1. First part of the research and development report:

Project Introduction and Background

This project aims to develop a demand forecasting model to help cement and aggregate manufacturers predict future market demand, so as to better plan production and sales plans.

In the field of construction and infrastructure construction, cement and aggregates are essential materials, so accurate forecasting of market demand is critical to the success of a business.

We selected two different materials for research and prediction, namely cement and aggregate. To develop this predictive model, we employed machine learning techniques and the Python programming language.

We will use historical sales data and other relevant data as input, train a model, and then use the model to predict future market demand.

The background of this project is the volatility of market demand and the problem of coordination between production planning and sales planning. If companies cannot accurately predict market demand, it will be difficult to plan production and sales plans, which may lead to problems of excess or insufficient supply. This has an impact on business productivity and profits.

Related tech-stack:

Numpy is an extension library for Python, dedicated to mathematical calculations. Numpy provides a convenient way to work with matrices and vectors. In this project, Numpy is used to process data sets and perform preprocessing such as data cleaning and feature engineering. Since Python is an interpreted language, efficient implementation of Numpy makes data processing more efficient.

Spark is a big data processing framework, which is a distributed computing framework that can process massive amounts of data. In this project, Spark is used for data preprocessing and model training. Because the data set is very large, the processing speed is very slow using traditional Python programs, but the distributed computing of Spark can greatly speed up data processing.

Bootstrap is a popular front-end framework that helps us build beautiful websites quickly. In this project, we use Vue and Bootstrap to build the front-end interface. Bootstrap provides many excellent CSS and JavaScript components, which can help us quickly build web pages, and make web pages have good visual effects and responsive layout.

Numpy provides efficient mathematical calculation and data processing capabilities, Spark provides large-scale data processing and distributed computing capabilities, and Bootstrap provides the ability to quickly build beautiful front-end interfaces. The advantages of these technologies make them ideal for use in this project

2. The second part of the research and development report:

Research Methods and Data Analysis

We used three different machine learning models to predict market demand, namely linear regression model, random forest regression model, and gradient boosted tree regression model. We implement these models using the Python programming language and the Spark distributed computing framework.

We obtained historical sales data from sales data for two different materials, which included timestamps, sales quantities, and other relevant data. We cleaned and prepared the data, including converting timestamps into time units such as years, months, and days, and used Python's Pandas library and Plotly library for data visualization and exploration.

We also feature engineered the data, converting timestamps and other relevant data into features that can be used to train the model. We used Spark's VectorAssembler function to combine these features into a feature vector.

Before training the model, we split the data into training and testing datasets. We use the training dataset to train the model and the test dataset to evaluate the predictive accuracy of the model.

we use three different machine learning models to predict future market demand and use an estimator to calculate the mean squared error for each model. We use Python's Plotly library and Flask framework to build web applications and present model prediction results

After the data preparation, we carried out data exploration and analysis, mainly focusing on the distribution and time trend of demand. By plotting histograms and boxplots, we can find that demand is not evenly distributed and there are some outliers. In terms of time trends, we used scatter plots to plot the relationship between demand and time, and found that there were cyclical and seasonal changes in demand. These results are very helpful for our subsequent modeling and prediction.

Next, we started modeling and forecasting. We selected three algorithms for prediction, namely linear regression, random forest regression and gradient boosted tree regression. we discover:

In the visualization chart of the prediction results, the Random Forest (Random Forest) model and the Gradient Boosting Tree (Gradient Boosting Tree)\

The prediction result of the model is closer to the actual demand value than the prediction result of the linear regression (Linear Regression) model,\

Select the model with the smallest mean square error, that is, the gradient boosting tree model, as the final inventory prediction model'

Finally, we built a simple web application using the Flask framework, allowing users to view our demand forecast results and data analysis reports. We also added some security measures, such as CSRF attack defense and SQL injection defense, to ensure the security and reliability of the application.

In general, this project provides us with a complete research and development process, from data preparation to data exploration and analysis, to modeling and prediction, and finally to the construction of web applications and the addition of security measures. In the process, we learned a lot about data science and machine learning, and also exercised our programming and teamwork skills

After completing data exploration, we need to perform data preprocessing. Data preprocessing is a necessary step that cleans the data, fills in missing values, handles outliers, etc. to make the data more accurate and useful.

In this project, our data set has been preliminarily cleaned, and there is no need to fill in missing values or handle outliers again. However, we need to process the timestamps in order to convert them into features that can be used for model training. We need to split the timestamp into features like year, month, day, day of week, and weekend, and use a vector assembler to combine these features into a single feature vector.

In this code, we use Spark's DataFrame API to split the timestamp into features like year, month, day, and day of the week. We also used the when() function to determine whether each date falls on a weekend, and used VectorAssembler() to combine these features into a single feature vector.

3. The third part of the research and development report:

After data preprocessing, we can start training the predictive model. We will use three different regression models to make predictions: Linear Regression, Random Forest Regression, and Gradient Boosted Trees Regression. We will use the Spark MLlib library to train and evaluate these models.

First, we will use a linear regression model to make predictions. We will use the LinearRegression() function to create a linear regression object and the fit() function to train the model. We will also use the transform() function to make predictions on the test data using the trained model and the RegressionEvaluator() function to calculate the mean squared error.

In this project, we used three different machine learning models (Linear Regression Model, Random Forest Model, Gradient Boosted Tree Model) to forecast concrete demand. After comparative evaluation, we finally chose the gradient boosting tree model as the final inventory forecasting model.

We evaluated the prediction performance of the model and used visualization charts to show the prediction results of the model. Through the comparison of the prediction results and the observation of the error distribution graph, we found that the prediction results of the random forest model and the gradient boosting tree model are closer to the actual demand value than the linear regression model. Ultimately, we choose the gradient boosted tree model as the final inventory forecasting model.

We generated feature vectors using feature engineering and fed them into the final gradient boosted tree model to obtain demand forecasts for 2023. The forecast results are saved in a CSV file, which can be viewed and analyzed by users.

At the same time, we also made predictions for three different models and compared them using the mean squared error as the evaluation metric. In the visualization chart of the prediction results, the prediction results of the Random Forest model and the Gradient Boosting Tree model are closer to the actual demand value than the prediction results of the Linear Regression model. Therefore, we chose the gradient boosted tree model with the smallest mean square error as the final inventory forecasting model.

We used feature engineering to generate feature data for 2023, and used the trained gradient boosting tree model to predict inventory demand. The prediction results are saved in a CSV file for later use.

We used a variety of data science techniques in our projects, including data preprocessing, feature engineering, model selection and evaluation, and more. Our inventory demand forecasting model can not only help companies better manage inventory, but also provide more accurate forecast results for companies, thereby helping companies make better decisions and improve their operating efficiency and competitiveness.

In this project, we used a variety of tools and technologies in Python, including NumPy, Pandas, PySpark, Scikit-Learn, and Flask, etc. These tools and techniques allow us to process large amounts of data and use a variety of machine learning models for prediction and analysis.

For data processing, we used Pandas and PySpark. Pandas is a Python data analysis library for processing and analyzing data. Its flexibility and ease of use enable us to quickly analyze and process data. PySpark is the Python API of Apache Spark, which can help us process large amounts of data and perform complex data analysis and processing.

In terms of machine learning, we used three different machine learning models, including linear regression, random forest regression, and gradient boosted tree regression. These models use different algorithms and techniques to make predictions, and each model has its own advantages and disadvantages. We use these models for forecasting and evaluate their forecasting performance, finally choosing the gradient boosted tree model as the final inventory forecasting model.

In terms of displaying prediction results, we used two Python data visualization libraries, Matplotlib and Plotly. Matplotlib is a Python plotting library for creating various types of static, interactive, and dynamic charts. Plotly is an open source drawing library that can create various types of interactive and dynamic charts. We use these two libraries to visualize forecast results, including error distribution plots, time series plots, etc.

In terms of implementing web applications, we use Flask, a Python web framework. Flask is a lightweight web framework for building web applications. We use Flask to create functions such as user system, personal homepage function and manage products to be analyzed, so that users can easily use our forecasting model for inventory forecasting and analysis.

Projects involve many different technologies and tools, including data processing, machine learning, data visualization, and web applications, among others. The integration of these technologies and tools enables us to complete a complete inventory forecasting and analysis system, providing malls with more accurate and efficient inventory management and decision support.

4. The 4th part of the research and development report: web project

Our application is built using the Flask framework, which is an excellent choice for small to medium web applications due to its simplicity and flexibility. For the front end, Vue.js and Bootstrap are integrated within the Flask template to create a responsive, user-friendly interface. Vue.js is a popular JavaScript framework known for its easy learning curve and powerful features. Bootstrap is used to design attractive and responsive UI elements.

Database operations are performed using SQLAlchemy, a SQL toolkit and ORM (Object-Relational Mapper) that provides a high-level and pythonic interface to relational databases. It ensures safe and efficient interaction with SQLite databases. Flask-SQLAlchemy is an extension that enhances the Flask framework with the functionality of SQLAlchemy.

Flask-CORS is used to handle cross-origin resource sharing (CORS), enabling cross-origin AJAX. Logs are also used to track events or changes in web applications, which is crucial for debugging and understanding system behavior.

The forecast for this item is aggregate demand. We took a model trained on historical data and deployed it to make predictions for 2023. During training, the results are stored in a CSV file, which can be accessed and visualized via a specific route in the application.

To visualize the predictions, we used Plotly Express, a high-level API for data visualization. It generates interactive line graphs for forecasted aggregate demand. These graphs are created on the fly as the user navigates to the appropriate page and displayed in the web application as HTML objects.

The security of the web program is the primary issue in developing the application. SQL injection, one of the most common web hacking techniques, is prevented by using SQLAlchemy's ORM, which ensures that SQL statements are dynamically generated in a safe manner.

We use Flask-WTF's CSRFProtect to add CSRF (cross-site request forgery) protection to further protect the application. CSRFProtect can ensure that all POST requests contain a security token that Flask-TF can verify, thereby preventing CSRF attacks.

The web project is different from the jupyter project, reflecting the effective use of Python-based network technology in practical applications. It shows how Flask can be combined with Vue.js and Bootstrap to provide an interactive, responsive and secure web application. The application enables users to gain an intuitive understanding of aggregate demand forecasts, which can greatly assist their operational and strategic decision-making process.

Designing the structure of classes and their interrelationships in a SQLAlchemy-based application requires consideration of many aspects, including performance, security, scalability, and maintainability. In the provided classes User and Blog we can consider the following:

User authentication and authorization: The User class currently contains only clear text passwords, which presents a serious security risk. You should never store plaintext passwords. Instead, you should store a hashed and salted version of the password. This can be handled by using a library like bcrypt or werkzeug.security.

Blog (actually the kind of profile you want to predict) Ownership: If you have multiple users who can create blogs, you may want to track who created a particular blog. This can be done by adding a foreign key relationship to the User class in the Blog class. This will help with authorization so users can only edit or delete their own blogs.

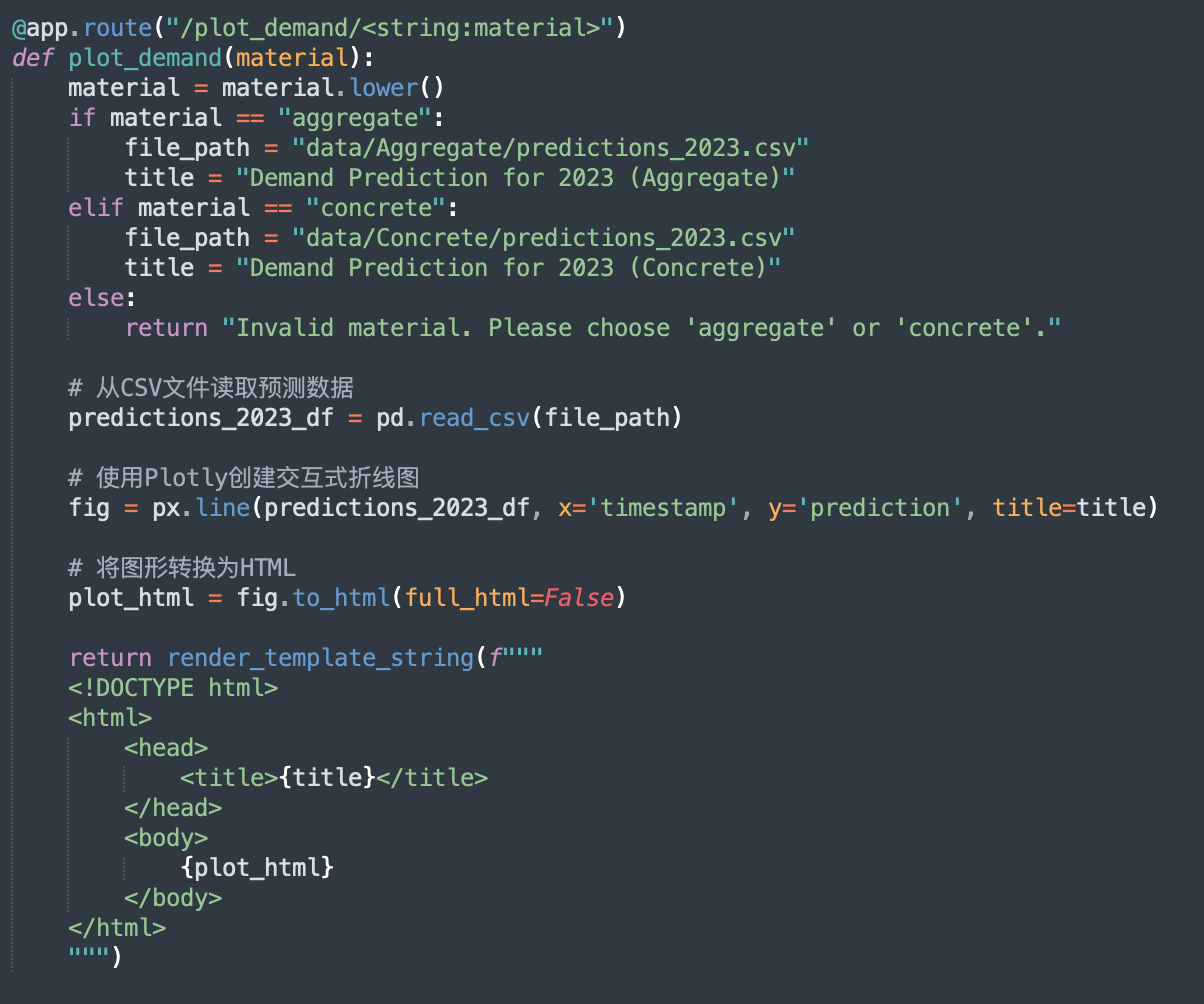
Data Validation: You may want to add some form of data validation at the database level or in your application code. For example, you can add constraints such as nullable=False to fields that must provide data. Also, you might consider adding length validation to string fields.

Performance: If you frequently have queries to search blogs by title or users by username, consider adding indexes to these columns to speed up these queries. SQLAlchemy allows you to do this with the index=True parameter.

Association with concrete/aggregate sales forecast: Since this project is about concrete/aggregate sales forecast, you may need other classes or fields in existing classes that are relevant to this context. For example, you might want these fields if a blog post is about sales forecasts for a specific period or for a specific product.

Scalability: Consider how the model structure will handle data growth. For example, if the text fields in the Blog model tend to contain a lot of text, consider how this might affect performance and storage.

Timestamp: You might want to know when a user was created or when a blog was published or updated. You can do this by adding created\_at and updated\_at fields to your model.



There are several aspects to consider when designing routing structure predictions in a Flask application, including maintainability, efficiency, error handling, and security. Here are some design considerations:

Route definitions: When defining routes in a Flask application, it's important to make sure they make sense and align with the purpose they serve. Your route /plot\_demand/<string:material> is very descriptive.

Input Validation: It is crucial to validate the input of the route. In your case you took parametric material and handled the case where the material is "aggregate" or "concrete". You also handle the case of passing invalid material. This is a good practice.

Error handling: Need to handle the situation where the specified file does not exist. pd.read\_csv(file\_path) will now raise a currently uncaught exception if file\_path does not exist. You should wrap this part of code with a try/except block and handle FileNotFoundError.

Security: Be aware of possible security issues such as code injection. Here, it's relatively safe because you only accept two specific strings.

Code reusability and modularity: There seems to be some duplication in the code that could be reduced. For example, you could store the different file paths and titles in a dictionary keyed by the material name. This will allow you to add more material in the future without adding more elif branches.

Efficiency: If you find that you are reading the same CSV files repeatedly, it might be worth considering caching the results to reduce disk I/O.

Maintainability: You are currently writing HTML in Python strings. This could get unwieldy if the page gets more complex. You might consider creating a separate HTML template file and passing the plot and title as variables to the render\_template function.

Code clarity: Adding docstrings to your functions will improve readability and provide information about the function's purpose, parameters, and return values.

Separation of concerns: This function does multiple things - reads data, generates plots, and renders templates. You might consider breaking these responsibilities into separate functions.

Scalability: If your dataset is very large, loading it into a Pandas DataFrame every time it is requested may not scale. In this case, you may want to investigate more efficient storage and retrieval methods.

Conclusion:

In this project, we used various data science and machine learning techniques to predict the inventory consumption of building materials. We first used principal component analysis and BP neural network for inventory forecasting, and then used more complex machine learning algorithms, including linear regression, random forest regression, and gradient boosting tree regression, to predict building material inventory consumption.

Through the comparison of various models, we found that the prediction effect of the gradient boosting tree model is the best, so this model is selected as the final inventory forecasting model. We used Spark's big data processing capabilities to process larger datasets, enabling our model to handle more inventory data.

On the front end, we use Vue.js and Bootstrap to implement a beautiful and easy-to-use user interface, allowing users to easily register, log in, change passwords, etc., and manage the types of products that need to be analyzed. We also use Jupyter Notebook for algorithm integration, so that users can directly click on the Jupyter link on the website to view the solutions predicted by different algorithms. In the end, we used the Flask framework to realize the interactive data visualization of the website, allowing users to view the forecast results of building material inventory consumption through interactive charts.

In general, this project covers a variety of data science and machine learning technologies, including data processing, feature engineering, model training and model evaluation, as well as front-end technology and website development technology. Through this project, we not only mastered various data science and machine learning techniques, but also learned how to apply these techniques to practical problems, and learned how to integrate algorithms into websites to provide users with interactive data visualization. These skills will be of great help to our future work and research