Mini course sales forecasting

Problem Description :

Background: In the rapidly growing e-learning industry, companies offer mini-courses on various topics to cater to learners' specific needs and preferences. These mini-courses are usually short in duration, covering specific topics in-depth, and are often provided by multiple countries, stores, and products. Understanding the sales patterns of these mini-courses is crucial for optimizing inventory, pricing strategies, and overall business planning.

Dataset Information: The dataset provided for this project contains historical sales data of mini-courses. The dataset is divided into two parts: the training set and the test set. The training set contains the sales records along with additional features, including date, country, store, product, and the number of courses sold (target variable). The test set contains similar features without the target variable, which is to be predicted using the trained model.

Features:

1. **Date:** The date when the sales occurred.
2. **Country:** The country where the mini-course was sold.
3. **Store:** The specific store from which the mini-course was purchased.
4. **Product:** The type of mini-course offered.
5. **Number of Courses Sold:** The target variable representing the quantity of mini-courses sold on a particular date.

Problem Statement: The goal of this project is to build a sales forecasting model using machine learning techniques to predict the future sales of mini-courses for different countries, stores, and products. The model should accurately predict the number of mini-courses that will be sold on specific dates to help the e-learning company make informed decisions regarding inventory management, marketing strategies, and revenue projections.

Objective: The primary objective of this project is to develop a robust forecasting model that minimizes the forecasting error and maximizes the accuracy of predicted sales. The model should account for seasonality, trends, and other relevant factors that impact the sales of mini-courses. The final model will be used to generate predictions for the test dataset, and the results will be submitted to evaluate its performance in the sales forecasting competition.

Expected Outcome: The successful completion of this project will result in a well-performing sales forecasting model that can predict the future sales of mini-courses accurately. This will enable the e-learning company to optimize its operations, improve revenue forecasting, and make data-driven decisions to enhance customer experience and business growth. Additionally, the project will also provide valuable insights into sales patterns across different countries, stores, and products, helping the company tailor its marketing strategies and offerings to specific regions and customer preferences.

Possible Framework :

Step 1: Data Preparation and Exploration

1. Import necessary libraries and packages.
2. Load the training and test datasets into Pandas DataFrames.
3. Explore the dataset to understand its structure and basic statistics.
4. Handle missing values and data types if required.
5. Visualize the sales distribution and trends over time using line plots and KDE plots.

Step 2: Feature Engineering

1. Identify categorical and numerical features in the dataset.
2. Create a list of categorical and numerical features for preprocessing.
3. Convert the date column to the appropriate datetime format.
4. Extract additional features from the date column, such as day, month, year, and day of the week.

Step 3: Preprocessing and Encoding

1. Separate the target variable (number of courses sold) from the input features.
2. Split the training dataset into features (X) and target (y).
3. Apply encoding techniques like GLMMEncoder to handle categorical variables.
4. Prepare the training dataset for model training.

Step 4: Model Selection and Evaluation

1. Define a function for evaluating the model using SMAPE (Symmetric Mean Absolute Percentage Error).
2. Initialize a list of regression models to be evaluated, such as LGBMRegressor, CatBoostRegressor, and HistGradientBoostingRegressor.
3. Implement cross-validation using TimeSeriesSplit to train and evaluate each model.
4. Compare the performance of each model using the mean SMAPE score and standard deviation.

Step 5: Model Training and Prediction

1. Select the best-performing model based on cross-validation results.
2. Train the selected model on the entire training dataset.
3. Make predictions on the test dataset using the trained model.
4. Apply multipliers to the predicted values based on specific countries' factors (optional).

Step 6: Submission

1. Create a submission file containing the predictions for the test dataset.
2. Save the submission file in the required format (CSV) for submission to the sales forecasting competition.

Step 7: Visualization of Predictions

1. Plot the predicted sales over time for different countries using line plots.
2. Visualize the forecasted sales trends to identify any country-specific patterns.

Step 8: Conclusion and Future Work

1. Summarize the findings and performance of the sales forecasting model.
2. Discuss potential improvements and future work for enhancing the model's accuracy.
3. Highlight the practical implications of the model's predictions for the e-learning company's business operations and strategies.

Step 9: Final Remarks

1. Conclude the project with final remarks and acknowledgments for contributions.
2. Provide suggestions for further research and development in sales forecasting for e-learning companies.

Note:

- Proper comments and documentation should be added throughout the code to improve code readability and understandability.
- The code should be modular and well-structured, following best practices for code organization.
- All steps should be executed in a logical sequence, ensuring each step's successful completion before moving to the next one.

- Data preprocessing and feature engineering steps are critical for model performance, so attention should be given to data cleaning and feature selection.
- Model evaluation and selection play a vital role in the project's success, and different models should be compared before finalizing the best one for predictions.

Code Explanation :

*If this section is empty, the explanation is provided in the .ipynb file itself.

Step 1: Importing Libraries and Loading Data The code starts by importing necessary libraries and packages required for data manipulation, visualization, and machine learning. Some of the key libraries used are pandas, numpy, seaborn, and scikit-learn. Next, it loads the training and test datasets into Pandas DataFrames.

Step 2: Data Exploration and Visualization After loading the data, the code performs data exploration to understand the structure and basic statistics of the datasets. It provides information like the number of records, number of unique values, percentage of unique values, and the number of missing values for each column in the training and test datasets. This helps us get a quick overview of the data quality and distribution.

Step 3: Feature Engineering Feature engineering involves creating new features or modifying existing ones to improve the model's performance. In this code, the date column is converted to the appropriate datetime format to work with time series data. Additional features like day, month, year, and day of the week are extracted from the date to capture temporal patterns.

Step 4: Preprocessing and Encoding Preprocessing prepares the data for model training. The code separates the target variable (number of courses sold) from the input features. It then applies encoding techniques like GLMMEncoder to handle categorical variables (country, store, and product). Encoding helps convert categorical data into numerical form, which machine learning models can work with.

Step 5: Model Selection and Evaluation Model selection involves choosing the best algorithm for making predictions. The code defines a function to evaluate the model's performance using SMAPE (Symmetric Mean Absolute Percentage Error), a metric suitable for sales forecasting tasks. Then, it initializes a list of regression models to be evaluated, such as LGBMRegressor, CatBoostRegressor, and HistGradientBoostingRegressor. The cross-validation technique TimeSeriesSplit is used to train and evaluate each model. This helps us understand how well each model performs on unseen data.

Step 6: Model Training and Prediction After evaluating different models, the code selects the best-performing model based on the cross-validation results. It trains the selected model on the entire training dataset and makes predictions on the test dataset. Optionally, the code applies multipliers to the predicted values based on specific factors for different countries. This step customizes the predictions for each country to account for different market conditions.

Step 7: Submission Once the model has made predictions on the test dataset, the code creates a submission file containing the predicted values. The file is saved in the required CSV format for submission to the sales forecasting competition.

Step 8: Visualization of Predictions The code includes visualizations to understand the forecasted sales trends over time for different countries. Line plots are used to show the predicted sales for each country, allowing us to identify any country-specific patterns in the forecasts.

Step 9: Conclusion and Future Work Finally, the code concludes with a summary of the model's performance and findings. It may discuss potential improvements and future work to enhance the model's accuracy further. Additionally, it highlights the practical implications of the model's predictions for the e-learning company's business operations and strategies.

Step 10: Final Remarks The last step of the code provides final remarks and acknowledgments for any contributions made during the project. It may suggest areas of further research and development in sales forecasting for e-learning companies.

Overall, this code demonstrates a comprehensive approach to sales forecasting using various machine learning models and techniques. It takes the data through data exploration, preprocessing, feature engineering, model selection, and evaluation to arrive at accurate predictions for the number of courses sold.

Future Work :

Sales forecasting is a dynamic process, and there are several avenues for future work to enhance the accuracy and applicability of the predictions. Below are some detailed steps and ideas for future work in this project:

Step 1: Data Collection and Enrichment:

- Obtain additional data related to marketing campaigns, seasonal events, and other external factors that may impact course sales.
- Incorporate economic indicators, such as GDP, inflation rates, and consumer confidence, to capture broader market trends.
- Consider including customer demographics and behavior data to better understand customer preferences and tailor forecasts accordingly.

Step 2: Feature Engineering and Selection:

- Explore advanced feature engineering techniques, such as time lag features, moving averages, and exponential smoothing, to capture temporal patterns and trends.
- Perform feature importance analysis to identify the most influential features for sales predictions.
- Implement feature selection methods to eliminate irrelevant or redundant features, which can improve model efficiency and generalization.

Step 3: Model Ensemble and Stacking:

- Experiment with model ensembles to combine the predictions of multiple models for more robust and accurate forecasts.
- Explore stacking techniques that use the outputs of multiple base models as input for a meta-model, allowing the meta-model to learn from the strengths of individual models.

Step 4: Hyperparameter Tuning:

- Conduct extensive hyperparameter tuning for each selected model to find the optimal parameter values that yield the best performance.
- Utilize advanced optimization algorithms like Bayesian optimization or genetic algorithms to automate the hyperparameter search process.

Step 5: Time Series Forecasting Models:

- Investigate the use of more specialized time series forecasting models like Prophet or ARIMA, which are designed to handle time-dependent data more effectively.
- Implement neural network-based models like Long Short-Term Memory (LSTM) networks or GRU (Gated Recurrent Unit) networks, which can capture long-term dependencies in time series data.

Step 6: Advanced Encoding Techniques:

- Experiment with different encoding techniques for categorical variables, such as Leave-One-Out (LOO) encoding or Target Encoding with smoothing.
- Explore entity embeddings to represent categorical variables as dense vectors in high-dimensional space, which can capture complex relationships between categories.

Step 7: Cross-Validation Strategies:

- Consider alternative cross-validation strategies like TimeSeriesSplit with different time window configurations to validate the models' performance under varying temporal contexts.
- Implement rolling cross-validation to simulate real-world forecasting scenarios, where the model is updated with new data over time.

Step 8: Outlier Detection and Treatment:

- Implement outlier detection techniques to identify and handle extreme sales values that may impact the model's accuracy.
- Use robust statistical methods or clustering algorithms to group similar sales patterns and handle outliers more effectively.

Step 9: Model Interpretability:

- Employ techniques like SHAP (SHapley Additive exPlanations) or LIME (Local Interpretable Model-agnostic Explanations) to gain insights into model predictions and understand the key drivers of sales forecasts.

Step 10: Business Scenario Simulations:

- Conduct scenario analysis and simulations to evaluate the impact of different business decisions on sales forecasts.
- Use the forecasts to optimize inventory management, marketing budgets, and resource allocation for the e-learning company.

Step-by-Step Guide:

1. Start by collecting additional relevant data to enrich the existing dataset.
2. Perform feature engineering to create new features and time-related variables from the date column.
3. Evaluate different feature selection methods to identify the most important features for sales forecasting.
4. Experiment with various time series forecasting models and model ensembles.
5. Conduct hyperparameter tuning for each selected model to optimize its performance.
6. Explore advanced encoding techniques for categorical variables.
7. Implement alternative cross-validation strategies to validate the models' performance.
8. Use outlier detection methods to handle extreme sales values.
9. Employ model interpretability techniques to gain insights into the models' predictions.
10. Conduct scenario analysis and simulations to apply the forecasts in practical business scenarios.

By following these steps and implementing the future work ideas, the sales forecasting model for e-learning can be continuously improved to provide more accurate and actionable predictions, assisting the e-learning company in making informed decisions and achieving its business goals.

Concept Explanation :

Welcome to the exciting world of CatBoost, where we'll embark on a journey to predict course sales like a pro! Imagine we are running an e-learning company, and we want to know how many students will enroll in our courses in the future. Our goal is to predict the number of course sales accurately so that we can prepare enough resources and ensure everyone gets a seat in our virtual classroom.

Now, CatBoost is not about training cats to predict sales (that would be a catastrophe!). Instead, it's an incredibly powerful algorithm designed to handle tabular data, like our sales dataset. And guess what? It's developed by the brilliant folks at Yandex, a Russian tech company. They must have thought, "Hey, let's build an algorithm that works as gracefully as a cat, with all its agility and power!"

So, what is CatBoost? CatBoost is a gradient boosting algorithm, which means it's like a team of energetic cats working together to solve our sales forecasting challenge. Each cat in the team is a decision tree, and they collaborate to make predictions.

But wait, what's a decision tree? Think of it as a flowchart where we ask a series of questions to arrive at a decision. For instance, our first question could be, "Is the course price below $50?" If the answer is "Yes," the cat (decision tree) may predict a certain number of sales, and if the answer is "No," it may predict a different number. So, each cat (tree) has its unique way of making predictions based on the questions it asks.

Okay, but why "gradient boosting"? Ah, this is where the magic happens! Gradient boosting is like training our team of cats to learn from their mistakes. Imagine our e-learning company hosts a cat competition, and each cat (tree) takes a turn making predictions. After each turn, the cats observe how well they did and learn from their errors. Then, in the next round, each cat focuses on the areas where they didn't perform well, getting better and better with each iteration.

That sounds like a smart bunch of cats! But what's special about CatBoost? Great question! CatBoost has a few extraordinary features that make it a superstar in the world of machine learning:

1. **Categorical Variable Handling:** Most algorithms don't like categorical variables (like "country," "store," and "product" in our dataset). But CatBoost can gracefully

handle these categorical cats! It can automatically convert them into numerical values without losing any valuable information.

2. **Built-in Cross-Validation:** CatBoost is smart enough to perform its own cross-validation, which is like checking how well the cats can predict sales on unseen data. This helps prevent overfitting, where the cats may become too fixated on our training data and lose sight of the bigger picture.
3. **Robust to Overfitting:** Overfitting can be a menace! It's like when our cats memorize all the answers instead of understanding the underlying patterns. But CatBoost is vigilant and knows how to prevent overfitting, so we get more accurate predictions.
4. **Handling Missing Data:** Cats hate missing data as much as we do! Luckily, CatBoost can handle missing values in our dataset like true professionals, making our job much easier.
5. **In Summary:** CatBoost is like a bunch of smart and agile cats working together to predict course sales. It handles categorical variables with ease, avoids overfitting, and even deals with missing data like a boss. It's an all-in-one package of speed, accuracy, and intelligence!

So, next time we need to predict course sales, let's call on our CatBoost team of furry data scientists. They'll make sure we're prepared with enough seats for all the eager learners in our virtual classroom. Happy predicting, and may the cats be with you! ⍰⍰

Exercise Questions :

1. **Question: What is the purpose of using CatBoost in the Mini Course Sales Forecasting project, and how does it handle categorical variables?**

Answer: The purpose of using CatBoost in this project is to predict course sales accurately. CatBoost handles categorical variables by automatically converting them into numerical values using techniques like Target Encoding and One-Hot Encoding, preserving the essential information in these variables.

2. **Question: How does the DateProcessor class in the code help in preparing the data for training the CatBoostRegressor?**

Answer: The DateProcessor class transforms the 'date' column in the dataset into separate columns for day, month, year, and day of the week. This transformation helps CatBoostRegressor to better understand the temporal patterns in the data, leading to more accurate predictions.

3. **Question: What is the significance of the TimeSeriesSplit used for cross-validation in this project?**

Answer: TimeSeriesSplit is used for cross-validation in this project because the data is time-dependent. It ensures that the training data comes before the validation data, simulating the real-world scenario where predictions are made on unseen future data.

4. **Question: Explain the smape function used in the code for evaluating the model's performance.**

Answer: The smape function calculates the Symmetric Mean Absolute Percentage Error (SMAPE) between the actual sales and the predicted sales. It measures the percentage difference between the actual and predicted values, making it suitable for evaluating forecast accuracy.

5. **Question: Why is it important to apply the log transformation to the target variable 'num_sold' before training the CatBoost model?**

Answer: Applying the log transformation to 'num_sold' helps in handling skewed target variables and stabilizes the variance. This makes the model more robust and improves its performance in predicting sales accurately.

6. **Question: What is the role of the multipliers function in the code, and how does it affect the final sales predictions?**

Answer: The multipliers function adjusts the predicted sales for different countries based on their unique characteristics. It allows us to fine-tune the model's predictions to match the specific market behavior in each country.

7. **Question: In the context of the Mini Course Sales Forecasting project, why is it necessary to avoid overfitting, and how does CatBoost prevent it?**

Answer: Overfitting occurs when the model becomes too complex and fits the training data too closely, resulting in poor performance on unseen data. CatBoost prevents overfitting through techniques like gradient boosting, regularization, and built-in cross-validation.

8. **Question: How would you modify the Mini Course Sales Forecasting project to handle additional features, such as 'student demographics' and 'course content'?**

Answer: To handle additional features, you would need to preprocess the new data and encode categorical variables appropriately using techniques like Target Encoding or One-Hot Encoding. Then, include the new features in the model training and prediction steps.

9. **Question: What other evaluation metrics could be used besides SMAPE to assess the performance of the CatBoost model in this project?**

Answer: Besides SMAPE, other evaluation metrics like Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Absolute Percentage Error (MAPE) could be used to assess the model's performance.

10. **Question: How would you interpret the feature importances obtained from the CatBoost model to gain insights into the sales forecasting?**

Answer: The feature importances indicate the relative importance of each feature in the model's predictions. Higher feature importance means the feature has a stronger influence on sales predictions. By analyzing these importances, we can identify the most influential factors affecting course sales and gain valuable insights for decision-making.

These intermediate-level exercise questions and answers cover various aspects of the Mini Course Sales Forecasting project and help deepen the understanding of CatBoost, cross-validation, evaluation metrics, and model interpretation. Happy learning and keep exploring the world of data science!