Meal Delivery Demand Forecasting

Project Overview:

Client Background

Our client operates a meal delivery company that serves multiple cities through various fulfilment centres. These centres are responsible for dispatching meal orders to customers, ensuring timely and efficient delivery.

Project Objective

The primary objective of this project is to assist the client in accurately forecasting the demand for meal orders in the upcoming weeks. This will enable the client to optimise the replenishment of raw materials and manage staffing requirements effectively, ensuring smooth operations and minimising waste.

Business Need

1. **Perishable Inventory Management**:
    - The majority of raw materials used in meal preparation are perishable. Accurate demand forecasting is crucial to maintaining the right inventory levels, reducing spoilage, and minimising waste.
2. **Staffing Optimization**:
    - Staffing at fulfilment centres must align with the forecasted demand to ensure efficient operations. Understaffing can lead to delays and customer dissatisfaction, while overstaffing increases operational costs.

Features:

- Forecast the food order-client side
- Dashboard - client side

- Generate human-readable explanations of demand forecasts -client side
- Summarise costumer reviews - client side
- Recommend the food - user side
- Chatbot-user side for understanding about meal which is popular meals at specific restaurant.

LLM use case:

Customer Insights:

- **Use LLM to analyse and summarise customer reviews.**
- **Extract key themes, sentiments, and preferences.**
- **Chatbot for user to understand about food and restaurant .**

Forecast Explanation:

- **Generate human-readable explanations of demand forecasts.**

Recommendation

- Recommend the food based on customer desire (like spacy,sugar,veg etc)
- Recommend food items based on the most frequently ordered dishes at a specific restaurant.

Dashboard

- **Sales Trends Over Time**: Analyse how sales have changed over different weeks or months using barchart.
- **Pie Charts for Each Product by City**: Visualise the distribution of sales for each product across different cities.
- **Food Type Analysis**: Determine which type of food is selling the most using barchart.

- **Price Analysis of Each Food**: Analyse the price distribution of each food item.

Technologies:

1.Time series
2.LLM model
3.RAG / Fine-tuning
4.Langchain
5.python,numpy,scikit learn,pandas,matplotlib,seaborn
6.Power bi / Tableau
7.EDA,ETL,Web scrap
8.Django,html,css
7.MLflow and DVC
8.AWS,DVC

Additional Technologies

To enhance the project's capabilities and ensure efficient data handling and deployment, we will incorporate:

- **VectorDB**: For efficient vector-based storage and retrieval.
- **MySQL and CassandraDB**: For robust database management.
- **CI/CD Pipelines**: For automated testing and deployment.
- **MLflow and DVC**: For managing machine learning experiments and version control.

Week 1: Foundation and Setup

Day 1-2: Project Setup

- **Define Scope:** Define project scope, milestones, and deliverables.
- **Version Control:** Set up version control using Git and create a repository on GitHub or GitLab.
- **Project Management:** Set up project management tools like Trello or Asana.
- **Pendings:**

  Have a proper idea on the entire project

  Research on RAG, langchain

  Look for other data that might be of use (information on food)

  Prompt Engineering

Day 3-4: Data Exploration and Preparation

- **Data Exploration:** Load and explore the dataset. Identify key features and assess data quality.
- **Data Cleaning:** Handle missing values, outliers, and data inconsistencies directly within the code.

Day 5-6: Learning and Setting Up MLflow and DVC

- **MLflow:** Follow tutorials to set up MLflow for experiment tracking and model management.
- **DVC:** Install and set up DVC for dataset and model version control. Follow tutorials to understand basic usage.

Day 7: Initial Dashboard Design

- **Design Layout:** Plan the dashboard layout and features using Power BI or Tableau. Outline key visualizations needed.

Week 2: Core Model Development and Technology Integration

Day 8-9: Time Series Forecasting

- **Model Selection:** Choose and implement time series forecasting models (e.g., ARIMA, Prophet, LSTM).
- **Implementation:** Start coding and training the models.

Day 10-11: Learning and Implementing VectorDB

- **VectorDB:** Set up a local instance of VectorDB. Integrate it for vector-based search and retrieval.

Day 12-13: Learning and Setting Up AWS

- **AWS:** Learn basic AWS services (S3 for data storage, EC2 for deployment). Set up an S3 bucket for storing data.

Day 14: Web Scraping

- **Web Scraping Setup:** Implement web scraping to gather additional data (e.g., customer reviews). Use libraries like BeautifulSoup or Scrapy.
- **Data Integration:** Manually integrate the scraped data into your existing data pipeline.

Week 3: Front-End Development and Advanced Features

Day 15-16: Django Setup and Basic Front-End

- **Django Setup:** Set up a Django project and create the basic project structure.
- **Basic Pages:** Develop basic pages for the application using HTML and CSS. Focus on layout and navigation.

Day 17-18: Advanced Front-End Development

- **Dashboard Integration:** Integrate dashboard visualizations into the Django application. Use Django templates to render data.
- **Styling:** Apply CSS to style the dashboard and other pages. Ensure a responsive design.

Day 19-20: Advanced Model Development

- **LLM Integration:** Integrate a Large Language Model (LLM) to analyze customer reviews and generate explanations for forecasts.
- **Forecast Explanations:** Develop features to generate human-readable explanations using LLM.

Day 21: Testing and Validation

- **Model Testing:** Test and validate the forecasting models. Refine them based on results.

- **Front-End Testing:** Test the Django application for usability, responsiveness, and functionality.

Week 4: Finalization and Deployment

Day 22-23: Final Adjustments

- **Feature Finalization:** Complete all features, ensure integration of front-end and back-end components.
- **Bug Fixes:** Address any issues identified during testing.

Day 24-25: Deployment

- **Deployment:** Deploy the Django application to AWS EC2. Ensure data is properly stored and accessible on AWS S3.

Day 26-27: Documentation and Training

- **Documentation:** Prepare detailed documentation including setup instructions, usage guides, and troubleshooting tips.
- **Client Training:** Provide a walkthrough and training for the client on using the dashboard and interpreting the results.

Day 28: Project Review and Feedback

- **Review:** Conduct a final review with stakeholders. Gather feedback and make any necessary adjustments.

This plan includes learning and implementing MLflow, DVC, and AWS, while focusing on the core functionalities of the project and ensuring deployment within the 4-week timeframe.