonoli, Simona. (2018). Social Media and Google Trends in Support of Audience Analytics: Methodology and Architecture.

[6] Schoen, Harald & Gayo-Avello, Daniel & Metaxas, Panagiotis & Mustafaraj, Eni & Strohmaier, Markus & Gloor, Peter. (2013). The power of prediction with social media. Internet Research: Electronic Networking Applications and Policy. 23. 10.1108/IntR-06-2013-0115.

[7] Google. (n.d.). *Google Trends*. Google trends. https://trends.google.com/trends/explore?geo=US&q=recipe&hl=en-US

[8] Urbansky, D. (n.d.). *Documentation*. spoonacular recipe and food API. https://spoonacular.com/food-api/docs#Extract-Recipe-from-Website

[9] *Instacart Market Basket Analysis*. Kaggle. (n.d.). https://kaggle.com/competitions/instacart-market-basket-analysis

[10] Dedhia, H. (2020, September 17). *Groceries dataset*. Kaggle. https://www.kaggle.com/datasets/heeraldedhia/groceries-dataset

[11] *Markt POS | Home*. (n.d.). Www.marktpos.com. Retrieved April 21, 2024, from https://www.marktpos.com/

[12] Toast. (n.d.). *Toast | Restaurant Point of Sale & Management System*. Pos.toasttab.com. https://pos.toasttab.com/

[13] Shahbaznezhad, H., Dolan, R., & Rashidirad, M. (2021). The Role of Social Media Content Format and Platform in Users' Engagement Behavior. Journal of Interactive Marketing, 53(1), 47-65. https://doi.org/10.1016/j.intmar.2020.05.001

[14] *Sklearn.preprocessing.MinMaxScaler*. scikit. (n.d.). https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html

[15] Brave Search API. (n.d.). Brave. Retrieved April 21, 2024, from https://brave.com/search/api/

[16] Sitikhu, P., Pahi, K., Thapa, P., & Shakya, S. (2019, October 30). *A Comparison of Semantic Similarity Methods for Maximum Human Interpretability*. ArXiv.org. https://doi.org/10.48550/arXiv.1910.09129

[17] Kennedy, J. (2024, March 15). *Who Is the TikTok Shopper?* The Business of Fashion. https://www.businessoffashion.com/articles/retail/who-is-the-tiktok-shopper/

[18] Pizza, H. R. I., & Mary. (2023, August 16). *Culinary Confessions: Cooking Habits of Gen Z & Millennials*. Home Run Inn Pizza. https://www.homeruninnpizza.com/news-blog/fun-facts/cooking-habits-gen-z-millennials/